TU WIEN Informatics

Pole-arina: Deep Learning–Based Coaching System for Pole Dancing Technique

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieurin

im Rahmen des Studiums

Visual Computing

eingereicht von

Katharina Scheucher, BSc
Matrikelnummer 11809620

an der Fakultät für Informatik

der Technischen Universität Wien

Betreuung: Dr. Peter Kán
Mitwirkung: Dr. Diana Marin

Wien, 8. September 2025

Katharina Scheucher	Peter Kán

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

TU WIEN Informatics

Pole-arina: Deep Learning–Based Coaching System for Pole Dancing Technique

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieurin

in

Visual Computing

by

Katharina Scheucher, BSc
Registration Number 11809620

to the Faculty of Informatics

at the TU Wien

Advisor: Dr. Peter Kán
Assistance: Dr. Diana Marin

Vienna, September 8, 2025

Katharina Scheucher | Peter Kán

Technische Universität Wien
A-1040 Wien ▪ Karlsplatz 13 ▪ Tel. +43-1-58801-0 ▪ www.tuwien.at

Erklärung zur Verfassung der Arbeit

Katharina Scheucher, BSc

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.
Ich erkläre weiters, dass ich mich generativer KI-Tools lediglich als Hilfsmittel bedient habe und in der vorliegenden Arbeit mein gestalterischer Einfluss überwiegt. Im Anhang „Übersicht verwendeter Hilfsmittel“ habe ich alle generativen KI-Tools gelistet, die verwendet wurden, und angegeben, wo und wie sie verwendet wurden. Für Textpassagen, die ohne substantielle Änderungen übernommen wurden, haben ich jeweils die von mir formulierten Eingaben (Prompts) und die verwendete IT- Anwendung mit ihrem Produktnamen und Versionsnummer/Datum angegeben.

Wien, 8. September 2025

Katharina Scheucher

Danksagung

Ich danke meinen Betreuern **Peter Kán** und **Diana Marin** von Herzen. Peter, mein Hauptbetreuer, hat mich mit schneller und durchdachter Beratung begleitet und seine Erfahrung großzügig eingebracht. Diana war eine außergewöhnliche Unterstützerin: Unsere wöchentlichen Besprechungen, ihr detailliertes Feedback und ihre Bereitschaft, jederzeit zu helfen (insbesondere in der Schlussphase) waren von unschätzbarem Wert.

Mein Dank gilt zudem allen Teilnehmenden, die zur Datensammlung und zur Nutzerstudie beigetragen haben, sowie der Studioleitung von *PoleDanceVienna* für die großzügige Bereitstellung der Studioräume für die Datenerhebung und Evaluation.

Abschließend gilt mein tiefster Dank meinem Partner, meinen Eltern und meinen Schwestern für ihre unerschütterliche Unterstützung, Geduld und Ermutigung.

Acknowledgements

I am sincerely grateful to my supervisors, **Peter Kán** and **Diana Marin**. Peter, as my primary supervisor, provided swift and thoughtful guidance, sharing his experience generously throughout this project. Diana was an exceptional supporter: our weekly meetings, her detailed feedback, and her readiness to help at any time (especially during the final stretch) were invaluable.

My thanks extend to all participants who contributed to the dataset and the user study, as well as to the owner of *PoleDanceVienna* for generously providing studio space for data collection and evaluation.

Finally, my deepest gratitude goes to my partner, my parents, and my sisters for their unwavering support, patience, and encouragement, always and without exception.

Kurzfassung

Diese Arbeit stellt **Pole-Arina** vor, ein markerloses Trainingssystem für statische Pole-Dance-Tricks, das Trainingsvideos analysiert, um den ausgeführten Trick zu erkennen und die Endpose mit transparentem, geometriebasiertem Feedback zu bewerten. Zu diesem Zweck wurde ein domänenspezifischer Datensatz kuratiert und annotiert. Er umfasst 836 Clips von 58 TeilnehmerInnen, die mit entsprechenden Labeln gekennzeichnet sind, welche die Erkennung mehrerer Tricks sowie passiver Zustände ermöglichen. Pole-Arina kombiniert ein leichtgewichtiges bidirektionales LSTM für die frameweise Erkennung mit einer regelbasierten Engine, die trickspezifische Ausrichtungen im Raum, Gelenkausrichtungen und Abstände bewertet. Diese Daten werden für NutzerInnen durch intuitive Visualisierungen und verständliche Tipps zugänglich gemacht. Das Modell erreichte eine accuracy von 93,82% pro Frame über alle Klassen hinweg und eine accuracy von 98,74% für trickspezifische Klassen. In einer kontrollierten between-groups Anwenderstudie wurde Pole-Arina mit der traditionellen Video-Selbstbewertung verglichen. Die TeilnehmerInnen, die Pole-Arina verwendeten, gaben an, dass sie dem Feedback deutlich mehr Vertrauen schenkten und mehr Klarheit darüber hatten, wie sie sich verbessern konnten. Außerdem wurde die Benutzerfreundlichkeit in der Pole-Arina Gruppe höher bewertet. Diese Ergebnisse zeigen, dass Pole-Arina eine genaue Erkennung und umsetzbares Feedback liefern kann, dem die Benutzer vertrauen und das sie verstehen, wodurch strukturiertes Coaching auch außerhalb des Studios zugänglich wird. Diese Arbeit schafft eine praktische Grundlage für KI-Coaching im Pole-Sport.

Abstract

This thesis presents **Pole-Arina**, a marker-less coaching system for static pole dancing tricks that analyzes training videos to recognize the performed trick and grade the final pose with transparent, geometry-based feedback. A domain-specific dataset was curated and annotated for this purpose. It includes 836 clips from 58 participants, labeled with a state scheme that supports multi-trick recognition and explicit background modeling. Pole-Arina combines a lightweight bidirectional LSTM for frame-wise recognition with a rule engine that evaluates trick-specific orientations, joint alignments, and proximities, rendering interpretable overlays and concise tips. The model achieved 93.82% per-frame accuracy across all classes and 98.74% trick-only accuracy on end-pose frames. A controlled between-groups user study compared Pole-Arina against traditional video self-review. Participants using Pole-Arina reported significantly higher trust in the feedback and greater clarity for how to improve, and rated usability higher. These results indicate that Pole-Arina can deliver accurate recognition and actionable feedback that users trust and understand, making structured coaching accessible outside the studio. This work establishes a practical baseline for AI coaching in pole sports.

Contents

CHAPTER 1

Introduction

Advancements in artificial intelligence (AI) and computer vision are reshaping how athletes and dancers train, offering new possibilities for personalized coaching and performance analysis. From fitness apps providing real-time form corrections [DWDW25] to AI referees and analytics in professional sports [PPW+24, GRRCR23], virtual coaches are increasingly emerging to enhance efficiency and reduce injury risks [MMN+24]. Such systems also aim to enhance exercise enjoyment and to incorporate social aspects [DWDW25]. However, as AI-driven coaching becomes more prevalent, it is crucial to address ethical considerations such as privacy, informed consent, and user trust. Ensuring transparent algorithms and respectful data handling not only meets ethical standards but also improves user acceptance of AI systems [LWHL24]. With thoughtful implementation, deep and machine learning-based tools hold powerful enhancements in complementing a human instructor by processing complex movement data and delivering instant feedback for all types of training.

AI-powered coaching systems have already proven their potential in sports and dance evaluation. For instance, computer vision models can evaluate Olympic sport performances and predict judges' scores, providing an objective measure of technique and consistency [PTM17]. In the fitness domain, vision-based assistant applications guide users through exercises such as yoga, weight training, or martial arts [TP23]. Furthermore, marker-less pose estimation can handle complex movements like full-body rotations and self-occlusions to evaluate dance performances [KK18]. Across various domains, pose estimation and deep learning enable the automatic analysis of motion and the generation of real-time feedback. Crucially, unlike traditional marker-based motion capture systems, computer vision approaches enable free movements, making suits or sensors unnecessary [MK23]. This is especially valuable for contact sports, where wearables are impractical to integrate.

1.1 Motivation & Problem Statement

One emerging domain that has received little attention from such AI coaching technologies is pole dancing. In recent years, pole dancing has shed much of its past stigma and evolved into a recognized athletic practice, with a growing community and global competitions. The International Pole Sports Federation (IPSF) was founded in 2009 and subsequently introduced a standardized set of rules and regulations for competitions, marking a fundamental step in establishing pole dance as a serious sport [Fed25]. Although the Global Association of International Sport Federations (GAISF) classified pole dancing as a professional sport, critics argue that hypersexualization is unavoidable [Wea20]. Treating pole dancing explicitly as a sport helps shift expectations toward an inclusive, technique-centered evaluation. This thesis contributes to that shift by developing and studying an objective, transparent coaching system for pole technique.

Pole dancing combines artistry with acrobatics, requiring strength, flexibility, and precise technique. Each pole trick involves complex, full-body movements that often include inversion and intricate transitions. To review their form independently, dancers most often record themselves, both inside and outside formal classes. Nonetheless, subtle body misalignments or incorrect technique often go unnoticed without proper assessment by expert instructors, which can slow progress and increase the risk of injury. A lack of feedback is not only frustrating but also causes falls or chronic strain. While instructors provide immediate guidance in the studio, they are costly and not always available, especially when training at home. Wearable-sensor systems overcome some visibility issues, but restrict movement and interfere with contact between the dancer's skin and the pole, making them unsuited for pole dancing. Recent advances in marker-less pose estimation and sequence models make purely video-based coaching feasible [Qu24].

However, building such a system for pole dancing comes with its own challenges. Pole tricks are highly dynamic and can appear very similar in their early stages, making it challenging to recognize which trick is being performed until key characteristics emerge. Furthermore, the naming conventions of pole tricks are not globally standardized or as established as in other sports. The same trick might have different names across studios or regions, causing dancers to struggle in identifying the trick or searching for a related tutorial. These factors motivate the need for an automated assistant that can accurately classify the performed pole trick, pinpoint technical mistakes, and provide interpretable feedback to improve the dancer's form. Another motivation stems from the author's direct experience teaching pole-dancing classes. Observations of numerous students revealed common struggles with self-review and clarified the value of consistent feedback. In summary, an accessible system outside the studio that does not require wearables and is tailored to the unique demands of pole athletes offers a clear value. It represents a necessary step towards making pole dancing training more accessible.

1.2 Aim of the Work

This thesis addresses the above-mentioned challenges by developing **Pole-Arina**, a marker-less, deep learning-based coaching system for static pole tricks. The main goals of Pole-Arina include: automatic pole trick classification, quality evaluation of the execution, and the delivery of corrective feedback in an intuitive visual format. To achieve this, the system leverages state-of-the-art pose estimation to extract the dancer's skeleton keypoints from ordinary training videos. It analyzes the motion sequence using a domain-specific model and rules. Therefore, Pole-Arina aims to act as an AI-powered pole coach, identifying the trick, evaluating it, and highlighting form deviations. It is not a replacement for real-world instruction when learning new tricks, but it provides guidance in training settings. The focus is on fundamental **static pole tricks** to provide educational value for beginners and intermediate dancers, to build correct technique. This work aims to (1) design a privacy-preserving, interpretable coaching pipeline for pole dancing technique, (2) develop a lightweight temporal recognizer for trick and phase detection, and (3) evaluate the system's effectiveness and usability in realistic training scenarios. Building on these objectives, the research questions can be summarized as follows:

1. How accurately can deep learning models classify pole dancing tricks?
2. To what extent can geometric-statistical scoring identify and quantify deviations from ideal execution of pole dancing tricks?
3. Which type of feedback from a marker-less coaching system can effectively improve a dancer's technique, and how can this improvement be quantified?

By answering these questions, the thesis aims to validate the feasibility of an AI-driven coaching tool in the context of pole sports. It further provides insights into designing effective automated feedback for complex athletic movements.

The evidence is collected along the three research questions:

- **RQ1** via quantitative recognition metrics on held-out data.
- **RQ2** via rule statistics and pose scores for detected end-poses, to identify and quantify deviations from ideal execution.
- **RQ3** via a controlled user study measuring trust & adoption, efficiency, understandability, and usability.

1.3 Contribution

The key contributions of this work include:

- **Pole Dancing Skeleton Dataset:** A novel, **domain-specific dataset** of pole dancing performances was created to support model training and evaluation. This dataset encompasses six fundamental static pole tricks, each performed by multiple volunteers of varying skill levels. The data collection process was carried out with strict attention to ethics and privacy. All participants provided informed consent, and the raw video data were handled confidentially. To secure privacy, only the extracted 3D skeleton joint coordinates (with a reconstructed depth value) are included in the released dataset and utilized for analysis. The final dataset consists of 836 video samples, covering both successful and failed attempts across varying experience levels. Each clip is annotated with the trick label and temporal progression.
- **Trick & Phase Recognition Model:** A bidirectional **Long Short-Term Memory (LSTM)** neural network was developed to classify pole tricks and detect their phase progression automatically. During training and inference, the model ingests the entire landmark sequence for a clip and produces a per-frame prediction. The trained model provides an accurate classification pipeline that serves as the backbone of Pole-Arina.
- **Pose-Quality Scoring and Feedback System:** Based on domain knowledge of proper pole technique, a **geometric rule-based scoring** system was designed to evaluate a dancer's performance. Each trick holds unique characteristics that specify body alignment and present opportunities for common instructor critiques. Based on these attributes, custom rules were derived to define a trick and apply geometric computations on the 3D joint coordinates. The output is a set of quantitative scores indicating the performer's deviation from the optimal pose. To aid understanding, the system visualizes the feedback directly on the dancer's video frames as an overlay. It highlights angles, alignments, and distances. The scoring system and its visualization are implemented as part of an interactive feedback interface, enabling users to review their performance frame by frame with guidance on how to improve.

Together, these contributions realize a lightweight, marker-less AI coaching prototype tailored to basic pole dancing tricks. Furthermore, this system offers potential for scalability as it is adaptable to other physical activities such as physiotherapy, gymnastics, or fitness routines.

1.4 Structure of this Thesis

The remainder of this thesis is organized as follows: first, Chapter 2 reviews the state-of-the-art in human motion capture and automated coaching systems. It contrasts marker-based motion capture with marker-less pose estimation approaches, and discusses relevant deep learning models for pose estimation and sequence analysis. Furthermore, it surveys prior work on automated feedback in sports and dance, highlighting the implementation of similar concepts in applied domains like gymnastics, yoga, and general

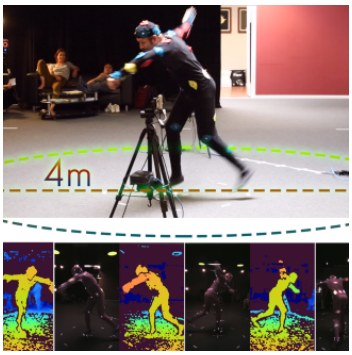
dance performance. Next, Chapter 3 presents a detailed description of the composition of the Pole-Arina dataset. It describes the selection of tricks, the data acquisition process, the annotation protocol for labeling phases, and the feature extraction pipeline using MediaPipe for skeletal data. This chapter also presents ethical considerations and dataset statistics, such as participant demographics and data characteristics. Chapter 4 provides a high-level overview of the Pole-Arina system's design and evaluation methodology. It formally states the problem and introduces the system's overall architecture. The chapter then describes the core methods for trick classification and pose-quality scoring. Chapter 5 dives into the technical implementation of each component. It covers data preprocessing steps, iterative model development, and parameter tuning of the LSTM model, specific rule calculations, and integration into a full-stack prototype. Chapter 6 evaluates the performance of the developed system. First, it presents the quantitative results of the trick recognition model, including accuracy, confusion matrices, and analysis of misclassifications. It further introduces the evaluation strategy by presenting a user study concept. It continues to report on the results, including both objective measurements and subjective feedback from questionnaires. Hypothesis tests and statistical analysis are used to determine the significance of observed improvements. Qualitative observations from the study are also discussed to gain insights beyond the tests. It further outlines potential directions for extending this research. Finally, Chapter 7 summarizes the thesis's findings, reflecting on the extent to which the research questions were answered and the overall success of Pole-Arina in addressing the initial problem.

CHAPTER 2

Related Work

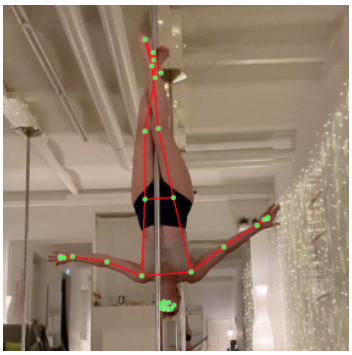
This chapter surveys prior work to set the stage for Pole-Arina. First, it contrasts marker-based, wearable, and marker-less motion capture approaches and summarizes their trade-offs for athletic and dance contexts. Next, it reviews human pose estimation and temporal sequence models, before covering automated coaching systems in sports and applications in dance. This synthesis motivates the choice of marker-less pose estimation with lightweight temporal modeling for pole-dance technique analysis.

2.1 Marker-Based vs. Marker-less Motion Capture



(a) Marker-based pose estimation example by [CZDK21]

(b) Wearable motion capture example, taken from [LX13]



(c) Marker-less motion capture using MediaPipe [Goo25]

Figure 2.1: Visual comparison of different motion capture technologies.

Marker-based optical systems. Optical motion capture (mocap) systems represent the gold standard for capturing human motion with high precision [STL24]. Such marker-based systems can achieve millimeter-level accuracy under controlled conditions, making

them well suited for detailed biomechanics analysis [CECS18]. They can be divided into active and passive marker systems [STL24]. While traditional motion capture systems like Vicon [Vic25] use passive markers that reflect light and are tracked by multiple cameras, active markers used by systems like Optotrak3020 [Nor25] emit light. Although active markers are generally more stable, they rely on additional power supplies and cables, which restrict movement compared to passive solutions.

However, both solutions have limited practicality outside of the lab. Setting up markers on a person is time-consuming and can interfere with natural movement. Markers might slip, require readjustment, recalibration during dynamic movements, or suffer from occlusion and constrained space [STL24]. Even if no cables are involved, attached markers and suits impose physical and psychological constraints on the performer, altering how they move [CECS18]. Therefore, when covered with sensors, dancers and athletes might not perform as usual [MK23]. Markers or wearable systems are particularly problematic in sports like pole dancing, where minimal sportswear is required for grip on the pole. Attached sensors or markers on the dancer's body interfere with the needed contact between skin and pole, while the pole can introduce magnetic interference. Thus, while marker-based systems are often unmatched regarding accuracy, their setup complexity and spatial limitations make them ill-suited to pole dancing.

Wearable inertial systems. IMU suits, or inertial measurement unit suits, are wearable systems that track body movements using sensors to capture data such as acceleration, rotation, and magnetic fields [STL24]. Key advantages are portability and robustness to occlusion. Since no cameras are needed, athletes can move freely in any environment without a direct line of sight. Wearable sensors allow large capture volumes and real-time tracking, making them well-suited for in-situ training like skiing or running. While they impose less interference on athletes, compared to markers, the accuracy is lower. Inertial sensors drift over time, require recalibration, and can be sensitive to magnetic distortions [STL24]. In pole dancing, any device must be tightly secured to withstand inversion and spins, while high-impact or aerial motions (e.g., tumbles, jumps, flips) might still cause a shift or data integration errors. Therefore, while IMU suits are effective for specific use cases, they are ineffective for free-form arts like dance, which require high fidelity and unrestricted movement.

Marker-less computer vision approaches. Advances in computer vision and deep learning have enabled marker-less motion capture using regular camera footage. Frameworks like OpenPose [CHS+19] and DeepPose [TS14] estimate human body positions from video frames without markers. This non-invasive method enables performers to move naturally while being recorded with ordinary cameras or smartphones [CECS18], using algorithms to reconstruct their skeleton motions. This is especially valuable in dance and other contexts, where attached equipment is undesirable. Fueled by deep learning, modern pose estimation models have dramatically improved accuracy and reliability. They achieve high recognition accuracy in various sports scenarios, from single-athlete skill analysis to multi-player game tactics [STL24]. However, limitations remain even

if marker-less methods are more flexible and scalable. Computer vision models require a direct line of view, struggle with occlusions, and are limited to the camera's field of view. Nonetheless, in contexts like pole dancing, marker-less methods represent the most viable approach, as it does not restrain the dancers' movements and leverages the use of simple video recording.

Table 2.1: Marker-based vs. marker-less comparison overview.

Method	Advantages	Drawbacks
Marker-Based	- High accuracy (mm-level) - Detailed 3D data - Well-validated for biomechanics	- Compromise natural movement - Occlusion & limited capture volume - Expensive & requires lab setup
Wearable Sensors (IMU)	- No line-of-sight - Possible real-time feedback - Minimal setup (small sensors)	- Drift & noise issues → recalibration - Can be uncomfortable & restrictive - Sensitive to magnetic interference
Marker-Less	- Enables free movement - Flexible setup - Scales to multiple people & large scenes	- Needs direct line-of-sight - Accuracy depends on algorithm - Real-time bottlenecks

2.2 Pose Estimation & Sequence Models

Human pose estimation. Early advances in computer vision enabled the automatic detection of human joint positions from images or videos. Toshev and Szegedy introduced DeepPose [TS14], a landmark work in formulating pose estimation as a deep neural network regression problem. Their work achieved state-of-the-art accuracy on benchmarks [TS14]. Furthermore, frameworks like OpenPose [CHS+19] have made real-time 2D multi-person pose tracking feasible by utilizing Part Affinity Fields to localize multiple people simultaneously. This and similar CNN-based methods became popular due to their robustness in various environments [ZWC+23]. Additionally, open-source libraries like Facebook AI's Detectron2 [WKM+19] provide keypoint detection models as part of its object detection toolkit. Google's MediaPipe Pose [Goo25] (based on BlazePose [BGR+20]) estimator further optimized pose estimation for mobile and edge devices, outputting 33 body landmarks over 30 FPS on a phone. It further reconstructs 3D pose coordinates (with a relative depth component) from a single RGB camera, enabling real-time posture analysis with only a smartphone camera. These advancements enable the calculation of highly accurate human skeleton data, in either 2D or estimated 3D, which can be obtained by marker-less pose estimation from ordinary video data.

Sequence models. After the computation of landmark sequences, the data can be fed to sequence models to solve classification problems. Traditional Recurrent Neural Networks (RNNs) suffered from vanishing or exploding gradients when learning long sequences [Zar21]. Hochreiter and Schmidhuber introduced the Long Short-Term Memory

(LSTM) network as a solution to overcome this limitation by enforcing constant error flow over time through gating mechanisms [HS97]. LSTMs can maintain long-range dependencies (over 100 time steps in the original work), making them well-suited for complex motion sequences or time-series related data. For instance, an LSTM can aggregate frame-by-frame pose data to recognize a tennis swing or a dance sequence as a whole. However, recent developments introduced an architecture that revolutionized sequence modeling. Vaswani et al. presented the Transformer model [VSP+17], which removes recurrence entirely and relies solely on self-attention mechanisms to capture temporal relationships. Transformers enable greater parallelization during training and achieve superior results in tasks like machine translation as well as motion analysis tasks [Qu24]. In human pose analysis, such models can attend to all timesteps simultaneously, capturing subtle movement patterns that RNNs might miss. However, as expected with deep learning models, they typically require a large dataset to generalize well. On smaller motion datasets, simpler recurrent models can sometimes rival or outperform large pretrained Transformers. For instance, Ezen-Can [EC20] found that a tuned bi-directional LSTM outperformed a fine-tuned BERT (a transformer-based model) on a small action classification task, while also being much faster to train. Thus, model choice should not only be decided on accuracy and performance but also on data size and context. Especially for sports and rehab applications, data is limited and might favor an LSTM-based approach over a data-hungry Transformer model.

Overall, modern pipelines often pair a CNN-based pose estimator with a temporal model to analyze a pose sequence [TP23]. Some systems utilize simple rule-based algorithms on landmark coordinates, while others train models directly on the data [TP23]. Furthermore, hybrid approaches are emerging, combining aspects of different models, to tailor the model architecture directly to the application domain. Qu et al. proposed a TransCNN-DSSS model to analyze dance movements with body dynamic and static streams [Qu24]. By decoupling quality dimensions via an attention mechanism, their model achieved about 90% accuracy in automatically scoring dance performances. This further highlights how different models can complement each other to form the technical foundation for modern motion analysis in various domains.

2.3 Automated Coaching & Feedback Systems in Sports

Towards coaching systems. Advances in pose estimation and sequence modeling have enabled a new generation of automated coaching and feedback systems across a wide range of sports. In traditional coaching, detailed movement analysis was limited to expert eyes or expensive motion-capture setups. With affordable hardware and recent advancements in AI, athletes can now receive real-time technique feedback on demand. As a result, deep-learning-enhanced fitness applications are growing rapidly in popularity, offering highly engaging and personalized coaching experiences to users [DWDW25]. Such apps leverage computer vision methods to monitor the user's movements and provide corrective feedback on form, count repetitions, and suggest workout adjustments. According to a recent user study, the appeal of such AI fitness systems lies in their

interactivity and tailored guidance, which can boost motivation and other aspects of gratification [DWDW25]. Technically, most automated coaching systems share a standard pipeline: pose estimation from video, followed by movement assessment and feedback generation [TP23]. Tharatipyakul and Pongnumkul's [TP23] review revealed that many systems rely on open-source frameworks, such as OpenPose, for human pose estimation to obtain skeleton data. The movement assessment can employ simple rule-based checks or a more complex comparison of an athlete's motion trajectory against an optimal model. Notably, researchers emphasize the importance of feedback clarity, correctness, and ethical consideration as users must trust and understand the AI coach for it to be effective [LWHL24].

Sports applications. In individual sports like weightlifting, yoga, or golf, computer vision systems guide users to refine their form by detecting asymmetries in a yoga pose [BNKB23] or the swing plane of a golf club [LHK22]. Comparatively, in team sports, analysis often goes beyond the single-athlete technique towards tactical insights. With abundant video data for popular sports, an AI-driven system can evaluate how a player's body orientation affects their passing options, or recommend tactical adjustments based on pattern recognition in movement data [PPW+24]. A systematic review by Pu et al. highlights that the explosion of data and the advancement of deep learning methods are transforming soccer analysis and training decisions [PPW+24]. Coaches can receive automated reports on metrics such as distance covered, joint load, or alignment during plays, thereby augmenting their expertise with objective data.

Evaluation & scoring. Another important application is performance evaluation and scoring. AI systems have been developed to score performance by analyzing pose sequences. Parmar and Morris [PTM17] implemented this by training models on Olympic events. Their system learned to predict judges' scores for diving, vault, and figure skating routines from video, using spatiotemporal features and regression models. A comparison between a Support Vector Regression (SVR) and an LSTM framework showed that while the SVR gave slightly better numeric scores, the LSTM was more natural for describing an action and thus better suited for giving qualitative feedback for improvement [PTM17]. Automated scoring is still an active research area. However, results so far show strong correlation with expert evaluations, suggesting AI can objectively standardize aspects of judging that are prone to human bias or error [KK18].

Injury prevention & rehabilitation. Motion analysis further enables the possibility of injury prevention and rehabilitation. By analyzing an athlete's movement pattern over time, models can detect risky mechanics or deterioration that coaches might miss. Recent reviews conclude that machine learning models can significantly improve the accuracy of injury risk assessments by processing complex biomechanical and workload data beyond human capacity [MMN+24]. For instance, such systems might learn that a certain gait asymmetry and jump landing force profile often precede Anterior Cruciate Ligament (ACL) injuries. In rehabilitation, pose estimation systems monitor patients doing therapy

exercises at home, ensuring compliance and correctness. This kind of augmented feedback loop can personalize training loads and prevent injuries by continuously adjusting to the athlete's posture [MMN+24].

In summary, automated coaching systems leveraging pose estimation are increasingly prevalent. They provide immediate, data-driven feedback on technique, reduce dependence on constant human supervision, and can enhance training efficiency and safety. While never aiming to replace human coaches, deep learning-based coaches fully act as intelligent assistants to reinforce proper form and measure performance.

2.4 Applications of Pose Analysis in Dance

Scoring dance performances. Applying pose estimation and automated feedback to dance presents unique challenges and opportunities. Unlike many sports, dance is an artistic performance where quality is judged not only by objective technique but also by expressiveness, musicality, and style. Despite this subjectivity, researchers have shown that computational pose analysis can effectively evaluate and even enhance dance training. A study by Kim and Kim [KK18] introduced a real-time dance evaluation system that utilizes marker-less pose estimation. They developed a camera-based pose tracker that is robust to fast rotations and self-occlusions. For evaluation, they defined a metric to compare a student's motion sequence to a reference sequence, ideal in terms of timing and accuracy. Remarkably, their system's scores had a 98% correspondence with professional judges' evaluations of the same performance [KK18]. Recent advances by Qu [Qu24] proposed a novel Transformer-CNN model that evaluated dance movement quality across multiple dimensions. Their approach breaks down dance quality into factors like accuracy of execution, fluidity of motion, and emotional expressiveness, using an attention-based mechanism to weight each factor. The combined model captures per-frame posture details and temporal dynamics to output an overall performance score. Again, tested against expert ratings, the system achieved an accuracy of 90% in predicting quality rating. While dancers could use such a system to get immediate feedback on form deviations, it also offers potential to provide rich recommendations targeting specific aspects of technique and performance quality.

Dance movement recognition & classification. Beyond scoring entire performances, pose analysis has been used for classifying and recognizing dance movements. Bera et al. [BNKB23] addressed the problem of fine-grained posture recognition in sports, yoga, and dance. They also highlight the scarcity of large public datasets in this domain. Therefore, they introduce a new image dataset for 102 sports actions and 12 dance styles. To solve the classification problem, they implemented a deep CNN with patch-based self-attention to classify poses and styles. Similarly, Agarwal et al. introduced POA-Net [AJJB24], a CNN model for classifying dance poses and activities, with a focus on ballroom dance forms. These classification models enable applications, such as an automatic dance coach,

to recognize the move a student is performing and evaluate it against a domain-specific syllabus.

Real-world employment. Notably, the entertainment industry has already embraced rudimentary pose-based dance feedback in the form of video games. Ubisoft's Just Dance [Ubi09] and similar rhythm games invite players to mimic on-screen choreography and score their performance based on the similarity of movements. Earlier versions relied on handheld motion controllers, but newer systems use camera-based full-body tracking (e.g., Kinect sensor) to evaluate dance moves. While these games are not as precise as state-of-the-art research pose estimators, they demonstrate a mass-market use case of marker-less motion analysis. Therefore, it is a short stretch to imagine a more serious dance training tool that provides dancers with real-time corrections during practice.

Application: Pole dance Despite this growing trend of AI coaching systems, pole dancing remains an area with relatively sparse technological assistance or other related work. In contrast to the evaluation and scoring system, PoeSpin [LCLX25] represents a human-AI collaborative system that turns pole dance movements into poetry. In Li et al.'s [LCLX25] work, they mapped the dancer's poses and motions to poetic verses by using movement as input for a generative art model. As a demonstration, the system captured live performances from a pole dancer to compose verses in real-time, blending physical expression with literary expression. This artistic application underscores that pose estimation is not limited to quantitative evaluation. At least one attempt placed pole dancing into the context of automatic evaluation. Yu [Yu20] proposed a virtual reality-based training system to help students learn pole routines by imitating a virtual instructor in an immersive environment. The system synchronized music and movement in a virtual scene to improve students' sense of rhythm and performance quality. While offering a new teaching medium, it does not provide automated pose feedback under commodity hardware constraints, which is precisely the advantage that Pole-Arina aims to provide.

In conclusion, marker-less pose estimation and AI analysis have shown great success in domains ranging from sports training to dance education. These technologies respect the performer's freedom of movement and capture rich data that can be translated into meaningful feedback. By building on existing methods and addressing pole-specific challenges, the related work sets the stage for Pole-Arina: a deep learning-based coaching system for pole dancing technique.

CHAPTER 3

Pole-Arina: Dataset

This chapter introduces the dataset for the Pole-Arina system. It motivates the selected set of static pole tricks and the recognition task. Furthermore, it outlines two acquisition paths, guided studio classes and open online submissions, with explicit consent and privacy safeguards, and details phase-aware annotation schemes. Next, it describes the pose-extraction pipeline that yields 3D skeleton sequences, along with lightweight preprocessing for temporal stability. The closing sections report key statistics, such as class balance, label integrity, pose coverage, and demographics. Last, it highlights practical biases that inform subsequent modeling and evaluation. The final dataset can be found here: Dataset.

3.1 Data Collection & Labeling

This section establishes the data foundation for Pole-Arina. The collection goal includes an ethical, realistic representation of fundamental static pole tricks. To that end, the dataset targets six foundational tricks and balances feasibility for novices with sufficient discriminability for modeling. Overall, the data collection follows two complementary paths: guided in-class recordings and open online submissions. Both routes use the same filming guidance and consent protocol and prioritize privacy by limiting released data to 2D skeleton keypoints. Annotations structure each clip into semantically meaningful temporal states. The remainder of the section motivates the trick set and task definition, presents data from both acquisition procedures, and discusses ethical concerns.

3.1.1 Trick Selection & Task Definition

Pole dance contains a broad repertoire of movement types, including *dance moves* (transitions where at least one foot stays on the floor), *spins* (rotational movements around the pole, while both feet are in the air), *floor work* (movements performed close

to the ground), and *tricks* (static or semi-static shapes on the pole). Poles themselves are either configured in *static* (no rotation) or *spinning* mode (bearing-mounted rotation). Since naming conventions and grouping of elements vary across federations and studios, this work uses the terminology presented by Spin City's Pole Bible [Cit25] and the International Pole and Aerial Sports Federation (IPSF) [Fed25].

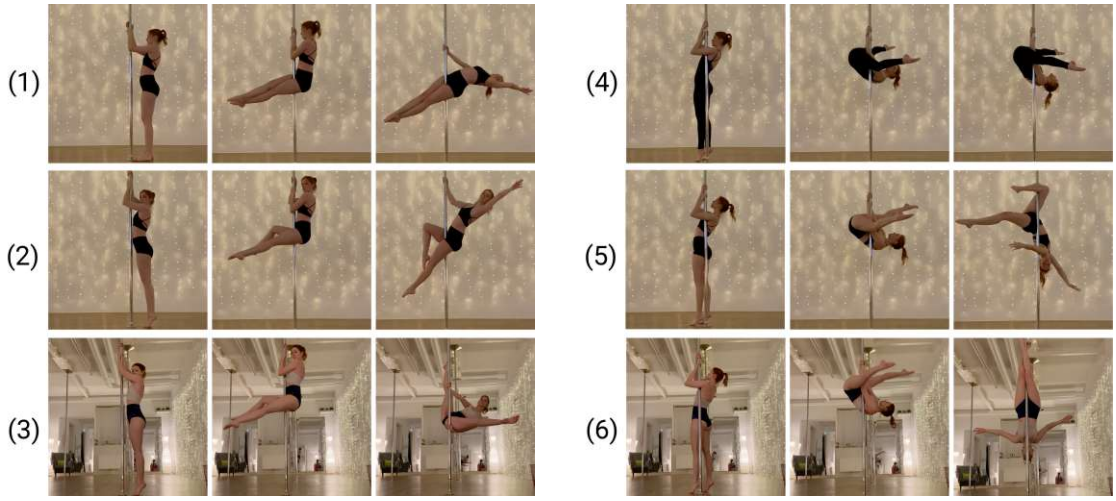
This work focuses exclusively on **static pole tricks**. Static execution allows the dancer to present the end position at a deliberate, consistent yaw to the camera/viewer. This minimizes foreshortening of limbs and self-occlusion and maximizes the aesthetic of the silhouette. Compared to spinning elements, this requirement makes static tricks more suited to consistent pose estimation and geometric measurement, as the reduced motion stabilizes the keypoint detection and limits motion blur on smartphones.

Task Definition. Each trick is decomposed into a three-phase movement sequence:

1. **Entry**: the performer is off the pole or in initial contact, preparing the entry.
2. **In transition**: the dancer is on the pole and moving toward the target position.
3. **End pose**: the final pose is established and held for a short interval.

For feedback generation, this thesis utilizes a geometric scoring system on the **end pose** phase only. They present the benefit of being not only time-stable but also reflecting other errors that manifest during the transition, making them suitable for precise, rule-based evaluation. Evaluation of the other phases is left as future work due to time constraints and the greater variability within the movement.

Selection criteria. The goal for the trick selection was to balance feasibility, safety for novice participants, and discriminability for the model. Therefore, the choice fell on six foundational tricks spanning two posture types: three upright beginner-level tricks and three intermediate inverted tricks. The upright set holds *Layout*, *Pin-Up*, and *Wrist Seat*. These tricks share similar entries (standing on the right side of the pole and pulling up into a sit) yet result in distinct end shapes. While the *Wrist Seat* is more distinguishable, the *Layout* and *Pin-Up* provide intentional "near-neighbor" classes to test the classifier's ability to separate subtle visual differences (see Figure 3.1a). The inverted set comprises *Straddle Invert*, *Gemini*, and *Crucifix*. Again sharing similar entries, these tricks begin from a basic invert grip, leading into a basic inverted position and transition to clearly differentiated end poses (see Figure 3.1b). Compared to the upright tricks, the inverted poses introduce greater biomechanical complexity, while remaining achievable for athletic beginners to lower-intermediate dancers.



(a) 1: Layout, 2: Pin-Up, 3: Wrist Seat (b) 4: Straddle Invert, 5: Gemini, 6: Crucifix

Figure 3.1: Progression of each trick, highlighting similar entries and transitions before the final pose.

3.1.2 Data Acquisition

Collecting high-quality and representative data was a crucial step in this thesis, as together with the annotations, it provides the ground truth for training and evaluating the Pole-Arina system. The aim was to assemble a dataset of approximately 600 clips, covering a balanced distribution of the six selected tricks and including both successful and failed attempts. Beyond the scale and diversity of the data, particular attention was given to the ethical aspects of data collection, since responsible handling of human-centered motion data is essential for trustworthy AI research [HZMY22].

Ethical considerations. Data-driven approaches such as Pole-Arina and other deep or machine learning-based systems offer clear benefits but also raise well-documented ethical concerns [HZMY22]. This paragraph briefly highlights possible issues and how they are addressed. In particular, recent publications discuss controversies around training on copyrighted or scraped content without consent (especially regarding generative AI). This further reinforces the need for explicit permission and transparent documentation [Lem24, Luc24, BP21]. With these concerns in mind, our data collection followed four principles:

1. Consent: A detailed protocol informed participants about the project goals, data handling, privacy measures, and withdrawal rights. Contributions were voluntary and limited to the intended purpose, consistent with GDPR (General Data Protection Regulation) requirements [PC16]. TU Vienna's data protection policy also enforces these regulations for lawful processing and data minimization [Wie25].

2. Privacy: To secure the participants' privacy, the released dataset only includes skeleton

joint coordinates. The raw video files remain private and are used solely for processing and quality control. Such privacy-enhancing approaches aim to minimize exposure of facial and background detail, while preserving the system's performance [HNR+25]. Although numerous participants gave their consent to publish their images in this thesis, the only human depicted in this work is the author.

3. Transparency: Standardized trick terminology is used throughout the execution of the data collection protocol. It further provides detailed information about the motivation, collection process, intended use, and limitations.

4. Discrimination and bias: While paying attention to balancing the data across tricks, experience levels, age, and gender, bias may arise from the used pose-estimation framework and demographic imbalance. Prior research highlights that fairness evaluation for human pose estimation is challenging due to missing demographic labels and data imbalance [LTNX23].

Online submissions. The first option to participate in the data collection was via an online form. Considering the ethical and project requirements, the form contained: project overview, participation and data-use terms, contact details, data collection consent, and detailed filming instructions for each trick, including a short tutorial playlist. General recording guidelines included:

- **Use a smartphone:** Participants record each trick with a smartphone, following the provided instructions.
- **Angle & framing.** The camera should capture the full body at all times, position it straight-on rather than from above or below.
- **Multiple attempts/videos per trick encouraged:** Multiple attempts per trick (including incomplete or failed tries) are encouraged to capture natural variation for the learning system.
- **Avoid background distractions:** Record in a space without bystanders or distracting movement visible in the frame.
- **Lighting.** Provide even, front-facing illumination so the body and movements are clearly visible, avoid strong backlight.
- **Clothing.** Wear form-fitting athletic wear (e.g., shorts and a sports bra/tank top) to keep key body positions visible.

The whole form was available in English and German. Once the video recording was finished, participants were able to upload the results directly through the online form.

In-class recordings. Because online recruitment alone did not reach the target goal, additional data were collected through dedicated pole classes. Similar to the online form, participants got all the relevant information beforehand and gave their consent with the right to withdraw participation at any time. An experienced instructor (myself) demonstrated each trick and supervised filming with a fixed camera setup that mirrored the online instructions. This setup ensured consistent angles, safer spotting for beginners, and more control to balance the data across the six target tricks.

Results. The final dataset comprises 836 clips from $N = 58$ participants. The protocol deliberately encouraged multiple attempts per participant to capture natural variability in the progression of a trick. Section 3.3 presents a detailed breakdown of the final dataset. Table 3.3 compiles concise, syllabus-aligned instructions and categories for the six selected tricks.

3.1.3 Annotation Protocol

To get into the final shape of a specific pole dancing trick, a sequence of movements is required. Therefore, the goal of the annotations was to create a ground truth that represents not only the final position but also identifies the trick's temporal progression. In parallel to the annotation process, the recognition model was already trained and tested on a subset, resulting in two different annotation schemes. The initial *multi-task* protocol (Protocol A) separates trick and phase labels, while the revised *single-task* scheme (Protocol B) merges them into a single per-frame label.

How the labeling was conducted. Hand-labeling data is often a slow, tedious, and repetitive process. A brief initial review of a small batch of clips resulted in a concise labeling guide and definitions. The aim was to keep labels measurable (frame-accurate start/end marks for phases) while avoiding subjective constructs. In particular, an early idea to label a per-phase "score" proved too subjective for consistent ground truth and was therefore replaced by a post hoc, rule-based scoring system. A custom Python tool accelerates annotation with keyboard shortcuts for frame stepping, instant annotation, and one-key phase assignment. This workflow kept labels consistent, adoptable, and scalable across hundreds of clips.

Why phase awareness matters. As described in section 3.1.1, each trick progresses through: *entry* (lifting off the floor) → *transition* (moving on the pole) → *end pose* (holding the final position). Phase awareness is crucial for three reasons. First, many tricks share similar entries, while discriminative cues often emerge only near the end of the transition. Therefore, phase context aids trick identification. Second, each phase holds different challenges (e.g., jump vs. pull-up during entry; loss of engagement during transition; angle deviations in the end pose). Third, end poses are comparatively more stable due to their static nature, enabling reliable geometric scoring methods (see Section 4.2.2 and 5.3).

Protocol A: multi-task labels. In the first scheme, each frame received two targets: a **trick** label from {`Layout`, `Pin-Up`, `Wrist Seat`, `Straddle Invert`, `Gemini`, `Crucifix`} and a **phase** label from {`Start`, `Transition`, `End`}. The definition of each label is as follows:

- **Start:** the dancer is still on the floor, touching the pole and preparing for the trick.
- **Transition:** both feet have left the floor, the dancer is on the pole, transitioning into a target position.
- **End:** the dancer has reached the final shape and holds it.

The phases match directly with the previously defined temporal progressions of a trick. A custom Python script was implemented to efficiently navigate through the video files and apply labels using keyboard shortcuts. The data were trimmed to the annotated interval (first `Start` phase frame to last `End` phase frame), and the labels were saved as a CSV with the following format:

```
filename,trick_name,start_frame,end_frame,phase
l1.mov,Layout,0,6,Start
l1.mov,Layout,7,112,Transition
l1.mov,Layout,113,162,End
...
```

Frame indices are zero-based, the start and end frames are inclusive, and segments are contiguous and non-overlapping within each file. This scheme aligns with a multi-task bidirectional LSTM, which will be discussed in section 5.2. As a result of this protocol, each video was separated into these exact same phases in the same order. This sequence prior helps suppress spurious fragments during trick detection. However, in real practice videos, idle time before starting, failed attempts, and immediate retries are very common. This rigid order suppressed more flexible patterns and restricted training signals for background frames. As illustrated in the top row of Figure 3.2 (A), frames outside the strictly defined window are trimmed, so idle time, retries, and dismounts are not represented in the labels.

Protocol B: single-task labels. To better support realistic training settings, where dancers might perform multiple tricks in one recording, a revision of the first scheme transformed the labels into single per-frame targets. The label set comprised two generic states and end pose labels for each of the six tricks:

{`floor`, `on_pole`} ∪ {`L_pose`, `P_pose`, `W_pose`, `V_pose`, `G_pose`, `C_pose`}.

The exact definition of each label is as follows:

- **floor:** the dancer is on the floor.
- **on_pole:** both feet have left the floor, the dancer is on the pole.
- ***_pose:** the dancer has reached the final shape and holds it.

Again, the phases match the defined temporal progressions of a trick. This subtle change merged the former trick and phase targets into one label space, enabling arbitrary sequences such as `floor` → `on_pole` → `floor` → `on_pole` → `L_pose` → `on_pole` → `floor`, thereby capturing idle segments and retries within a single clip. A semi-automatic Python script supported an efficient adaptation from the old to the new annotation scheme. While the `floor` label extends the previously used `Start` label to also cover idle time before and after the dancer is on the pole, the remaining labels are directly mapped to: `Transition` → `on_pole` and `End` → `{L,P,W,V,G,C}_pose`. Again, stored in a CSV, the labels were transformed into:

```
filename,state,start_frame,end_frame
l98.MOV,floor,0,20
l98.MOV,on_pole,21,140
l98.MOV,L_pose,141,154
l98.MOV,on_pole,155,308
l99.MOV,floor,0,83
l99.MOV,on_pole,84,151
l99.MOV,L_pose,152,158
p1.mov,floor,0,10
p1.mov,on_pole,11,97
p1.mov,P_pose,98,137
...
```

The bottom row of Figure 3.2 (B) shows how the state labels capture idle (`floor`), generic interaction (`on_pole`), explicit end poses (e.g., `L_pose`), and the dismount (`on_pole` → `floor`) within a single recording, enabling multi-trick detection and robust background modeling. Model-wise, this simplified the objective from a two-head (trick, phase) to a single-head problem over eight states. Significantly, the model trained on Protocol-B could detect multiple tricks per video and distinguish background and generic interaction from explicit poses. However, the introduction of generic labels by protocol B enhanced class imbalance, which was addressed by using a weighted cross-entropy.

Conclusion. Protocol A provided clean, phase-aware supervision but constrained clips to a single trick and rigid phase order, limiting realism and background coverage. Protocol B preserved phase awareness implicitly, scaled labeling to real practice behavior, and enabled multi-trick detection within a single video. A side-by-side example on the same video is shown in Figure 3.2, contrasting the rigid `Start`→`Transition`→`End` segmentation of Protocol A with the richer state timeline of Protocol B.

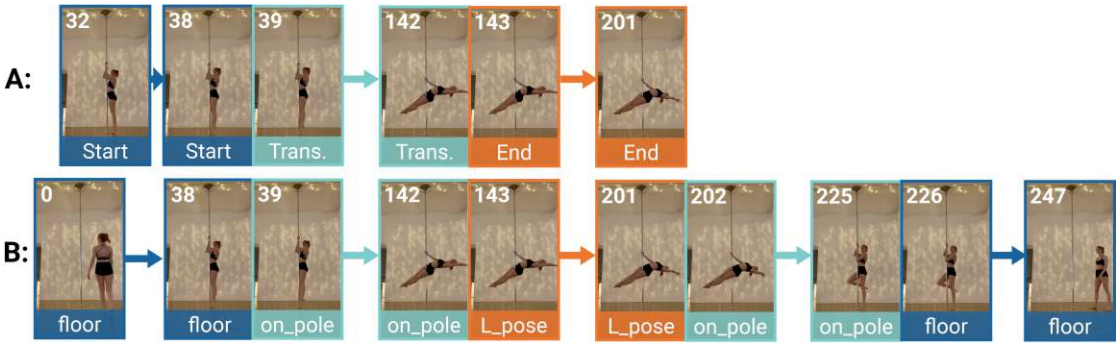


Figure 3.2: Side-by-side comparison of applying both protocols to the same video.

3.2 Feature Extraction & Preprocessing

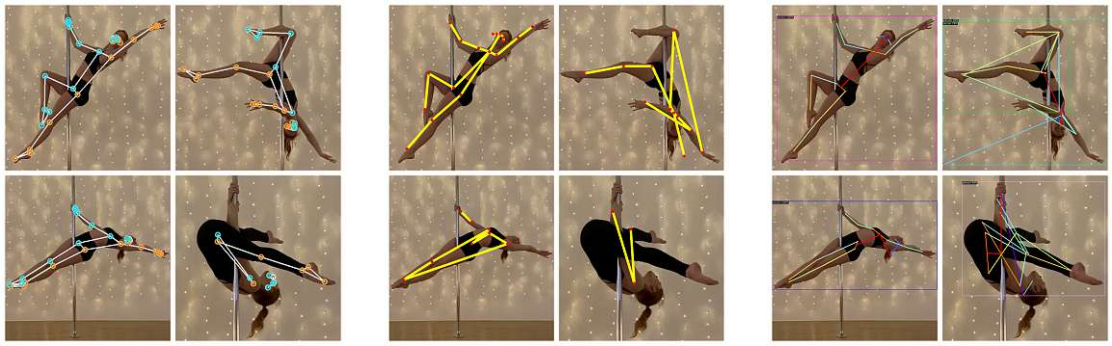
This section explains how the pipeline converts raw training videos into stable, privacy-preserving inputs for learning and evaluation. Instead of operating on pixels, the system extracts 2D body skeletons per frame and stores them as a compact tensor. A lightweight preprocessing stage then mitigates artifacts, such as high jitter, brief occlusions, and occasional dropped detections. These steps, together with normalization and optional data augmentation, produce temporally coherent landmark sequences that match the training distribution.

3.2.1 Skeleton Extraction

Pose-estimator selection. Three marker-less pose estimators were considered with a focus on real-time application and mobile feasibility: OpenPose [CHS+19], MediaPipe Pose [Goo25], and Detectron2 [WKM+19]. MediaPipe is based on the BlazePose [BGR+20] architecture and offers a lightweight, on-device pipeline with a 33-landmark body topology. OpenPose provides robust multi-person parsing but carries a heavier runtime cost with only 18 joints. Detectron2's Keypoint R-CNN achieves high accuracy on COCO's 17-keypoint scheme, but requires GPU resources and offers 17 joints.

A qualitative comparison of these estimators on four representative pole tricks provides insight into their limitations. MediaPipe produced stable and complete landmark predictions even in inverted positions, while Detectron2 and OpenPose frequently lost joints or misaligned the skeleton. Simple poses, such as the Pin-Up, were estimated consistently across all frameworks, while inverted tricks, like the Straddle Invert, proved more challenging, especially for Detectron2 and OpenPose (see Figure 3.3).

A quantitative runtime analysis on Google Colab further highlights their performance differences. As shown in Table 3.1, MediaPipe achieves real-time performance, even on a CPU, while Detectron2 and OpenPose are far slower and therefore impractical for real-time analysis.



(a) MediaPipe (b) OpenPose (DNN/COCO) (c) Detectron2

Figure 3.3: Qualitative comparison of skeleton overlays across four pole tricks.

Besides runtime, deployment support also played a role. MediaPipe offers pip-installable packages with stable CPU/GPU support. Detectron2 requires GPU resources and a heavier installation. In contrast, OpenPose suffers from compatibility issues with current CUDA/Caffe toolchains and only supports Ubuntu and Windows systems.

Considering runtime performance, number of joints, robustness in complex poses, and ease of integration, MediaPipe Pose was selected as the estimator for this thesis.

Table 3.1: Runtime benchmark on a five-second, 360×640 video (164 frames). CPU = Colab CPU runtime; GPU = Colab T4. OpenPose results use the COCO-18 model via OpenCV DNN.

Framework	**CPU (Colab)**		**T4 GPU (Colab)**		**Joints**	**Supports**
	Time [s]	FPS	Time [s]	FPS		
MediaPipe Pose[Goo25]	7.7	21.3	5.5	30.0	33	Windows, Linux, Android, iOS, macOS
Detectron2 (KPRCNN)[WKM+19]	1382.9	0.12	23.4	7.00	17	Linux, macOS, Windows (limited)
OpenPose (COCO)[CHS+19]	691.6	0.24	618.9	0.26	18	Ubuntu, Windows

MediaPipe configuration. The extraction script implements MediaPipe Pose with the following specifications:

- `MEDIAPIPE_DISABLE_GPU=1`
- `static_image_mode=false`

- `min_detection_confidence=0.5`
- `min_tracking_confidence=0.5`

For each frame, MediaPipe returns 33 landmarks as an array $(x, y, z, \text{visibility})$. Coordinates x, y are normalized to $[0, 1]$ (origin at the top-left), where z is a relative depth value in the same normalized scale (negative towards the camera). Visibility is a per-landmark confidence in $[0, 1]$.

Output. For each input video of length T frames, the extractor produces a NumPy array of shape $T \times 33 \times 4$ (`float32`) saved as `<video_name>.npy`. Optionally, for quality checks, the script saves a video with a skeleton overlay for each frame.

Script notes. Extraction is implemented in `extract_skeleton.py` (batch processing over folders). The tool also supports optional data augmentations (add noise, in-plane rotation, sequence time-warp) to create additional samples.

3.2.2 Preprocessing Techniques

Temporal smoothing. Pose estimation might suffer from jitter caused by detecting noise and small motions. To address this issue, the skeleton extractor applies an Exponential Moving Average (EMA) to each landmark. Let $\ell_{i,j} \in \mathbb{R}^3$ be the coordinates (x, y, z) of joint j at frame i and $\hat{\ell}_{i,j}$ the smoothed value. With smoothing factor $\alpha \in (0, 1]$, the smoothing process is governed by:

$$\hat{\ell}_{i,j} \;=\; \alpha\, \ell_{i,j} \;+\; (1-\alpha)\, \hat{\ell}_{i-1,j}, \quad \text{with } \alpha = 0.3 \text{ in our case.}$$

EMA is a first-order infinite impulse response (IIR) low-pass filter. Compared to the Simple Moving Average (SMA), EMA is more efficient as it does not require a buffer to store previous data, and the weight is not distributed equally. Instead, it emphasizes the most recent data samples while the previous data decays exponentially but never reaches zero [FHC19].

Low-visibility handling. MediaPipe provides a confidence for the visibility of each landmark. If the value falls below a threshold $\tau = 0.3$, the landmark keeps the previous frame value rather than updating, to reduce jitter during brief occlusions. This usually occurs when the pole covers certain body parts. If no pose is detected for an entire frame, the extractor saves a zero array placeholder. In the later introduced preprocessing pipeline, such frames will be linearly interpolated.

These additions stabilize joint trajectories while keeping the computation lightweight. Savitzky-Golay filters, or Kalman filters [FHC19], are valid alternatives but introduce either additional latency or extra state assumptions [Sch11, FHC19].

3.3 Final Dataset Statistics

This section summarizes the composition and quality of the dataset. It documents the number of recordings per trick, the distribution of frames across labels, and the reliability of the pose extractor in tracking joints. Beyond simple counts, the statistics highlight practical biases that arise in real practice footage and explain the compensating measures used during learning.

The reporting proceeds as follows:

- **Label & class balance:** per-video counts by trick and frame-level distributions under Protocol B, with a side-by-side comparison to the legacy Protocol A to expose phase proportions.
- **Label integrity:** coverage ratios (#labeled / #total frames) and checks for overlaps or gaps to ensure consistent alignment between labels and skeletons.
- **Video & skeleton properties:** capture characteristics (FPS, portrait resolutions) and pose-tracking reliability (per-clip coverage, per-joint visibility patterns), confirming suitability for temporal modeling and geometric rules.

Together, these statistics provide a transparent view of the dataset's strengths and limitations.

3.3.1 Label & Class Balance

This section provides a summary of the chosen label properties to provide insight and validate the quality.

Per-video class balance. Figure 3.4 shows the number of videos that contain at least one end-pose for each of the six target tricks. Upright tricks like the Layout and Pin-Up are more common compared to the inverted ones. These results were expected given the increasing difficulty of the intermediate tricks. Especially for beginners, sitting on the pole in a Layout or Pin-Up is easier than hanging upside down from a Gemini. This class imbalance was addressed by applying targeted data augmentation and a weighted loss for the model. Notably, a few video files include failed and multiple attempts of a trick, but the visualization contains the number of videos, not segments. The actual number of frames per phase will be discussed in the section below. Across the complete list, **812/836** videos contain one end-pose, **9/836** include a second try, and **24/836** are failed attempts.

Phase distribution. The following plots show the frame distribution for each label and include a comparison against the legacy Protocol A. Figure 3.5 presents the actual phase per frame distribution used for training the final recognition model. Phase `on_pole` immediately stands out, as it holds the majority share of 60.0% of all frames. Followed

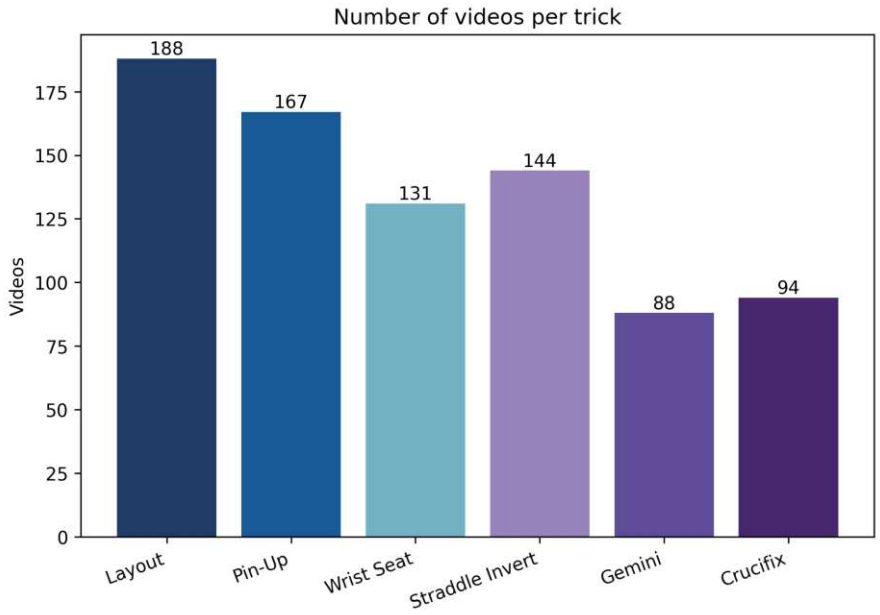


Figure 3.4: Per-trick class balance. Number of videos containing at least one end-pose for each target trick.

by `floor` with 20.5%, while the trick-specific labels contribute only a small fraction individually, with around 2-4% and a combined value of 19.5%. This shows that the label imbalance is stronger than the number of tricks per video imbalance seen in Figure 3.4. By design, each recording typically shows a single target trick but always includes background, idle, and transition segments. As a result, `on_pole` and `floor` dominate frame counts.

For context and comparison, Figure 3.6 shows the label distribution from the legacy Protocol A (see Section 3.1.2 for more details on this labeling scheme). Since this scheme yielded two separate sets of labels (trick name, phase), the results are represented as a stacked bar chart. The stacks show the absolute frames per trick label, while the percentages inside indicate the share of each phase label (`Start`, `Transition`, `End`). Aggregated over all tricks, `Transition` accounts for 63.2% of labeled frames, `Start` for 16.4%, and `End` for 20.4%. While not being as noticeable at first glance, the results present a similar class imbalance to the chosen labeling method. Figure 3.6 also reveals how long each trick tends to last in the transition phase. Measured as the share of labeled frames within each trick, Straddle Invert shows the shortest transition (40.4%), whereas Crucifix shows the longest (74.8%).

In the end, Protocol B was chosen for training the final model, because it keeps the background explicit and can capture retries and idle time without data trimming, while preserving trick-specific end states for evaluation.

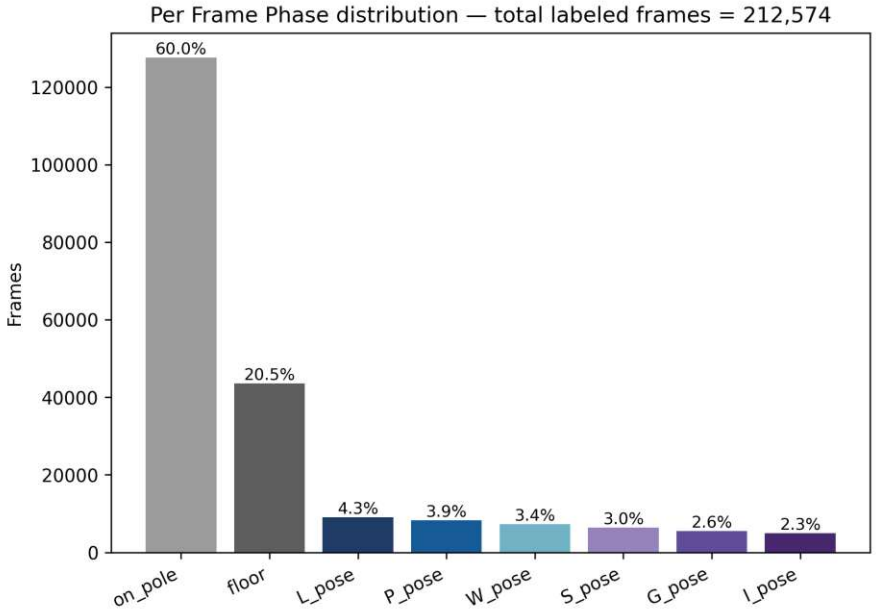


Figure 3.5: Protocol B. Percentages above bars show the relative contribution of each label to the total of 212,574 labeled frames.

At a glance, the label distribution is:

- **Protocol B:** 20.5% `floor`, 60.0% `on_pole`, 19.5% `*_pose` (combined)
- **Protocol A:** 16.4% `Start`, 63.2% `Transition`, 20.4% `End`

Label integrity. All labels underwent basic quality checks. First, the label coverage ratio is calculated as the labeled frames divided by the total number of frames. It is important to note that not all frames are labeled, especially long idle segments at the beginning or end of a clip. The ratios are overall high, with 54.9% having perfect coverage (1.0) and 75% over 0.957 (see Figure 3.7). No overlapping segments or internal gaps were detected after final curation. Quality control and label integrity enable consistent alignment between labels and skeleton data.

3.3.2 Video Data & Skeleton Data

To preserve privacy, the public dataset contains only skeleton sequences rather than raw videos. The extraction script stores the data as NumPy arrays of shape $N \times 33 \times 4$ with MediaPipe's 33 joints and the channels $x, y, z, visibility$ in normalized image coordinates. Furthermore, an EMA filter slightly smoothed the data, and joints with a visibility lower than 0.3 were treated as missing (see Section 3.2.2).

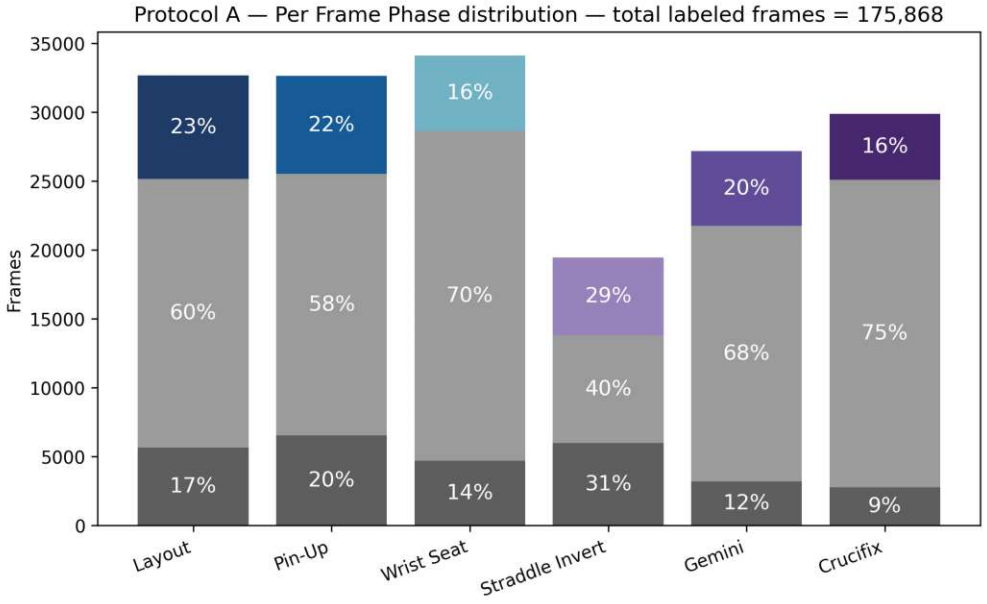


Figure 3.6: Protocol A. Bar height shows the absolute number of labeled frames per trick. The phase labels occur in the following stack order: bottom=Start, middle=Transition, top=end (highlighted in a trick-specific color). Percentages inside the bars indicate the relative share of each phase.

FPS & resolution. Participants of the data collection process used their smartphones for recording. The videos are predominantly at 30 FPS (∼96%) and a small minority at 60 FPS (∼4%). Resolutions cluster around portrait format with a 16:9 aspect ratio. After normalizing orientation, 29.3% of the clips are exactly 1080x1920, and 69.3% are close to Full High Definition (FHD). Table 3.2 summarizes the most common resolutions in more detail. Generally, these characteristics match a typical home and studio setup and align with the intended use case.

Skeleton detection quality. Skeleton reliability was evaluated to confirm that MediaPipe consistently tracks across frames. Pose coverage is defined as the fraction of frames with a valid skeleton. Out of 836 clips, 771 (92.2%) achieve a perfect score of 1.0. Only seven videos fall below 0.95, with the minimum at 0.767 for a recording of a Pin-Up. In total, non-perfect coverage occurs in 65 files, most often in more complex tricks such as the Straddle Invert (20 files) and the Wrist Seat (15 files). Due to their simplicity, Layout and Pin-Up remain the most stable, with only a handful of affected videos. Figure 3.8a highlights all non-perfect coverage clips.

The individual joint visibility was also evaluated by counting frames where the landmark confidence fell below 0.3. The results show a generally high visibility, with a median of 0.0% for frames below this threshold. Looking at a per-joint visibility heatmap across all frames, Figure 3.8b indicates uniformly high ratios across the body, with slightly lower

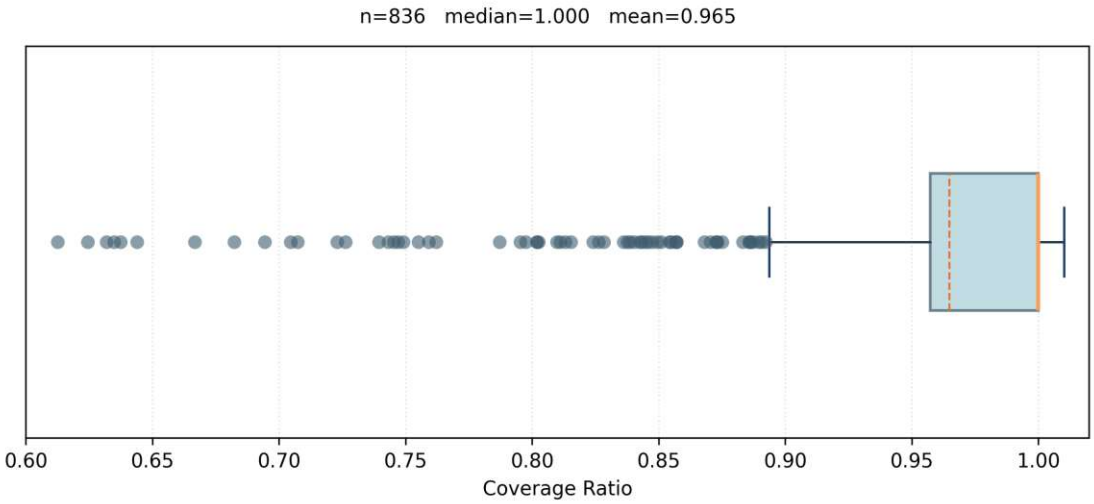


Figure 3.7: Box plot of coverage ratio with most labels achieving near-perfect coverage.

Table 3.2: Most common portrait resolutions within the dataset.

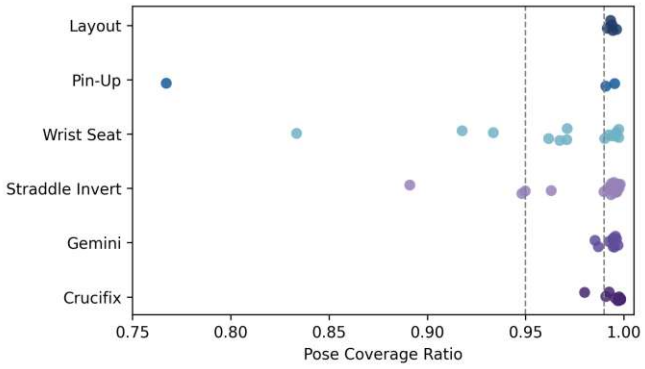
Resolution	**Count**	**% of 836**
1080×1920	245	29.31%
720×1280	22	2.63%
1010×1796	20	2.39%
1028×1828	19	2.27%
1020×1814	18	2.15%
1018×1810	18	2.15%
478×850	16	1.91%
1034×1838	14	1.67%
1016×1806	14	1.67%
1014×1804	13	1.56%

values (0.99-0.95) for the right elbow and hand cluster. This pattern matches recording issues for upright tricks such as Layout and Pin-Up, where the arm extends above the head and may leave the camera view.

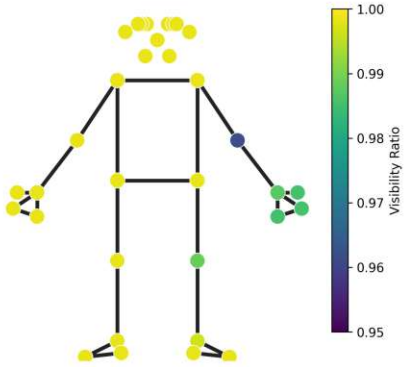
Overall, skeleton detection offers near-perfect coverage and stable joint visibility, ensuring reliable alignment between labels and skeleton data.

3.3.3 Demographics

The data collection aimed to strike a balance that still accurately reflects the student demographics in real-world pole classes. Another priority was to capture broad variations in execution quality so the model learns every deviation from failed to perfect attempts.



(a) Non-perfect pose coverage per video, grouped by trick. Dotted lines mark thresholds at 0.95 and 0.99.



(b) MediaPipe skeletal schematic colored by landmark visibility ratio.

Figure 3.8: MediaPipe coverage information.

Experience balance. The dancers' experience has a greater impact on variability than age or height. Because the selected tricks range from beginner to intermediate, the dataset deliberately emphasized novice attempts. Ideally, half of all contributors should have limited to no prior pole experience. This selection enhances the diversity of entries, transitions, and final shapes, thereby improving the robustness and generalizability of the recognition model. Therefore, as shown in Figure 3.9, the total number of 58 participants comprised 34 non-dancers (58.6%) and 24 dancers (41.4%).

Figure 3.9: Experience balance bars (Non-dancer vs. Dancer).

Gender context. Adult female students mostly visit local studios in Austria and likely around the world. Many classes are restricted to women only to maintain a comfortable environment, especially given the low-coverage attire required for a secure grip on the pole. At the same time, pole is a gender-inclusive sport with a growing number of men. The collected data reflects this reality while remaining open to all participants. Gender counts are: Female $n = 43(74.1\%)$, Male $n = 15(25.9\%)$ (see Figure 3.10).

Figure 3.10: Gender balance bars (Female vs. Male).

Age distribution. Although not as crucial for data diversity, the age distribution, as displayed in Figure 3.11, ranges from 20 to 58 years old, with a median of 28 years. Again, this accurately reflects real-class distributions of pole dancing students.

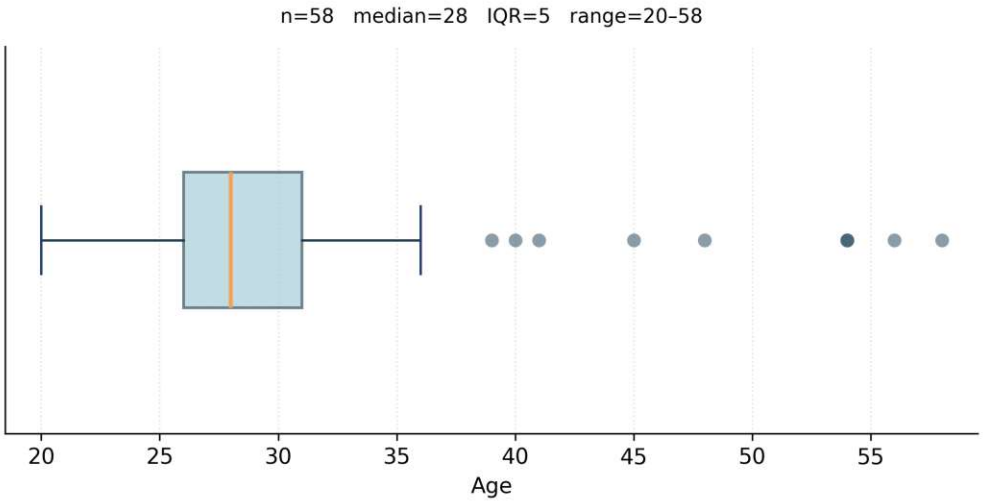


Figure 3.11: Age distribution.

Table 3.3: Compact summary of the selected tricks; terminology aligned with IPSF and Spin City [Fed25, Cit25].

Thumbnail	Trick	Description	Category / Level
	Layout	1. Stand on the right side of the pole 2. Pull up, cross legs at ankles 3. Lean back, arch, push hips up	Upright Beginner
	Pin-Up	1. Stand on the right side of the pole 2. Pull up, right toes to left knee 3. Lean slightly back and arch	Upright Beginner
	Wrist Seat	1. Stand on the right side of the pole 2. Pull up, right toes to left knee 3. Place left hand underneath the thigh 4. Lean back, open legs into a V shape	Upright Beginner
	Straddle Invert	1. Stand behind the pole, facing left 2. Stronghold grip, pull up, lean back 3. Open legs into a V shape	Inverted Inter.
	Gemini	1. Start with Straddle Invert 2. Hook the outside leg, other leg down 3. Chest up, arch and release hands	Inverted Inter.
	Crucifix	1. Start with Straddle Invert 2. Place legs into crucifix hold 3. Upper body low, release arms	Inverted Inter.

CHAPTER 4

A Coaching System for Pole Dancing Technique

Pole-Arina addresses a common challenge in pole training: subtle misalignments and unsafe form, which often remain undetected outside guided classes. The thesis delivers a marker-less, video-based coaching application that recognizes the performed trick, identifies its temporal progression, and grades the final pose with transparent, geometry-based feedback. The design prioritizes privacy, interpretability, and practical deployment on user hardware.

4.1 System Overview

Pipeline overview. Figure 4.1 summarizes the end-to-end process of the Pole-Arina system. First, the user records themselves performing one pose or a combination of tricks. From a single uploaded RGB clip, the pipeline extracts a sequence of 33×4 MediaPipe landmarks per frame. Next, it applies preprocessing (e.g., smoothing and normalization) before feeding the sequence to the LSTM-based recognition model. The model produces a frame-by-frame timeline of semantic states, such as generic phases or specific trick poses. Detected end-pose frames are then passed to a rule-based engine, which evaluates trick-specific geometric checks and renders visual overlays along with textual cues for feedback.

In summary, the model identifies the performed trick, and the rule-based module provides feedback on the pose correctness, indicating where adjustments are needed. The following sections formalize the recognition and scoring components at a high level, while Chapter 5 provides implementation details and Chapter 6 presents the evaluation and results.

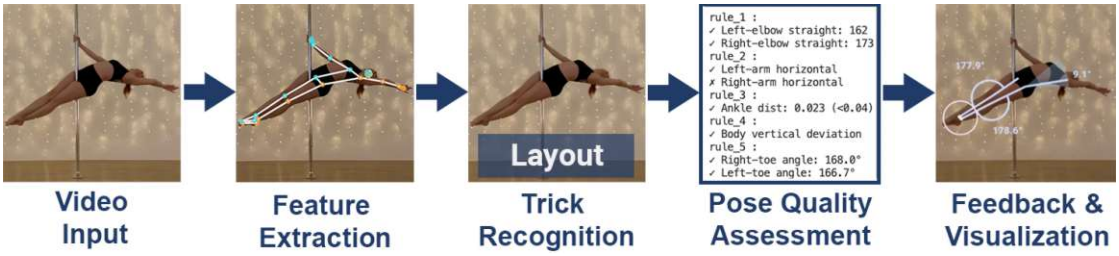


Figure 4.1: Pole-Arina end-to-end pipeline.

4.2 Trick Recognition & Pose Analysis

Pole-Arina requires a temporal model that turns a video into a sequence of semantic states interpretable for coaching. Given per-frame landmarks, the task is defined as a multi-class classification problem that assigns one label to every frame. In computer vision, this falls under the broad field of action recognition or action segmentation. This section formalizes the frame-wise label space and input representation, then presents a bidirectional LSTM for sequence labeling, including architecture, class-imbalance handling, and the training objective. Next, it describes decoding and post-processing to obtain stable, contiguous end-pose segments. Finally, it introduces the rule-based pose-quality analysis that converts the classifier output into interpretable coaching feedback.

4.2.1 LSTM Model

The primary task of the recognition model is to map a time sequence of skeletons to a time sequence of semantic states that support coaching feedback. Each input frame provides a 33 $\times$4 feature tensor (33 landmarks $\times$ $(x, y, z, \text{visibility})$). The model outputs a label for every frame, covering both generic movement phases and trick-specific end poses.

Model decision. A lightweight, bidirectional Long Short-Term Memory (LSTM) model efficiently processes temporal context on modest datasets and hardware [HS97]. On small to medium datasets, enhanced recurrent models can rival or outperform heavier Transformer variants while offering lower latency [EC20]. Due to latency and deployment constraints, the implementation of Pole-Arina favored a Bi-LSTM over larger models. As illustrated in Figure 4.2, one or more LSTM layers, with a moderate number of hidden units, take the normalized skeleton coordinates as input. The following time-distributed fully connected layer produces a classification score for each frame. The final layer is a softmax over the defined label classes, so the network outputs a probability distribution for the frame's state at each time step. Compared to the original LSTM formulation [HS97] and the classical Bi-LSTM concept [GFS05], the deployed model is *explicitly bidirectional* with concatenated directions, uses a *time-distributed fully connected head* rather than a CRF/CTC decoding layer, and includes *dropout* between recurrent output

and the classification head for regularization. This compact design preserves temporal context while remaining efficient on consumer hardware.

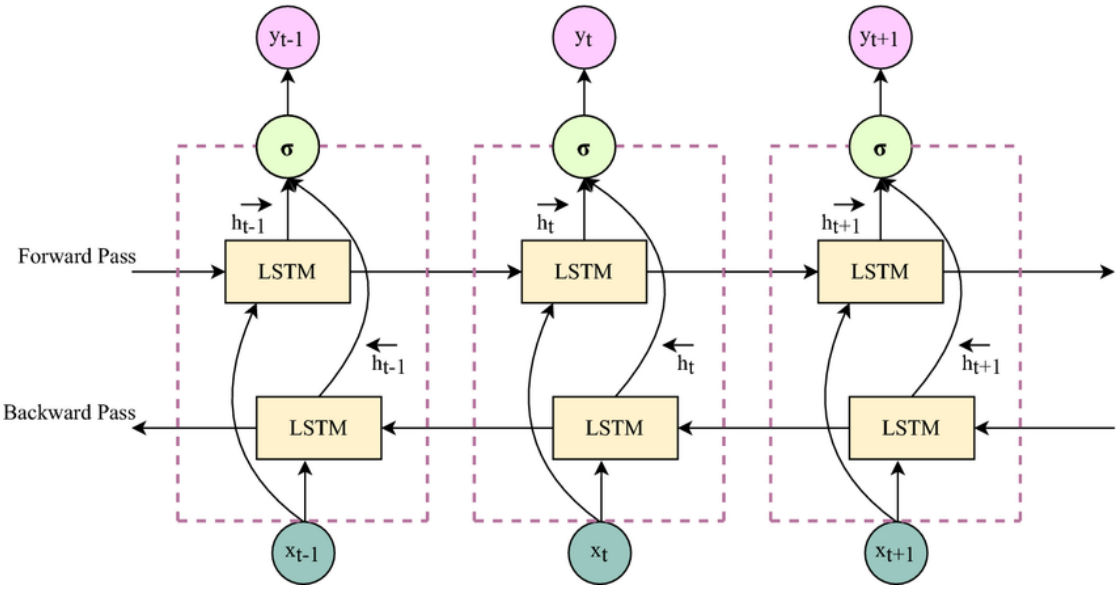


Figure 4.2: Bidirectional LSTM architecture, taken from [NJ22].

Label design. The final label set combines generic phases with trick end poses:

$$\{\mathit{floor}, \mathit{on\_pole}\} \cup \{\mathit{L\_pose}, \mathit{P\_pose}, \mathit{W\_pose}, \mathit{S\_pose}, \mathit{G\_pose}, \mathit{I\_pose}\}.$$

These end pose labels indicate the frames where the performer holds the final position, which presents the most trick characteristics and will be used for evaluation. To smooth out momentary prediction errors, a short median filter is applied to the model's per-frame outputs. This removes brief misclassifications before decoding the sequence into contiguous segments for each recognized phase or pose.

Handling imbalance. As presented in Chapter 3.3, end pose classes occur far less frequently than the generic phase classes, and some tricks are less represented than others. To address this imbalance, weights are added to the training loss, and data augmentation balances rare tricks. This encourages the network to learn the infrequent classes despite skewed data distribution. The exact weighting and optimization settings are described in Section 5.1.2 and 5.2.2. The network was optimized with standard procedures (using the Adam optimizer and early stopping on a validation set) to ensure good generalization.

Decoding & post-processing. The raw output of the LSTM is a per-frame label sequence, which may still contain occasional flicker or brief frame misclassifications within a phase. To obtain stable, meaningful segments, two post-processing steps are applied. First, using the model's confidence, only results above a specified threshold are accepted.

Second, a median filter over a short window merges isolated misclassified frames into the surrounding classes. After these steps, the timeline is segmented and labeled by phase or end pose. In particular, only sufficiently long end pose segments are kept for analysis to avoid fleeting motions. This decoding process yields a cleaner interpretation for the dancer's entry into the final pose of a trick.

4.2.2 Pose-Quality Rule Design

Once the system has identified an end pose, the next step is to grade the quality of that pose using explicit geometric rules on the skeleton. The module turns joint coordinates into transparent feedback, including pass/fail values, an overall score, and textual suggestions. The design focuses on interpretable checks, like body orientation, limb alignment, and joint proximity. It measures angles or normalizes distances with tolerances specific to the trick. The pipeline uses normalized image coordinates, enforces a visibility threshold, and evaluates all frames of the end phase. This subsection explains the concept of the feedback mapping in support of **RQ2**. The full implementation and rule design specifics are located in Section 5.3.

Post-processing vs. direct pose evaluation. An early design choice was whether to have the model directly learn pose correctness or to evaluate in a post-processing step. One initial consideration included the first LSTM iteration to output a pose-quality score. The implementation included scoring each phase using a predefined error list and matching it with the viewed performance. For each identified error, the score would be reduced by one, going from: perfect → good → ok → fail. However, this approach proved impractical due to the subjective nature of labeling and the limited time for expanding and annotating the training dataset. Instead, the LSTM focuses solely on trick and phase recognition, while the pose quality evaluation was left for the post-processing stage. This separation simplifies model training and makes evaluation criteria more transparent and adjustable, without requiring retraining of the model.

Focus on the final pose. As a reminder, each pole trick can be broken down into individual phases. For instance, the *mount* or *entry* into the trick, the *transition* where the dancer is in motion, and the *final pose*. In practice, the execution quality can be evaluated at each of these steps. Some general pointers include:

- **Entry:** Controlled entry on the pole, including the right muscle engagement without unnecessary jumps and swings.
- **Transition:** Correctness of contact points, technique, and body alignment while moving towards the final position.
- **End position:** Present common trick characteristics, often including pointed toes, specific angles, and body orientation.

Initially, all phases were considered for evaluation, but due to time and data constraints, the solution was optimized based on the end position. This phase holds the richest information about the trick's execution because errors during transition are usually reflected in the final state. Moreover, the dancer holds the trick statically at the end, which further has a specific presentation angle to the viewers. This focus enables a clear view, less motion blur, and consistent positioning.

Defining correctness. A key challenge in designing pose-quality metrics is the absence of a strict "rulebook" for executing each pole trick. Unlike gymnastics or ballet, where technique is codified in detail, pole dancing is guided by general best practices rather than exact prescriptions. Judges and instructors look for proper form, body alignment, and precision, but there is room for personal style. For instance, having fully extended legs and pointed toes is universal for clean lines, but exact angles and distances between joints can vary with individual anatomy. Therefore, the evaluation criteria should be specific, yet flexible to account for differences in body proportions or styles. Additionally, factors such as height, limb length, age, or gender should not result in penalties. Using the normalized skeleton data provided by MediaPipe accounts for scale differences and focuses on the geometry of the pose. Each rule is specified in terms of an angle or a distance between joints, along with a tolerance range that accounts for minor variations. The tolerances ensure that performances are not unfairly penalized, when deviating slightly due to personal style. In summary, correctness is defined by a set of geometrical conditions that reflect good form, with allowances made for normal variability between different performers.

Rule families & scoring concept. Each trick holds defining characteristics represented as rules with targets and tolerances. During evaluation, each rule returns a boolean pass/fail for each frame of the end pose segment. The overall pose score is calculated as the fraction of rules passed out of the total rules for that trick. For example, if a particular trick has 5 rules and the dancer satisfies 4 of them, the pose would score 4/5 or 80%. The scoring system defined three geometric rule families to cover a broad range of possible checks:

- **Body Orientation:** orientation of a segment relative to ground.
- **Limb Alignment:** internal joint angles.
- **Joint Proximity:** normalized distances between landmarks.

Exact formulas, thresholds, and per-trick targets are specified in Section 5.3.

4.2.3 Feedback & Visualization

In addition to the written feedback, Pole-Arina also provides visual feedback in the form of overlays, drawn on top of the evaluated frame. Each rule type maps to a distinct visual primitive:

- **Orientation → Arcs:** appears as circular arcs around a reference joint to highlight the accepted angle sector.
- **Alignment → Angles:** traditionally represented as the angle between two vectors, connecting the affected joints.
- **Proximity → Circles:** shows a circle centered between two landmarks with a radius equal to the measured distance.

This mapping separates the rule concepts effectively, while maintaining consistent feedback across the tricks. Regarding the implementation, the same primitives port cleanly to SVG overlays to enable exploration through interactivity. Figure 4.3 illustrates the three overlays in a simple style on a Pin-Up pose.

Together, the rule configuration and its overlays form a transparent scoring system on top of the recognizer. In short, the model identifies **what** was performed, the scoring system explains **why** it received this value, and the visuals show **where** to adjust. The **how** is directly given by the evaluation message, explaining the needed adjustments.

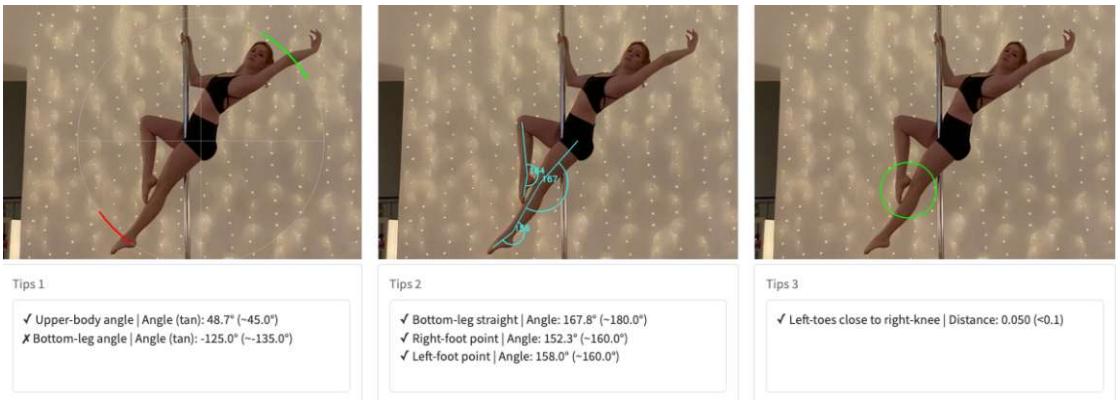


Figure 4.3: Rules mapped to visual overlays on a Pin-Up pose.

4.3 Pole-Arina Evaluation

Pole-Arina is evaluated along two paths. First, quantitative tests on a held-out split assess recognizer accuracy and the stability of end-pose detection (RQ1) and validate geometric scoring against trick definitions (RQ2). Second, a controlled user study examines effectiveness and usability in practice, combining objective improvement with subjective measures of trust, clarity, and SUS usability (RQ3). The full methodology and results are presented in Chapter 6.

CHAPTER 5

Pole-Arina: Implementation

This chapter provides detailed information on the end-to-end implementation of **Pole-Arina**. Following standard practice in motion pipelines [TP23], data preprocessing stabilizes pose trajectories and handles missing detections. For that, it employs temporal smoothing and gap interpolation before preparing the data into a train/validation/test split. Next, data augmentation balances the dataset and increases the sample size to improve generalization. A bidirectional LSTM is trained to produce per-frame predictions. Section 5.2 documents the architectural evolution and the final formulation with class-weighted cross-entropy. Three distinct rule families turn landmark geometry into pass/fail checks with specified tolerances for each check. Finally, the application prototype wraps the pipeline behind a single analysis endpoint and renders interactive overlays, including scores and improvement tips.

5.1 Data Preprocessing & Augmentation

Effective preprocessing is essential to convert raw pose data into reliable inputs. Real-world motion-capture data often contains noise, missing values, and other inconsistencies. If left unaddressed, these issues can lead to training malfunctions and degrade the model's performance. Preprocessing techniques clean and refine the data by transforming noisy and inconsistent inputs into a clean format suitable for machine/deep learning [OGK+24]. Typical preprocessing steps include: noise reduction, outlier removal, and handling of missing data. Once cleaned, the labels are aligned with the skeleton data and split into train, test, and validation sets. Figure 5.1 presents a visual overview of this pipeline.

Figure 5.1: Preprocessing pipeline overview.

5.1.1 Data Preprocessing

Clean-up. As a reminder, the video data was captured through single RGB cameras, more specifically, the ones integrated in smartphones. MediaPipe Pose [Goo25], a pose estimation framework, extracted the skeleton data, which can introduce jitter and minor errors in the detected joint coordinates. Noisy pose data causes visible flicker effects in skeleton visualization, which may complicate learning but also appear less trustworthy. To address this, the pre-processing pipeline applied temporal smoothing to the sequence of detected keypoints. The result is a calmer, more realistic motion sequence where high-frequency noise is suppressed. Along with noise, missing detections are another issue, which yield zero or null coordinates for all joints. Instead of leaving blank frames, the pipeline performs a simple interpolation to fill the gaps. Therefore, short occlusions or tracking failures do not result in inaccurate labels or fragmented input data. Both of these techniques are thoroughly discussed in Section 3.2.2. However, to make the impact visible and measurable, Figure 5.2 overlays per-frame jitter before and after temporal smoothing on a representative Straddle Invert.

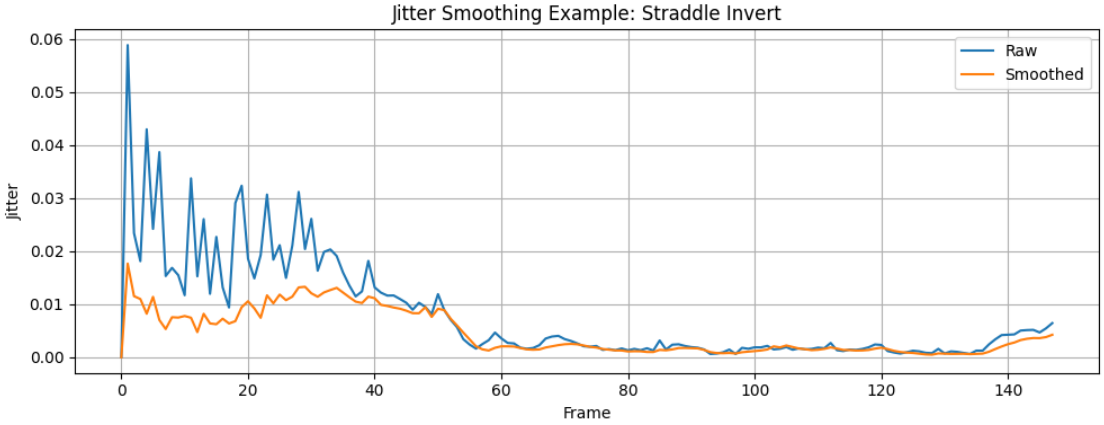


Figure 5.2: EMA reduces high-frequency jitter on a representative Straddle Invert clip.

Measuring jitter. Per-frame jitter quantifies high-frequency motion in the 2D landmarks. For frame t, we compute the mean Euclidean displacement across N tracked landmarks in normalized image coordinates. The calculation only uses landmarks above a visibility threshold ($v_i \geq \tau,\ \tau = 0.3$). The first frame is excluded since J_1 is undefined.

Let T be the number of frames in a clip and let $\mathbf{p}_{i,t} = (x_{i,t}, y_{i,t})$ denote the 2D (normalized) image coordinates of landmark i at frame t.

Per-frame jitter is computed as:

$$J_t = \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{p}_{i,t} - \mathbf{p}_{i,t-1}\|_2, \qquad \mathbf{p}_{i,t} = (x_{i,t}, y_{i,t}).$$

The series $\{J_t\}$ is collapsed to a single, comparable number. The arithmetic mean is calculated over frames for both raw and EMA-smoothed landmarks using the **sequence-level averages:**

$$J^{\text{raw}} = \frac{1}{T-1}\sum_{t=2}^{T} J_t^{\text{raw}}, \qquad J^{\text{smooth}} = \frac{1}{T-1}\sum_{t=2}^{T} J_t^{\text{smooth}}.$$

The percentage reduction reports how much the EMA suppresses jitter:

$$\text{Reduction} = \left(1 - \frac{J^{\text{smooth}}}{J^{\text{raw}}}\right) \times 100.$$

At the dataset level, Figure 5.3 summarizes the percentage reduction in jitter achieved by the EMA filter and grouped by trick. As expected, more complex or inverted poses exhibit higher raw jitter and therefore show larger reductions in jitter, compared to simpler tricks such as the Layout. Together, the plots illustrate that the smoother input sequences are produced by the preprocessing pipeline by removing high-frequency noise.

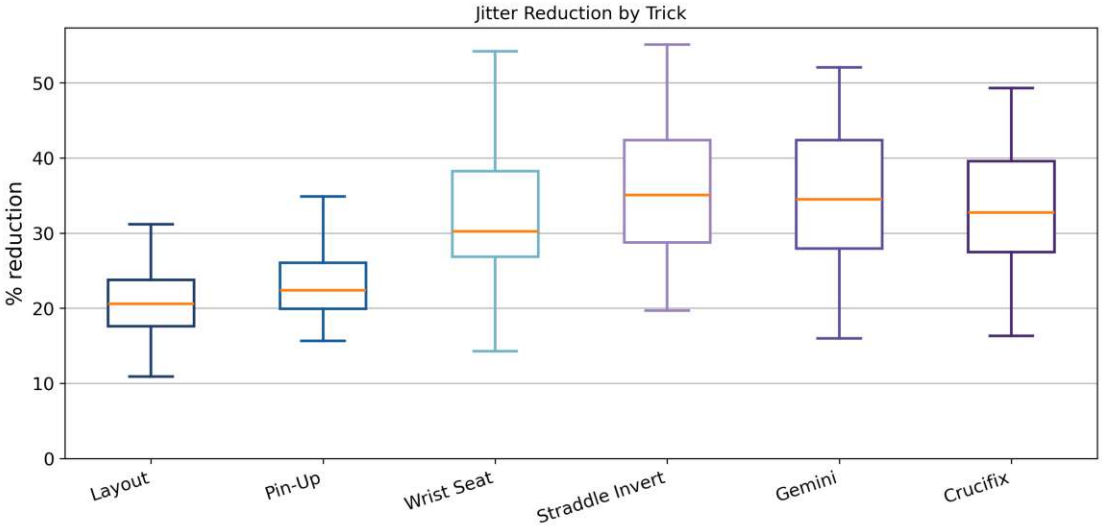


Figure 5.3: Dataset-level jitter reduction by trick.

Label alignment & segmentation. Once the skeleton data is verified and cleaned, the data gets aligned with the ground-truth phase annotations from the dataset. Each video was labeled in segments, providing the start and end frames of each trick phase (e.g., `floor, on_pole, L_pose, etc.`). The preprocessing script takes those segments to match them with the continuous skeleton sequence and assign a class label to each frame. First, the label CSV provides all annotations for a given video and is sorted by their start frame. Next, the skeleton data is sliced to span from the start to the end frame of each labeled phase. Previously, the data was tested for any gaps or overlapping labels. This alignment process ensures each skeleton frame is paired with the correct target label. The resulting collection of synchronized data-label pairs is a matrix of shape frames×joints×coordinates, accompanied by a matching sequence of class labels for each frame.

Dataset split. The final step divides the processed dataset into separate subsets for training, validation, and testing. After aggregating all labeled sequences, the set is randomly split into three parts:

- **Train:** ∼70%
- **Validation:** ∼15%
- **Test:** ∼15%

Notably, the dataset separation does not split per-frame but per-video. Therefore, the model will always see a full sequence without interruption. The test set provides hold-out data for final evaluation, while the validation split finetunes the model. Finally, the cleaned, aligned, and augmented (as discussed next) data is prepared and ready for the model.

In summary, smoothing and interpolation enhance the data quality by reducing noise and filling gaps. It aligns each sequence with the ground-truth labels and splits the data for the training of the model. This enables the model to learn from high-quality inputs and correct targets, which is foundational to achieving proper output.

5.1.2 Data Augmentation

After preprocessing, the next step deliberately expands the dataset by introducing diversity through data augmentation. It is a standard concept where new synthetic training examples are derived from existing ones by applying various transformations. The goal is to introduce controlled variability by increasing the dataset size, thereby reducing overfitting and improving the model's ability to generalize, especially when the available data is limited. Data augmentation techniques can range from basic image manipulation to deep learning approaches by generating new data samples through Generative Adversarial Networks (GANs) [SK19] or other models. In this thesis, the applied techniques are categorized into spatial augmentations and temporal augmentations.

The first modifies the geometric attributes of the pose, while the latter alters the time dimension of the sequence. Both types aim to simulate realistic variations, including diverse viewing angles, varying execution speeds, and minor sensor noise.

Augmentation techniques. Three methods were implemented and applied to the skeleton sequence in random combinations:

1. **Gaussian noise :** adding small Gaussian noise to joint coordinates.
2. **Random rotation:** rotating the 2D pose trajectory by a few degrees.
3. **Time warping:** randomly stretching or compressing the temporal duration.

These methods produce realistic variations for human pose sequences and align with known related work methods [XKCP24].

Gaussian noise. A slight Gaussian noise is added independently to each joint's coordinates. Therefore, each coordinate is perturbed by a tiny random offset. This effect simulates minor positional errors or measurement noise that naturally occurs in pose estimation. By training the model on noisy skeletons, the model becomes more robust to jitter while maintaining focus on the overall pose, rather than on landmark positions ([XKCP24]).

Random Rotation. A rotation of between -5° and +5° is applied to the whole skeleton in 2D. This spatial transformation changes the global orientation of the skeleton but preserves the relative pose. The aim is to make the model invariant to viewpoint or orientation changes. In practice, this happens frequently, as dancers might not pay attention to the horizontal alignment of the camera view. The rotation range is limited to small angles, as tricks are often characterized by body orientation which is utilized for evaluation. Figure 5.4 displays such constraints for split-style shapes. The different presentations impose distinct geometric targets. Augmentation, therefore, restricts rotation to small angles to preserve trick-defining orientation cues.

Figure 5.4: Split-style shapes: horizontal, diagonal, and vertical presentations.

Time Warping. This augmentation alters the speed of the motion sequence. The data can either be randomly stretched or squeezed (between a factor of 0.8 and 1.2) before getting resampled to the original number of frames. Therefore, some frames are either skipped or interpolated while the original sequence length remains to match the existing labels. Training on time-warped data enables the model to learn variations in execution speed and temporal dynamics.

The augmentation can be applied individually or in combination. For the final data set, multiple augmented versions were generated by randomly selecting one, two or all three of the transformations. By augmenting each original sequence into several variants, the size of the training set was effectively multiplied, while introducing a rich variety of poses. Each clip was multiplied by three random augmentations, except for the *Gemini* trick, which received five per video. Additionally, it addressed class imbalance by generating more samples for underrepresented tricks such as the Gemini.

5.2 Recognition Model (LSTM)

This section covers the recognition model, which takes skeleton data as input and outputs semantic states for feedback and scoring. First, Subsection 5.2.1 **LSTM Iterations** outlines the architectural progression, from a real-time system to a multitask prediction model, and finally to an eight-class single-head solution. Each step reveals different strengths and limitations through experiments, which lead to the development of the final model. Subsection 5.2.2 **Final Single-Output LSTM** specifies the LSTM architecture used for the Pole-Arina implementation.

Task Definition The output evolved with the project goals:

1. {Start, Transition, End}×{fail, ok, good, perfect} for real-time grading;
2. {Start, Transition, End}×{L, P, W, S, G, I} for automatic trick identification;
3. {floor, on_pole} ∪ {L_pose, P_pose, W_pose, S_pose, G_pose, I_pose} to enable multi-trick detection.

Several constraints influence the model choice and labels: privacy (skeletons only), data scale, single-camera capture, and class imbalance. Success criteria focus on accurate end-pose detection, reliable segmentation across multiple tricks in one clip, and controlled errors concentrated in transitional frames rather than in the final poses.

5.2.1 LSTM Iterations

This section presents the explored model iterations before settling on the final design. All iterations run in PyTorch and consume skeleton vectors of size 33 ×4 (33 landmarks × $(x, y, z,$ visibility$)$). They also share the same backbone: a bidirectional LSTM with two

layers, a hidden size of 64, and a dropout rate of 0.2. Furthermore, they output per-frame logits and employ an Adam optimizer with a learning-rate scheduler. All models were trained with Google Colab on an NVIDIA T4 GPU.

Initial Multi-Task LSTM (phase + score) The initial model was an LSTM-based network with several training enhancement configurations and a double-head output layer. Key features of this model included:

- **Learning-rate scheduler:** A dynamic learning-rate scheduler adjusted the step size during training, to sustain steady convergence as training progressed.
- **Dropout regularization:** A dropout layer tackles overfitting by randomly omitting units during training, encouraging the model to generalize better.
- **Bidirectional:** The activated bidirectional LSTM layer enables the model to process the sequence in both forward and reverse time directions. Considering past and future context, the model can capture richer dependencies in the sequence.
- **Multi-task outputs:** The LSTM was expanded into a multi-task model that computes predictions for two separate output heads: first, a performance **score** {perfect, good, ok, fail} and second, the current temporal **phase** of the trick {Start, Transition, End}. This design allowed the network to learn both what the dancer was doing and how well they did it in parallel. Multi-task learning can improve the generalization by leveraging a shared representation for related tasks.

The labeled dataset is closely related to the one introduced by Protocol A (see Section 3.1.3) but holds an additional label for the score. Each phase segment was annotated with a grade (perfect, good, ok, fail) based on the number of execution mistakes. Standard errors were identified in advance, and each match would result in a one-level downgrade of the score. For instance, if two mistakes were identified during the transition phase, the score would be labeled as *ok* instead of *perfect*. This first concept was inspired by dance video games (e.g. *Just Dance* [Ubi09]), which provide immediate and similar feedback to players during performance. The initial multi-task LSTM performed well, achieving a test accuracy of **91.58%** on the score prediction task and **96.08%** on the phase classification.

While the real-time model was conceptually appealing for regular dancing, it had some practical issues, especially for pole dancing. First, it required the user to pre-select which trick they were going to perform, limiting the system's autonomy. Second, delivering real-time feedback to a pole dancer has some usability issues, as dancers often hang upside-down on the pole without the ability to look at their phones. An augmented-reality mirror could overlay feedback visually, but such hardware is not commonly available to users and would complicate the system. Considering these drawbacks, instead of a "real-time coach", the system would focus on recognizing and identifying the trick and current progression (phase) and transfer the grading to a post-analysis system.

Phase and trick detection LSTM. The next iteration removed the scoring component to focus on the recognition task. The two prediction heads were reconfigured first to classify the current **phase** {Start, Transition, End} as before, and second, to identify the performed trick at each frame. The user no longer needed to tell the system which trick would be performed, as the LSTM would automatically detect it from the motion sequence. Furthermore, this opened the possibility to handle **multiple different tricks in one session**, which is standard practice in training where dancers train combinations of moves in succession.

This implementation fully realized the concept of the labeling Protocol A, with each frame carrying a phase and trick label. Again, the model learned this in parallel and continued to utilize the previously introduced features (bidirectional, dropout, learning-rate scheduler, etc.) to maintain the already satisfying performance. Trained on the full dataset, this model achieved a test accuracy of **96.83%** for phase classification and **99.95%** for trick classification. These results were auspicious for building the final Pole-Arina system.

Despite its excellent performance on single-trick videos, this architecture revealed some limitations when tested for multi-trick detection within a single recording. The issue was caused by the labeling scheme and how the LSTM learned temporal patterns. First, the initial training data assumed a strict phase order per video: start on the floor, transition on the pole, and end in the final pose. Next, if the dancer performed a second trick in the same recording, the sequence of labels might repeat, but the dancer would first need to get down from the pole. Effectively, this approach did not provide the required flexibility to detect multiple tricks in a single video. To address this, a final strategy was implemented to enhance the labeling strategy and increase the model's flexibility, which is discussed in section 5.2.2.

Evaluation protocol. To verify that each model learned a meaningful representation and generalizes beyond its training split, every iteration underwent the same evaluation:

- **Training-Validation plots:** to compare training and validation curves. If the training performance drastically exceeds validation performance, the model might overfit, resulting in poor generalization.
- **Confusion matrices:** to reveal if certain classes are more frequently confused.
- **Per-frame timelines:** to inspect temporal consistency on held-out clips.

Figure 5.5 illustrates a concise panel summarizing these checks for the second model iteration. This protocol documents the training progression and guides changes between iterations.

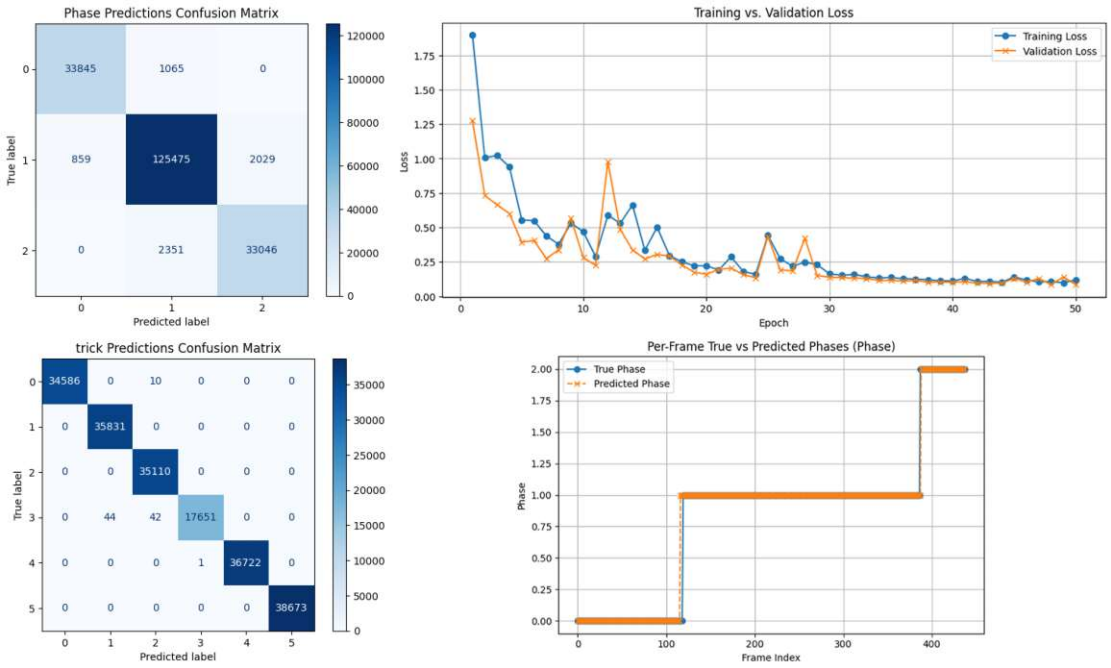


Figure 5.5: Evaluation diagnostics for the second model iteration, serving as an overview.

5.2.2 Final Single-Output LSTM

For the final refinement, the previous problem was reformulated into a single-head classification task with a mix of trick-specific and generic labels. A semi-automated re-labeling script, enforced by Protocol B (see Section 3.1.3), transformed the data to support multiple trick detection per video. Instead of using two separate outputs for phase and trick, the labels were combined into a single label set that encoded both the phase and trick. Generic phase labels covered the common positions at the start, middle, and idle time of any trick, while specific end-pose labels represented the final trick progression.

Improvements & challenges. This combination simplified the model output to a single classification at each frame, realistically capturing the structure of pole dance training videos. It further leverages the insight that early phases are similar across tricks, with no immediate need to classify the trick until a distinctive position is reached. However, this adaptation introduced significant class imbalance in the training data. While every attempt at a trick will contain generic labels, end-pose labels appear only when the particular trick is performed. Additionally, more challenging tricks, such as the Gemini, are underrepresented compared to beginner tricks. This imbalance is prone to introducing a data bias towards frequent classes to the classifier, while struggling to recognize rare poses. A countermeasure introduced weights to the model's loss function. The model training applied weighted cross-entropy by assigning higher weights to the

minority classes, to enforce penalties for errors on the rare classes. This is a standard technique to improve learning for imbalanced data, as it encourages the model to devote more capacity to learning those classes [ADAdCB19].

Class-weighted cross-entropy. Let K be the number of classes, $\mathbf{z}_t \in \mathbb{R}^K$ the logits at frame t, and $y_t \in \{0, \dots, K-1\}$ the ground-truth label. Let $\mathcal{T} = \{\, t \mid y_t \neq -100 \,\}$ be the set of valid frames (`ignore_index`$= -100$ in PyTorch). The loss is

$$\mathcal{L} = \frac{1}{|\mathcal{T}|} \sum_{t \in \mathcal{T}} w_{y_t} \left(-\log \frac{\exp(z_{t,y_t})}{\sum_{c=0}^{K-1} \exp(z_{t,c})} \right). \tag{5.1}$$

Class weights are inversely proportional to the square root of the empirical frame counts n_c, normalized to keep the average weight at 1:

$$w_c = \frac{(n_c + \varepsilon)^{-1/2}}{\frac{1}{K} \sum_{j=0}^{K-1} (n_j + \varepsilon)^{-1/2}}, \qquad \varepsilon = 10^{-8}. \tag{5.2}$$

This schedule reduces the dominance of frequent classes without the instability of strict inverse-frequency weighting. Rare end poses receive a stronger and more stable learning signal while the overall loss scale remains comparable.

Evaluation & fine-tuning. The following protocol and visualizations aim to verify the model's accuracy, frame-wise phase/trick recognition, reliable end pose identification, and strong generalization while handling class imbalance. The training kept the checkpoint with the lowest validation loss, while the best score was at epoch 29 with val_loss=0.121 and val_acc=93.83%. The test accuracy yields 93.82%, closely matching the validation accuracy and indicating good generalization ability. While fine-tuning the hyperparameters, different learning-rates were tested, as shown in Figure 5.6. This comparison identifies $5{\times}10^{-4}$ as the most reliable setting as it achieves the highest validation accuracy with stable late-epoch behavior.

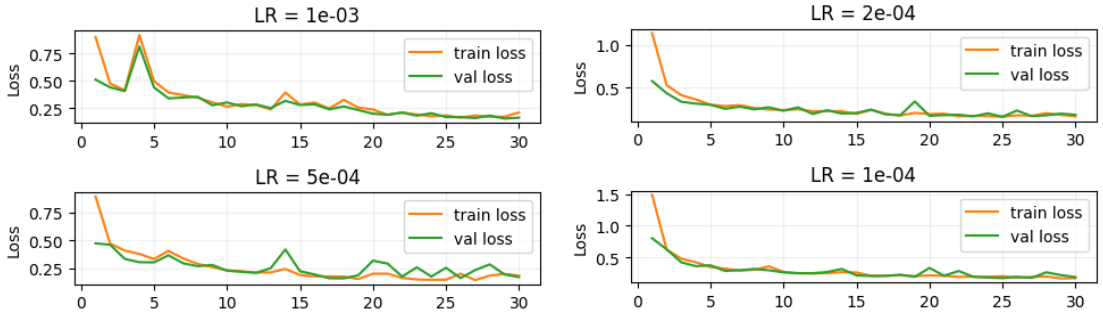


Figure 5.6: Learning rate sweep: over 30 epochs for different learning rates.

A confusion matrix helps to identify common misclassifications for each class. It counts how often each true label appears on the diagonal (correct) and how often it is predicted

as another label off the diagonal (errors). Therefore, a deeply colored diagonal suggests accurate classifications. The coloring in Figure 5.7 suggests frequent errors for all classes except for *floor* and *on_pole*. However, this is mainly due to its high representation in the dataset, as *on_pole* holds the most mislabeled entries. This aligns with the data, as transitions morph directly into the final position, so border frames near the end can already resemble the end pose. Per-class recall (see Figure 5.7) confirms this pattern: *floor* 97.8%, *on_pole* 91.1%, and all end poses between 98.1-99.4%. Trick-only accuracy reaches 98.74%, confirming dependable recognition once a trick sequence begins. Overall, the errors concentrate in transitional *on_pole* frames, while the crucial final poses are detected reliably.

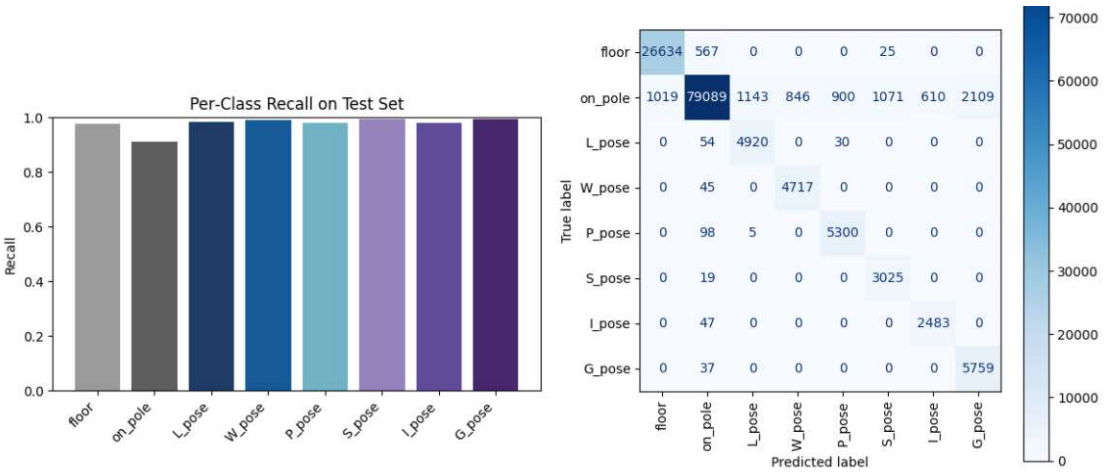


Figure 5.7: Left: per-class recall on the test set. Right: confusion matrix on the test set.

Multi-trick detection. A comparison on a challenging test case reveals the effectiveness of this solution. The test includes a single video in which a pole dancer performs nine tricks in succession. Before the weights were applied, the LSTM correctly detected 6 out of 8 tricks, with both missed tricks being instances of the Gemini. The improved version correctly recognized all 8 tricks in the sequence. As a result, the final model successfully overcame this challenge, reliably detecting all tricks, including those with fewer samples.

Figure 5.8: Successful detection of 8/8 tricks in one video.

Together, these results indicate that the single-head formulation, the chosen hyperparameters, and the class-weighting scheme deliver accurate frame-wise phase recognition and robust trick identification without resorting to a more complex architecture. Table 5.1 provides a final overview of the LSTM iterations.

Table 5.1: Bidirectional LSTM, 2 layers, hidden size 64, and dropout rate 0.2.

Iteration	**Label set**	**Settings**	**Data**	**Test acc.**
Two-head: phase+score	{Start, Transition, End}/ {fail, ok, good, perfect}	2× cross-entropy; no class weights; no decoding	50 (Pin-Up)	P: 96.1% S: 91.6%
Two-head: phase+trick (Protocol A)	{Start, Transition, End}/ {L, P, W, S, G, I}	2× cross-entropy; no class weights; no decoding	510 (Mixed)	P: 96.8% T: 99.9%
Single-head: (Protocol B)	{floor, on_pole} ∪ {L_pose, P_pose, W_pose, S_pose, G_pose, I_pose}	cross-entropy with class weights; median filter	836 (Final)	93.82% multi-trick

Training data across iterations. The metrics between model iterations are not strictly comparable, as the training samples grew during data collection. To provide a brief overview, the introduced LSTM models were trained on the following subsets:

1. **Phase/Score LSTM:** trained on **50** Pin-Up data samples only.
2. **Phase/Trick LSTM:** trained on **510** mixed videos covering all tricks, with ∼3 random data augmentations per video.
3. **Final single-head LSTM:** trained on complete dataset of **836** videos with ∼3-5 augmentations per clip, weighted by trick frequency (e.g., more augmentation for Gemini, fewer for Layout).

5.3 Geometric Scoring System

An initial idea was to use a data-driven approach, such as an autoencoder [TBL18], to learn ideal poses and quantify deviations from the ideal pose. Out-of-place body parts could be identified by analyzing the reconstruction error. However, this implementation requires a large set of "perfect" trick examples, while still being susceptible to bias. Instead, a geometric rule-based approach is realized, as this thesis already provides a working recognition model with a well-labeled dataset. This direct and interpretable solution circumvents the black-box nature of Deep Learning models and the potential bias inherent in subjective labeling. This approach aligns with how instructors critique

form via visual checks of angles and body alignment. Explicit checks ensure consistency and address the known issue that human judgment can be biased and inconsistent in dance evaluation, underscoring the need for objective measures [Qu24].

Rule design. Specific characteristics enable viewers to identify a performed trick. Instructors, therefore, establish those features as guidance to achieve this trick. Based on this observation, the evaluation set consists of **five to seven geometric rules** capturing its most characteristic requirements. The following categories group the rules as follows:

- **Body Orientation:** requires the dancer to orient the body at a particular angle relative to the pole or ground. For example, in a Layout, the upper body should be angled towards the floor, whereas in the Crucifix, the entire body should be upside-down.
- **Limb Alignment:** defines how straight or aligned a specific body part is. Joint angle calculations at elbows, knees, and other joints verify full extension or specific angles. For instance, straight legs are required in the Layout, Wrist Seat, or straight arms for the Straddle Invert and Crucifix.
- **Joint Proximity:** ensures correct distances between parts of the body. For example, in a Layout, the ankles should be closed, or in a Pin-Up, the right toes should be close to the left knee.

Each rule is defined by a target value and an accepted tolerance. During evaluation, the rules are applied to each frame of the end position to compute all the defined angles and distances. If a required joint holds a low confidence and is therefore not visible, the rule is marked and does not pass due to insufficient data. The evaluation script outputs a list of results, including a boolean for pass/fail, the measure value, and a short description.

Geometric rule definitions. Let $\mathbf{p}_i = (x_i, y_i, z_i, v_i)$ denote MediaPipe landmarks in normalized image coordinates, where $x, y \in [0, 1]$ and v is the visibility score. A visibility threshold $\min(v_i) \geq \tau$ (with τ=0.75) must pass for the joints to be evaluated.

Body orientation. For an oriented segment $B \to A$, define

$$\theta_{\text{orient}}(A, B) = \text{atan2}(y_B - y_A,\ x_A - x_B) \cdot \frac{180}{\pi},$$

A rule passes if $|\theta_{\text{orient}} - \theta^\star| \leq \Delta$, where $\theta^\star$ is the target (e.g., 180° for straight) and Δ is the tolerance.

Limb alignment. Given points A, B, C, define $\mathbf{u} = \mathbf{a} - \mathbf{b}$, $\mathbf{w} = \mathbf{c} - \mathbf{b}$ (2D). The internal angle is

$$\theta_{\text{align}}(A, B, C) = \arccos\left(\frac{\mathbf{u} \cdot \mathbf{w}}{\|\mathbf{u}\| \, \|\mathbf{w}\| + \varepsilon}\right) \cdot \frac{180}{\pi}, \quad \varepsilon = 10^{-8}.$$

Again, a rule passes if $|\theta_{\text{align}} - \theta^{\star}| \leq \Delta$.

Joint Proximity. For points A, B, the normalized 2D distance is

$$d(A, B) = \sqrt{(x_A - x_B)^2 + (y_A - y_B)^2}.$$

A rule passes if $d(A, B) \leq \Delta$ (e.g., 0.05).

The end pose score is computed as the fraction of rules passed, score $= \frac{\#\text{passed}}{\#\text{evaluable}}$. The system reports per-rule messages with measured values and targets for interpretability.

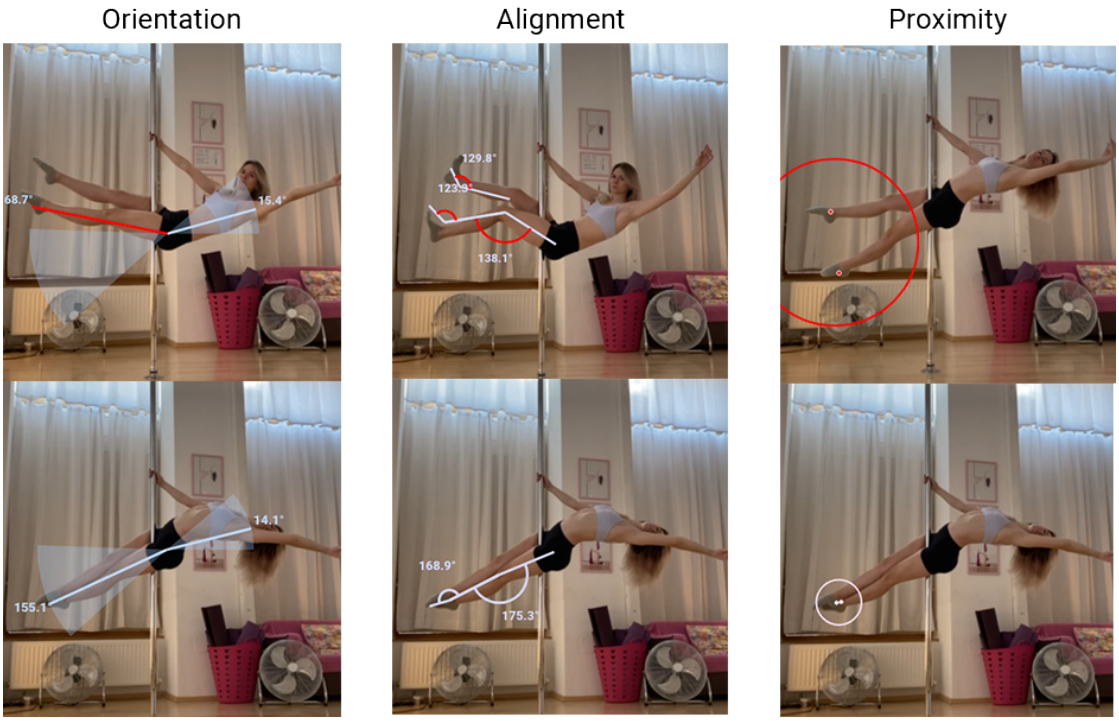


Figure 5.9: Examples of passed and failed rules displayed through interactive overlays. Top: failed, bottom: passed.

5.4 Pole-Arina: Application

This section presents the final prototype to realize Pole-Arina and enable user interaction. It is implemented as a web application to support heterogeneous hardware, enable fast interactive overlays, and centralize computational workloads on a server, while still running on consumer laptops and smartphones. The frontend (React/Next.js) runs in the browser, manages uploads, and renders interactive overlays. The backend (FastAPI + PyTorch) exposes a single analysis endpoint that extracts skeletal data, performs sequence classification with the Bi-LSTM, evaluates end poses with geometric rules, and returns a structured result. Videos are uploaded through the website's file picker (drag-and-drop supported). In the user study, recordings were made on a smartphone and then uploaded via the laptop browser. The full source code can be found here: Pole-Arina.

5.4.1 Backend

This section describes the server-side components to transform an uploaded video into trick phases and interpretable pose-quality feedback. The backend exposes a single analysis endpoint, runs the recognition pipeline, evaluates all end-pose frames, and returns a structured result.

API endpoint. The service employs FastAPI (`api.py`) with one primary route:

- **POST** `/analyze`
- **Input:** a single .mp4/.mov uploaded via the web interface (desktop or mobile browser)..
- **Optional query field:** confidence threshold for phase recognition and median filter kernel size.
- **Output:** JSON structure with detected tricks, per-trick feedback items, per-trick end-pose frames, and processing metadata.

At a high level, the route:

1. stores the upload in a temporary path,
2. calls `evaluate_video()` from `pole_arina.py`,
3. collects end-pose sequences, feedback, end-pose frames, and normalized skeleton sequences,
4. and serializes results to JSON.

This compact interface keeps the client simple and reduces integration effort in the frontend.

Trick & phase recognition module. The recognition pipeline is implemented in `pole_arina.py` and follows a six-step formula designed for clarity and robustness:

1. **Skeleton extraction.**
   The system reads frames via OpenCV and extracts 33x4 landmarks per frame using MediaPipe. It keeps both normalized image coordinates for model input and world/absolute coordinates available for display. To maintain consistency with the training distribution, the backend applies the same Exponential Moving Average (EMA) and interpolation to MediaPipe landmarks prior to classification.

2. **Input normalization.**
   Landmark sequences are stored as a tensor of shape (T,33,4), where T is the number of frames in the clip, and normalized consistently with the training setup. This preserves privacy and yields a compact representation for inference.

3. **Model loading.**
   The backend loads the trained lightweight bidirectional LSTM from a checkpoint and either selects a GPU if available or CPU otherwise.

4. **Per-frame inference.**
   The model outputs per-frame logits over the single-head output layer, while a softmax converts them to probabilities.

5. **Temporal smoothing.**
   A median filter with a configurable kernel (default 7) suppresses brief misclassifications from being detected as a fully realized end pose.

6. **Decoding.**
   The sequence decodes into contiguous runs with labels, start/end frame, and a confidence summary. The pipeline only passes stable end poses (confidence over 0.75) to the evaluation module.

The model returns the full timeline and skeleton sequences to the frontend to avoid re-running inference.

Feedback module. Pose scoring is implemented in `trick_evaluator.py` as a transparent rule engine over MediaPipe joints. The rules mirror the specification in Section 5.3 and cover three geometric evaluations, each mapping directly to a distinct overlay technique:

- **Body Orientation→Arc:** body angle relative to a horizontal line.
- **Limb Alignment→Angle:** internal angle at a joint, e.g., knee or elbow.
- **Joint Proximity→Proximity:** normalized 2D distance between landmarks.

For each detected end-pose frame, the evaluator:

- checks trick-specific rule configurations,
- applies a visibility threshold,
- computes pass/fail per visible rule given a target and tolerance,
- aggregates a pose score as the fraction of passed rules over all evaluable rules.

This ensures interpretability, as every score is derived from a readable checklist.

By combining these modules, the backend provides a compact and reliable analysis service for trick recognition and evaluation. A single `/analyze` call turns a raw clip into a trick timeline and an interpretable feedback list.

5.4.2 Frontend

Design & technology. The interface aims to be intuitive, straightforward, and coach-like. Guided interaction minimizes friction for non-technical users and keeps the focus on the trick analysis. An early Gradio prototype validated the pipeline workflow, but limited interactivity led to a full React/Next.js implementation. The fully realized prototype is a lightweight single-page web application with Material UI (MUI) for accessible, consistent components and D3.js for interactive overlays.

User interface. A stepper UI organizes the workflow into four screens:

1. **Upload:** initial video upload and analysis.
2. **Summary:** overview of all detected tricks.
3. **Detail:** interactive single-trick view with detailed feedback and overlays.
4. **Dashboard:** session summary for all detected tricks and scores.

Upload. A single video upload triggers model analysis, starts a new training session, and resets previous results. First, the user selects a single trick video and hits "analyze" to send it to the backend. While waiting for a response, the UI displays a loading progress bar. After a few seconds, the frontend receives the results and the user advances to the summary view. Each initial upload resets the frontend and starts a new training session.

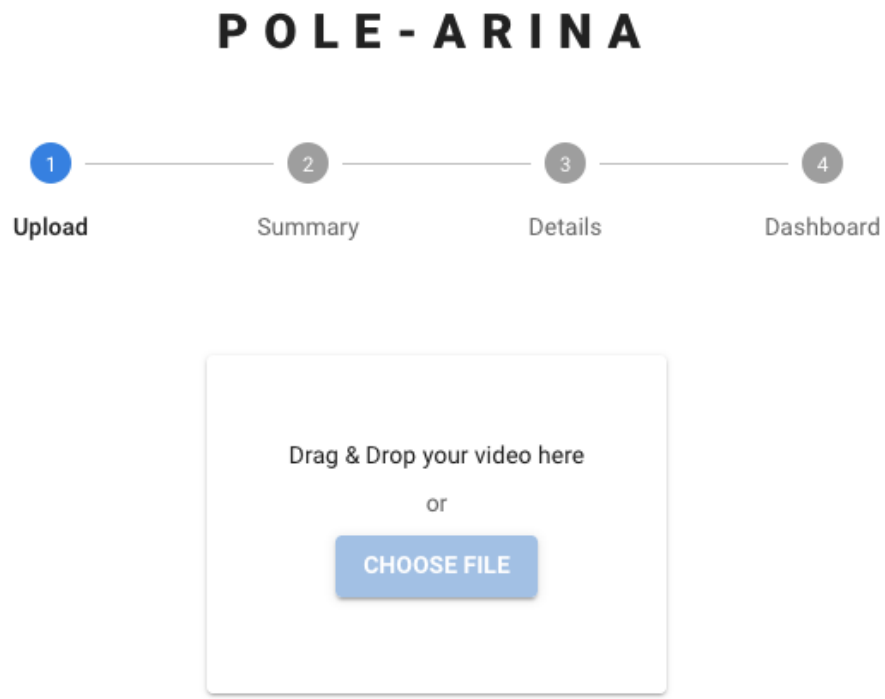


Figure 5.10: Upload & analyze: single-video upload starts a new session.

Summary. This tab provides an overview of all detected tricks in the current training session. Analyzed tricks appear as cards grouped by trick. Each card shows the trick name, the middle frame of the detected end sequence (thumbnail), and the performance score. A floating "+" button at the bottom right corner allows users to add further attempts to the current session without clearing prior results. Videos of the same trick are displayed in chronological order with a simple indicator for improvement, no change, or decline between the cards. Clicking a card lets the user advance to the detailed view of the selected trick.

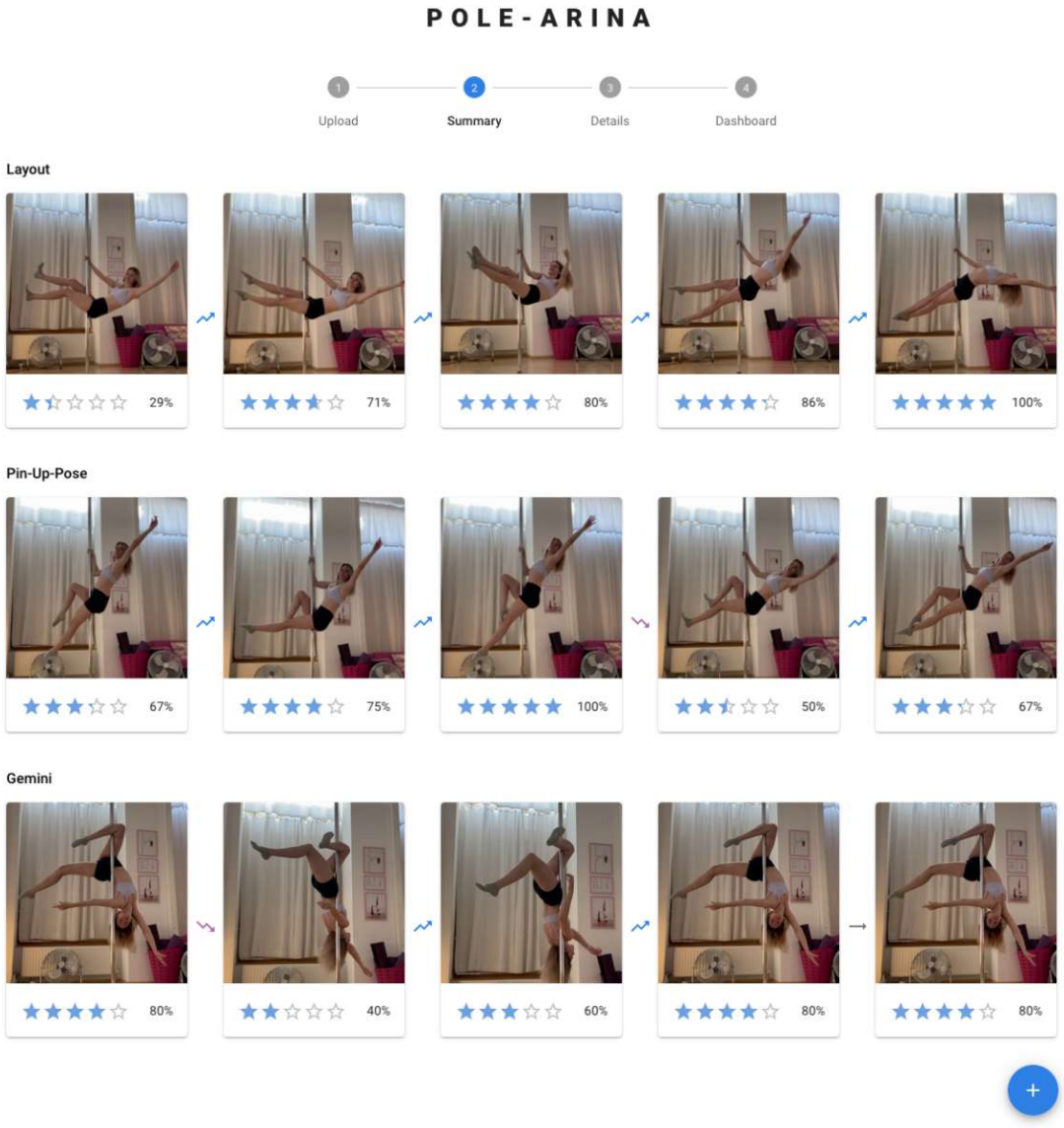


Figure 5.11: Summary: trick cards with thumbnails, scores, and an *add-video* button.

Detail. The selected trick opens a control panel. First, the trick evaluation returns two scores: the visibility score and the performance score. The first score is defined as the percentage of evaluable rules, while the second calculates the percentage of passed rules out of all evaluable rules. The left pane displays the current frame, initially chosen based on highest visibility and then highest performance. The right pane offers overlay toggles by rule family (orientation, alignment, proximity) and a rule list. Each rule item includes a status icon (pass, fail, or not visible), a rule description, an improvement tip (if failed), a score with a target range, and an overlay toggle button. By default, the panel activates all overlays for failed rules, but provides global control through the top buttons or single control for each rule item. Two MUI rating components display each score, including the exact value. A frame slider at the bottom enables manual frame selection to explore changes over time.

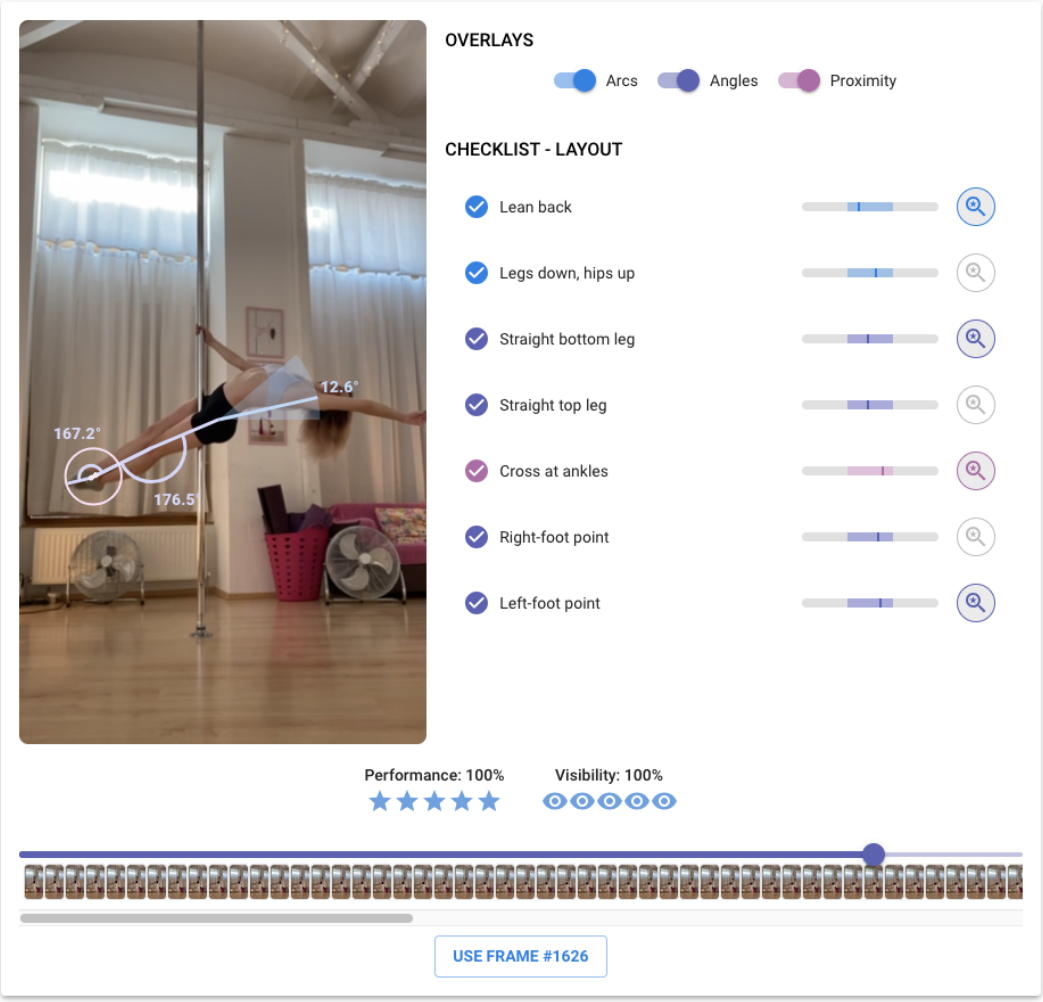


Figure 5.12: Detail: frame viewer with overlay controls and per-rule feedback.

Dashboard. The session dashboard summarizes the training progress across all tricks. It displays the average performance and visibility scores, and the number of performed tricks. An interactive line chart plots score over attempts per trick, with a thumbnail display at the side. The display (see Figure 5.14) can either show the best, worst, or a selected attempt through the line chart, for direct visual comparison.

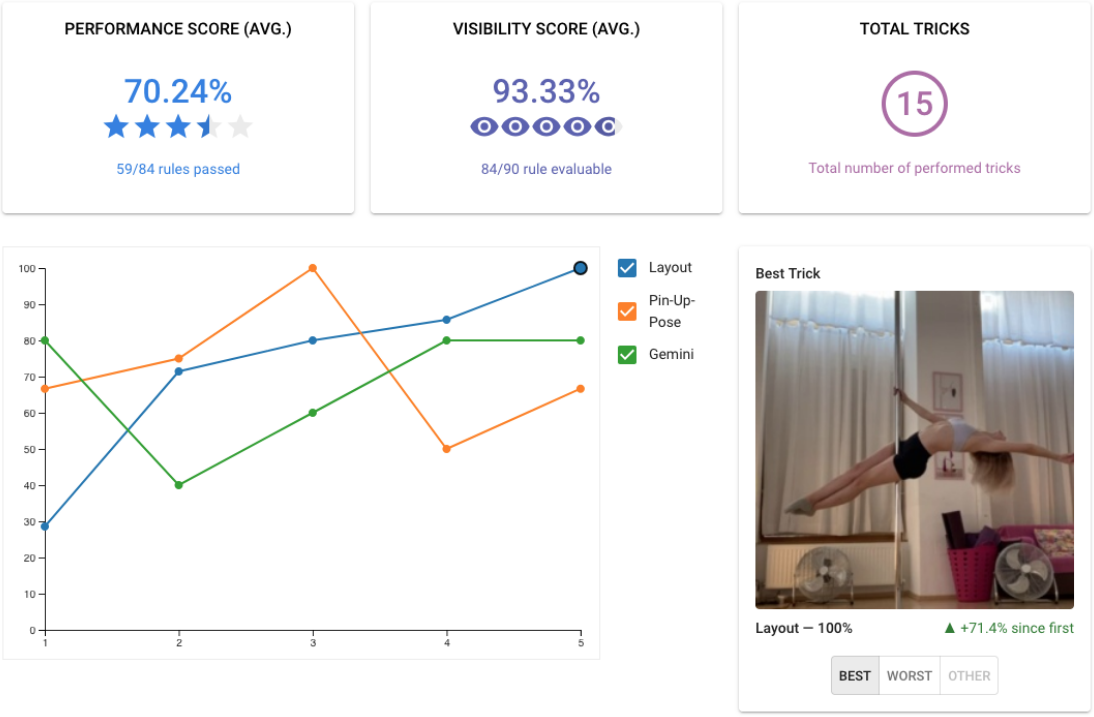


Figure 5.13: Dashboard: session-level statistics.

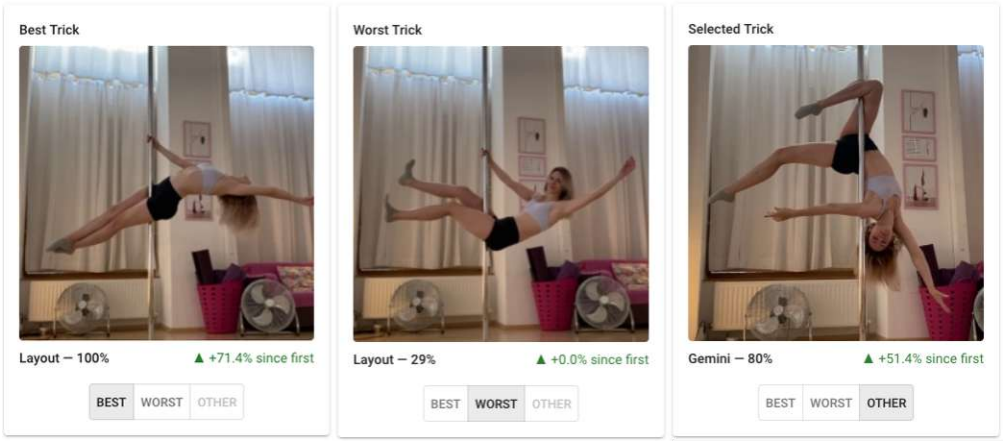


Figure 5.14: Best/Worst display example.

To summarize, the frontend guides the user through a simple, coach-like flow: upload a clip, review a trick gallery, inspect per-rule feedback with overlays, and reflect on progress in the dashboard. Together with the backend, the interface turns model outputs into actionable tips and session-level insights, enabling the user-study assessment that answers **RQ3**. Table 5.2 lists the complete rule catalog used by the evaluator across all six tricks, including targets and tolerances.

Table 5.2: Complete rule catalog across all tricks.

Trick	Rule	Type	Joints (idx)	Target	Tol.
Layout	Lean back	Orientation	(11, 23)	22.5°	±22.5°
	Legs down, hips up	Orientation	(27, 23)	−155°	±20°
	Straight bottom leg	Alignment	(23, 25, 27)	180°	±20°
	Straight top leg	Alignment	(24, 26, 28)	180°	±20°
	Cross at ankles	Proximity	(28, 27)	0	≤ 0.05
	Right-foot point	Alignment	(26, 28, 32)	165°	±15°
	Left-foot point	Alignment	(25, 27, 31)	165°	±15°
Wrist Seat	Straight left leg	Alignment	(23, 25, 27)	180°	±20°
	Straight right leg	Alignment	(24, 26, 28)	180°	±20°
	Lean back	Orientation	(11, 23)	20°	±20°
	Right-foot point	Alignment	(26, 28, 32)	165°	±15°
	Left-foot point	Alignment	(25, 27, 31)	165°	±15°
Pin-Up	Lean slightly back	Orientation	(11, 23)	45°	±15°
	Straight leg down	Orientation	(27, 23)	−135°	±10°
	Toe to knee	Proximity	(32, 25)	0	≤ 0.10
	Top leg into passé	Alignment	(23, 25, 27)	180°	±20°
	Right-foot point	Alignment	(26, 28, 32)	165°	±15°
	Left-foot point	Alignment	(25, 27, 31)	165°	±15°
Straddle Invert	Push hips up	Orientation	(23, 11)	112.5°	±22.5°
	Lean back, straight arms	Alignment	(11, 13, 15)	160°	±20°
	Straight left leg	Alignment	(23, 25, 27)	180°	±15°
	Straight right leg	Alignment	(24, 26, 28)	180°	±15°
	Head-back tilt	Alignment	(7, 11, 23)	160°	±20°
	Right-foot point	Alignment	(26, 28, 32)	165°	±15°
	Left-foot point	Alignment	(25, 27, 31)	165°	±15°
Gemini	Push hips up	Orientation	(23, 11)	112.5°	±22.5°
	Back-leg straight	Alignment	(23, 25, 27)	160°	±20°
	Back-leg horizontal	Orientation	(27, 23)	180°	±15°
	Right-foot point	Alignment	(26, 28, 32)	165°	±15°
	Left-foot point	Alignment	(25, 27, 31)	165°	±15°
Crucifix	Left-arm straight	Alignment	(11, 13, 15)	180°	±20°
	Right-arm straight	Alignment	(12, 14, 16)	180°	±20°
	Cross at ankles	Proximity	(28, 27)	0	≤ 0.05
	Body upside-down	Orientation	(0, 27)	−90°	±15°
	Right-foot point	Alignment	(26, 28, 32)	165°	±15°
	Left-foot point	Alignment	(25, 27, 31)	165°	±15°

CHAPTER 6

Evaluation & Results

This chapter presents the evaluation strategy and outcomes for the Pole-Arina system. The evaluation encompasses both a quantitative assessment of the pose recognizer and scoring model, as well as a controlled user study to examine the system's effectiveness and usability in practice. Quantitative tests on a held-out dataset validate the recognizer (**RQ1**) and the geometric scoring (**RQ2**). A controlled user study assesses effectiveness and usability in practice (**RQ3**). The study evaluates the system in real-world training scenarios, comparing AI-assisted feedback with traditional video self-review in terms of user trust, improvement efficiency, feedback understandability, and overall usability.

6.1 Quantitative Model Performance

The recognizer was evaluated on a held-out test split to estimate generalization. The protocol reported:

- **Per-frame accuracy** on the full label set and **trick-only accuracy** on end-pose classes (RQ1).
- **Per-class precision/recall** and **confusion matrices** to reveal systematic misclassifications.
- **Temporal stability** via post-processing with a fixed confidence detection threshold and median kernel chosen on validation data.
- **Multi-trick robustness** on sequences containing several tricks in succession.

The classifier achieved a per-frame accuracy of **93.82%** across all classes and a trick-only accuracy of **98.74%** when considering only the final trick poses. Per-class precision and recall were analyzed, and a confusion matrix revealed that the most common trick

misclassification occurred for the `on_pole` label. Between tricks, the two visually similar tricks *Layout* and *Pin-Up* proved most confusing. The system was also tested on video sequences containing multiple tricks in succession, and it demonstrated robust performance by correctly segmenting and recognizing each trick in order. Overall, these results indicate that the recognizer provides accurate and stable identification of pole tricks, forming a solid foundation for the feedback mechanism.

6.2 User Study Design

This study's hypotheses evaluate whether Pole-Arina's feedback would be trusted and understood, whether it would improve form efficiently, and whether it would be rated as usable. To answer **RQ3**, a controlled between-groups experiment compared Pole-Arina feedback against traditional self-review. Participants practiced a single preselected pole trick for five trials. The *Experimental* condition used Pole-Arina, while the *Control* condition used a standard video with self-assessment. Measurement combined per-trial Likert items, a post-session questionnaire, the System Usability Scale (SUS), and open-ended questions for qualitative insights.

6.2.1 User Study Methodology

Evaluation goals. To evaluate the system's performance in real training scenarios, the user study verifies the following hypotheses:

- **H1: Trust & Adoption:** Participants in the *Experimental* condition will report higher trust in the accuracy of feedback and greater confidence about what to improve next than those in the *Control* condition.
- **H2: Efficiency:** Participants in the *Experimental* condition will show greater improvement across five trials than those in the *Control* condition.
- **H3: Understandability:** Participants in the *Experimental* condition will rate the clarity and helpfulness of feedback higher than those in the *Control* condition.
- **H4: Usability:** The *Experimental* condition will receive a higher System Usability Scale (SUS) score than the *Control* condition.

The chosen criteria align with the project's research questions while testing the technology's acceptance. Factors related to trust also include: demographic variables, privacy protection, robustness, transparency, and performance [LWHL24], which are all considered throughout this thesis. In particular, fostering user trust is crucial, as prior studies have indicated the importance of this criterion for the effective utilization of AI systems [LWHL24]. This study's hypotheses evaluate if Pole-Arina's feedback would be trusted, understood, improve their form efficiently, and be rated as a usable system.

Study design. The experiment targeted $N \approx$20-30 participants, ranging from non-dancers to advanced pole dancers. The user study followed a between-groups design. The study randomly assigned one of two conditions to each participant and completed all trials under that feedback method. The options are:

- **Control:** using traditional self-review through video recording and post hoc replay.
- **Experimental:** using the Pole-Arina application for deep-learning-based evaluation and feedback about pose correctness.

To minimize expectancy effects, participants remained unaware of the alternative condition and study aims until the debriefing. The study assigned a trick based on the dancer's experience. Beginners would practice the Layout while intermediate and advanced levels performed the Pin-Up. The two tricks share similar geometric structure and hold the same difficulty, yet the Pin-Up enforces stricter pose rules. They also formed the most frequent confusion trick pair in the recognition model (see Figure 5.7). Therefore, the feedback rules would be tested on similar form requirements, additionally challenging the model's recognition abilities.

User study protocol. All sessions took place in the same pole studio to maintain consistency. A smartphone camera recorded the participants at a fixed position and angle. For better accessibility, a two-screened laptop station hosted Pole-Arina and served as a reviewing platform for each trick attempt. Therefore, between each review, the recording was transferred to the laptop via Apple AirDrop. Each participant booked a one-hour time slot, which was typically used for 45 minutes. An online form guided the user through each step with follow-up questions. The study protocol was structured as follows, with identical flow for both the Control and Experimental conditions:

1. **Orientation & Consent:** Participants were briefed on the study goals and what to expect, including a caution about physical strain and possible bruises (also known as pole kisses). After participants signed the informed consent, the first section of the online form gathered demographic data. This established the context of each user and ensured a mix of backgrounds.
2. **Task Assignment & Demonstration:** The condition (Control or Experimental) was assigned at random while keeping it balanced throughout the study and the dancers' pole experience. As the setting simulated pole practice at home (or at least without an instructor), the participant was shown a demonstration video, instead of a real-life tutorial. Additionally, they received a walkthrough of the selected reviewing method. Both groups were allowed to watch the demonstration video unrestricted. A free attempt was allowed to get familiar with the trick and setup.
3. **5-Trial Loops with Feedback:** The core of the study is executed through a 5-trial practice loop. The participant attempted the assigned trick five times, aiming

to improve with each repetition. After each attempt, the participant reviewed their performance. Either they watched their video playback or examined the feedback provided by Pole-Arina. After each review, the participant filled out a per-trial survey. The form encouraged them to reflect on their performance, identify mistakes, and adjust the trick accordingly.

4. **Post-Session Questionnaire:** After completing all trials, the participant filled out a post-study survey about the employed feedback method. It includes Likert-scale statements that directly address the evaluation hypotheses. Additional open-ended questions allowed for qualitative feedback, and a standard ten-item System Usability Scale (SUS) was used to evaluate overall usability. The SUS is a widely used survey that provides a global measure for usability, through ten Likert items [B+96]. All questions were answered with respect to the assigned condition, to allow baseline comparison of traditional versus AI-assisted review methods.

5. **Pole-Arina specific Feedback:** After the main evaluation, all participants, regardless of their group, were offered a chance to try out Pole-Arina. Therefore, everyone had the opportunity to experience the AI-coaching system and fill out a final questionnaire to provide qualitative feedback.

6. **Debrief:** The study concluded with a debrief, during which the purpose and conditions were thoroughly explained, and participants were allowed to ask questions and share additional remarks.

Evaluating the results. By design, the user study produced both quantitative and qualitative data. Quantitatively, each trial provided an objective performance score from the implemented system and a subjective self-assessment from the participant. This allowed for a comparison of self-perception vs. actual performance. Using the evaluation scores, an improvement metric estimates the participant's trick execution across the trials. The primary hypothesis for efficiency was to demonstrate that the experimental group showed greater improvement than the control group, indicating faster and more effective learning. For subjective responses, such as clarity of feedback and confidence after each trial, trends were analyzed across the five attempts. All Likert scales were matched and summarized according to the hypotheses. The SUS responses were both statistically compared and converted to a 0-100 score per standard guidelines [B+96]. Finally, the qualitative open-ended answers were analyzed by open coding methodology to identify important codes.

Overall, this evaluation approach combines a quantitative check on the model's performance with a human-centered assessment of the system's effectiveness and user experience.

Question-hypothesis mapping & aggregates. The underlying hypotheses grouped each questionnaire item and appropriately aggregated it into simple composites. Each composite is identified by a short code for cross-referencing in Chapter 6.

- **H1 (Trust & Adoption).**
  - **H1A1 (per-trial):**
    Q: *How confident are you that you know what to improve next?*
    → **Aggregate:** mean across trials.
  - **H1A2 (post-session):**
    Q: *The feedback I received was accurate.*
    Q: *I felt confident that this digital review method correctly reflected my performance.*
    → **Aggregate:** mean over both items.
- **H2 (Efficiency).**
  - **H2A1 (performance slope):**
    Q: *How would you rate your performance of this trial?*
    → per-participant least-squares slope of the five self-ratings.
  - **H2A2 (performance delta):**
    Q: *How would you rate your performance of this trial?*
    → per-participant change of the same self-ratings.
  - **H2A3 (self vs. system agreement):**
    Q: *How would you rate your performance of this trial?*
    → per-participant averages of self-ratings and Pole-Arina scores for correlation and paired comparison.
- **H3 (Understandability).**
  - **H3A1 (per-trial):**
    Q: *How clear was the feedback you received from the digital review method?*
    → **Aggregate:** mean across trials.
  - **H3A2 (post-session):**
    Q: *I understood how to interpret this feedback to improve my form*
    Q: *The digital review method helped me identify mistakes.*
    → **Aggregate:** mean over both items.
- **H4 (Usability).**
  - **H4A1 (SUS):**
    System Usability Scale score (0-100) computed per standard rules [B+96].

6.3 User Study Results

This section reports the outcomes of the between-groups user study. It first profiles participants' demographics to contextualize subsequent analyses. It then presents the hypothesis tests for H1-H4 with corresponding visualizations, highlighting where differences between conditions are statistically reliable and where effects were not detected. The section closes with qualitative feedback that complements the quantitative findings and surfaces design implications for Pole-Arina.

6.3.1 Demographics.

By design, the user study proposed two different groups (dancers or non-dancers) with two distinct conditions (Experimental or Control). To ensure meaningful results, the protocol required a sample size of between 20 and 30 participants. In the end, a total of 33 participants completed the study: 17 in the Experimental group and 16 in the Control group (see Figure 6.1). Similar to the dataset demographics, the final set resembles the composition of regular pole classes while covering a broad range of execution quality.

Figure 6.1: Condition balance bars. Bars show absolute counts.

Experience balance. Depending on the prior pole experience, a matching practice trick was assigned for the trials. Balancing experience across conditions was therefore essential to avoid confounding when comparing feedback methods. The stacked balance bar in Figure 6.2 shows a near-equal distribution of non-dancers and dancers within both the Experimental and Control groups, enabling fair analyses of improvement and feedback quality across skill levels.

Figure 6.2: Experience balance bars. Bars show absolute counts.

Gender context. Participation reflected typical studio demographics, with most respondents identifying as female, fewer male participants, and no respondents selecting the alternative option. The distribution is shown in Figure 6.3.

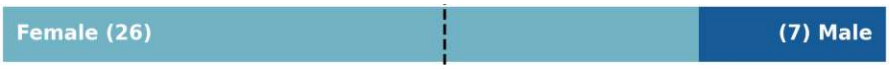


Figure 6.3: Gender balance bars. Bars show absolute counts.

Age distribution. Figure 6.4 presents the age spread using a horizontal boxplot. The ages span between 19 and 56 years old, with a median of 30, which aligns with the local studio's age distributions.

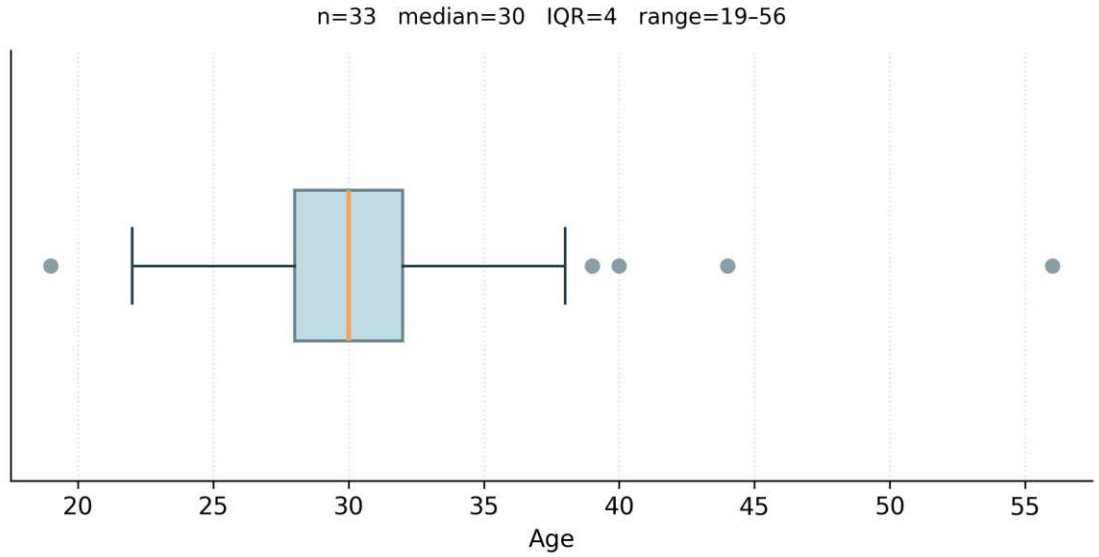
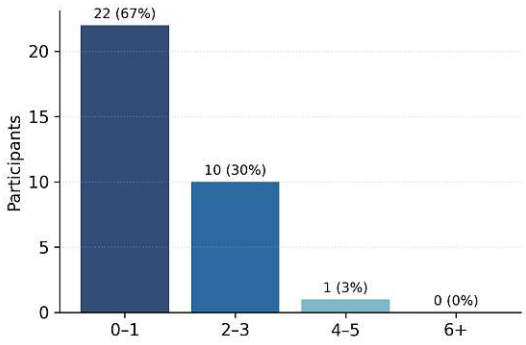
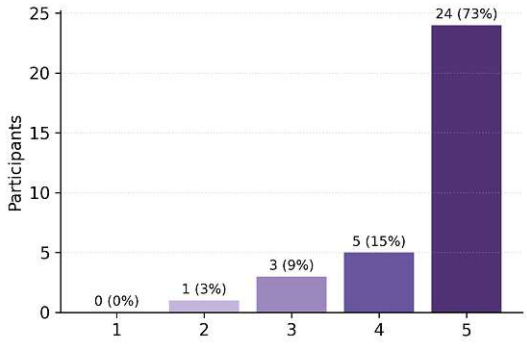


Figure 6.4: Age distribution boxplot.

Training frequency & technology comfort. To contextualize prior exposure and likely learning dynamics, Figure 6.5a presents the weekly pole-training frequencies, and Figure 6.5b shows a self-estimated technology comfort rating on a 1-5 Likert scale.



(a) Weekly pole training frequency: 0-1, 2-3, 4-5, or 6+.



(b) Technology comfort: higher equals more comfortable with apps/websites.

Figure 6.5: Participant distributions for (a) weekly training frequency and (b) technology comfort. Category labels match the questionnaire options. Bars show absolute counts.

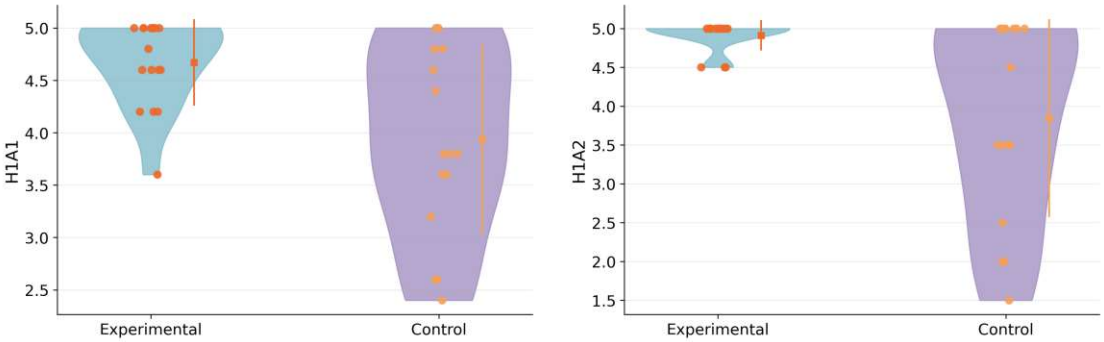
6.3.2 Hypothesis Tests

All analyses were run in IBM SPSS Statistics Version 31.0.0.0. For each composite, normality was assessed with Kolmogorov-Smirnov and Shapiro-Wilk tests. As all cases violated normality, the Mann-Whitney U was chosen to perform all conditional comparisons (Experimental N=17 vs. Control N=16).

H1: Trust & Adoption (H1A1, H1A2).

- **H1A1 - Results:** There was a statistically significant difference in H1A1 between Experimental (mean rank = 20.88) and Control (mean rank = 12.88) condition, U=70.00, Z=−2.430, p=.015.
- **H1A2 - Results:** There was a statistically significant difference in H1A2 between Experimental (mean rank = 20.79) and Control (mean rank = 12.97) condition, U=71.50, Z=−2.702, p=.007.

Interpretation: Participants using Pole-Arina reported higher accuracy for the feedback and greater confidence that the method reflected their performance, and they felt it was clearer what to improve next.



(a) H1A1: per-trial confidence, 1-5.

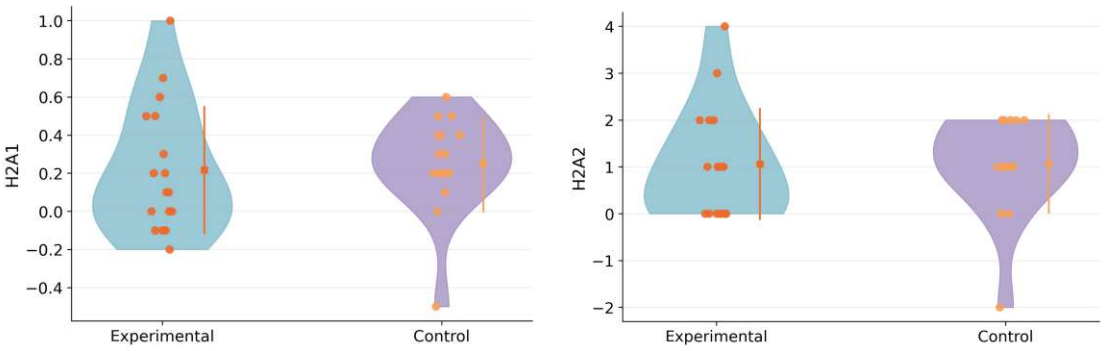
(b) H1A2: post-session trust, 1-5.

Figure 6.6: Value distributions by condition, H1.

H2: Efficiency

- **H2A1 - Results:** There was no statistically significant difference in H2A1 (slope-based improvement) between Experimental (mean rank = 15.41) and Control (mean rank = 18.69) condition, U=117.50, Z=−.698, p=.485.
- **H2A2 - Results:** There was no statistically significant difference in H2A2 (delta-based improvement) between Experimental (mean rank = 15.91) and Control (mean rank = 18.16) condition, U=117.50, Z=−.698, p=.485.

Interpretation: Both groups improved over five trials to a similar extent. On average, participants' self-ratings systematically differed from the system's ratings (inspection of per-participant differences indicated a tendency to rate themselves lower than the tool). However, the rank-order association between the two was weak.



(a) H2A1: improvement slope per participant. (b) H2A2: improvement delta per participant.

Figure 6.7: Value distributions by condition, H2.

H3: Understandability

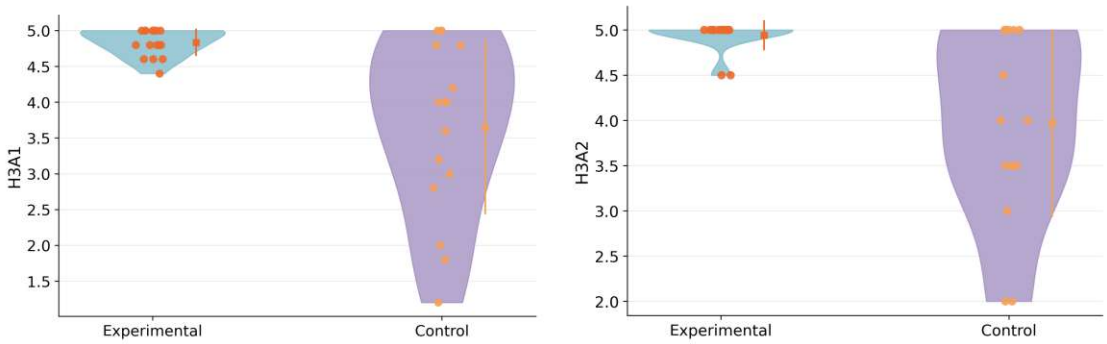
- **H3A1 - Results:** There was a statistically significant difference in H3A1 between Experimental (mean rank = 21.94) and Control (mean rank = 11.75) condition, U=52.00, Z=−3.100, p=.002.
- **H3A2 - Results:** There was a statistically significant difference in H3A2 between Experimental (mean rank = 21.59) and Control (mean rank = 12.13) condition, U=58.00, Z=−3.266, p=.001.

Interpretation: Pole-Arina's overlays and explanations were rated as significantly clearer and more helpful for understanding how to fix mistakes, compared to traditional video analysis.

H4: Usability

- **H4A1 - Results:** There was a statistically significant difference in H4A1 between Experimental (mean rank = 20.76) and Control (mean rank = 13.00) condition, U=72.00, Z=−2.335, p=.020.
- **H4A1 - Descriptives:** Experimental M=95.44, SD=4.07, Median = 95.0; Control M=86.41, SD=11.14, Median = 88.75.

Interpretation: Both reviewing methods achieved high SUS scores, with Pole-Arina rated significantly higher ("Best imaginable" usability) by benchmark guidelines (see Table 6.1).



(a) H3A1: per-trial clarity, 1-5. (b) H3A2: post-session understandability, 1-5.

Figure 6.8: Value distributions by condition, H3.

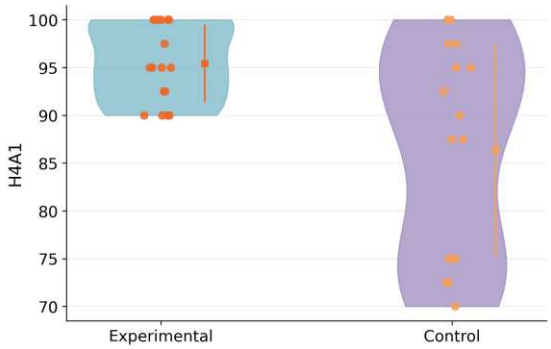


Figure 6.9: Value distributions by condition, H4A1: SUS score, 0-100.

Table 6.1: SUS benchmark guidelines (after Bangor et al.[BKM09]).

SUS score	Adjective rating	Grade
≥ 90	Best imaginable	A+
80.3–89	Excellent	A
74–80	Good	B
68–73	OK–Good	C
51–67	OK / Marginal	D
< 51	Poor / Not acceptable	F

6.3.3 Qualitative Results

A Pole-Arina-specific post-study questionnaire was used to collect qualitative feedback from all 33 participants. The responses provide rich insight into how users perceived the system's usefulness, accuracy, and areas for improvement, which helps in evaluating our hypotheses.

Would you use Pole-Arina regularly in your training? Overall, the feedback was overwhelmingly positive: **all participants indicated that they would use the application in their training routine**. Many said they would use it *definitely* or *regularly* to improve their form, with a few noting they might not use it every single attempt, but certainly every training session or for challenging moves. For example, one participant wrote, "I would definitely use it, it's a wonderful and invaluable tool to correct even moves that you thought you knew how to do". This strong usage intention demonstrates a high acceptance of the system. Participants were enthusiastic about incorporating the app into solo practice at home, and even as a complement during classes.

Did the overlays help you understand how to improve your form? All participants found the feedback rules and visual overlays helpful in understanding and correcting their technique. In particular, dancers reported that the system's precise, objective feedback helped them notice details they would otherwise overlook. "The graphical representation of the lines is really helpful to see where and how you can improve the movement" noted one user.

Do you think Pole-Arina would be useful in beginner and/or advanced classes? Many highlighted that the tool was especially beneficial for beginners. Simultaneously, most participants agreed it would "be useful to all levels", with advanced dancers using it to "give the pose the final touch" and perfect their form. A few novices did caution that a complete beginner might feel overwhelmed using the app without any prior instruction. However, after learning the basics in class, they felt the app would be handy for independent practice.

How accurate did you find Pole-Arina's performance analysis? Participants generally reported that the system's analysis was accurate and reliable, giving them confidence in the feedback. Many described the pose analysis as "very accurate" noting that the app correctly identified their form errors. However, a few minor inaccuracies were observed. For instance, a few users mentioned the system occasionally had trouble recognizing a fully pointed foot or confused two very similar tricks (Layout vs. Pin-Up) on the first try. One participant wrote: "Some accuracy problems in identifying similar poses, but other than that it seemed quite accurate." Others noted that if a body part was hidden from the camera, the system sometimes missed that joint, leading to a less optimal frame selection or an incomplete evaluation. Nonetheless, participants mainly understood these issues as minor limitations of the current prototype (often related to camera angle or body positioning) rather than fundamental flaws. This overall trust in the accuracy of the system is critical for its validity and was reflected in comments like "Absolut akurat! Ich stimm der App vollkommen zu, was die Bewertung meiner Posen betrifft." (Absolutely accurate! I fully agree with the app regarding the evaluation of my poses.).

What did you like most about using Pole-Arina? Participants highlighted several aspects of the application that they liked best. A dominant theme was the visual and detailed nature of the feedback. Nearly all users praised the overlay of angles, lines, and highlighted body segments on their pose images, which made it "immediately clear what to improve and how." They also appreciated the simple, intuitive interface and workflow. Several described the tool as "easy to use" and the feedback presentation as "clear and specific." The ability to scrub through recorded video frames and see feedback for each attempt in a summary was also frequently mentioned. Other answers mentioned that the tool introduced a game-like or self-competitive element to practice: "it was a fun, gamified experience ... it motivated me to improve each time to look better," wrote one participant. Such comments suggest the system can increase engagement and enjoyment in training, potentially improving adherence.

What would you change or improve about Pole-Arina? While the overall feedback was positive, participants also provided valuable suggestions and pointed out current limitations, which helped identify avenues for future work. For example, many participants wanted to see a reference of the "perfect" pose for comparison. Similarly, participants asked for integrated tutorials or tip videos. Another highly requested feature was real-time feedback. For instance, the app could give an audio cue whenever the dancer achieves the correct form or if a major mistake occurs. Users found the idea of live feedback exciting, as it could help them adjust their pose immediately rather than only correcting on the next attempt.

In summary, the qualitative results show that participants overwhelmingly found the system beneficial, easy to use, and effective. At the same time, users provided constructive feedback highlighting the current limitations. The current system has a limited set of moves with offline feedback and minor detection and scoring quirks. These limitations, however, directly point to concrete improvements that form the basis of future work.

6.4 Discussion

Summary of findings. The study examined whether a deep learning-based coaching system improves trust, learning efficiency, understandability, and usability compared with traditional video self-review. Overall, three out of four hypotheses were supported:

- **H1 (Trust & Adoption)**
- **H3 (Understandability)**
- **H4 (Usability)**

H2 (Efficiency) was not supported within the five-trial protocol: improvement slopes and deltas did not differ significantly between conditions. Together, these results indicate that **RQ3** is answered positively for trust, understandability, and usability, while efficiency advantages were not detected under the present design.

Interpretation & likely causes. Two factors explain the absence of a measurable efficiency difference over five trials. First, traditional video replay represents a strong baseline for immediate improvement, especially if one is familiar with the tricks. Second, efficiency advantages from structured cues often manifest over longer practice horizons and across various mistakes. Furthermore, participants described overlays as precise and helpful for understanding *how* to adjust form, and adoption intent was uniformly high, suggesting that benefits may accumulate with continued use.

Alignment with model results. The quantitative model's performance established a reliable basis for the user experience. Per-frame accuracy and trick-only accuracy were high, and end pose detection remained robust across multi-trick sequences. The rule-based scoring provided transparent, geometric justifications, while the qualitative remarks are consistent with these properties and help explain the observed advantages in trust and understandability.

In summary, the study validates Pole-Arina's primary goal: pairing accurate recognition with transparent, geometry-based explanations yields feedback that users trust and understand, laying the groundwork for measurable skill gains as practice extends beyond a single session.

CHAPTER 7

Conclusion

This thesis introduced Pole-Arina, a novel marker-less deep learning-based coaching system designed for static pole dancing tricks. The development and evaluation of Pole-Arina addressed a clear gap in technology for dismissed sports, providing feedback and analysis without the need for wearable sensors. The system recognizes the performed trick, isolates end poses, and grades form using transparent geometric rules rendered as visual overlays.

A primary contribution is a domain-specific dataset tailored to pole dancing technique. It includes: **836** clips from **58** participants, annotated for phases and end poses, and released as 3D skeleton (with an estimated depth value) sequences to protect privacy. A revised single-head label scheme supports multi-trick recordings and explicit background modeling. Building on this foundation, a lightweight bidirectional LSTM performs frame-wise recognition over six static tricks, and a rule engine converts landmark geometry into pass/fail checks and an overall pose score. A full-stack prototype implements the pipeline and surfaces interpretable feedback through interactive overlays.

The research questions were addressed as follows:

- **RQ1:** was met with strong results. **93.82%** per-frame accuracy across all classes and **98.74%** trick-only accuracy on end-pose frames, with robust behavior on multi-trick sequences.
- **RQ2:** was realized via explicit, trick-specific geometric rules that map angles, orientations, and proximities. Feedback and visual cues enable consistent, transparent grading.
- **RQ3:** was examined in a controlled user study (N=33): compared to traditional video self-review, Pole-Arina achieved significantly higher ratings for trust/accuracy and understandability, and a higher SUS usability score (*best imaginable* by benchmark guidelines).

These findings suggest that Pole-Arina can provide meaningful support for independent practice in pole sports. The approach generalizes to other domains where end poses encode most of the instructional signal and where transparent rule checks promote trust.

Current implementation limitations & improvements. The present prototype is optimized for single-camera, offline analysis of static tricks. This choice simplifies deployment but introduces practical constraints. First, robustness can degrade under strong occlusions, extreme viewpoints, or low light. Next, geometric rules emphasize alignment and orientation but do not yet cover fine stylistic criteria. Furthermore, the workflow involves smartphone capture and laptop-based processing, so near–real-time feedback on-device is not evaluated. Immediate improvements include on-device/mobile inference for faster computation, camera-guidance prompts (viewing angle, distance, lighting) to reduce failure modes, confidence/uncertainty indicators in the UI, and incremental rule catalogs with editable tolerances.

User study limitations & future work. It is important to note the following limitations:

- **Sample and setting:** The study took place in a single studio with a modest sample size (N=33), which bounds external validity.
- **Protocol length:** Five attempts in one session may be insufficient to expose efficiency differences that require consolidation over longer practice intervals.
- **Set-up:** The user study employed a laptop-based workflow with smartphone recordings and post-hoc transfer. Therefore, mobile deployment or real-time evaluation was not tested.
- **Task scope:** Only a fixed set of tricks and rule configurations was evaluated. Generalization to a broader repertoire remains to be demonstrated.

Finally, qualitative reasoning followed a lightweight thematic approach. While themes were consistent with quantitative outcomes, future work could include multi-coder reliability checks to strengthen interpretive claims.

Altogether, Pole-Arina demonstrates that a compact Bi-LSTM paired with interpretable geometric scoring and a privacy-first dataset can deliver accurate recognition, clear feedback, and high user trust. This establishes a practical baseline for AI coaching in pole dance and points to an expandable pathway for accessible, marker-less coaching across movement disciplines.

Overview of Generative AI Tools Used

Generative AI tools (OpenAI ChatGPT and Grammarly) were used solely as writing aids for surface-level editing (grammar, punctuation, and minor rephrasing) and for translating the abstract/acknowledgements. No AI tools were used to generate research ideas, design the study, create or analyze data, write technical content, or produce figures or results. All scientific claims, methods, datasets, and conclusions are my own.

Übersicht verwendeter Hilfsmittel

Generative KI programme (OpenAI ChatGPT und Grammarly) wurden im Schreibprozess zur Kontrolle der Grammatik und Zeichensetzung, sowie in geringem Ausmaß als Formulierungshilfe eingesetzt. KI wurde nicht zur Entwicklung wissenschaftlicher Ideen, Studienplanung, der Verarbeitung von Daten, Generierung von Tabellen, Grafiken oder technischer Texte genutzt. Alle wissenschaftlichen Behauptungen, Methoden, Datensätze und Schlussfolgerungen sind meine eigenen.

List of Figures

List of Tables

List of Algorithms

Bibliography

[ADAdCB19] Yuri Sousa Aurelio, Gustavo Matheus De Almeida, Cristiano Leite de Castro, and Antonio Padua Braga. Learning from imbalanced data sets with weighted cross-entropy function. *Neural processing letters*, 50(2):1937–1949, 2019.

[AJJB24] Aditya Agarwal, Parth Jha, Ojas Jain, and Asish Bera. Poa-net: Dance poses and activity classification using convolutional neural networks. In *2024 IEEE Region 10 Symposium (TENSYMP)*, pages 1–6. IEEE, 2024.

[B+96] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[BGR+20] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. Blazepose: On-device real-time body pose tracking. *arXiv preprint arXiv:2006.10204*, 2020.

[BKM09] Aaron Bangor, Philip Kortum, and James Miller. Determining what individual sus scores mean: Adding an adjective rating scale. *Journal of usability studies*, 4(3):114–123, 2009.

[BNKB23] Asish Bera, Mita Nasipuri, Ondrej Krejcar, and Debotosh Bhattacharjee. Fine-grained sports, yoga, and dance postures recognition: A benchmark analysis. *IEEE Transactions on Instrumentation and Measurement*, 72:1–13, 2023.

[BP21] Abeba Birhane and Vinay Uday Prabhu. Large image datasets: A pyrrhic win for computer vision? In *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1536–1546. IEEE, 2021.

[CECS18] Steffi L Colyer, Murray Evans, Darren P Cosker, and Aki IT Salo. A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system. *Sports medicine-open*, 4:1–15, 2018.

[CHS+19] Zhe Cao, Gines Hidalgo, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity

fields. *IEEE transactions on pattern analysis and machine intelligence*, 43(1):172–186, 2019.

[Cit25] Spin City. *The Ultimate Pole Bible*. Spin City Aerial Fitness Ltd, 2025.

[CZDK21] Anargyros Chatzitofis, Dimitrios Zarpalas, Petros Daras, and Stefanos Kollias. Democap: Low-cost marker-based motion capture. *International Journal of Computer Vision*, 129(12):3338–3366, 2021.

[DWDW25] Zhao Du, Shan Wang, Ziyan Deng, and Fang Wang. Unveiling the power of ai fitness apps: a uses and gratifications perspective. *Journal of Global Information Management (JGIM)*, 33(1):1–28, 2025.

[EC20] Aysu Ezen-Can. A comparison of lstm and bert for small corpus. *arXiv preprint arXiv:2009.05451*, 2020.

[Fed25] International Pole Sports Federation. Code of points 2025 – 2027. `https://ipsfsports.org/downloads/Uncategorised/ipsf_pole_sports_code_of_points_2025-2027_final_070120240.pdf`, 2025. Accessed: 2025-08-16.

[FHC19] Muhammad Fikri, Samiadji Herdjunanto, and Adha Cahyadi. On the performance similarity between exponential moving average and discrete linear kalman filter. In *2019 Asia Pacific Conference on Research in Industrial and Systems Engineering (APCoRISE)*, pages 1–5. IEEE, 2019.

[GFS05] Alex Graves, Santiago Fernández, and Jürgen Schmidhuber. Bidirectional lstm networks for improved phoneme classification and recognition. In *International conference on artificial neural networks*, pages 799–804. Springer, 2005.

[Goo25] Google. Mediapipe pose landmarker. `https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker`, 2025. MediaPipe Solutions, Google AI Edge. Accessed: 2025-08-22.

[GRRCR23] Indrajeet Ghosh, Sreenivasan Ramasamy Ramamurthy, Avijoy Chakma, and Nirmalya Roy. Sports analytics review: Artificial intelligence applications, emerging technologies, and algorithmic perspective. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 13(5):e1496, 2023.

[HNR+25] Wenjun Huang, Yang Ni, Arghavan Rezvani, SungHeon Jeong, Hanning Chen, Yezi Liu, Fei Wen, and Mohsen Imani. Recoverable anonymization for pose estimation: A privacy-enhancing approach. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 5239–5249. IEEE, 2025.

[HS97] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[HZMY22] Changwu Huang, Zeqi Zhang, Bifei Mao, and Xin Yao. An overview of artificial intelligence ethics. *IEEE Transactions on Artificial Intelligence*, 4(4):799–819, 2022.

[KK18] Yeonho Kim and Daijin Kim. Real-time dance evaluation by markerless human pose estimation. *Multimedia Tools and Applications*, 77:31199–31220, 2018.

[LCLX25] Yihua Li, Hongyue Chen, Yiqing Li, and Yetong Xin. Poespin: A human-ai dance to poetry system for movement-based verse generation. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 8(3):1–13, 2025.

[Lem24] Mark A. Lemley. How generative ai turns copyright upside down. *Stanford Technology Law Review*, 25(1):21–48, 2024.

[LHK22] Chen-Chieh Liao, Dong-Hyun Hwang, and Hideki Koike. Ai golf: Golf swing analysis tool for self-training. *IEEE Access*, 10:106286–106295, 2022.

[LTNX23] Julienne LaChance, William Thong, Shruti Nagpal, and Alice Xiang. A case study in fairness evaluation: Current limitations and challenges for human pose estimation. In *Association for the Advancement of Artificial Intelligence 2023 Workshop on Representation Learning for Responsible Humancentric AI (R2HCAI), Washington, DC*, volume 1, 2023.

[Luc24] Nicola Lucchi. Chatgpt: a case study on copyright challenges for generative artificial intelligence systems. *European Journal of Risk Regulation*, 15(3):602–624, 2024.

[LWHL24] Yugang Li, Baizhou Wu, Yuqi Huang, and Shenghua Luan. Developing trustworthy artificial intelligence: insights from research on interpersonal, human-automation, and human-ai trust. *Frontiers in psychology*, 15:1382693, 2024.

[LX13] Jianchao Lv and Shuangjiu Xiao. Real-time 3d motion recognition of skeleton animation data stream. *International Journal of Machine Learning and Computing*, 3(5):430, 2013.

[MK23] Marina Mikami and Noriyuki Kida. Categorizing rhythmic jumping motion using motion capture without markers. *Advances in Physical Education*, 13(2):93–105, 2023.

[MMN+24] Carmina Liana Musat, Claudiu Mereuta, Aurel Nechita, Dana Tutunaru, Andreea Elena Voipan, Daniel Voipan, Elena Mereuta, Tudor Vladimir

Gurau, Gabriela Gurău, and Luiza Camelia Nechita. Diagnostic applications of ai in sports: a comprehensive review of injury risk prediction methods. *Diagnostics*, 14(22):2516, 2024.

[NJ22] Dinesh Naik and CD Jaidhar. A novel multi-layer attention framework for visual description prediction using bidirectional lstm. *Journal of Big Data*, 9(1):104, 2022.

[Nor25] Northern Digital Inc. Optotrak3020. `https://tsgdoc.socsci.ru.nl/images/e/eb/Optotrak_Certus_User_Guide_rev_6%28IL-1070106%29.pdf`, 2025. Optotrak3020. Accessed: 2025-09-08.

[OGK+24] Bengie L Ortiz, Vibhuti Gupta, Rajnish Kumar, Aditya Jalin, Xiao Cao, Charles Ziegenbein, Ashutosh Singhal, Muneesh Tewari, and Sung Won Choi. Data preprocessing techniques for ai and machine learning readiness: Scoping review of wearable sensor data in cancer care. *JMIR mHealth and uHealth*, 12(1):e59587, 2024.

[PC16] European Parliament and Council. Regulation (eu) 2016/679 (general data protection regulation). `https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng`, 2016. Accessed: 2025-08-17.

[PPW+24] Zhiqiang Pu, Yi Pan, Shijie Wang, Boyin Liu, Min Chen, Hao Ma, and Yixiong Cui. Orientation and decision-making for soccer based on sports analytics and ai: A systematic review. *IEEE/CAA Journal of Automatica Sinica*, 11(1):37–57, 2024.

[PTM17] Paritosh Parmar and Brendan Tran Morris. Learning to score olympic events. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 20–28, 2017.

[Qu24] Jiping Qu. A dance movement quality evaluation model using transformer encoder and convolutional neural network. *Scientific Reports*, 14(1):32058, 2024.

[Sch11] Ronald W Schafer. What is a savitzky-golay filter?[lecture notes]. *IEEE Signal processing magazine*, 28(4):111–117, 2011.

[SK19] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

[STL24] Xiang Suo, Weidi Tang, and Zhen Li. Motion capture technology in sports scenarios: a survey. *Sensors*, 24(9):2947, 2024.

[TBL18] Michael Tschannen, Olivier Bachem, and Mario Lucic. Recent advances in autoencoder-based representation learning. *arXiv preprint arXiv:1812.05069*, 2018.

[TP23] Atima Tharatipyakul and Suporn Pongnumkul. Deep learning-based pose estimation in providing feedback for physical movement: A review. 2023.

[TS14] Alexander Toshev and Christian Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1653–1660, 2014.

[Ubi09] Ubisoft Paris. Just dance, 2009. Video game.

[Vic25] Vicon Motion Systems Ltd UK. Vicon. `https://www.vicon.com/`, 2025. Vicon. Accessed: 2025-09-08.

[VSP+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[Wea20] Charlene Weaving. Sliding up and down a golden glory pole: Pole dancing and the olympic games. *Sport, Ethics and Philosophy*, 14(4):525–536, 2020.

[Wie25] TU Wien. Data protection at tu wien. `https://www.tuwien.at/en/tu-wien/organisation/central-divisions/data-protection-and-document-management/data-protection-at-tu-wien`, 2025. Accessed: 2025-08-17.

[WKM+19] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[XKCP24] Chu Xin, Seokhwan Kim, Yongjoo Cho, and Kyoung Shin Park. Enhancing human action recognition with 3d skeleton data: A comprehensive study of deep learning and data augmentation. *Electronics*, 13(4):747, 2024.

[Yu20] Hongbo Yu. Application research and analysis of college pole dance teaching based on virtual reality technology. In *International Conference on Application of Intelligent Systems in Multi-modal Information Analytics*, pages 602–610. Springer, 2020.

[Zar21] S Zargar. Introduction to sequence learning models: Rnn, lstm, gru. *Department of Mechanical and Aerospace Engineering, North Carolina State University*, 37988518, 2021.

[ZWC+23] Ce Zheng, Wenhan Wu, Chen Chen, Taojiannan Yang, Sijie Zhu, Ju Shen, Nasser Kehtarnavaz, and Mubarak Shah. Deep learning-based human pose estimation: A survey. *ACM computing surveys*, 56(1):1–37, 2023.