# SKATE SENSOR: A MOBILE APPLICATION AND IMU SYSTEM TO ASSIST IN FIGURE SKATING TECHNIQUES THROUGH INFORMATION AND AI

Jocelyn Zhang [1] , Andrew Park [2]

[1] University High School, 4771 Campus Dr, Irvine, CA 92612
[2] University of California, Irvine, Irvine, CA 92697

***ABSTRACT***

*Figure skating is a physically demanding and expensive sport, and beginners often lack enough accessible support to practice safely and improve efficiently. Skate Sensor addresses this problem through a wearable sensing system paired with a mobile application that helps users analyze skating techniques and access educational resources [1]. The project combines two shin-mounted sensor pods, a Flutter mobile app, and machine learning models developed with Python and PyTorch [2]. Together, these components collect motion data, synchronize left and right leg activity, classify techniques, and evaluate whether an attempt matches successful motion patterns. Several implementation challenges had to be considered, including sensor synchronization, limited training data, and memory use within the app's media features. These issues were addressed through timeline alignment, expanded data collection strategies, and more efficient loading of visual content. Experimental design focused on model accuracy and the impact of informational screens. Overall, Skate Sensor demonstrates strong potential as a practical tool for safer, more informed, and more independent beginner training.*

***KEYWORDS***

*Figure Skating, Coaching, Injuries, Skating techniques, Artificial Intelligence*

## 1. INTRODUCTION

Figure skating is a beautiful but highly demanding sport, especially for beginners. In the United States, participation has been growing through rink programs and Learn to Skate programs, and interest in the sport continues to rise after the 2026 Winter Olympics [3]. More skaters are entering the sport, but many of them quickly discover that figure skating requires constant practice, access to coaching, and the ability to repeat skills over and over again with proper technique. Beginner skaters often do not have enough affordable and accessible support while training. Figure skating is expensive because families must pay for private lessons, ice time, equipment, competition fees, and off-ice training. This makes it difficult for many skaters to receive frequent, personalized feedback. The sport is also physically and mentally draining. Skaters must train balance, strength, flexibility, and precision all at once, while also dealing with mental blocks, pressure from coaches and parents, and the frustration of inconsistent progress. Too much repetition can also lead to fatigue and injury. This problem is important because it affects not only skaters, but also the future of sport. When beginners lack support, they lose confidence, get injured too frequently, or quit early. Figure Skating needs more accessible resources and tools to help beginners learn more safely and effectively.

David C. Wyld et al. (Eds): WiMNeT, NLDM, EDUPT, SIP, AISO, COMIT – 2026
pp. 143-164, 2026. CS & IT - CSCP 2026 DOI: 10.5121/csit.2026.1601013

Shi, Ozaki, and Honda used five wearable IMUs to study figure skating jump mechanics in detail [4]. Their work was strong in mathematical precision, but it focused on only one technique and did not provide automatic coaching feedback for everyday training. Wang and Sun developed a skate mounted wearable system that recorded motion data and later replayed it in a Unity application even though the ground plane wasn't big enough. Although their method supported movement review, it required post session data transfer, tracked only one skate, placed more emphasis on playback than on direct analysis, and made some incorrect assumptions. Hardegger et al. showed that IMUs and machine learning could be applied successfully in skating environments. However, their work was more focused on recognition of athletic movement than on practical coaching assistance, and it was centered on hockey rather than figure skating. Skate Sensor improves on these methods by prioritizing portability, live analysis, beginner accessibility, and automated feedback for independent figure skating practice.

Skate Sensor is a mobile app with a hardware companion device designed to help skaters better analyze and improve their techniques on the rink. Using a combination of accelerometer sensors connected to ESP32 boards, the user's movements can be transmitted via Bluetooth to the app for analysis [5]. The sensors themselves are to be strapped to the user's shins, facing outward to allow for accurate recordings with minimal interference. The app itself is designed to have a combination of algorithmic and machine learning models to help determine the success of a technique being performed. Users will be able to view their recorded activity history based on the collected sensor data and analysis, all of which is saved and processed completely offline. Additionally, the app features guides on both technique execution and injury prevention exercises that continue to grow over time. The technique screen displays varied techniques including jumps and spins with video demonstration for users, and detailed analysis of each technique with exercises that can help users improve on specific techniques. As for the injury screen, it features different exercises that target specific body parts that are vulnerable to injury, stretches that increase mobility and help prevent injuries, as well as a variety of ways to recover from injuries. In order to maximize accessibility, the app is developed in Flutter to enable easy maintenance and export across both Android and iOS [6].

A. Model accuracy on the rink
a. Sensors strapped to a skater and Axel, Salchow, and Loop are each performed 10 times
b. Recording also done for each attempt and evaluated by human judges
c. Models give their predictions and the result is compared to the human evals
B. User interaction + impact for info screens
a. Have two groups: one who uses skate sensor and one that doesn't
b. After a 3-week window, outcomes are compared via survey responses

Section 4 evaluated two important areas of Skate Sensor: model performance during real skating and the practical value of the app's educational content. The first experiment tested whether the system could accurately recognize axel, salchow, and loop attempts on the rink. A skater wore both sensors on the outer shins and performed each technique multiple times while each attempt was also recorded for later human review. The model's predictions were then compared against those human evaluations. This experiment is important because the system's success depends on reliable recognition and scoring.

The second experiment examined whether the app's technique and injury-prevention screens improved user outcomes. A user group and control group were compared over three weeks through survey data. This experiment helps measure whether the informational side of the system could support safer practice and better technique development.

## 2. CHALLENGES

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

### 2.1. Synchronizing Dual Skate Sensor Data Streams

A major challenge would be keeping the left and right skate sensors accurately synchronized. If the two devices send data at slightly different times, the system could misinterpret the timing between leg movements, which is critical in figure skating. Delays or unequal sampling rates could reduce the accuracy of technique detection. To address this, I could design the system to timestamp every reading as it arrives and align both streams onto a shared timeline. I could also average repeated readings within short intervals and fill small gaps with nearby values, so each model input remains complete and consistent.

### 2.2. Improving Training Data Diversity for Skating Models

Another challenge would be collecting enough real skating data to properly train and evaluate both the technique classifier and the success scoring model. Machine learning systems depend on large and varied data, so limited data could cause errors when identifying and scoring techniques for skaters. Differences in coaching style and execution could also make patterns less consistent. To address this, I could gather data from many practice sessions and from different skaters and include both successful and unsuccessful attempts. I could also expand the dataset over time so newer model versions become more reliable and better generalized.

### 2.3. Managing Memory for Technique GIF Playback

Another challenge could be managing memory on the techniques screen, especially since the app includes GIFs to demonstrate skating techniques for users. Since the media files would take up a lot of space, loading too many at once could make the app crash. This would be a huge problem if users scroll through several techniques in one session. To solve this, I could make the media files smaller, load them only when a skater selects a technique, and remove them from memory when they are no longer being viewed.

## 3. SOLUTION

The main structure of my program consists of three major components, the hardware sensor system, the mobile application, and the AI analysis. These three parts work together to collect skating motion data, organize it, and turn it into useful feedback for the skater.

The program begins with the hardware layer. Two wearable sensor pods are attached to the outside of the skater's shins, one on each leg. Each pod uses an ESP32-S3 QT Py microcontroller and a 9 DoF IMU to collect motion data such as acceleration, angular velocity, and elevation. These sensors continuously measure how each skate is moving during practice [7].

Next, the data is sent to the mobile app, which was built using Flutter. The app receives the live sensor streams, timestamps them, and aligns the left and right skate data into one synchronized timeline. The app also displays practice information, technique results, and other session features for the user.

Finally, the synchronized data is passed into the AI component. Python and PyTorch are used to process the raw sensor data into fixed time windows and train machine learning models. One model identifies which skating technique is being performed/attempted, while another evaluates

whether the attempt matches a successful motion pattern. The output is then returned as technique detections, success or fail judgments, and session summaries.

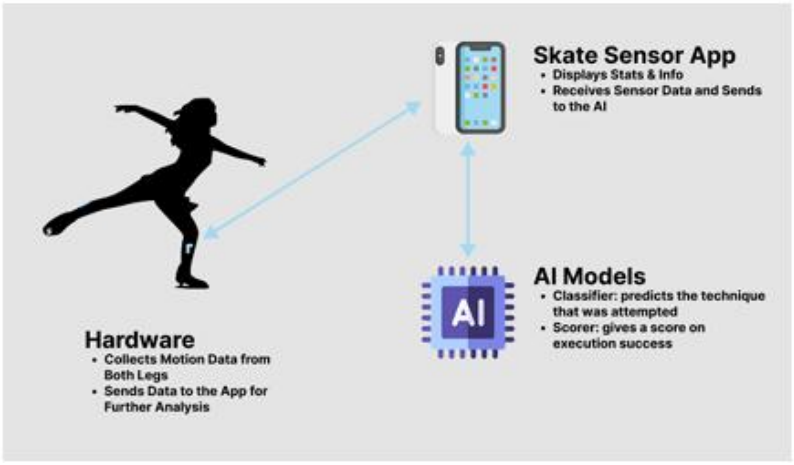


Figure 1. Overview of the solution

The hardware component's purpose is to collect skating motion data from both legs during practice. The sensors consist of ESP32-S3 QT Py boards, IMU sensors, and a battery [8]. This component relies on motion sensing through fixed measurement units, which detect acceleration, rotation, and elevation. The hardware acts as the input layer of the program by capturing the skater's movements and sending that data to the mobile app for further processing.

Figure 2.Hardware Architecture of the Dual IMU Sensor Pods

```
class SensorState:
    def __init__(self):
        self.elevation = 0.0
        self.jump_active = False
        self.jump_peak = 0.0
        self.jump_phase = 0

        self.spin_active = False
        self.spin_duration = 0
        self.spin_rpm = 0

    def get_simulated_data(self):
        """Generates a single data packet."""
        # --- Jump Simulation ---
        if self.jump_active:
            # Ascending phase
            if self.jump_phase < 10:
                self.elevation += self.jump_peak / 10
            # Descending phase
            else:
                self.elevation -= self.jump_peak / 10

            self.jump_phase += 1
            if self.elevation < 0:
                self.elevation = 0
                self.jump_active = False
        else:
            # Randomly trigger a new jump
            if random.random() < 0.01: # 1% chance each tick to jump
                self.jump_active = True
                self.jump_phase = 0
                self.jump_peak = random.uniform(0.1, 0.5) # 10cm to 50cm jump height
                print(f"--- SIM ({DEVICE_NAME}): Starting a jump to {self.jump_peak*100:.0f} cm ---")

        # --- Spin Simulation ---
        gyro_y = 0.0
        if self.spin_active:
            # Maintain spin speed for a duration
            gyro_y = self.spin_rpm * (2 * 3.14159) / 60 # Convert RPM to rad/s
            if random.random() < 0.2: # Add some noise
                gyro_y *= random.uniform(0.8, 1.2)
            self.spin_duration -= 1
            if self.spin_duration <= 0:
                self.spin_active = False
                print(f"--- SIM ({DEVICE_NAME}): Spin finished ---")
        else:
            # Randomly trigger a new spin
            if random.random() < 0.008: # 0.8% chance each tick to spin
                self.spin_active = True
                self.spin_rpm = random.randint(100, 400)
                self.spin_duration = random.randint(15, 30) # for 1.5-3.0 seconds
                print(f"--- SIM ({DEVICE_NAME}): Starting a {self.spin_rpm} RPM spin ---")
```

Figure 3.Simulated Sensor State Generation Logic

This code shows the sensor component of Skate Sensor. The Sensor state class stores the current state of the skater's motion. This code runs whenever the program needs to create sample motion data for testing. In the init method, variables such as elevation, jump_active, spin_active, etc. were created to track jump and spin behavior (elevation, whether a jump or spin is currently happening, peak of a jump height, spin speed, etc). The get simulated date method updates this information. For example, if a jump is active, the skater's elevation would increase and then decrease, which models the motion of a real jump. This same process is used for spins, where gyro is used to calculate the spin speed. This method helps the app test the analysis of motion data.

The mobile app component's purpose is to receive sensor data, organize it, and display information to the skater. I used Flutter to build this part of the system. This component also relies on synchronization, since the program must align data coming from the left and right skate sensors onto the same timeline. The app serves as the center of the program. It connects the hardware to AI, manages live practice sessions, stores exported data, and presents results in a format the user can understand and adapt to. It also acts as an information hub where users can look up information on techniques and how to perform them as well as injury prevention exercises to minimize harm during training.

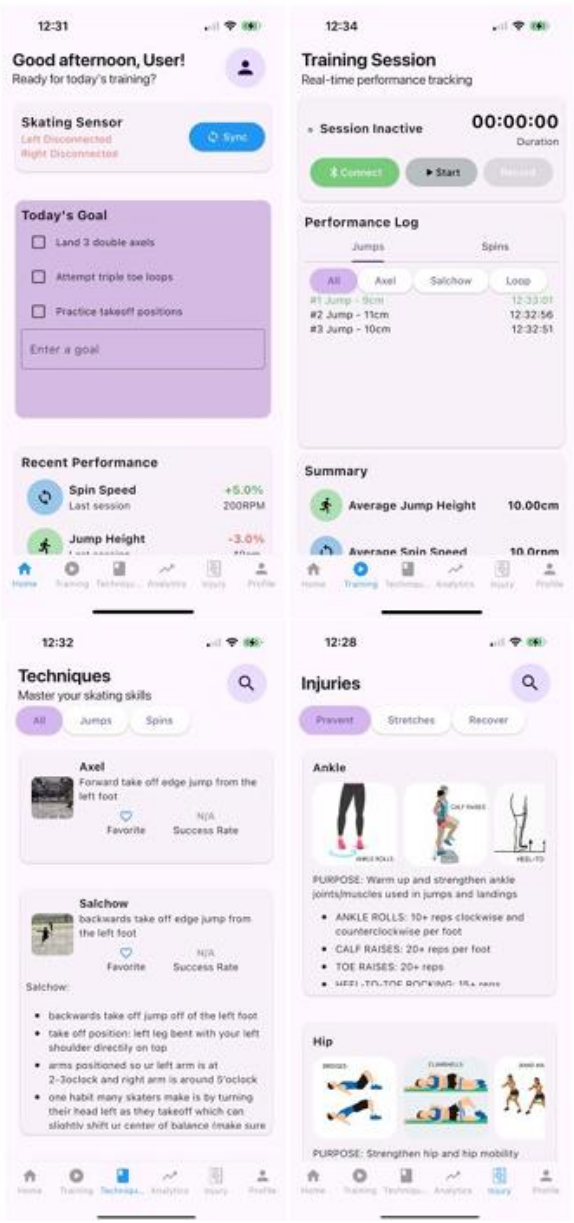


Figure 4. Flutter-Based Mobile Application Interface

```
class _MediaPopUpState extends State<MediaPopUp> {
@override
Widget build(BuildContext context) {
  return Dialog.fullscreen(
    backgroundColor: const Color.fromARGB(0, 0, 0, 0),
    child: InkWell(
      onTap: () {
        Navigator.of(context).pop();
      },
      child: Container(
        color: const Color.fromARGB(126, 0, 0, 0),
        width: SizeConfig.horizontal! * 100,
        height: SizeConfig.vertical! * 100,
        child: Center(
          child: ClipRRect(
            borderRadius: BorderRadius.circular(8.0),
            child: (widget.imagePath.contains(".gif"))
                ? Gif(
                    image: AssetImage(widget.imagePath),
                    autostart: Autostart.loop,
                    placeholder: (context) => const Text("Loading..."),
                  )
                : Image.asset(widget.imagePath),
          ),
        ),
      ),
    ),
  );
}
}
```

Figure 5.Full-Screen Media Popup Implementation in Flutter

This code is part of the mobile app interface in Skate Sensor. It is responsible for showing video demonstrations of techniques in a full screen pop. It runs when a user taps on the media in the app and wants to view it more closely. The _MediaPopUpState class controls the appearance and behavior of the techniques screen. The main method shown is the build method, which creates what the user sees in the pop up. The method first makes a full screen with a transparent background so the media can appear on top of the current screen. It also adds an Inkwell which lets the user tap anywhere on the screen to exit the demonstration and return to the previous screen. A semitransparent container then covers the background to make the media easier to focus on. The video/gif/media is then centered and clipped, so it has rounded corners using ClipRRect [9]. The variable widget.imagePath stores the file path of the video shown and checks whether that path contains .gif. If it does, the app uses the gif widget to display the video in a loop. If it does not, it uses an Image.asset to display just an image. This code does not send data to a back-end server and is only used to create the app's interface.

The AI component's purpose is to interpret the synchronized sensor data and turn it into meaningful skating feedback. I used Python and PyTorch to build and train this part of the system. This component relies on neural networks, which are machine learning models that learn patterns from examples. The AI is the final part of program [10]. It identifies which technique is being performed and evaluates whether the motion matches a successful attempt, allowing the app to provide useful feedback to the skater.

```
(venv) Jocelyn@admins-MacBook-Pro-7 ai_code % python3 train.py
Data shape: (2, 320), Labels shape: (2,)
Data shape: (2, 320), Labels shape: (2,)
Data shape: (1, 320), Labels shape: (1,)
Training on device: mps
Epoch 1/10, Loss: 1.1518
Epoch 2/10, Loss: 0.7627
Epoch 3/10, Loss: 0.5610
Epoch 4/10, Loss: 0.4208
Epoch 5/10, Loss: 0.3035
Epoch 6/10, Loss: 0.2101
Epoch 7/10, Loss: 0.1394
Epoch 8/10, Loss: 0.0894
Epoch 9/10, Loss: 0.0565
Epoch 10/10, Loss: 0.0351
Training on device: mps
Epoch 1/10, Loss: 0.5495
Epoch 2/10, Loss: 0.3959
Epoch 3/10, Loss: 0.3039
Epoch 4/10, Loss: 0.2394
Epoch 5/10, Loss: 0.1896
Epoch 6/10, Loss: 0.1467
Epoch 7/10, Loss: 0.1117
Epoch 8/10, Loss: 0.0830
Epoch 9/10, Loss: 0.0599
Epoch 10/10, Loss: 0.0423
```

Figure 6. AI-Based Technique Classification Pipeline

```
uture<String> classifyTechnique(List<List<double>> inputData) async {
  final shape = [1, 320];
  final flattenedData = inputData.expand((i) => i).toList();
  final inputOrt = OrtValueTensor.createTensorWithDataList(flattenedData, shape);
  final inputs = {'input': inputOrt};
  final runOptions = OrtRunOptions();
  final outputs = await session.runAsync(runOptions, inputs);
  inputOrt.release();
  runOptions.release();
  String technique = 'unknown';
  outputs?.forEach((element) {
    if (element != null) {
      final outputData = element.dataAsList<double>();
      final maxIndex = outputData.indexWhere((e) => e == outputData.reduce((a, b)
=> a > b ? a : b));

      final confidence = 0.5;
      if (outputData[maxIndex] >= confidence) {
        switch (maxIndex) {
          case 0:
            technique = 'axel';
            break;
          case 1:
            technique = 'loop';
            break;
          case 2:
            technique = 'salchow';
            break;
          default:
            technique = 'unknown';
        }
      }
    }
    element?.release();
  });
  return technique;
 }
}
```

Figure 7. Neural Network Technique Classification Logic

This code is part of the AI component. Its purpose is to take processed sensor data and predict

which skating technique the athlete performed. This code runs after the app has collected and synchronized the motion data from the two sensors. The classifyTechnique method shown is called when the program is ready to analyze the skating attempt and send it through the trained classifier method. The method first takes in inputData, which is a 2D list of data from the sensors. Since the model expects one vector, flattenedData is used to combine all the smaller lists into one single list. Then it creates it into the shape of [1,320], so that the data matches the format required by the model [14]. The program then runs the model using session.runAsync, where the model returns the output scores for different techniques. The code then finds the highest score and checks where it is above the confidence threshold of 0.5. If it is, the program labels the motion as an axel, loop, or sal. However, if the confidence is below 0.5, it returns unknown.

# 4. EXPERIMENT

## 4.1. Experiment 1

One possible blind spot is the model's accuracy on the rink. This is important because the app must correctly identify skating techniques in order to provide personalized and accurate feedback.

To properly test the model accuracy, the sensor devices are first turned on and strapped to the skater's outer shins before then being paired to the mobile app via bluetooth. The skater must then perform the following techniques: axel, salchow, and loop. For each of these, they must perform the technique ten times. For each attempt, there will also be a recording done so that human evaluation can be performed independent of any AI input. For each of the two models, they will then be evaluated based on their intended objective. For the classifier, the accuracy is measured based on whether or not it successfully detected when a given technique was actually attempted and also whether or not there were any false positives. As for the scorer model, the accuracy is based on how closely it aligns to the human evaluation of the recorded techniques.


Figure 8. Comparison Between AI Technique Classification and Human Evaluation Results

## 4.2. Experiment 2

Another blindspot to measure is the effectiveness of the information screen present in the app, particularly toward technique success and injury mitigation.

For both information categories, the control group will be the skaters who have no interaction with the Skate Sensor system. For the techniques screen, the effectiveness is measured based on the success rate of the featured techniques among users compared to the control group after a three week period. As for the injury screen, we will be measuring the rate of the injuries suffered by the users relative to the control within the same three-week window. For the sake of logistics, the data will be collected from the participating athletes through a survey system so that individuals can log their rates of success on their attempted techniques as well whether or not they suffered any injuries. The survey participants from the user group will also be asked which entries of both of the information screens they read if applicable.

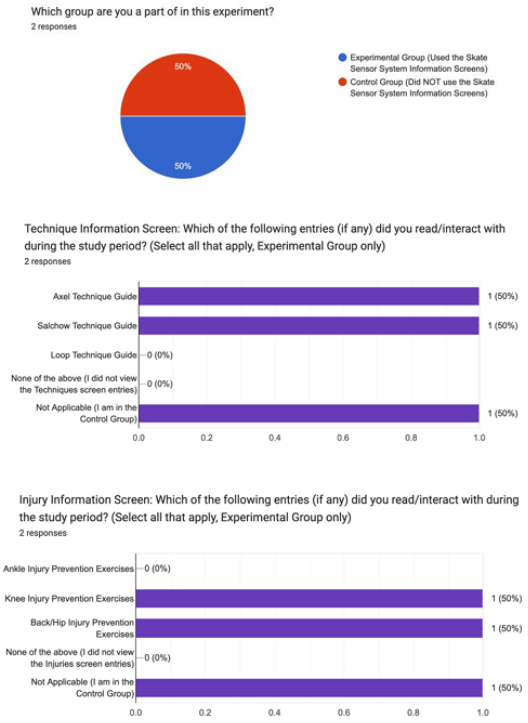


Figure 9. Actual Chart Titles Related to Skate Sensor User Experiments

## 5. RELATED WORK

Shi, Ozaki, and Honda address a similar problem by using five wearable IMUs to study the mechanics of figure skating jumps [11]. One strength of their work is its rigorous mathematical approach, which analyzes a wide range of variables, including take-off angle, to understand jump performance in detail. Their system is similar to mine in that it is wearable and sends data wirelessly to a phone. However, their method has several limitations. It only examines one technique, flip jumps, requires a full-body setup with large sensors, and depends on manual analysis of the data rather than an autocoaching system. In addition, the study uses a relatively small sample size. My project builds on this idea by making the system more practical and more focused on automated feedback for independent practice.

Wang and Sun (2023) developed a skate-mounted wearable system that records motion data and later replays the movement in a Unity app [12]. Their approach is strong because it captures skate specific movement and gives skaters a visual way to review technique. However, the system requires data to be transferred after practice through an SD card, so it does not provide immediate

feedback. The app also emphasizes playback more than direct analysis. In addition, parts of the experimental design rely on simplifying assumptions that reduce the accuracy of the results, especially during more complex motion. The positioning on the foot can introduce additional variability that could interfere with accuracy. Sensor placement can also introduce variability, and because the system tracks only one skate, the range of techniques it can display is limited. My project improves this by providing a more practical, coaching focused system with automated feedback.

Hardegger et al. show that wearable sensors and machine learning can successfully be used to analyze skating-related movement [13]. Their work is strong because it applies IMU-based sensing in a realistic athletic environment and demonstrates the broad potential of this technology for sports training. However, the system seems to focus more on motion recognition than on giving direct coaching feedback, and it has not been developed into a fully deployed autocoaching tool. My project extends this idea by placing greater emphasis on automated feedback that can help skaters improve technique during practice.

## 6. CONCLUSIONS

The safety emphasis of the Skate Sensor app means that the breadth of information that needs to be presented both for technique execution and injury is going to continually grow and evolve. With that being said, the effort to do so would be diminished if we don't also implement incentives for the users to read and retain the information. We can start off by adding quizzes and an earnable scoring system which can in turn be used to reward cosmetics. Such a feature would also necessitate a social environment where users can add one another through the app and compare their skill profiles with each other. This would require the introduction of a cloud-based backend to hold all user information, which is a notable departure from the current local only design. The skill expression portion of such a feature expansion would require the models that classify and score the techniques to be all the more accurate. Therefore, we would also have to expand the amount of metrics that are measured per data point as well as collecting more of such data to continue training the AI models to create better versions [15]. Another general improvement that can be made is by shaving off the unnecessary parts of the current prototype hardware and creating a custom minified PCB which would further reduce the footprint and ease of use for the end user.

Skate Sensor shows how wearable sensing and mobile AI can make figure skating practice more accessible, informative, and supportive for beginners. Although the system still has a lot of room to grow, it establishes a strong foundation for safer training, more consistent feedback, and a more independent learning experience in the sport.

## REFERENCES

[1] Jones, Michael, et al. "Training, children, and parents: Coach perspectives on wearable sensor data in sub- elite figure skating in the United States." International Journal of Human-Computer Studies 183 (2024): 103184.

[2] Miller,HannahH.,DanaK.Voelker,andDanielGould."AMixed-MethodsInvestigationofEliteUSFigure Skaters' SocialMedia Use and Its Implications on Their Psychological Well-Being and Performance as Athletes." Journal of Clinical Sport Psychology 1.aop (2026): 1-21.

[3] King, Deborah L., and Jason D. Vescovi. "Integrating technology in figure skating." The Science of Figure Skating. Routledge, 2018. 122-138.

[4] Wang, Yirina, and Yu Sun. "An Intelligent and Data-Based Skate Analyzer to Assist in Analyzing Movements of Skate on Ice." CS & IT Conference Proceedings. Vol. 13. No. 5. CS & IT Conference Proceedings, 2023.

[5] Hardegger, Michael, et al. "Sensor technology for ice hockey and skating." 2015 IEEE 12th International Conference on Wearable and Implantable Body Sensor Networks (BSN). IEEE, 2015.
[6] Drazdova, Victoria. "Exploring the roles of artificial intelligence and wearable feedback technologies in figureskatingperformanceanalysis:Ascopingreview."JournalofHumanSportandExercise21.2(2026):618-628.
[7] Novikova,Irina,etal."ComputerVisioninAssessmentofComplexityofExercisesofFigureSkating."2024 IEEE Ural-Siberian Conference on Biomedical Engineering, Radioelectronics and Information Technology (USBEREIT). IEEE, 2024.
[8] Hirosawa,Seiji."CanMachineLearningRobustlyPredictGradeofExecutioninFigureSkatingJumpsfrom Kinematic Features Across Competitions—A Case Study of Ladies' Double Axel at the World Championships."(2025).
[9] Souaifi, Marouen, et al. "Artificial intelligence in sports biomechanics: A scoping review on wearable technology, motion analysis, and injury prevention." Bioengineering 12.8 (2025): 887.
[10] Salonen, Rosa-Maria. "Socio-economics of figure skating: Finnish figure skating families' socio-economic standing and perceptions of their child's participation in figure skating." (2020).
[11] Lee, Michelle Nicole. Does Early Performance Predict Later Success in Figure Skaters?. MS thesis. University of Toronto (Canada), 2024.
[12] Furgeson, Duncan O. Determining Figure Skating Jump Under-Rotation in Real-Time Using IMU Sensors During Practice. MS thesis. Brigham Young University, 2022.
[13] Bruening, Dustin A., et al. "A sport-specific wearable jump monitor for figure skating." PloS one 13.11 (2018): e0206162.
[14] Shi, Yuchen, Atsushi Ozaki, and Masaaki Honda. "Kinematic analysis of figure skating jump by using wearable inertial measurement units." Proceedings. Vol. 49. No. 1. MDPI, 2020.
[15] Han,JulieS.,EllenT.Geminiani,andLyleJ.Micheli."Epidemiologyoffigureskatinginjuries:areviewof the literature." Sports health 10.6 (2018): 532-537.