DESIGN AND IMPLEMENTATION OF AXON AI: A SMART HELMET SYSTEM FOR REAL - TIME CONCUSSION DETECTION IN FOOTBALL PLAYERS USING IOT SENSORS AND MOBILE INTEGRATION

Jiahao Chang [1], Soroush Mirzaee [2]

[1]St. Margaret's Episcopal School, 31641 La Novia Ave, San Juan Capistrano, CA92675
[2]California State Polytechnic University, Pomona, CA 91768

ABSTRACT

Axon AI is a helmet system designed to help detect concussions in football players [1]. Every year, players can get long-term brain damage because they don't realize they are hurt or they are not checked out quickly. Our system uses small hardware parts like Particle Boron microcontrollers, gyroscopes, and pressure sensors, along with a Firebase database and a mobile app made with Flutter Flow, to gather and analyze impact data instantly [2]. When there's a dangerous hit, the system immediately alerts coaches and trainers, so they don't have to rely solely on how players feel or what they say they are hurt. We faced some challenges like making sure the helmet fits, fixing network issues, and setting the right thresholds for alerts. We improved the design through testing and adjustments. Tests showed that the sensors were accurate and that the data was sent reliably under good network conditions. Overall, Axon AI looks promising as a tool to detect concussions quickly, helping coaches make faster decisions and keeping players safer from long-term injury

KEYWORDS

Hardware, Artificial Intelligence, Flutter, Football Players

1. INTRODUCTION

Each year, countless football players take heavy contact to the head, and are often unaware of the serious damage that they're brains are experiencing. Axon AI offers a smart solution to the long-standing issue of undiagnosed concussions during a football game [3]. For many decades, many people have been misdiagnosed, whether positively or negatively. Many players who are often driven by adrenaline, even after experiencing a traumatic brain injury, will continue to play due to their need to perform, often making their injuries worse and risking even worse injuries.

This is a widespread issue across all ages and levels of sports. Impacts are frequent in Football, and symptoms are often delayed or just not checked at all, and athletes are put at risk because they are unable to receive immediate attention. A 2022 survey of 411 youth (ages 8–14) athletes in contact sports found that 12.7% of players admitted not reporting a suspected concussion, and 13.1% continued playing after a suspected concussion. This statistic shows how prevalent this issue is in our sports world.

David C. Wyld et al. (Eds): CNSA, CCSEIT, NLAI, SIGML, ICCIOT, AIAP – 2026
pp. 109-117, 2026. CS & IT - CSCP 2026 DOI: 10.5121/csit.2026.160512

These statistics are scary, as playing through brain injuries will lead to long-term consequences such as chronic traumatic encephalopathy, commonly known as CTE, memory loss, depression, and early-onset dementia [4]. Axon AI tackles this issue by giving coaches real-time data and artificial intelligence to detect concussions quickly and efficiently [5]. Ensuring that the player gets the data they need to succeed in the athletic world, healthily. Axon AI has the potential to change how concussions are treated and protect people from long-term dangers.

The first method looked at using scalp-mounted EEG sensors to find changes in brain activity from concussions [6]. While this method worked, it raised safety concerns because impacts could cause discomfort or injury due to direct scalp contact. Our project improved this by utilizing noninvasive sensors built into helmets, which eliminated risks such as punctures or skin irritation. The second method reviewed tools such as SCAT tests, impact sensors, and biomarkers. These had issues such as being subjective, lacking clear thresholds, or requiring lab tests that took time. Our project fixed this by providing quick, real-time results that could be viewed on a mobile device, helping with faster decisions on the sideline. The third method looked at voice biomarkers for detecting concussions. These can be affected by loud noise and need baseline samples. Axon AI doesn't rely on human speech, making it more reliable in noisy game environments and not requiring any prior setup.

Axon AI uses hardware and an AI system to predict when a concussion may have occurred using graphs and measuring acceleration to achieve this. Our solution solves this problem by collecting data and analyzing it to give real-time predictions. Our data collection uses the gyroscope for rotation and tilt of the head/helmet of the player, and an accelerometer for sudden rises in Gforce and pressure. Axon AI's solution collects data using four resistor pressure sensors from the front, sides, and back of the helmet, giving us a full view of the force and acceleration in all directions. Additionally, the AI will collect all the force data and process the probability of concussions and display readable data inside the mobile app. Consequently, this will make their report more accessible and more understandable for the coach. Our app interface presents the data in a simple and accessible way, allowing coaches, trainers, and medical staff to understand whether a player might be at risk quickly. By providing real-time alerts and easy-to-understand reports, Axon AI ensures faster response times and smarter decisions, ultimately protecting athletes from the dangers of undiagnosed brain injuries. An alternative solution would've been a rating of concussion severity by player and by coach. This solution is deemed inapplicable as the coach wouldn't accurately decipher whether a hit was a heavy impact or if it just appeared to be a blow without lasting effects. The player aspect would also be flawed, as someone could have a concussion but be dazed and just won't admit it. Another example of this would be if it were a minor concussion that they didn't feel, but it could still give them lasting problems if it goes untreated. There are many other reasons that the feedback-based AI system wouldn't work, but Axon AI takes the human error out of it and gives straight data to analyze whether something is severe or minor.

Two experiments tested Axon AI's sensing system and data quality. We first looked at how pressure sensors in helmets affect impact detection. We used three setups and dropped a 1-poundweight three times each to see how placement matters. Results showed clear differences, with means from 2.53 to 3.2. Setup C had the highest values, likely because it was placed where the padding was less dense. Setup A had the lowest, as softer padding reduced force. Variability was low within each setup, showing consistent readings. The second experiment assessed the accuracy of the sensors by comparing pressure data with head movement using the BNO085sensor. We used the best placement from the first test and recorded peak acceleration and angular velocity at three impact levels, five trials each. Pressure readings closely matched the head movement data, with a correlation of around 0.986 for acceleration and 0.988 for velocity.

The means increased steadily from about 6.1 g to 18.2 g for acceleration and 105 to 331 degrees/sec for velocity. This shows that setting thresholds can give reliable, real-time alerts

2. CHALLENGES

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

2.1. System Safety and Compatibility

The significant problems that could potentially lead to issues would be the system's safety. Based on testing, important issues such as helmet incompatibility, uncomfortable fit, and hazardous placements are the most important concerns. In terms of helmet incompatibility, we could have a model tweaked based on the format of the football helmet, as many helmets have different structures, padding, facemasks, etc. Regarding uncomfortable fit, we could use testing to figure out which material or where to place the central master system to create a more convenient fit. When it comes to hazardous placement, it could be extremely dangerous to have sharp materials next to the spinal cord or brain, and through different testing, we could use a gel-like substance to encase the compartment, therefore making it soft and making it less hazardous.

2.2. Hardware Component Selection

Major problems came up regarding the parts of the system. Our most significant problem was to figure out which microcontroller to use. There were many issues that we came across, especially in terms of fit, cellular connection, and stock availability. We could use the particle boron, which has both the small size component and the cellular network, to pair up. Another issue was the gyroscope and the pressure pads. We faced major availability issues, and we could use parts that were almost identical from different marketplaces and try to compensate for the stock. We could have preordered parts, but the company we could have bought from was known for late deliveries, therefore making us compensate.

2.3. Data Management and Connectivity

One of the other challenges of the project was deciding on how the data would be handled. Since there are multiple athletes that the coach will need to monitor at the same time, technologies suchas Bluetooth would be out of the door since they have limited connection capabilities. As a result, we decided to streamline the data from all of the helmets to a database that will then be accessed through the phone application. This way, the stream of information will not be cut off, and the coach will have immediate access to the information of all of the users

3. SOLUTION

The three major components of the program consist of the hardware, the database, and the application. The hardware consists of the Particle Boron, a gyroscope, pressure detection pads, and the casing of the Boron. The particle boron is a processing unit, and it uses cellular technology to transmit data. It is the head of systemic operations. The gyroscope detects acceleration and a shift in axes. The pressure detection pads measure the amount of force per impact. For the database part of this unit, we used Firebase Firestore. Firestore is a No SQL database, which is a non-relational database that offers a flexible data model [7]. The application is an incorporation of data-displaying/history, team-player interface, and it also includes artificial intelligence as a part of estimation and safety suggestion*. The application displays data from the most recent and the timeline. The team-player interface allows the coach to add players from their team to their online team, and it allows the coaches to see data for everyone on the team. This allows for real-time threat management of all players and for coaches to see real data and not just players' opinions about how hurt or not hurt they are. How the program works is that it is a system including a hardware and software component. Firstly, the sensors/gyroscope accumulate data. Then the data will be sent to the Particle Boron, and the data from the gyroscope will be read through the library's file in the Boron, then it will be compressed into a file and sent off to Firebase in JSON format [8]. Then, after Firebase receives that data, it will be sent into a digital package to the application you have signed into

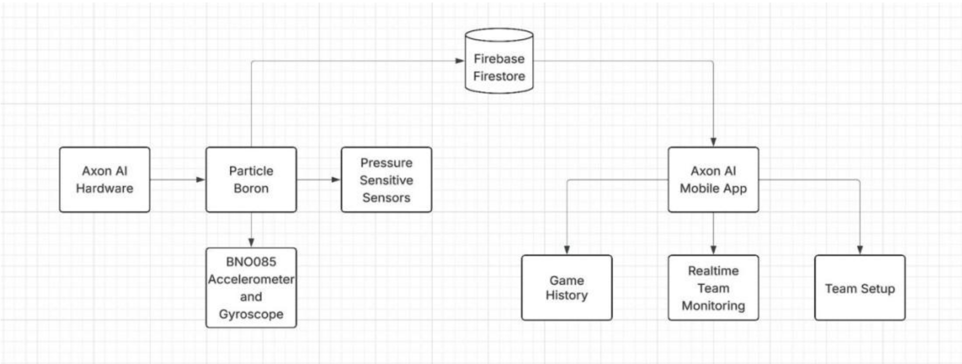


Figure1. Overview of the solution

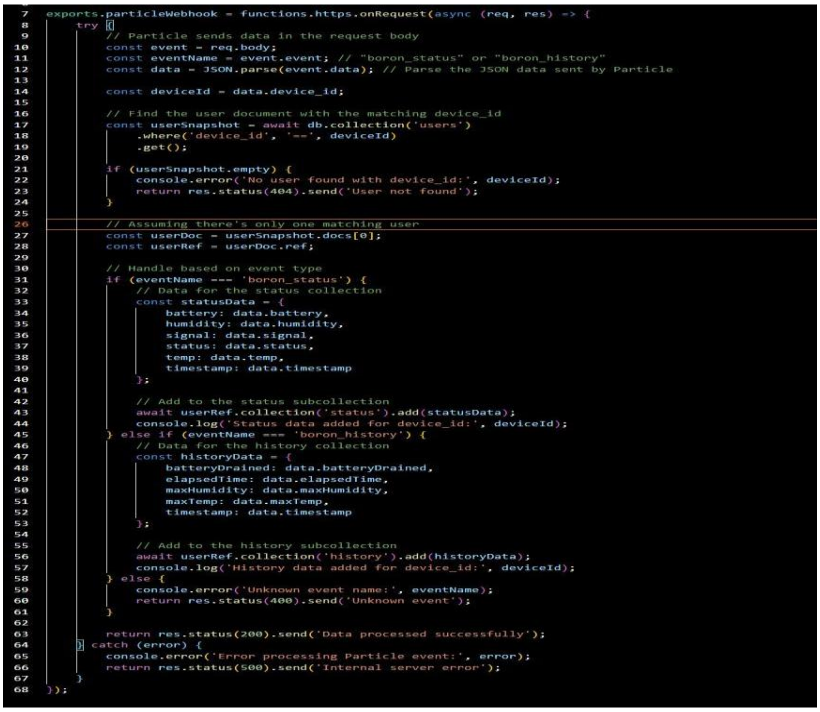
The Boron's purpose is to transmit data and translate data via libraries to Firebase. Then, the component relies on the cell's special purpose and ability to transmit data. The broad scheme of this component is to receive data from the gyroscope and pressure pads and send data to our database.

```
void publishMaxValues(){
    char jsonBuffer[256];
    snprintf(jsonBuffer, sizeof(jsonBuffer),
    "{\"maxAccel\":%.2f, \"rotation\": %.4f \"maxFSR\": %.2f}",
    maxAccelVal,
    maxRotationVal,
    maxFSRVal
    );

    Particle.publish("impact_data", jsonBuffer, PRIVATE);
}

void loop() {
    delay(10ms);

    if (bno08x.wasReset()) {
        Particle.publish("BNO08x", "sensor was reset", PRIVATE);
        setReports();
    }

    if (!bno08x.getSensorEvent(&sensorValue)) {
        return;
    }

    readAccel();  // Call the acceleration checking function
    static unsigned long lastPublish = 0;
    const unsigned long publishInterval = 1000;


    switch (sensorValue.sensorId) {
        case SH2_GAME_ROTATION_VECTOR:
            maxRotationVal = max(maxRotationVal, abs(sensorValue.un.gameRotationVector.real));
            break;

        case SH2_LINEAR_ACCELERATION:
            float accelX = sensorValue.un.linearAcceleration.x * MS2_TO_MILES_S2;
            float accelY = sensorValue.un.linearAcceleration.y * MS2_TO_MILES_S2;
            float accelZ = sensorValue.un.linearAcceleration.z * MS2_TO_MILES_S2;

            char accelBuffer[128];
            snprintf(accelBuffer, sizeof(accelBuffer),
                    "x: %.6f y: %.6f z: %.6f miles/s^2",
                    accelX, accelY, accelZ);
            Particle.publish("BNO08x_Accel", accelBuffer, PRIVATE);
            break;
    }

    if (millis() - lastPublish >= publishInterval){
        publishMaxValues();
        lastPublish = millis();
    }

    delay(100ms);  // Respect Particle rate limit
}
```

Figure 2. Screenshot of code 1

At the top of the code, we start out with a loop command, continuously looping the underlying code with a 10 millisecond delay. The code below first takes the max values that the system has, then it processes them, resets the previous sensor readings, and then sends them over to Firestore database through a webhook. The code runs continuously and is always checking for inertial measurement unit (IMU) data from the sensor [9]. The first part of the loop checks whether the sensor has been reset. If the sensor is reset, it publishes a message to Particle Cloud and performs a reinitialization of the sensor report settings, then it checks for fresh sensor data. When new data is available, the code determines the type of sensor reading —either rotational data or linear acceleration. Based on the type, it processes and stores the values, updating the maximum values recorded for acceleration, rotation, and force sensing. The program formats these max values into a JSON package every second and publishes them to the cloud. This systemenables real-time motion monitoring and remote data access through the Particle platform.

The database selected for this project is Firebase, a cloud-based platform developed by Google. Firebase is the central hub for receiving, storing, and transmitting data throughout the system. One of its most important features is its NoSQL database format, which allows data to be stored in flexible, document-based structures rather than rigid, table-based schemas. This structure makes it especially well-suited for handling dynamic or real-time data, such as the sensor readings generated by the Particle Boron. Because Firebase is scalable and highly responsive, it supports fast read and write operations, which is critical for applications requiring real-timetracking and historical data access. In the broader scheme of the system, Firebase acts as the "middleman " between data collection and data visualization. Once data is gathered from the Particle device, it is processed and stored in Firebase, which can later be retrieved by apps or dashboards for display and analysis. This setup not only simplifies data management but also ensures the entire system operates smoothly, with minimal delays between data capture and availability. Overall, Firebase is a key component that provides efficient, reliable, and scalable data flow across the entire application.



```
exports.particleWebhook = functions.https.onRequest(async (req, res) => {
    try {
        // Particle sends data in the request body
        const event = req.body;
        const eventName = event.event; // "boron_status" or "boron_history"
        const data = JSON.parse(event.data); // Parse the JSON data sent by Particle

        const deviceId = data.device_id;

        // Find the user document with the matching device_id
        const userSnapshot = await db.collection('users')
            .where('device_id', '==', deviceId)
            .get();

        if (userSnapshot.empty) {
            console.error('No user found with device_id:', deviceId);
            return res.status(404).send('User not found');
        }

        // Assuming there's only one matching user
        const userDoc = userSnapshot.docs[0];
        const userRef = userDoc.ref;

        // Handle based on event type
        if (eventName === 'boron_status') {
            // Data for the status collection
            const statusData = {
                battery: data.battery,
                humidity: data.humidity,
                signal: data.signal,
                status: data.status,
                temp: data.temp,
                timestamp: data.timestamp
            };

            // Add to the status subcollection
            await userRef.collection('status').add(statusData);
            console.log('Status data added for device_id:', deviceId);
        } else if (eventName === 'boron_history') {
            // Data for the history collection
            const historyData = {
                batteryDrained: data.batteryDrained,
                elapsedTime: data.elapsedTime,
                maxHumidity: data.maxHumidity,
                maxTemp: data.maxTemp,
                timestamp: data.timestamp
            };

            // Add to the history subcollection
            await userRef.collection('history').add(historyData);
            console.log('History data added for device_id:', deviceId);
        } else {
            console.error('Unknown event name:', eventName);
            return res.status(400).send('Unknown event');
        }

        return res.status(200).send('Data processed successfully');
    } catch (error) {
        console.error('Error processing Particle event:', error);
        return res.status(500).send('Internal server error');
    }
});
```

Figure3. Screenshot of code2

This section of the code represents the endpoint of a custom webhook that serves as a bridge between the Particle Boron device and Google Firestore. When the particle boron finishes collecting and processing sensor data, this webhook is triggered to transmit the results. The webhook listens for incoming data sent from the Particle Cloud in JSON format. Once received, the endpoint parses and processes this data, ensuring it is correctly structured and ready for storage. After processing, the webhook securely transfers the data into a Firestore database, where it can be organized, queried, and used for further analysis or display in applications [10]. In simpler terms, this endpoint acts as the final step in the data flow pipeline—once the Particle Boron completes its task of gathering environmental or movement data, this web hook takes that data and moves it into Firestore for long-term storage and accessibility. This setup allows developers and users to track historical data, perform analytics, and integrate the information into dashboards or mobile apps. The use of Firestore also ensures scalability, reliability, and real-timesyncing across platforms. Overall, this endpoint is crucial for maintaining a smooth, automated flow of information from the physical device to the cloud database.

The application is built using Flutter Flow, a development platform that rapidly creates mobile applications with minimal manual coding. One of the standout features of Flutter Flow is its user friendly interface, making it accessible even to those with limited programming experience. This makes it ideal for quickly building dynamic apps that need to handle complex data interactions in real time. In the context of this application, Flutter Flow plays a key role in enabling real-time data management for sports teams [14]. Coaches can view live data as the game is happening, allowing them to make quick, strategic decisions based on current performance metrics. Meanwhile, players gain access to this data after the game ends, enabling them to review their performance and track progress over time. This dual-access system supports real-time and historical data tracking, giving the player/coach a complete picture of individual and team performance. The flexibility of Flutter Flow also allows for easy updates and feature expansions as the app evolves. By integrating real-time tracking with a clean, responsive interface, the app becomes an essential tool for modern coaching and athletic development. Overall, Flutter Flow provides the foundation for an interactive experience, enabling the user to bridge the gap between live performance and data analysis.

4. EXPERIMENT

4.1. Experiment 1

The placement of resistive pressure sensors is crucial due to the amount of padding that Football helmets have. Placing them in the right place is very important in identifying the pressure that is being put on the player's head.

We placed layered pressure sensors in three distinct configurations inside the football helmet to capture a range of pressure readings. Each setup was tested by wearing the helmet and dropping a 1 lb weight from a fixed height. This controlled setup ensures that variations in sensor readings come primarily from sensor placement rather than inconsistencies in external force. To validate the results, each configuration was tested three times, and all measurements were recorded. This method reduces error by averaging results across multiple trials. The data collected from these trials provided control points for comparing sensor performance across placements.

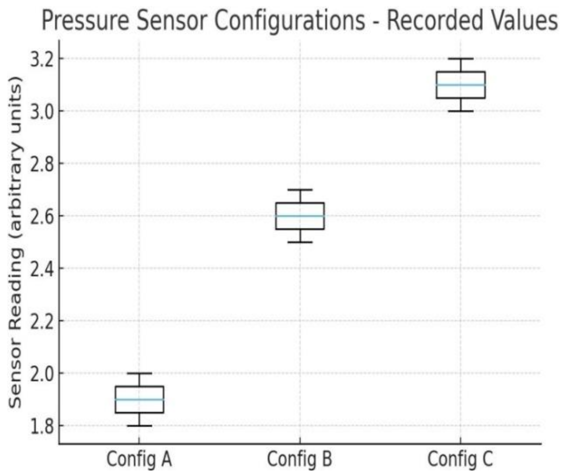


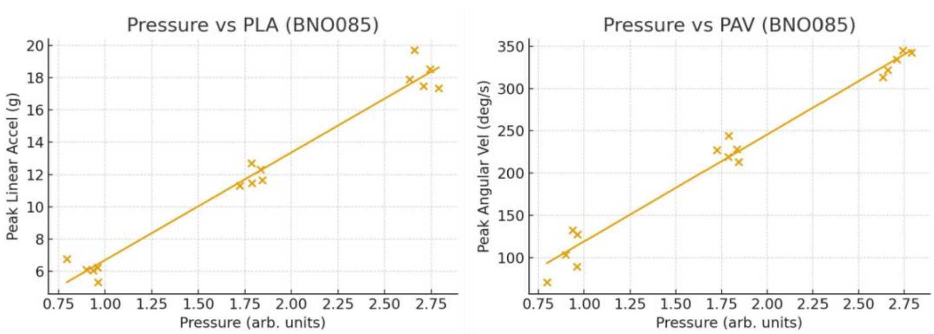
Figure 4. Figure of experiment 1

The experiment produced a mean of 2.53, a median of 2.6, with the lowest value being1.8andthe highest 3.2. These results show that sensor placement significantly affects the readings, with Config A producing the lowest values and Config C yielding the highest. The variation across trials within each configuration was minimal, suggesting the system is consistent once placement is fixed. The most surprising result was the degree of separation between configurations—Config C consistently reported higher values than anticipated, likely due to its positioning closer to high impact zones where padding compresses less. Conversely, Config A's lower readings may reflect placement in areas heavily cushioned by padding, reducing sensor sensitivity to impact. The largest effect on results is clearly the positioning relative to padding density and helmet curvature, confirming the need for precise placement to ensure accurate concussion risk detection in players..

4.2. Experiment 2

Validate whether BNO085 head-kinematics (peak linear acceleration, PLA, and peak angular velocity, PAV) scale with impact severity and correlate with pressure-sensor readings at the head.

Use the most accurate placement for pressure sensors (on padding, touching head). Mount aBNO085 (accelerometer + gyro) at the crown, rigidly coupled to the helmet liner. Apply three controlled impacts by dropping 0.5 lb, 1.0 lb, and 1.5 lb weights from a fixed height; run5trialsper level. For each trial, log PLA (g) and PAV (deg/s) from the BNO085 and the peak pressure from the resistive sensor. This isolates impact magnitude while keeping height, placement, and fit constant. We test linearity, dynamic range, and cross-sensor agreement to confirm that pressure values represent the same physical event captured by the IMU.



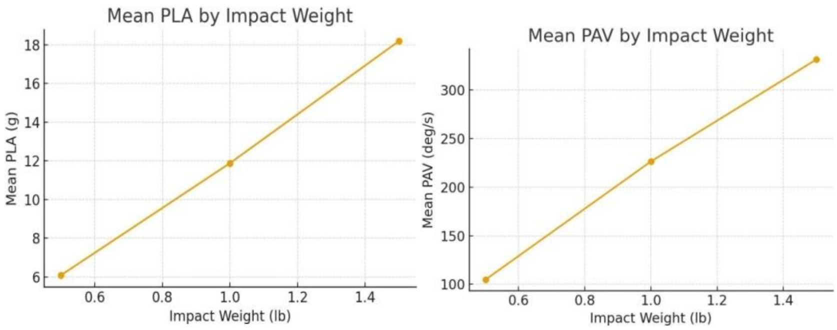


Figure 5. Figure of experiment 2

Across 15 trials, pressure strongly tracked both PLA and PAV. The correlation between pressure and PLA was r ≈ 0.986; between pressure and PAV, r ≈ 0.988. Mean values rose monotonically with impact weight: PLA increased from ~6.1 g (0.5 lb) to ~11.9 g (1.0 lb) and ~18.2 g (1.5lb); PAV increased from ~105 deg/s to ~226 deg/s and ~331 deg/s. This pattern indicates good linearity and dynamic range for the BNO085 within this regime, with modest trial-to-trial variance consistent with drop placement and padding compression noise. The tight relationships suggest the pressure sensor (on-padding) reflects the same impact energy seen by the IMU, supporting its placement choice for head-proximal forces. Small residuals likely come from micro-slip, helmet fit, and slight angle differences on impact. Practically, combining pressure + PLA/PAV provides a more robust trigger for potential concussion events than either modality alone (e.g., require elevated pressure and PLA threshold).

5. Related Work

The University of Columbia tried to solve this problem, and they had something more tangible [11]. They had used physical EEG machine sensors that physically attached them to the scalp, and they would send a signal to the computer if the variables reached a certain threshold. There are major concerns coming from our perspective. The first would be safety concerns, if a football player were to drop their head accidentally or intentionally into another person, and an impact comes after that, the barbs on the EEG could potentially puncture and damage the scalp, leading to potential infections if untreated due to Adrenaline. Another issue would be just the natural position of it. It seems to be extremely uncomfortable, and even if the player had an excellent game without coming into contact with anything, it would most likely still induce itchiness/pain.

The technologies covered in the article include the CNS Neurosurgery journal, which specified the Sports Concussion Assessment Tool (SCAT), head impact sensors, blood-based biomarkers, vestibulo-ocular/eye-tracking, and mobile apps—face limitations that hinder their effectiveness[12]. SCAT relies on subjective symptom reporting and requires trained personnel, making it inconsistent in chaotic sideline settings. Head impact sensors measure force but lack validated thresholds to confirm concussions. Blood-based biomarkers are not yet practical for real-time use due to lab processing delays. Vestibulo-ocular and eye-tracking tools are promising but often expensive and require specialized equipment. The article, coming from the IEE publishing site, had made an app for spotting concussions using voice biomarkers [13]. Its tech has issues: The app's speech tests, like saying tricky words or repeating syllables, get messy if users hesitate or feel off, making results iffy. Pulling 38voicefeatures, like pitch, gets distorted by background noise, which is common on the sidelines. Even good mics can't handle loud crowds, messing up recordings. The speech recognition tech trips over slurred or fast words, leading to mistakes. It needs a baseline voice sample, which isn't always there, and picking up tiny voice changes is tough. These problems make it unreliable for quick concussion checks

6. CONCLUSIONS

While Axon AI shows significant promise, it faces several limitations. A key issue is compatibility across different football helmet brands and models. Inconsistent helmet shapes make it difficult to ensure accurate sensor placement, which may affect data reliability. Additionally, the system relies on cellular connectivity via the Particle Boron, which may not function well in remote or crowded areas, leading to delayed or failed data transmission. Another challenge is the early-stage AI model, which currently uses generalized thresholds for concussion detection [15]. Since concussions vary between individuals, this limits accuracy. If we had more time, future improvements to address these issues would include expanding compatibility testing across helmet types and improving sensor calibration methods. Upgrading to better Wi-Fi and cellular capabilities that would strengthen connectivity and further develop the AI model using personalized player data could also enhance predictive accuracy. These changes would improve Axon AI's reliability and real-world effectiveness in protecting athletes.

REFERENCES

[1] Khalil, Tawfik B., Werner Goldsmith, and J. L. Sackman. "Impact on a model head-helmet system."International Journal of Mechanical Sciences 16.9 (1974): 609-625.
[2] Fadaee, Mohsen. "Building the foundation for an ai-integrated career coaching application withflutterflow." (2024).
[3] Meehan III, William P., et al. "The prevalence of undiagnosed concussions in athletes." Clinical Journal of Sport Medicine 23.5 (2013): 339-342.
[4] McKee, Ann C., et al. "The neuropathology of chronic traumatic encephalopathy." Brain pathology25.3(2015): 350-364.
[5] Tien, James M. "Internet of things, real-time decision making, and artificial intelligence." Annals of DataScience 4.2 (2017): 149-178.
[6] Tyagi, Aruna, Sunil Semwal, and Gautam Shah. "A review of EEG sensors used for data acquisition."Journal of Computer Applications (IJCA) (2012): 13-17.
[7] Han, Jing, et al. "Survey on NoSQL database." 2011 6th international conference on pervasive computingand applications. IEEE, 2011.
[8] Lv, Teng, Ping Yan, and Weimin He. "Survey on JSON data modelling." Journal of Physics: ConferenceSeries. Vol. 1069. No. 1. IOP Publishing, 2018.
[9] Fang, Zhou, et al. "Conversion of upper-limb inertial measurement unit data to joint angles: a systematicreview." Sensors 23.14 (2023): 6535.
[10] Kesavan, Ram, et al. "Firestore: The nosql serverless database for the application developer." 2023 IEEE39th International Conference on Data Engineering (ICDE). IEEE, 2023.
[11] Poellabauer, Christian, et al. "Challenges in concussion detection using vocal acoustic biomarkers." IEEEAccess 3 (2015): 1143-1160.
[12] Yue, John K., et al. "Sideline concussion assessment: the current state of the art." Neurosurgery 87.3 (2020): 466-475.
[13] Bretzin, Abigail C., et al. "Concussion nondisclosure in youth sports." Journal of athletic training57.7(2022): 688-695.
[14] Diallo, Ousmane, Joel JPC Rodrigues, and Mbaye Sene. "Real-time data management on wireless sensor networks: A survey." Journal of Network and Computer Applications 35.3 (2012): 1013-1021.
[15] Chen, Tsung-Hsing, et al. "Artificial intelligence model for a distinction between early-stage gastric cancer invasive depth T1a and T1b." Journal of C