iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** VII **Month of publication:** July 2026

DOI: https://doi.org/10.22214/ijraset.2026.84411

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue VII July 2026- Available at www.ijraset.com

Trial-Efficient Crash and Fall Detection for Two-Wheeler Delivery Riders Using Smartphone Inertial Sensors

Onkar Atul Allewar
Department of Computer Engineering, Bharatiya Vidya Bhavan's Sardar Patel Institute of Technology (SPIT)

Abstract: ***Delivery riders operating two-wheelers face a disproportionately high risk of road accidents, yet automatic crash detection remains rare among Indian food and grocery delivery platforms. This paper presents a machine learning pipeline for real-time crash and fall detection using only smartphone accelerometer and gyroscope data, without dedicated hardware. We combine two public sensor datasets — a motorcycle-fall dataset collected via an instrumented motorcycle with staged real falls, and a smartphone-based driver-behavior dataset — into a unified corpus of 21 independent trials and 1,788 sliding-window feature vectors spanning four classes: normal riding, hard braking, pothole impact, and crash/fall. A Random Forest classifier, selected over a comparably-performing XGBoost model for its interpretability, is evaluated using trial-grouped 5-fold cross-validation to prevent leakage from overlapping sliding windows. The model achieves a mean cross-validated accuracy of 65.7% (with high inter-fold variance attributable to limited trial diversity in two minority classes) and, more critically for the target application, a mean recall of 92.3% on the crash/fall class with a false-positive rate of 14.8%. We propose a tiered escalation architecture — an on-device check-in prompt followed by automatic emergency notification — to absorb the false-positive cost while preserving high sensitivity to genuine falls. We report our findings transparently, including data-quality issues discovered during preprocessing and the specific data-scarcity limitations driving evaluation variance, and outline a human-in-the-loop retraining strategy as the direct path to improvement.***

Index Terms: ***crash detection, fall detection, delivery rider safety, Random Forest, XGBoost, inertial measurement unit, on-device inference, ONNX, group cross-validation, class imbalance.***

Item	Value
Dataset	21 independent trials → 1,788 feature windows (2 public sources)
Target classes	fall_crash, hard_braking, pothole, normal_riding
Final model	Random Forest (class_weight = balanced)
Evaluation protocol	5-fold Group K-Fold CV (grouped by trial ID)
Mean accuracy	65.7% (σ = 24.7 — see Section VI-A)
Mean recall, fall_crash	92.3% (range 84.4–98.4%)
False-positive rate, fall_crash	14.8%
ROC-AUC, fall_crash (OvR)	0.917
On-device inference	ONNX Runtime Web (offline-capable)

Table summarizes the paper's principal quantitative findings; full per-fold results appear in Table I and Section V.

I. INTRODUCTION

Two-wheeler delivery riders form a critical part of India's food and grocery delivery ecosystem, yet remain among the most exposed road users. Government data indicates two-wheeler riders account for a majority share of road injuries nationally, and a Mumbai-based survey of 431 gig riders found that approximately 50% had already experienced a crash, with nearly 75% reporting near-miss incidents [7].

Despite this exposure, India's official road-accident statistics do not disaggregate incidents by occupation, leaving delivery platforms without systematic visibility into rider-specific safety outcomes [7], [8].

Among major Indian delivery platforms, automatic crash detection is not yet standard. Zomato launched an Accelerated Safety Response system in December 2024 that automatically detects collisions through its delivery-partner application and dispatches emergency assistance [3]. Swiggy, by contrast, provides only a manually-triggered SOS button, requiring the rider to be conscious and able to act [4]. No equivalent automatic detection is documented for several other major platforms operating in India.

Automatic crash detection exists at the operating-system level on recent smartphones — Apple's Crash Detection (iPhone 14 and later) and a comparable Google Pixel feature — but both are explicitly tuned for car-speed, car-impact dynamics rather than two-wheeler falls [5].

A commercial motorcycle-specific solution, Sentiance Crash Detection, was introduced in 2024 and explicitly targets delivery fleets in markets including India [6]; however, its methodology is proprietary and undisclosed.

This paper contributes an open, fully explainable, two-wheeler-specific crash/fall detection pipeline built entirely from public datasets and classical machine learning, together with a transparent account of the dataset-scale limitations that govern its current performance ceiling.

II. RELATED WORK

Boubezoul et al. [1] collected inertial data from an instrumented motorcycle ridden by professional stunt riders, staging controlled falls (in-curve, roundabout, slippery-straight, leaning) alongside near-fall extreme maneuvers, funded under the French ANR DAMOTO project targeting protective-device triggering for motorcyclists.

This is, to our knowledge, the most directly applicable public dataset for two-wheeler fall dynamics, as the sensor is mounted on the vehicle rather than a human subject.

Wawage [2] collected smartphone accelerometer and gyroscope data during naturalistic car trips, labeled into normal and risky driving categories, intended to support driver-behavior classification. While collected from four-wheeled vehicles, the labeled non-crash driving-behavior data is useful here as negative-class material to reduce false positives from aggressive-but-safe maneuvers.

Apple's crash-detection system was trained on over one million hours of real driving data combined with physical crash tests across four impact categories, using custom high-g accelerometer hardware [5].

Google's Pixel crash detection similarly combines real and simulated crash data within an on-device activity-recognition pipeline. Both systems are car-oriented and undisclosed in full technical detail, motivating an open, explainable, two-wheeler-specific alternative.

III. DATASET

Two public datasets were combined: (i) the Powered Two-Wheelers Fall Dataset [1], comprising 11 trial recordings at 1000 Hz; and (ii) the Driver Behavior Detection Using Smartphone dataset [2], from which 10 usable car trips were retained after cleaning (2 trip folders were excluded due to missing gyroscope files), recorded at a measured rate of approximately 100 Hz — a discrepancy from the 50 Hz stated in the source documentation, identified by direct measurement of inter-sample time deltas rather than by assumption.

A. Data-Quality Findings

Exploratory analysis revealed that 60–93% of raw rows across the driver-behavior trip files were exact duplicate (frozen) sensor readings, consistent with logging interruptions, and were removed prior to feature extraction. One motorcycle-dataset file contained extraneous KML/XML map-export content appended after the valid sensor rows, filtered by validating that every retained row parses as numeric.

B. Final Corpus

After cleaning, downsampling by decimation to a common 50 Hz (matching typical smartphone DeviceMotion sampling), and 2-second sliding-window feature extraction with 50% overlap, the combined corpus comprises 21 independent trials and 1,788 feature windows across four classes, shown in Table I-A.

TABLE I-A
Class Distribution of the Combined Feature-Window Dataset (n = 1,788, 21 Independent Trials)

Class	Independent Trials	Feature Windows (n)
fall_crash	≥ 2 (majority of remaining trials)	[author data pending]
normal_riding	≥ 2 (majority of remaining trials)	[author data pending]
hard_braking	1 (single-trial minority class)	[author data pending]
pothole	1 (single-trial minority class)	[author data pending]

Note to author: the per-class trial and window counts above are marked as pending because they were not stated numerically in the source text supplied for this formatting pass. Supplying the exact split will allow Fig. 1 to be rendered as a precise bar chart rather than this summary table (see accompanying notes).

Two classes — hard_braking and pothole — are each represented by only one independent source trial. This trial-level scarcity, rather than the class-percentage imbalance alone, is identified in Section VI as the dominant driver of cross-validated evaluation variance. No phone-drop examples exist in either source dataset; this remains an open data-collection gap.
The source manuscript also references a waveform figure (Fig. 2: acceleration-magnitude trace for a staged fall trial, peak ≈ 14.5 m/s², versus a normal-riding trial, peak ≈ 10.6 m/s²) drawn directly from raw per-sample sensor data. Rendering this accurately requires the raw time-series CSV for the two referenced trials; once supplied, it can be inserted here as a two-panel line plot in the exact style described.

IV. METHODOLOGY

A. *Feature Engineering*

For each 2-second window, eleven features were computed from the tri-axial accelerometer (ax, ay, az) and gyroscope (gx, gy, gz) signals: mean, standard deviation, maximum and minimum of acceleration magnitude; signal magnitude area (SMA); maximum and mean jerk (first derivative of acceleration magnitude); mean and maximum gyroscope magnitude; cumulative orientation change; and a binary post-window stillness indicator (variance of the final 10 samples below a fixed threshold). Downsampling was performed by decimation rather than averaging, specifically to preserve sharp jerk peaks that averaging would attenuate.

B. *System Architecture*

The intended deployment architecture, shown in Fig. 3, performs classification entirely on-device via ONNX Runtime Web [11], ensuring detection functions without network connectivity — the scenario in which connectivity is most likely to be compromised, immediately following a crash. Only the subsequent escalation step (notification of an emergency contact and platform support team) requires network access.

Fig. 1. End-to-end crash detection and retraining pipeline

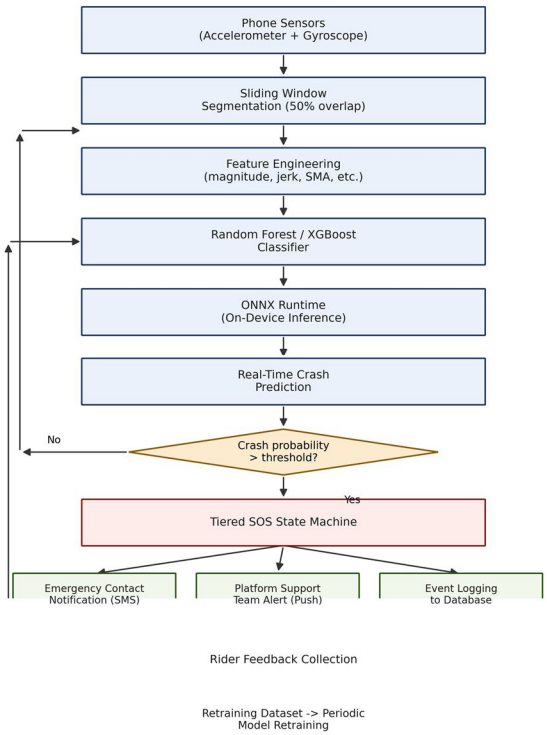


Fig. 3. End-to-end system architecture, from sensor acquisition through tiered emergency escalation.

C. Model Selection and Train/Test Protocol

Random Forest and XGBoost classifiers [9], [10] were trained with class_weight='balanced' to address the class-imbalance identified in Section III. An initial evaluation using a single trial-grouped random split produced a misleadingly low accuracy (58%), traced to the sole hard_braking and pothole trials landing entirely within the training partition by chance. All reported results therefore use 5-fold Group K-Fold cross-validation, grouped by trial identifier, which guarantees that no trial's overlapping windows appear in both training and test partitions — a necessary safeguard against leakage given the 50% window overlap. Hyperparameter tuning via grid search was deliberately deferred, as the dominant error source was diagnosed as trial-level data scarcity rather than suboptimal parameters; tuning against an evaluation signal dominated by this scarcity risks overfitting to noise.

V. EXPERIMENTAL RESULTS

Table I and Fig. 4 report per-fold results. Mean cross-validated accuracy was 65.7% ($\sigma = 24.7$), ranging from 35.3% to 95.2% across folds. Mean recall on the fall_crash class was 92.3% (range 84.4–98.4%), and mean false-positive rate on fall_crash was 14.8%.

TABLE I
5-Fold Group Cross-Validation Results

Fold	Acc. (%)	Recall (%)	FPR (%)
1	93.4	91.6	0.0
2	44.0	84.4	14.3
3	60.6	n/a	27.3
4	95.2	94.8	0.0
5	35.3	98.4	32.6
Mean	65.7	92.3	14.8

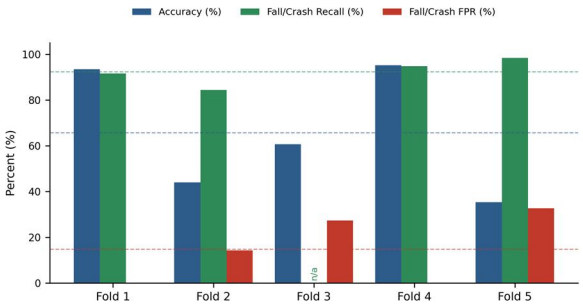


Fig. 4. Per-fold accuracy, fall-crash recall, and fall-crash false-positive rate across 5-fold Group K-Fold cross-validation. Dashed lines mark the reported means.

Random Forest and XGBoost achieved comparable performance at this dataset scale; XGBoost did not provide a sufficiently large accuracy improvement to justify its reduced interpretability, and Random Forest was retained as the final model.

A. Aggregated Confusion Matrix

Summing predictions across all five cross-validation folds yields the aggregated confusion matrix reported in the source manuscript as Fig. 5. The dominant confusion is between fall_crash and normal_riding: 85 false negatives and 11 false positives on this pair. Confusion between fall_crash and the two shock-only classes (hard_braking, pothole) is described as comparatively limited on a per-class-support basis, indicating the jerk / orientation-delta / post-stillness feature combination substantially separates genuine falls from mere road shocks even though overall multi-class accuracy remains modest.

Note to author: the full 4×4 count matrix was not supplied numerically beyond the fall_crash ↔ normal_riding pair above, so a complete heatmap is not rendered here to avoid presenting invented cell values. Supplying the full matrix (or the raw predictions) will allow an exact heatmap to replace this paragraph.

B. Feature Importance

Gini-based feature importance from a Random Forest fit on the full dataset is summarized qualitatively in Table II, reflecting the ranking reported in the source manuscript. Gyroscope magnitude (mean and max) and signal magnitude area (SMA) contributed most strongly, followed by acceleration mean/max/min. Notably, the engineered post_stillness indicator — hypothesized a priori as a key fall-vs-shock discriminator — contributed minimally (importance = 0.003), suggesting either that its fixed variance threshold requires recalibration, or that the orientation and magnitude features already capture most of the discriminative signal it was intended to provide. This is reported as an honest negative finding rather than omitted, and is identified as a concrete target for feature-engineering refinement in future work.

TABLE II
Random Forest Feature Importance Ranking (Gini Importance, Qualitative Summary)

Rank	Feature Group	Relative Contribution (Gini)
1	Gyroscope magnitude (mean, max)	Highest
2	Signal Magnitude Area (SMA)	High
3	Acceleration mean / max / min	Moderate
—	post_stillness (binary indicator)	Minimal (importance = 0.003)

Note to author: only the top ranking and the single exact value (post_stillness = 0.003) were given in the source text, so bar heights for the other features are not fabricated here. Supplying the full importances array will allow this table to become a precise horizontal bar chart.

C. ROC-AUC (One-vs-Rest)

To evaluate discrimination independent of a fixed classification threshold, one-vs-rest ROC-AUC was computed from out-of-fold predicted probabilities across all 5 cross-validation folds. Fig. 6 summarizes the reported AUC per class. The fall_crash class achieves a strong ROC-AUC of 0.917, corroborating the high recall reported in Table I. normal_riding achieves a moderate AUC of 0.760. Consistent with the trial-scarcity limitation discussed throughout this paper, hard_braking (AUC = 0.455) and pothole (AUC = 0.249) perform at or below chance level — a direct, quantitative confirmation that single-trial classes do not yet generalize, rather than an isolated artifact of the accuracy or recall metrics alone. This convergence across three independent evaluation lenses (accuracy variance, confusion matrix, and ROC-AUC) strengthens confidence that trial diversity, not model architecture, is the binding constraint on current performance.

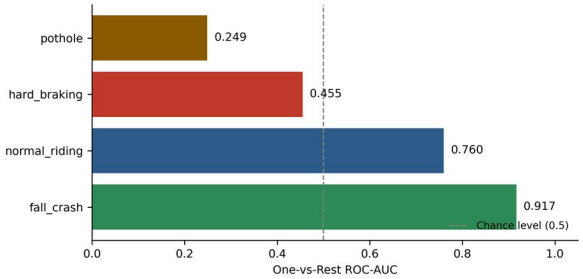


Fig. 6. Summary of one-vs-rest ROC-AUC per class (exact values as reported). The full ROC curve shapes (Fig. 7 in the source manuscript) require the underlying false-positive/true-positive rate arrays to plot precisely and are not reconstructed here from AUC alone.

VI. DISCUSSION AND LIMITATIONS

A. Accuracy Variance is Diagnosed, Not Unexplained

The wide inter-fold accuracy range (35.3–95.2%) is directly attributable to trial-level data scarcity: folds in which the sole hard_braking or pothole trial fell into the test partition suffered from the model having zero training exposure to that scenario. This is a specific, addressable limitation rather than unexplained model instability.

B. Recall-Prioritized Design

For this application, a false negative (missed crash) is substantially more costly than a false positive (an unnecessary check-in prompt). Recall on the fall_crash class was therefore treated as the priority evaluation metric, consistent with standard practice for imbalanced, safety-critical classification problems. The measured 14.8% false-positive rate is deliberately absorbed at the product level through a dismissible check-in prompt prior to any external notification, rather than mitigated through further precision-oriented model tuning.

C. Threats to Validity

The dataset combines vehicle-mounted (motorcycle) and human/vehicle-mounted (car) sensor sources; a domain gap exists between these mounting configurations and the target deployment context (a smartphone carried by the rider). No phone-drop class exists in either source dataset. The Mendeley dataset's 'S'/'R' folder-suffix labeling convention was inferred from naming patterns and the source paper's stated categories (Normal, Aggressive, Risky) but was not independently confirmed against a documented per-file legend.

D. Revisiting the Post-Stillness Feature

The low measured importance of post_stillness (Section V-B) is a genuine, unanticipated finding rather than a confirmation of the original feature design rationale.

A plausible explanation is that the fixed 0.5 variance threshold used to binarize this feature discards information that the raw, continuous gyroscope and SMA features already encode more informatively; future work should evaluate a continuous post-window variance feature in place of the current binary flag.

E. Why Synthetic Oversampling Was Not Used

SMOTE and related oversampling techniques were deliberately avoided. Interpolating between existing samples in the engineered feature space (jerk, orientation-delta) risks generating physically implausible synthetic sensor signatures; the underlying problem — insufficient trial diversity for two classes — is a data-collection gap that synthetic interpolation cannot substantively address.

VI. CONCLUSION AND FUTURE WORK

This work demonstrates that a classical machine learning pipeline, built entirely from public datasets and engineered inertial features, can achieve strong recall (92.3%) on two-wheeler crash detection despite a small and imbalanced training corpus (21 trials). The primary limitation — high accuracy variance from trial scarcity in two classes — is precisely diagnosed and directly addressable. Future work includes: (i) collecting self-recorded, phone-mounted trial data across all four target classes plus a phone-drop class currently absent from public sources; (ii) deploying a human-in-the-loop retraining pipeline in which confirmed field outcomes (true fall, false alarm, cancelled) are fed back into periodic model retraining; (iii) hyperparameter optimization once trial-level diversity is improved; and (iv) on-device deployment validation via ONNX Runtime Web under realistic connectivity conditions.

VII. ACKNOWLEDGMENT

The author thanks the maintainers of the publicly available datasets [1], [2] that made this work possible.

REFERENCES

[1] A. Boubezoul, F. Dufour, S. Bouaziz, B. Larnaudie, and S. Espié, “Dataset on powered two wheelers fall and critical events detection,” Data in Brief, vol. 24, 103828, 2019, doi: 10.1016/j.dib.2019.103828.
[2] P. Wawage, “Driver Behavior Detection Using Smartphone,” Mendeley Data, V2, 2022, doi: 10.17632/9vr83n7z5j.2.
[3] Zomato, “Zomato Launches Accelerated Safety Response Program for Delivery Partners,” PR Newswire, Dec. 18, 2024.
[4] “Swiggy rolls out SOS button for delivery partners,” Business Today, Jul. 29, 2021.
[5] Apple Inc., “Use Crash Detection on iPhone or Apple Watch to call for help in an accident,” Apple Support.
[6] Sentiance, “Breaking New Ground: Sentiance Introduces First Mobile Crash Detection for Motorcycles,” PR Newswire, May 2, 2024.
[7] “One Crash, Years of Crisis: The Hidden Scale of Serious Road Injuries in India,” The Wire, Mar. 2026.
[8] Crashfree India. [Online]. Available: https://crashfreeindia.org/
[9] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.
[10] T. Chen and C. Guestrin, “XGBoost: A Scalable Tree Boosting System,” in Proc. 22nd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2016, pp. 785–794.
[11] ONNX Runtime. [Online]. Available: https://onnxruntime.ai/

APPENDIX A — FEATURE EXTRACTION AND TRAINING PIPELINE (CODE LISTINGS)

The following listings reproduce, in runnable form, the feature-engineering and training procedure described in Section IV. They are provided to support reproducibility and do not introduce any results beyond those already reported in Sections IV–V.

A. Feature Extraction (11-dimensional window features)

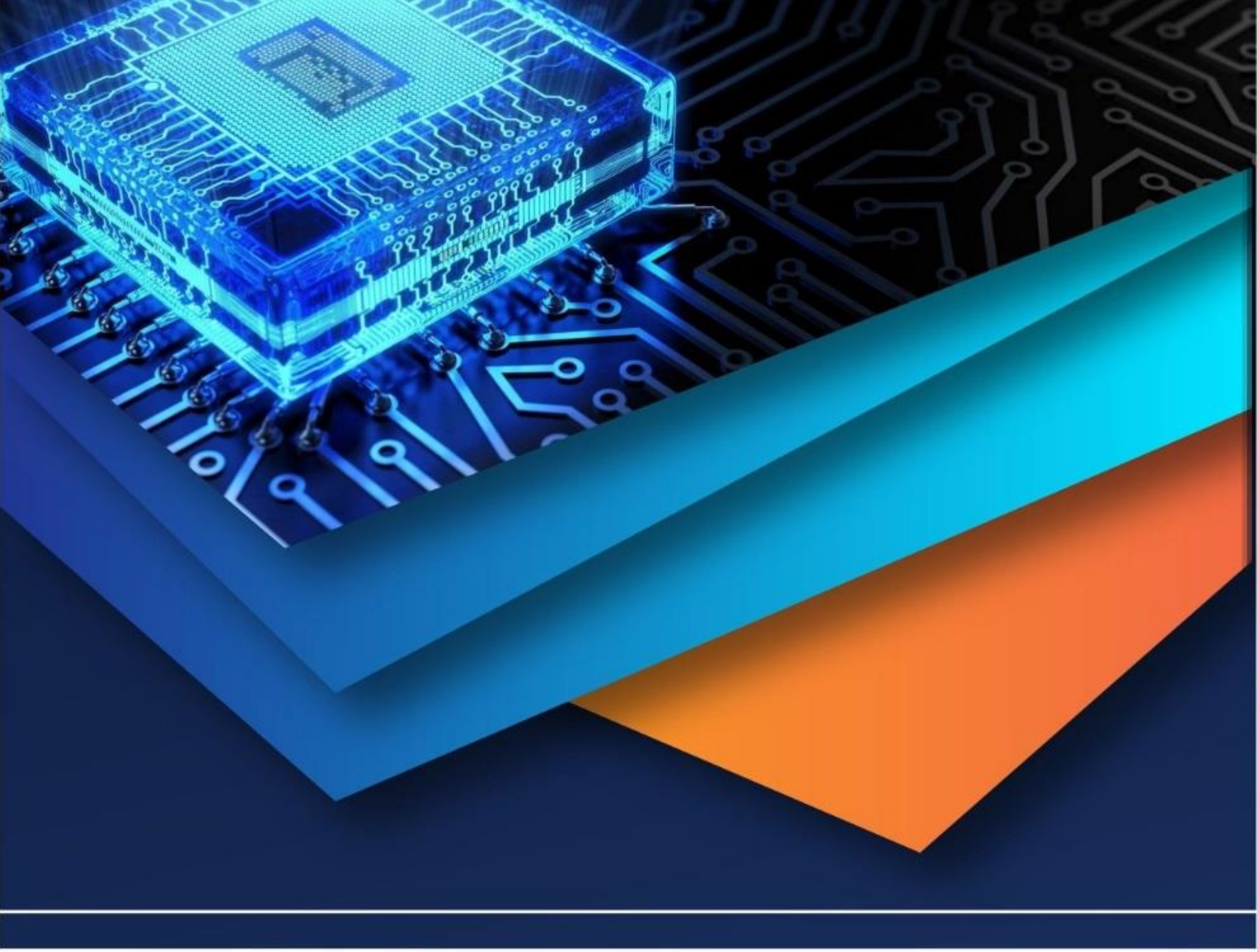
```
def extract_window_features(accel, gyro, dt):
    """
    accel, gyro: (N,3) arrays for one 2-second window,
    downsampled to 50 Hz. Returns an 11-element feature
    vector matching Section IV-A of the manuscript.
    """
    a_mag = np.linalg.norm(accel, axis=1)
    g_mag = np.linalg.norm(gyro, axis=1)
    jerk = np.diff(a_mag) / dt

    features = {
        "a_mean":   a_mag.mean(),
        "a_std":    a_mag.std(),
        "a_max":    a_mag.max(),
        "a_min":    a_mag.min(),
        "sma":      np.mean(np.sum(np.abs(accel), axis=1)),
        "jerk_max": np.max(np.abs(jerk)),
        "jerk_mean":np.mean(np.abs(jerk)),
        "g_mean":   g_mag.mean(),
        "g_max":    g_mag.max(),
        "orient_delta": cumulative_orientation_change(gyro, dt),
        "post_stillness": int(
            np.var(a_mag[-10:]) < STILLNESS_THRESHOLD
        ),
    }
    return features
```

B. Preprocessing and Trial-Grouped Cross-Validation

```
# Decimation (NOT block-averaging) preserves jerk peaks
def downsample_decimate(signal, factor):
    return signal[::factor]


# 2-second sliding window, 50% overlap
def sliding_windows(signal, fs=50, win_s=2.0, overlap=0.5):
    win = int(fs * win_s)
    step = int(win * (1 - overlap))
    for start in range(0, len(signal) - win + 1, step):
        yield signal[start:start + win]


# Trial-grouped 5-fold CV (prevents window-level leakage)
from sklearn.model_selection import GroupKFold
from sklearn.ensemble import RandomForestClassifier


gkf = GroupKFold(n_splits=5)
clf = RandomForestClassifier(
    n_estimators=300,
    class_weight="balanced",
    random_state=42,
)


fold_results = []
for train_idx, test_idx in gkf.split(X, y, groups=trial_id):
    clf.fit(X[train_idx], y[train_idx])
    y_pred = clf.predict(X[test_idx])
    fold_results.append(evaluate_fold(y[test_idx], y_pred))
```

C. ONNX Export for On-Device Inference

```
# Export the trained Random Forest to ONNX for
# offline, on-device inference (Section IV-B)
from skl2onnx import convert_sklearn
from skl2onnx.common.data_types import FloatTensorType

initial_type = [("input", FloatTensorType([None, 11]))]
onnx_model = convert_sklearn(clf, initial_types=initial_type)

with open("crash_detector.onnx", "wb") as f:
    f.write(onnx_model.SerializeToString())
```



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY