A Two-Layer Risky Driver Recognition Model With Context Awareness

KE WANG (Member, IEEE), AND JIAN JOHN LU
Key Laboratory of Road and Traffic Engineering of the State Ministry of Education, College of Transportation Engineering, Tongji University, Shanghai 201804, China
Corresponding author: Jian John Lu (jianjohnlu@tongji.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 71871165, and in part by Jiangsu Technology Innovation and C-V2X Academy Open Topics under Grant KC-V2X2021002.

ABSTRACT For autonomous vehicles and intelligent connected vehicles, the real-time recognition of risky drivers can play an important role in traffic accident prevention. However, the external environment substantially impacts driving behavior and driving risk and is usually costly to acquire. Existing risky driver recognition models often ignore external environment information or assume this information is given. We propose two hierarchical two-layer context-aware machine learning structures. The first layer can speculate external context, for example, traffic states. The second layer recognizes risky drivers based on the contextual information speculated from the first layer. The German Highway Drone Dataset is used to establish risky driver recognition and traffic state recognition models. Rear-end collision risk and side collision risk are evaluated for each vehicle. Drivers with high collision risk are labeled as risky drivers. By analyzing vehicle trajectory data from three traffic states: free-flow, saturated, and congested, we find that traffic states have a significant influence on vehicle’s longitudinal speed, lateral speed, longitudinal acceleration/deceleration, and collision risk. Six classifiers, including SVM, KNN, RF, Adaboost, Extra trees, and XGBoost, are applied to train recognition models. Results show that the proposed structures can significantly improve model’s ability to recognize risky drivers.

INDEX TERMS Driving style, risky driving, context awareness, machine learning, vehicle trajectory.

I. INTRODUCTION New Information and Communication Technologies (ICT) are redefining the automobile industry. Autonomous Vehicles (AV) and Intelligent Connected Vehicles (ICV) provide massive driving behavior and driving environment data and demand a deep understanding of human driving behavior. Dangerous driving behavior is the leading cause of traffic accident in historical data [1]. Identifying an individual’s driving style and dangerous driving behavior can enhance AV and ICV’s ability to perceive risk from the interaction with surrounding vehicles, adjust driving strategy to driving conditions, and improve road traffic safety.

There is no universally accepted definition of driving style. Sagberg et al. [2] reviewed the definitions of driving style in the literature and summarized that driving style represents a habitual and relatively permanent way of driving that differs across individuals or between groups of individuals. A driver could consciously or subconsciously choose a driving style. By contrast, Martinez et al. [3] stated that driving style is strongly influenced by external driving conditions, like traffic, road type, time in the day, weather, then the same driver could switch driving style under different driving conditions. The driving style studies in most papers could be either relatively permanent or changing with external driving conditions, depending on the measurement methods of the driving style chosen by researchers. For example, aggressive driving is one of the heavily studied driving styles, and it can be measured by self-report questionnaires and observations of driving behavior. Self-report questionnaires [4] reflect drivers’ self-assessment of their driving style, and self-assessment is generally stable. Aggressive driving indicators extracted from driving behavior observations include speed profile, hard braking, lateral control, and tailgating [5]. The measured driving style could be influenced by the driving condition under which the driving behavior data was observed. For many studies that rely on driving behavior data collected from naturalistic driving experiments, driving
simulators, and vehicle trajectory data, the external driving condition cannot be ignored to identify driving styles accurately.

Many external driving conditions were found to influence driving style. Ericsson [6] found that traffic, road type, and driver’s demographic factors influence driving patterns, among which road type has the most significant impact. Driving patterns may differ between city road and freeway. Freeways are designed to have continuous traffic flow, and city roads have turning and traffic control infrastructure [7]. Dixit et al. [8] found that drivers tend to behave more conservatively when the pavement was wet, and drivers behave more aggressively during the morning peak period compared to midday and evening peak periods. Therefore, it is necessary to be aware of the external driving conditions as the contextual information for driving style recognition and adapting AV and ICV’s driving strategy to suit the specific driving condition.

We classify existing driving style recognition models into three categories, based on the role of contextual information in the model:

- **Context-Irrelevant**: Although contextual information is essential to driving style recognition, most conventional driving style recognition models do not consider contextual information in the system by assuming all observations share the same contextual information [9], [10].
- **Context-Dependent**: Some models consider certain contextual information that sensors can acquire. For example, Yi et al. [11] proposed a personalized driving state recognition model that each driver has two individual driving state recognition models, one for motorway and another for secondary road. Before predicting the driving state for a new sample, the driver’s identity and road type should be given. However, the collection of all contextual information could be difficult and costly in practice. Some contextual information can only be estimated with uncertainty. This drawback limits the application of context-dependent models.
- **Context-Aware**: A context-aware system can extract and interpret context information that is important to characterize the situation of an entity, including people, place, device, and things [12]. Bejani and Ghathe [13] developed a context-aware system that can recognize the context information, including traffic levels, car types, and driving events, with which the driving style is evaluated using a rule-based fuzzy inference model. Suzdaleva and Nagy [14] proposed a two-layer pointer model that estimates driving conditions, namely urban, rural, and highway, in the inner pointer and estimates driving style in the outer pointer. To the authors’ knowledge, there are few studies that establish a context-aware driving style recognition model using machine learning. Applying machine learning recognition model for uncertain contextual information could improve the system’s performance on driving style recognition.

Except for context awareness, another challenge of driving style recognition is data labeling. Normal samples and dangerous/aggressive/risky samples can be classified based on illegal driving behavior [15] or accident [16] in naturalistic driving data. Subjective data labeling methods include experts scoring [17] and driver self-assessment questionnaires [18]. These methods are not suitable for large sample size vehicle trajectory data. Xue et al. [19] used collision surrogate measurements, such as Time to Collision (TTC) and Margin to Collision (MTC), to label drivers in dangerous car-following state and normal car-following state. Clustering algorithms [20] or semi-supervised learning [21], [22] are designed for unlabeled or partially labeled data, but the result is often hard to verify. Moreover, vehicle collision risk assessment, which is related to driving style, has been studied using machine learning algorithms [23], [24].

The contributions of this paper are as follows. (1) We propose a risky driver labeling method using vehicle trajectory data based on rear-end collision risk and side collision risk. (2) We analyze vehicle trajectory data in three traffic states: free-flow, saturated, and congested, and confirm their impact on driving behavior and driving risk. (3) We propose two hierarchical two-layer context-aware structures that can recognize both driving style and contextual traffic states. (4) We test the proposed structures’ performance to prove their superiority over conventional context-irrelevant machine learning.

This paper’s remainder is organized as follows: Section II describes the vehicle trajectory data used for modeling. Section III introduces the framework of the risky driver recognition modeling and the methodology of each part in the framework. Section IV presents the results. Section V discusses our findings. Section VI concludes.

## II. DATA

The vehicle trajectory data used in this paper is from the Highway Drone Dataset (highD) [25]. HighD contains vehicle trajectory data recorded at six German highways using Unmanned Aerial Vehicles (UAV). The trajectory of 110,500 vehicles was extracted from UAV videos by computer vision algorithms. The UAV camera can cover a 420-meter length of highway with a typical vehicle positioning error of less than 10cm. The vehicle’s information, including position, speed, acceleration, lane-changing, and car-following was detected and tracked every 0.04 s and smoothed using Bayesian smoothing.

The data used in this paper contains three files, which were recorded on a 6-lane highway at 8:55 AM, 10:12 AM, and 17:21 PM in a working day, respectively. The trajectory of 7950 vehicles was recorded over 55 minutes 46 seconds. The details of the three files are shown in Table 1.

The traffic state of each file is determined by its volume, mean speed, and density. The design capacity of the 6-lane highway is 13200 passenger car units (PCU) per hour. The volume of the first file is close to the design capacity; therefore, the flow state is saturated. The second file has a traffic...
volume lower than the design capacity and density higher than the density in the saturated flow; therefore, the flow state of the second file is congested. The third file has traffic volume lower than the design capacity and density lower than the density in the saturated flow; therefore, the flow state of the third file is free-flow.

### III. METHOD

We summarize the methodology framework as a flowchart in Figure 1. First, vehicle trajectory data is used to calculate each vehicle’s rear-end collision risk and side collision risk, based on which each driver is labeled as risky driver or normal driver. The collision risk evaluation method and driver labeling are introduced in Sections III-A and III-B.

Once the driving style is labeled, we try to establish a risky driver recognition model. Section III-C describes how to extract and select features from the vehicle trajectory data. Since the number of risky drivers is often much smaller than the number of normal drivers, we apply SMOTE (Synthetic Minority Oversampling TTechnique) oversampling to relieve the imbalance issue. The proposed two-layer risky driver recognition structures are introduced in Sections III-D. Single-layer Risky-driver-recognition without context-awareness (SR) is the conventional machine learning structure that does not consider contextual information, which is traffic state in this paper. Two-layer Risky-driver-recognition structure with Context-awareness (TRC) is hierarchical and contains traffic state recognition model and risky driver recognition models. Probability-based Two-layer Risky-driver-recognition structure with Context-awareness (PTRC) is an extension of TRC that prediction result is based on probabilities.

Section III-E covers machine learning classifiers used in recognition models and hyperparameter tuning. Section III-F introduces probability calibration. Section III-G explains how we evaluate model performance and conduct the statistical test.

#### A. COLLISION RISK EVALUATION

Given the target vehicle and surrounding vehicle’s trajectory data, we need to calculate the collision risk of the target vehicle through the whole driving process. We consider rear-end collision risk and side collision risk in the measurement of high-risk drivers in this paper. We categorize the target vehicle’s moving state into car-following, lane departure, and lane-changing and calculate collision risk based on the following rules:

1) **Car-Following:** When the gap between the target vehicle and its leading vehicle is smaller than 50 m, calculate the rear-end collision risk (shown in Figure 2(a)).

2) **Lane Departure:** If the lateral distance between the target vehicle and the left (or right) lane line is smaller than 0.5m and the target vehicle is moving toward the left (or right) lane line, calculate the side collision risk between the target vehicle and the adjacent vehicle in the left (or right) lane (shown in Figure 2(b)). If the target vehicle is at car-following state and lane departure state simultaneously, then both rear-end collision risk and side collision risk should be calculated.

3) **Lane-Changing:** When the target vehicle’s center crosses a lane line (shown in Figure 2(c)), keep calculating the rear-end collision risk between the target vehicle and its leading vehicle and between the target vehicle and its following vehicle until the target vehicle completes land-changing. The completion of land-changing is determined when the target vehicle’s distance to the lane line is greater than 0.5m (shown in Figure 2(d)).

#### 1) REAR-END COLLISION RISK

Rear-end collision risk at time $t$ is calculated using Difference of Space distance and Stopping distance (DSS):

$$DSS(t) = \frac{V_l^2(t) - V_f^2(t)}{2\mu g} + d(t) - \tau V_f(t)$$

where $V_l$ and $V_f$ are the longitudinal speed of the leading and following vehicles, respectively;
μ is the fraction rate, set to 0.7; g is the acceleration of gravity, 9.8 m/s²; d is the longitudinal gap between the leading and following vehicles; 
τ is the reaction time of driver. Suggested by Green [26], when the following vehicle is accelerating, τ is set to 1.5 s; when the following vehicle is decelerating or idling, τ is set to 0.7 s.

When DSS ≥ 0, it means the following vehicle has enough time to decelerate and avoid a collision. When DSS < 0, the following vehicle does not have enough time to react to the leading vehicle’s abrupt deceleration and then has a collision risk.

2) SIDE COLLISION RISK
To evaluate the side collision risk between two vehicles, we need to calculate the lateral location of vehicle’s corner that is closest to the adjacent vehicle (denoted as yR). The direction parallel to the lane line is the x axis, and the direction vertical to the lane line is the y axis. The vehicle’s heading direction is determined by its lateral speed $V_y$ and longitudinal speed $V_x$. $y_R$ can be calculated as follows:

$$
\theta(t) = \arctan \left( \frac{-V_y(t)}{V_x(t)} \right) \tag{2}
$$
$$
y_d(t) = \sqrt{w^2 + h^2} \cos \left( \frac{\pi}{2} - \arctan \left( \frac{h}{w} \right) - \theta(t) \right) \tag{3}
$$
$$
y_R(t) = y_c(t) - y_d(t) \tag{4}
$$

where $\theta(t)$ is the angle between the heading direction and the x axis; $y_d(t)$ is the lateral distance between the corner point and center of vehicle; $w$ and $h$ are the vehicle’s length and width, respectively; $y_c(t)$ is the lateral location of vehicle’s center.

In Figure 2(b), the adjacent vehicle is on the left side of the target vehicle. The side collision risk is evaluated using time-to-collision (TTC) in equation (5).

$$
\text{TTC}(t) = \frac{V_{Ay}(t) - V_y(t)}{|y_R(t) - y_A(t)|} \tag{5}
$$

where $V_{Ay}$ is the lateral speed of the adjacent vehicle; $y_A$ is the lateral location of the adjacent vehicle’s center.

If the adjacent vehicle is on the right side, the side collision risk should be evaluated using equation (6).

$$
\text{TTC}(t) = \frac{V_y(t) - V_{Ay}(t)}{|y_R(t) - y_A(t)|} \tag{6}
$$

The threshold of TTC has been studied heavily in literature. Brown et al. [27] and Das and Maurya [28] considered that 3 s as the threshold of TTC to warn rear-end collision. Bella and Russo [29] suggested using 2.5 s or 3 s as the threshold for abnormal car-following. Qu et al. [30] and Minderhoud and Bovy [31] also adopted 3 s as the TTC threshold in their
traffic safety evaluation research. Therefore, this paper uses 3 s as the threshold for side collision.

**B. RISKY DRIVER LABELING**

To establish a risky driver recognition model, we need to label all observed vehicles as either normal drivers or risky drivers. The labels are assigned based on the vehicle’s overall collision risk and used as the ground truth for machine learning model training. Average collision risk (ACR) can measure the overall collision risk exposed to the target vehicle over the whole trajectory:

\[
ACR = \frac{1}{T} \int_0^T \left( 1 \{ \text{DSS}(t) < 0 \} + 1 \{ \text{TTC}(t) < 3 \} \Delta t \right) dt
\]

where \( T \) is the observation duration of the target vehicle; \( \Delta t \) is the time interval of observations; \( 1 \{ \} \) is an indicator function. For example, \( 1 \{ \text{DSS}(t) < 0 \} \) is 1 when \( \text{DSS}(t) < 0 \) and 0 when \( \text{DSS}(t) \geq 0 \).

Risky drivers are labeled based on their ACRs. The ACR threshold is determined using the Interquartile Range (IQR) method [32], which is an outlier detection method proposed by Laurikkala et al. The threshold can be calculated as follows.

\[
\text{Threshold} = Q_3 + 1.5(Q_3 - Q_1)
\]

where \( Q_3 \) and \( Q_1 \) are the upper and lower quartile of the non-zero ACR distribution, respectively. IQR method can calculate the threshold of abnormal data under various distributions [33].

**C. FEATURE EXTRACTION AND SELECTION**

Features are extracted from vehicle’s longitudinal speed, lateral speed, and gap using Discrete Fourier Transform (DFT). DFT converts time series of driving parameters to signal amplitude in the frequency domain and has been proved to be an efficient feature extraction method in driving behavior studies [16]. The DFT of a given time series \( (x_1, x_2, \ldots, x_N) \) is defined as a sequence of \( N \) complex numbers \( (DFT_0, DFT_1, \ldots, DFT_{N-1}) \):

\[
DFT_k = \sum_{n=0}^{N-1} x_ne^{-\frac{2\pi in}{N}}
\]

where \( i \) is the imaginary unit.

The mean, standard deviation, coefficient of variation, and 20 DFT coefficients of longitudinal speed, lateral speed, and gap are extracted as features. There are 72 features in total.

Recursive Feature Elimination (RFE) [34] is a feature selection algorithm. First, all 72 features are used in model training. Second, features are ranked based on their contributions to model performance. Third, the least important feature is iteratively eliminated as long as the model’s performance can be improved.

**D. CONTEXT-AWARENESS STRUCTURE**

This Section proposes two context-awareness structures and compares them with the commonly used risky-driver-recognition structure without context-awareness. All context recognition and risky driver recognition models discussed in this Section use the same 72 features extracted from the vehicle trajectory data as the inputs. The final output of all risky driver recognition models is a 0-1 indicator showing whether a given driver is a risky driver.

1) **RISKY-DRIVER-RECOGNITION WITHOUT CONTEXT-AWARENESS**

The most commonly used Single-layer Risky-driver-recognition (SR) structure is shown in Figure 4. The structure is straightforward. Through feature extraction and selection, vehicle trajectory data is transformed to the input of the risky driver recognition model. The recognition model often utilizes supervised machine learning algorithms, such as decision tree, support vector machine, neural network, and ensemble learning. Given an input, the well-trained recognition model can generate a prediction of driving style.

In the whole process, contextual information (traffic state, for instance) is not involved.

![Figure 4. Single-layer risky-driver-recognition without context-awareness (SR).](image)

2) **RISKY-DRIVER-RECOGNITION WITH CONTEXT-AWARENESS**

The first proposed 2-layer context-awareness structure is shown in Figure 5, namely Two-layer Risky-driver-recognition with Context-awareness (TRC). TRC includes \( K + 1 \) recognition models, where \( K \) is the number of context categories. Figure 5 considers three context categories: free traffic flow, saturated traffic flow, and congested traffic flow; therefore, we have one context recognition model and three risky driver recognition models, one for each context category.

The input features are first used to recognize the context (traffic state in this paper), in which the target vehicle is moving, and then the input features are delivered into the corresponding specialized model to recognize risky drivers.
K. Wang, J. J. Lu: Two-Layer Risky Driver Recognition Model With Context Awareness

**FIGURE 5.** Two-layer risky-driver-recognition structure with context-awareness (TRC).

Based on the predicted context category. For example, if a given vehicle is recognized as moving in the context of congested flow, the risky driver recognition model under congested traffic flow is chosen to predict the driver’s driving style.

3) **PROBABILITY-BASED RISKY-DRIVER-RECOGNITION WITH CONTEXT-AWARENESS**

The second proposed 2-layer context-awareness structure is shown in Figure 6, namely Probability-based Two-layer Risky-driver-recognition with Context-awareness (PTRC). Same as RC, PRC has \( K + 1 \) recognition models. However, instead of outputting predicted labels, the \( K + 1 \) recognition models generate predicted probabilities for each label. The probability-based 2-layer risky-driver-recognition model can be expressed as follows:

\[
\hat{p}(y_1) = \hat{p}(y_1|c_1) \hat{p}(c_1) + \hat{p}(y_1|c_2) \hat{p}(c_2) + \hat{p}(y_1|c_3) \hat{p}(c_3)
\]

(10)

where:
- \( \hat{p}(y_1) \) is the predicted probability of being risky driver for a given input;
- \( \hat{p}(c_1) \), \( \hat{p}(c_2) \), \( \hat{p}(c_3) \) are the predicted probability of context 1, 2, and 3, respectively, reported by the context recognition model;
- \( \hat{p}(y_1|c_1) \), \( \hat{p}(y_1|c_2) \), \( \hat{p}(y_1|c_3) \) are the predicted probability of being risky driver, reported by the risky driver recognition models under context 1, 2, and 3, respectively.

If \( \hat{p}(y_1) > 0.5 \), then the predicted driving style for the given input is risky; otherwise, the predicted driving style is normal.

**FIGURE 6.** Probability-based two-layer risky-driver-recognition structure with context-awareness (PTRC).

**E. CLASSIFIER**

The establishment of context recognition model and risky driver recognition models need machine learning classification algorithms. The classifier candidates are described below.

1) **SUPPORT VECTOR MACHINE**

Support vector machine (SVM) algorithm [35] is to find a hyperplane in a multi-dimensional space that distinctly classifies the data points with the maximum margin. The hyperparameters of SVM are the \( C \) parameter and gamma.

2) **K-NEAREST NEIGHBORS**

K-nearest neighbors (KNN) algorithm [36] is a non-parametric supervised machine learning method that is widely used in classification problems. An instance is assigned to the class most common among its \( k \) nearest neighbors. KNN has three important hyperparameters: the number of neighbors, power parameter for the Minkowski distance metric, and weight function used in prediction.

3) **AdaBoost**

AdaBoost [37], short for Adaptive Boosting, fits a classifier on the original dataset and then increases the weights of incorrectly classified instances such that subsequent classifiers focus more on these instances. The hyperparameters of AdaBoost include the number of estimators and learning rate.
the base classifier is set to decision tree with maximum depth equals 1.

4) RANDOM FOREST
Random Forest (RF) is an ensemble learning method consisting of multiple decision trees trained independently on various sub-samples of the dataset [38]. The class prediction with the most votes from decision trees becomes the final prediction of the random forest. The hyperparameters of RF include the number of estimators, maximum depth, the minimum number of samples required to split an internal node, maximum features, etc.

5) EXTRA TREE
Extra trees [39] and random forest are two similar ensemble methods. One difference is that random forest uses bootstrap replicates while extra trees use the whole original sample. Another difference is that RF chooses the optimum split for nodes while extra trees choose it randomly. The hyperparameters of extra trees include the number of estimators, maximum depth, the minimum number of samples required to split an internal node, maximum features, etc.

6) XGBoost
XGBoost, short for eXtreme Gradient Boosting, is a gradient tree boosting-based software that is superior in performance, fast in training time, and has an easy-to-use interface [40]. The hyperparameters of XGBoost include the number of estimators, learning rate, maximum depth, subsample ratio, etc.

We apply hyperparameter tuning in the training of each classifier using HyperOpt. HyperOpt [41] is an open-source Python library for hyperparameter tuning. It allows for the automatic search of the optimal value of hyperparameters mention above by Tree of Parzen Estimators (TPE). It is more powerful than manual search and grid search because of its speediness, stability, and accuracy compared to manual tuning and grid search.

F. PROBABILITY CALIBRATION
All classifiers listed in the last Section can predict class with a probability score for each class. The probability scores are necessary for the PTCR structure. However, these probability scores are usually biased for two reasons: first, SVM and boosted trees are not trained using a probabilistic framework and then product biased probabilities [42]; second, imbalanced data and sampling impacts the predicted probabilities for the majority and minority class instances [43].

There are two common probability calibration methods: Platt scaling [44] and isotonic regression [45], [46]. Platt scaling trains a logistic regression to map the original classifier’s output to the true class probability. Isotonic regression is a non-parametric approach that fits a piecewise constant non-decreasing function, where predicted probabilities are monotonically increasing over bins. Platt scaling is applied in this paper since isotonic regression may have an overfitting issue when the calibration data sample size is small.

G. CROSS-VALIDATION AND EVALUATION
This paper uses precision rate, recall rate, F1 score, and Area under the Precision-Recall Curve (AUPRC) to evaluate recognition model’s performance.

Precision rate is defined as follows:

\[ \text{Precision} = \frac{TP}{TP + FP} \]  

where TP is the number of risky drivers correctly identified; FP is the number of normal drivers wrongly identified as risky drivers.

Recall rate is defined as follows:

\[ \text{Recall} = \frac{TP}{TP + FN} \]  

where FN is the number of risky drivers wrongly identified as normal drivers.

The F1 score is the harmonic average of precision rate and recall rate. A high F1 score represents high values in both precision rate and recall rate.

\[ F1 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \]  

The precision-recall curve is a plot of the precision rate and the recall rate for different probability thresholds. Area Under Precision-Recall Curve (AUPRC) is more appropriate than Area Under Receiver Operating Characteristic curve (AUROC) to measure the model’s performance when the dataset is imbalanced [47].

Machine learning algorithms are commonly evaluated using k-fold cross-validation, and their evaluation metrics, such as mean accuracy scores, are compared directly. Statistical significance tests are designed to test whether the difference between evaluation metrics is statistically significant or the result of a statistical fluke. The null hypothesis is that metric scores observed from two algorithms were drawn from the same distribution. If this assumption is rejected, it suggests that the difference in metric scores is statistically significant. Otherwise, the two algorithms’ performances are statistically equal.

K-fold cross-validated paired Student’s t-test is the most used statistical test for machine learning algorithms comparison. However, the calculation of the t-statistic in the test is misleading since the metric scores in each sample are not independent [48]. In k-fold cross-validation, a given observation will be used in the training dataset k-1 times. This means that the estimated metric scores are dependent.

Dietterich [48] recommended a resampling method called 5 × 2-fold cross-validation that involves five repeats of 2-fold cross-validation. Two-fold cross-validation can ensure that each observation appears only in the train or test dataset once. A paired Student’s t-test is used on the results.

\[ t = \frac{\mu}{\sqrt{\frac{1}{5} \sum_{i=1}^{5} \left( \left( \Delta_i^{(1)} - \mu \right)^2 + \left( \Delta_i^{(2)} - \mu \right)^2 \right)}} \]  

\[ t = \frac{\mu}{\sqrt{\frac{1}{5} \sum_{i=1}^{5} \left( \left( \Delta_i^{(1)} - \mu \right)^2 + \left( \Delta_i^{(2)} - \mu \right)^2 \right)}} \]  

\[ t = \frac{\mu}{\sqrt{\frac{1}{5} \sum_{i=1}^{5} \left( \left( \Delta_i^{(1)} - \mu \right)^2 + \left( \Delta_i^{(2)} - \mu \right)^2 \right)}} \]
where:
\[ \Delta_i^{(1)} \] is the scores difference of two algorithms for the first fold of the \( i \)-th 2-fold cross-validation;
\[ \Delta_i^{(2)} \] is the scores difference of two algorithms for the second fold of the \( i \)-th 2-fold cross-validation;
\[ \mu = \frac{\Delta_i^{(1)} + \Delta_i^{(2)}}{2} \] is the mean of scores difference for the first 2-fold cross-validation.

Under the null hypothesis that two algorithms are statistically equal, \( t \) is assumed to follow a Student’s t-distribution with 5 degrees of freedom. If \( t \) stays close enough to 0, then the null hypothesis is satisfied. The threshold is 2.571 at the 95% confidence level. \( 5 \times 2 \) cross-validation is used in this paper to compare algorithms’ performance.

We use stratified \( 5 \times 2 \)-fold cross-validation to evaluate the classification algorithm’s performance. The average AUPRC, F1 score, precision rate, and recall rate are used to compare the performance of models. To find the optimal hyperparameters of each classifier, we iterate the hyperparameter optimization algorithm 500 times. The hyperparameter values that can reach the highest average AUPRC are the optimal hyperparameters.

**IV. RESULTS**

The computation platform is a laptop with an AMD Ryzen 7 8-core CPU. Analysis of collision risk, risky driver labeling, and driving behavior parameters was done on Matlab 2018b. All machine learning work was done using Python with supported libraries, including pandas, numpy, xgboost, imblearn, hyperopt, and scikit-learn.

First, we analyze the labeling result in Section IV-A; second, we compare the driving parameters in three traffic states and justify the necessity of recognizing traffic states.

**A. COLLISION RISK AND RISKY DRIVERS**

Using the method introduced in Section III-A, we calculate the ACR of drivers and plot the distributions in Figures 7-9. Only samples satisfying the following rules are kept in the dataset:

- The vehicle type is private car.
- The vehicle’s total duration in car-following state is longer than 10 s.

As shown in Figure 7, in free flow, more than 50% of drivers have a zero or near-zero ACR value indicating they are driving at a safe state; 20% of drivers have an ACR between 1.25 s and 2.5 s. About 10% of drivers have an ACR greater than 5 s.

As shown in Figure 8, in saturated flow, 22% of drivers have a zero or near-zero ACR, while the percentage of drivers with ACR between 1.25 s and 2.5 s rises to 31%. About 25% of drivers have an ACR greater than 5 s.

As shown in Figure 9, in congested flow, 22% of drivers have a zero or near-zero ACR, while the percentage of drivers with ACR between 1.25 s and 2.5 s rises to 45%. About 20% of drivers have an ACR greater than 5 s.

Although the distributions of ACR vary across different traffic states, we want to determine universal criteria for all traffic states in order to label risky drivers. Based on the IQR method introduced in Section III-B, we calculate the upper and lower quartile of all ACRs and find ACR = 0.5 as the universal risky driver threshold for three traffic states. Drivers with ACR > 0.5 are labeled as risky drivers. The data information after labeling is shown in Table 2. There are 99 risky drivers in the free flow file, 155 risky drivers in the saturated flow file, and 86 risky drivers in the congested flow file. The saturated flow file has the highest percentage of risky drivers, but still, the data is highly imbalanced, and the imbalance ratio is 10.1:1. The imbalance ratio of the free
flow file is 14.4:1, and the imbalance ratio of the congested flow file is 22.3:1.

We find that drivers tend to drive more safely in congested flow due to the congestion. For example, the average lane change per vehicle decreases as traffic volume increases. The percentage of risky drivers in congested flow is the lowest among all three flow states.

### B. DRIVING BEHAVIOR PARAMETERS IN THREE TRAFFIC STATES

We analyzed the differences in driving risk of three traffic states in the last Section. This Section compares the distributions of individual’s four driving behavior parameters in three traffic states. The individual’s driving behavior parameters include each vehicle’s longitudinal speed, lateral speed, longitudinal acceleration, and longitudinal deceleration.

1) **LONGITUDINAL SPEED**

Unsurprisingly, the free-flow has a relatively higher mean longitudinal speed than congested flow and saturated flow. It is noteworthy that many drivers in the free-flow have a mean longitudinal speed higher than the speed limit, which is 33.33 m/s, or 120 km/h. In congested flow, more than 56% of drivers have a mean longitudinal speed lower than 25 m/s. The distribution of mean longitudinal speed in the saturated flow is between that in the free flow and congested flow.

![FIGURE 10. Distribution of mean longitudinal speed.](image)

The free-flow has the lowest longitudinal speed fluctuation. About 38% of drivers in free-flow have a near-zero longitudinal speed fluctuation. The fluctuation in the saturated flow is higher than that in the free flow, while the congested flow has the highest fluctuation.

![FIGURE 11. Distribution of standard deviation of longitudinal speed.](image)

![FIGURE 12. Distribution of mean lateral speed.](image)

![FIGURE 13. Distribution of standard deviation of lateral speed.](image)

2) **LATERAL SPEED**

Due to congestion and fewer lane changes, more than 90% of drivers in the congested flow have a mean lateral speed close to zero, while this percentage in the saturated flow and free flow is about 80% and 65%, respectively.

Although drivers in congested flow have the highest fluctuation in longitudinal speed, they have a more stable lateral speed than drivers in free flow and saturated flow. The distributions of the standard deviation of lateral speed in free flow and saturated flow are similar.

---

**TABLE 2. Driving style information of three data files.**

| Flow state | Sample size | No. of normal drivers | No. of risky drivers | Imbalance ratio | Average lane changes |
|------------|-------------|-----------------------|---------------------|----------------|---------------------|
| Free-flow  | 1529        | 1430                  | 99                  | 14.4:1         | 0.12                |
| Saturated | 1720        | 1565                  | 155                 | 10.1:1         | 0.08                |
| Congested | 2004        | 1918                  | 86                  | 22.3:1         | 0.05                |
3) LONGITUDINAL ACCELERATION AND DECELERATION
Since the drivers’ longitudinal speed in free flow is more stable than those in saturated flow and congested flow, their distributions of mean longitudinal acceleration and deceleration are more concentrated around zero. By contrast, the distributions of mean longitudinal acceleration and deceleration in congested flow are more diverse.

In summary, we find that congested traffic flow has more drivers with unstable longitudinal speed, acceleration, and deceleration. However, as shown in Table 2, the percentage of risky drivers in congested flow is lower than the other two flow states. It is inconsistent with common sense that speed instability and abrupt acceleration/deceleration are indicators of risky driving. This contradiction implies that drivers should not be compared directly based on their driving parameters across different traffic states. In order to justify risky drivers, traffic state should be clarified first, and then drivers in the same context (traffic state) are compared to recognize risky drivers.

C. MODEL RESULTS
This Section aims to test the performance of three risky-driver-recognition structures: SR, TRC, and PTRC. SR is the baseline structure, and TRC and PTRC are our proposed structures. SR has only one recognition model: the risky driver recognition model (results are shown in Table 3). TRC and PTRC have four recognition models: one context recognition model (results are shown in Table 4) and three risky driver recognition models (results are shown in Tables 5-7), one for each traffic state.

For each model, we compare the performance of six classifiers: SVM, KNN, Adaboost, RF, Extra tree, and XGBoost. Table 3 shows that XGBoost is the best classifier for risky driver recognition model in the SR structure with the highest AUPRC, 0.781, and the highest F1, 0.730. SVM and KNN classifiers are much worse than ensemble learning algorithms, including Adaboost, RF, Extra tree, and XGBoost.

For the traffic state recognition model, XGBoost is the best choice. As shown in Table 4, XGBoost reaches the highest AUPRC, 0.804, the highest precision, 0.805, the highest recall, 0.804, and the highest F1, 0.804, among all classifiers. The second-best classifier is RF, with the second-highest AUPRC, precision, recall, and F1.

For the risky driver recognition model in each traffic state, XGBoost is still the best classifier. As shown in Tables 5-7, XGBoost’ AUPRC is 0.764 in free flow, 0.802 in saturated flow, and 0.756 in congested flow; XGBoost’ F1 is 0.728 in free flow, 0.768 in saturated flow, and 0.766 in congested flow. According to F1, the second-best classifier is RF, with the second-highest AUPRC, precision, recall, and F1.

Comparing the XGBoost results from Table 3 with those from Table 5-7, we find that awareness of traffic state can increase model’s risky driver recognition ability. For example, without knowing the driving context (traffic state), the precision and recall of the risky driver recognition model are 0.763 and 0.715, respectively. Knowing that a driver is in saturated flow, the precision and recall of risky driver

| Table 3. Performance of risky driving recognition model without context recognition. |
|---|---|---|---|
| Classifier | AUPRC | Precision | Recall | F1 |
| SVM | 0.519 | 0.058 | 0.900 | 0.109 |
| KNN | 0.459 | 0.327 | 0.726 | 0.443 |
| Adaboost | 0.703 | 0.678 | 0.694 | 0.671 |
| RF | 0.728 | 0.708 | 0.700 | 0.693 |
| Extra tree | 0.733 | 0.776 | 0.632 | 0.682 |
| XGBoost | 0.781 | 0.763 | 0.715 | 0.730 |

| Table 4. Performance of traffic state recognition model. |
|---|---|---|---|
| Classifier | AUPRC | Precision | Recall | F1 |
| SVM | 0.401 | 0.202 | 0.401 | 0.246 |
| KNN | 0.554 | 0.595 | 0.554 | 0.561 |
| Adaboost | 0.714 | 0.726 | 0.714 | 0.717 |
| RF | 0.778 | 0.780 | 0.778 | 0.779 |
| Extra tree | 0.717 | 0.729 | 0.717 | 0.721 |
| XGBoost | 0.804 | 0.805 | 0.804 | 0.804 |

| Table 5. Performance of risky driving recognition model in free flow. |
|---|---|---|---|
| Classifier | AUPRC | Precision | Recall | F1 |
| SVM | 0.499 | 0.058 | 0.900 | 0.109 |
| KNN | 0.410 | 0.333 | 0.619 | 0.414 |
| Adaboost | 0.622 | 0.617 | 0.729 | 0.646 |
| RF | 0.727 | 0.765 | 0.739 | 0.723 |
| Extra tree | 0.705 | 0.767 | 0.649 | 0.685 |
| XGBoost | 0.764 | 0.813 | 0.698 | 0.728 |
recognition model are 0.834 and 0.730, respectively; knowing that a driver is in congested flow, the precision and recall of risky driver recognition model are 0.800 and 0.783, respectively; knowing that a driver is in free flow, the precision and recall of risky driver recognition model are 0.813 and 0.698, respectively. All three traffic state-specific risky driver recognition models have higher precision and recall than the general risky driver recognition model, except the model in free flow, which has a slightly lower recall score than the general model.

Using XGBoost as the classifier for all recognition models in SR, TRC, and PTRC, we list the final risky driver recognition evaluation results in Table 8. The results for SR are the same as the results for XGBoost in Table 3 since the SR is a single-layer structure and has only one recognition model. The results for TRC and PTRC are the combination of context (traffic state) recognition and risky driver recognition. Given a vehicle without knowing its context, TRC and PTRC need to predict its context and corresponding probabilities and then using traffic state-specific risky driver recognition models to predict driving style. PTRC (calibrated) applies Platt scaling to calibrate predicted probabilities of XGBoost, instead of using uncalibrated probabilities directly.

Table 8 shows that TRC, PTRC, and PTRC (calibrated) all outperform SR according to AUPRC and F1 scores. We conduct Student’s t-test to see whether the difference between SR and other structures is statistically significant. TRC, PTRC, and PTRC (calibrated) all pass the Student’s t-test, which means the two-layer structure is significantly better than the commonly used single-layer structure in the field of risky driver recognition. PTRC (calibrated) has the highest AUPRC, 0.820, and the highest F1, 0.762.

### V. DISCUSSION

The impact of traffic state on driving behavior is evident. As traffic congestion increases, the vehicle’s longitudinal speed becomes unstable, lateral speed’s variation decreases, abrupt acceleration/deceleration occurs more frequently. Moreover, we find that saturated flow has the highest percentage of risky drivers and congested flow has the lowest based on collision risk evaluation. Therefore, we believe that traffic state is an essential factor in risky driver recognition, and the relationship between traffic state and collision risk is complicated that congestion does not always cause traffic risk.

Our goal in this paper is to test whether we can increase machine learning’s ability of risky driver recognition by using novel two-layer structures without introducing any new features. Since the input features of machine learning have no traffic state information, the two-layer structures need to predict traffic state and recognize risky drivers. There are two main advantages of two-layer structure compared to single-layer structure:

- **More Flexibility in Machine Learning Models**: Two-layer structures have $K+1$ models, and each recognition model can apply different classifiers and different hyperparameter values to reach the highest performance possible. By contrast, a single-layer structure has only one model and can only apply only one classifier with one hyperparameter setting.

- **Adaptable to New Data**: Traffic states and driving behavior is complex. Different locations could have different compositions in traffic states and driving styles. The data we applied in this paper have 1529 vehicles in free flow, 1720 vehicles in saturated flow, and 2004 vehicles in congested flow. The percentage of risky drivers in each traffic state is also different. For example, a heavily used highway is always congested, and its driving risk is different from a highway that is congested only at peak hours. The single-layer structure requires a retraining of the whole risky driver recognition model, while the two-layer structure only needs to retrain the risky driver recognition model for congested flow.

The two-layer structure’s disadvantage is the relatively high computational cost. SR structure only needs to train one risky driver recognition model, while TRC and PTRC need to train one traffic state recognition model and three traffic state-specific risky driver recognition models. Since the training of traffic state-specific models involves a subsample of the data, the training time of each state-specific model is much less than that of the model in the SR structure.

### TABLE 9. Computational cost of risky driving recognition structures.

| Structure          | Time (s) |
|--------------------|----------|
| SR                 | 8.5      |
| TRC                | 19.0     |
| PTRC               | 19.0     |
| PTRC (calibrated)  | 19.1     |
Table 9 lists the computational cost of each structure through 10-fold cross-validation. The computation time of SR is 8.5 s, and the computation time of TRC and PTRC are both 19.0 s. The PTRC (calibrated) takes 19.1 s since it needs a little extra time for probability calibration. Although the two-layer structures’ computational cost is much higher than the single-layer structure, it may not be a major concern in practice, thanks to the growing computation power of electronic devices. Moreover, parallel computation can help to reduce the computational cost of two-layer structures since the K + 1 models can be trained independently.

**VI. CONCLUSION**

This paper proposes two risky driver recognition structures: Two-layer Risky-driver-recognition with Context-awareness (TRC) and Probability-based Two-layer Risky-driver-recognition with Context-awareness (PTRC). Using vehicle trajectory data, we are able to demonstrate the impact of traffic state on driving behavior parameters and risky driving. Therefore, as an essential type of context information, traffic state should be considered in the risky driver recognition model. We confirm that TRC and PTRC are significantly better than the single-layer structure by achieving higher AUPRC and F1 scores through the statistical test. PTRC’s performance can be further increased after probability calibration.

This research has a few limitations. The performance improvement of proposed structures is moderate. Future improvement could be achieved by considering more traffic states. However, the optimal number of traffic states and traffic states classification problems are not discussed in this paper. The performance evaluation of the proposed structure uses a single source dataset recorded on a German highway. To prove the value of TRC and PTRC, we need more performance tests with new datasets.

For future research, more detailed context information should be considered, such as road type, weather, and vehicle type. Driving behavior data from other sources, such as naturalistic driving data and driving simulator data, are valuable to further test the performance of proposed structure.

**REFERENCES**

[1] J. J. Rolison, S. Regev, S. Moutari, and A. Feeney, “What are the factors that contribute to road accidents? An assessment of law enforcement views, ordinary drivers’ opinions, and road accident records,” *Accident Anal. Prevention*, vol. 115, pp. 11–24, Jun. 2018.

[2] F. Sagberg, Selpi, G. F. B. Piccinini, and J. Engström, “A review of research on driving styles and road safety,” *Hum. Factors*, vol. 57, no. 7, pp. 1248–1275, 2015.

[3] C. M. Martinez, M. Heucke, B. Gao, D. Cao, and F.-Y. Wang, “Driving style recognition for intelligent vehicle control and advanced driver assistance: A survey,” *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 666–676, Mar. 2018, doi: 10.1109/TITS.2017.2706978.

[4] O. Taubman-Ben-Ari, M. Mikulincer, and O. Gillath, “The multidimensional driving style inventory—Scale construct and validation,” *Accid. Anal. Prevention*, vol. 36, no. 3, pp. 323–332, May 2004, doi: 10.1016/S0001-4575(03)00010-1.

[5] T. H. Itkonen, E. Lehtonen, and Selpi, “Characterisation of motorway driving style using naturalistic driving data,” *Transp. Res. F: Traffic Psychol. Behav.*, vol. 69, pp. 72–79, Feb. 2020, doi: 10.1016/j.trf.2020.01.003.

[6] E. Ericsson, “Variability in urban driving patterns,” *Transp. Res. D, Transp. Environ.*, vol. 5, no. 5, pp. 337–354, 2000, doi: 10.1016/S1361-9200(99)00003-1.

[7] H. Xia, K. Boriboomsmsin, and M. Barth, “Dynamic ECO-driving for signalized arterial corridors and its indirect network-wide energy/emissions benefits,” *J. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 31–41, 2013, doi: 10.1080/15472450.2012.712494.

[8] V. V. Dikit, V. V. Gayah, and E. Radwan, “Comparison of driver behavior by time of day and wet pavement conditions,” *J. Transp. Eng.*, vol. 138, no. 8, pp. 1023–1029, Aug. 2012, doi: 10.1061/(ASCE)TE.1943-5436.0000400.

[9] W. Han, W. Wang, X. Li, and J. Xi, “Statistical-based approach for driving style recognition using Bayesian probability with kernel density estimation,” *IET Intell. Transp. Syst.*, vol. 13, no. 1, pp. 22–30, Jan. 2019, doi: 10.1049/iet-its.2017.0379.

[10] K. Wang, Q. Xue, X. Y. Xing, and C. Li, “Improve aggressive driver recognition using collision surrogate measurement and imbalanced class boosting,” *Int. J. Environ. Res. Public Health*, vol. 17, no. 7, p. 2375, Mar. 2020.

[11] D. Yi, J. Su, C. Liu, M. Quddus, and W.-H. Chen, “A machine learning based personalized system for driving state recognition,” *Transp. Res. C, Emerg. Technol.*, vol. 105, pp. 241–261, Aug. 2019, doi: 10.1016/j.trc.2019.05.042.

[12] J.-Y. Hong, E.-H. Suh, and S.-J. Kim, “Context-aware systems: A literature review and classification,” *Expert Syst. Appl.*, vol. 36, no. 4, pp. 8509–8522, May 2009, doi: 10.1016/j.eswa.2008.10.071.

[13] M. M. Bejani and M. Ghatee, “A context aware system for driving style evaluation by an ensemble learning on smartphone sensors data,” *Transp. Res. C, Emerg. Technol.*, vol. 89, pp. 303–320, Apr. 2018, doi: 10.1016/j.trc.2018.02.009.

[14] E. Suzdaleva and I. Nagy, “Two-layer pointer model of driving style depending on the driving environment,” *Transp. Res. B, Methodol.*, vol. 128, pp. 254–270, Oct. 2019, doi: 10.1016/j.trb.2019.08.009.

[15] G. S. Aoude, V. R. Desaraju, L. H. Stephens, and J. P. How, “Driver behavior classification at intersections and validation on large naturalistic data set,” *IEEE Trans. Intell. Transp. Syst.*, vol. 13, no. 2, pp. 724–736, Jun. 2012.

[16] R. Kluger, B. L. Smith, H. Park, and D. J. Dailey, “Identification of safety-critical events using kinematic vehicle data and the discrete Fourier transform,” *Accident Anal. Prevention*, vol. 96, pp. 162–168, Nov. 2016.

[17] M. M. Bejani and M. Ghatee, “A context aware system for driving style evaluation by an ensemble learning on smartphone sensors data,” *Transp. Res. C, Emerg. Technol.*, vol. 89, pp. 303–320, Apr. 2018.

[18] N. Zhao, B. Mehler, B. Reimer, L. A. D’Ambrosio, A. Mehler, and J. F. Coughlin, “An investigation of the relationship between the driving behavior questionnaire and objective measures of highway driving behavior,” *Transp. Res. F: Traffic Psychol. Behav.*, vol. 15, no. 6, pp. 676–685, Nov. 2012.

[19] Q. Xue, K. Wang, J. J. Lu, and Y. Liu, “Rapid driving style recognition in car-following using machine learning and vehicle trajectory data,” *J. Adv. Transp.*, vol. 2019, pp. 1–11, Jan. 2019.

[20] F. Wang, J. Zhang, S. Wang, S. Li, and W. Hou, “Analysis of driving behavior based on dynamic changes of personality states,” *Int. J. Environ. Res. Public Health*, vol. 17, no. 2, p. 430, Jan. 2020.

[21] T. Liu, Y. Yang, G. B. Huang, Y. K. Yeo, and Z. Lin, “Driver distraction detection using semi-supervised machine learning,” *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 4, pp. 1108–1120, Apr. 2016.

[22] W. Wang, J. Xi, A. Chong, and L. Li, “Driving style classification using a semisupervised support vector machine,” *IEEE Trans. Human-Mach. Syst.*, vol. 47, no. 5, pp. 650–660, Oct. 2017.

[23] L. Peng, M. A. Sotelo, Y. He, Y. Ai, and Z. Li, “Rough set based method for vehicle collision risk assessment through inferring Driver’s braking actions in near-crisis situations,” *IEEE Trans. Intell. Transp. Syst. Mag.*, vol. 11, no. 2, pp. 54–69, Mar. 2019, doi: 10.1109/MITS.2019.2903539.

[24] L. Peng, Y. Wang, F. Zhang, J. Zhang, and Z. Li, “Evaluation of emergency driving behaviour and vehicle collision risk in connected vehicle environment: A deep learning approach,” *IET Intell. Transp. Syst.*, vol. 15, no. 4, pp. 754–764, Apr. 2021, doi: 10.1049/iet-its.2020.12153.

[25] R. Krajewski, J. Bock, L. Kloecker, and L. Eckstein, “The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems,” in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2118–2125.

[26] M. Green, “‘How long does it take to stop?’ Methodological analysis of driver perception-brake times,” *Transp. Hum. Factors*, vol. 2, pp. 195–216, Sep. 2000, doi: 10.1012/STHPF2003_1.
[27] T. L. Brown, J. D. Lee, and D. V. Mcgehee, “Human performance models and rear-end collision avoidance algorithms,” *Hum. Factors*, vol. 43, no. 3, pp. 462–482, 2001.

[28] S. Das and A. K. Maurya, “Defining time-to-collision thresholds by the type of lead vehicle in non-lane-based traffic environments,” *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 12, pp. 4972–4982, Dec. 2020.

[29] F. Bella and R. Russo, “A collision warning system for rear-end collision: A driving simulator study,” *Proc.-Social Behav. Sci.*, vol. 20, pp. 676–686, Jan. 2011.

[30] X. Qu, Y. Kuang, E. Oh, and S. Jin, “Safety evaluation for expressways: A comparative study for macroscopic and microscopic indicators,” *Traffic Injury Prevention*, vol. 15, no. 1, pp. 89–93, 2014.

[31] M. M. Minderhoud and P. H. Bovy, “Extended time-to-collision measures for road traffic safety assessment,” *Accident Anal. Prevention*, vol. 33, no. 1, pp. 89–97, Jan. 2001.

[32] J. Laurikkala, M. Juhola, and E. Kentala, “Informal identification of outliers in medical data,” in *Proc. Workshop Notes 14th Eur. Conf. Artif. Intell. (ECAI): 5th Workshop Intell. Data Anal. Med. Pharmacol.* (IDAMAP), N. Lavrac, S. Miksch, and K. Branko, Eds. Amsterdam, The Netherlands: IOS Press, 2000, pp. 20–24.

[33] I. Vasconcelos, R. O. Vasconcelos, B. Olivier, M. Roriz, M. Endler, and M. C. Junior, “Smartphone-based outlier detection: A complex event processing approach for driving behavior detection,” *J. Internet Services Appl.*, vol. 8, no. 1, pp. 1–30, Sep. 2017.

[34] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik, “Gene selection for cancer classification using support vector machines,” *Mach. Learn.*, vol. 46, nos. 1–3, pp. 389–422, 2002, doi: 10.1023/A:1012487302797.

[35] B. E. Boser, I. M. Guyon, and V. N. Vapnik, “A training algorithm for optimal margin classifiers,” in *Proc. 5th Annu. Workshop Comput. Learn. Theory (COLT)*, 1992, pp. 144–152, doi: 10.1145/130385.130401.

[36] E. Fix and J. L. Hodges, “Discriminatory analysis, nonparametric discrimination: Consistency properties,” USAF School Aviation Med., Randolph Field, TX, USA, Tech. Rep. 4, 1951.

[37] Y. Freund and R. E. Schapire, “A decision-theoretic generalization of on-line learning and an application to boosting,” *J. Comput. Syst. Sci.*, vol. 55, pp. 119–139, Aug. 1997, doi: 10.1006/jcss.1997.1504.

[38] L. Breiman, “Random forests,” *Mach. Learn.*, vol. 45, no. 1, pp. 5–32, 2001, doi: 10.1023/A:101093404324.

[39] P. Geurts, D. Ernst, and L. Wehenkel, “Extremely randomized trees,” *Mach. Learn.*, vol. 63, no. 1, pp. 3–42, 2006, doi: 10.1007/s10994-006-6226-1.

[40] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794, doi: 10.1145/2939672.2939785.

[41] J. Bergstra, D. Yamins, and D. D. Cox, “Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures,” in *Proc. Int. Conf. Mach. Learn.*, Jun. 2013, pp. 115–123.

[42] A. Niculescu-Mizil and R. Caruana, “Predicting good probabilities with supervised learning,” in *Proc. 22nd Int. Conf. Mach. Learn. (ICML)*, 2005, pp. 625–632.

[43] B. C. Wallace and I. J. Dahabreh, “Class probability estimates are unreliable for imbalanced data (and how to fix Them),” in *Proc. IEEE 12th Int. Conf. Data Mining*, Dec. 2012, pp. 695–704.

[44] J. C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in *Advances in Large Margin Classifiers*. Cambridge, MA, USA: MIT Press, 1999, pp. 61–74.

[45] B. Zadrozny and C. Elkan, “Obtaining calibrated probability estimates from decision trees and naive Bayesian classifiers,” in *Proc. Int. Conf. Mach. Learn.*, Jun. 2001, pp. 609–616.

[46] B. Zadrozny and C. Elkan, “Transforming classifier scores into accurate multiclass probability estimates,” in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2002, pp. 694–699, doi: 10.1145/775047.775151.

[47] T. Saito and M. Rehmsmeier, “The precision-recall plot is more informative than the ROC plot when evaluating binary classifiers on imbalanced datasets,” *PLoS ONE*, vol. 10, no. 3, Mar. 2015, Art. no. e0118432.

[48] T. G. Dietterich, “Approximate statistical tests for comparing supervised classification learning algorithms,” *Neural Comput.*, vol. 10, no. 7, pp. 1895–1923, 1998, doi: 10.1162/089976698300017197.