Automatic Unusual Driving Event Identification for Dependable Self-Driving

Hongyu Li
WINLAB, Rutgers University
North Brunswick, New Jersey
hongyuli@winlab.rutgers.edu

Hairong Wang
WINLAB, Rutgers University
North Brunswick, New Jersey
hairong@winlab.rutgers.edu

Luyang Liu
WINLAB, Rutgers University
North Brunswick, New Jersey
luyang@winlab.rutgers.edu

Marco Gruteser
WINLAB, Rutgers University
North Brunswick, New Jersey
gruteser@winlab.rutgers.edu

ABSTRACT

This paper introduces techniques to automatically detect driving corner cases from dashcam video and inertial sensors. Developing robust driver assistance and automated driving technologies requires an understanding of not just common highway and city traffic situations but also a plethora of corner cases that may be encountered in billions of miles of driving. Current approaches seek to collect such a catalog of corner cases by driving millions of miles with self-driving prototypes. In contrast, this paper introduces a low-cost yet scalable solution to collect such events from any dashcam-equipped vehicle to take advantage of the billions of miles that humans already drive. It detects unusual events through inertial sensing of sudden human driver reactions and rare visual events through a trained autoencoder deep neural network. We evaluate the system based on more than 120 hours real road driving data. It shows 82% accuracy improvement versus strawman solutions for sudden reaction detection and above 71% accuracy for rare visual views identification. The detection results proved useful for re-training and improving a self-steering algorithm on more complex situations. In terms of computational efficiency, the Android prototype achieves 17Hz frame rate (Nexus 5X).

CCS CONCEPTS

• **Human-centered computing** → **Ubiquitous and mobile computing systems and tools**; • **Computing methodologies** → **Scene anomaly detection**; • **General and reference** → *Design*;

KEYWORDS

Self-Driving, Unusual Driving Identification, Autoencoder, Inertial sensing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
SenSys '18, November 4–7, 2018, Shenzhen, China
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5952-8/18/11…$15.00
https://doi.org/10.1145/3274783.3274838

ACM Reference Format:
Hongyu Li, Hairong Wang, Luyang Liu, and Marco Gruteser. 2018. Automatic Unusual Driving Event Identification for Dependable Self-Driving. In *The 16th ACM Conference on Embedded Networked Sensor Systems (SenSys '18), November 4–7, 2018, Shenzhen, China.* ACM, New York, NY, USA, 13 pages. https://doi.org/10.1145/3274783.3274838

1 INTRODUCTION

While automated driving technology has made great strides, ensuring dependability over a broad set of unusual traffic situations and corner cases remains a key challenge [1–3]. Current automated driving products largely require human supervision (NHTSA level 2) [4], with only few systems that allow taking eyes off the road under very limited conditions (level 3) [5]. Self-driving without supervision in select geographic areas environments (level 4) appears to be emerging but it is still under active development and testing (e.g., [6]).

Validating such technology requires understanding the unusual events and corner cases (e.g., objects on the roadway, pedestrian crossing highway, deer standing next to the road, etc.) that one could encounter in billions of miles of driving. Such a large number of miles is needed since the goal is to achieve safety levels far above average human drivers and human drivers in the United States achieve almost 100 million vehicle miles traveled in between fatalities [7]. This motivates collecting a catalog of unusual driving events that represent challenging situations expected in billions miles of driving to accelerate the development of truly dependable level 4 and level 5 systems.

Most existing efforts collect driving data with a small fleet of tens to hundreds of highly instrumented vehicles that are continuously operated with test drivers, but it is challenging to cover billions of miles with such a small fleet. Road testing is therefore augmented with stress testing with corner cases on proving grounds and in simulation. This helps, but it remains uncertain whether a comprehensive set of corner cases was tested. Such a comprehensive set of unusual events and corner cases can be more easily obtained by scaling data collection to large numbers of minimally instrumented (camera-equipped) human driven vehicles, as previously advocated by BigRoad [8]. This, however, would still require identifying the unusual events in such a vast dataset to create test cases for proving grounds or simulators.

Manual inspection of collected data to flag unusual driving events is one possible solution, but will require plenty of extra effort, amplify privacy concerns, and increase storage and networking overhead for collecting all data. When vehicles are primarily human-driven, we cannot simply watch for human interventions as in current self-driving prototypes to identify corner cases that the system cannot handle well. Due to the many degrees of freedom in navigating a road simply detecting discrepancies between human drivers steering and speed input and self-driving system's choice is not reliable. Although there exists a large body of work on abnormal driving event detection [9–22], this work focus on detecting specific, known situations but cannot detect previously unknown unusual road events that are missing in the current set of test cases for automated vehicles. Therefore, automatically identifying unusual driving events remains a challenge.

To address this challenge, we propose an automatic unusual driving events identification system, which can detect unusual situations through in-vehicle algorithms and can easily be scaled for wide deployment. It identifies unusual situations through a two-pronged approach involving inertial monitoring of driver reactions and an autoencoder-based technique for detecting unusual video scenes. It detects sudden driver reactions (e.g.hard braking and swerving), since situations that challenge human drivers are also likely to be interesting test cases for automated vehicles. Since sudden driver reactions usually involve accelerations and angular speed, a three stage inertial sensing approach is proposed to detect unusual braking and swerving events. The rationale for the second video-based detection algorithm is that not all corner cases which may confuse self driving system will elicit a response from a human driver. Since previously unseen corner cases are more likely to differ from the training samples, we propose autoencoder-based approaches to identify these unfamiliar views, including a detector that can run on the vehicle side on low cost devices with 71.43% accuracy and a detector partially running in the cloud with 80.3% accuracy. The performance is evaluated based on 120 hours road driving data collected by about 10 drivers. To further illustrate the efficiency of our proposed system, the vehicle end unfamiliar view detector is implemented on an android phone and can process video frames at 17Hz, which is sufficient for flagging unusual scenes. Moreover, the unusual driving events detected by our approaches have been useful for re-training and improving the performance of the self driving model. This performance improvement on more complex road situations demonstrates the potential to accelerate the development of robust self driving systems with this unusual event detection framework. The contributions of this work can be summarized as follows:

- Introducing, to our knowledge, the first scalable unusual events identification and collection approach for self driving research and development, which employs human driven vehicles, instead of highly-instrumented vehicles.
- Developing an unusual events identification system to detect a variety of corner cases including both challenging situations for human drivers and unusual video for self driving models.
- Designing efficient unusual imagery detectors for low cost in-vehicle devices to limit the necessary video uploads, to conserve bandwidth and reduce privacy concerns.
- Analyzing more than 120 hours of driving data to evaluate the accuracy of unusual events identification and demonstrate the potential of these detected events to improve the performance of self-driving algorithms.

2 UNUSUAL EVENTS AND DESIGN OBJECTIVES

This paper focuses on unusual events and corner cases that can confound automated driving systems. For example, these include traffic scenarios that blind sensors, scenarios where key traffic participants are occluded or obscured by other objects, emerging objects that are difficult to identify, and unexpected movements by traffic participants. The ultimate goal is to understand the long tail of such traffic scenarios, meaning those that one would expect to encounter only after millions of miles of driving but that still need to be handled by the system to achieve a level of robustness that far exceeds human drivers. Specifically, this parper targets a subcategory of unusual events which can be inferred from sudden human driver reactions and unfamiliar views and are likely to affect the performance of self driving systems.

2.1 Current Approaches

Current approaches to collect data and test self driving systems on unusual events can be categorized as follows. **Public road testing** has been conducted by multiple companies through a small fleet of highly instrumented vehicles across different areas. Waymo has tested their vehicle on roads for 5 million miles[23] and Uber for 2 million miles[24]. **Closed course testing** is able to stage challenging driving cases (such as people jumping out of canvas bags or porta potties on the side of the road, skateboarders lying on their boards, thrown stacks of paper in front of sensors, etc.[23]) at the test facilities, like the Castle of Waymo and the Mcity of University of Michigan. Such testing allows recreate difficult situations more frequently than they occur during public road driving and can capture data with the sensor suite of an autonomous vehicle. **Simulation testing** allows simulated testing of corner cases, and further extended testing on the variations of such corner cases by tuning many different moving angles and different speeds of vehicles for example.

The robustness challenge: The accumulated public road miles still fall far short of the number of miles needed to demonstrate a lower fatality rate than above-average human drivers. Besides, the recent fatal accident during public road testing illustrates the challenge with correctly handling a broad set of road situations[25]. Current testing regimes seek to amplify this testing of 'known' corner cases through a combination of proving ground testing and simulations. While certainly useful, it remains unclear whether this extrapolation from known corner cases can lead to the desired level of robustness or whether new categories of unknown corner cases exist that will still need to be discovered in the next billions of miles of driving. The approach that this paper proposes is meant to complement these current techniques and addresses precisely this question.

2.2 Scaling Data Collection with BigRoad

As previously argued [8], current testing efforts could be accelerated through large-scale road-data collection involving hundreds of thousands of minimally instrumented, human-driven vehicles. While such vehicles may not generate sensor data that is directly usable in automated vehicles, it allows identification of new corner cases that can then be recreated on proving grounds and in simulation, as outlined above. Although [8] accurately records internal driver inputs (i.e., steering wheel angles, driving speed and acceleration) and external perceptions of road environments (i.e., road conditions and front-view video), it remains challenging to upload and store rich video data from such a large number of videos. In addition, drivers may have privacy concerns when video data is collected on their complete trips. This raises the question of whether it is possible to identify the small fraction of useful data that represents corner cases and challenging situations through preprocessing of the data inside the vehicle, which would allow uploading only these critical events.

2.3 Design Goals

Based on the drawback of current approaches and the scaling requirement discussion above, we identify the following key design goals for a scalable automated unusual driving event collection system.

- **Sense from human driven vehicles.** Rapid scaling to hundreds of thousands of vehicles requires making use of human-driven vehicles to cover a wider range of driving areas and cumulatively gather the rare happen unusual situations. The driving event identification system can therefore not rely on a full self-driving sensor suite but should expect minimal infrastructure and data source such as dashcams.
- **Minimize data uploads.** As the scale of data collection increases to hundreds of thousands of vehicles, tremendous wireless bandwidth usage would be required for uploading all video data into the cloud. Full uploads also increase privacy concerns. Thus, the system should be able to identify relevant events while most of the rich video data remains in the vehicle.
- **Build on off-the-shelf devices and stay within their computational limits.** Large scale data collection based on human driven vehicles will benefit from affordable off-the-shelf devices, which provide limited the computational power. Therefore, both time and space efficiency of the system should be optimized to guarantee a real time processing or near real-time computation with limited resources.

The design of our system will try to benefit automated driving systems as follows. For automatic driving components that only take front view videos and inertial readings as input, our system may be able to provide data that can be directly used as training or testing inputs. For systems which require other sensor inputs, such as radar or lidar, this data may not be available from human-driven vehicles but the detected unusual events can still help uncover traffic situations of interest and previously unknown corner cases. This information can then be used to define test cases and stage them on testing sites to collect the necessary data for other sensors to fully test the system. Besides, since our goal is to enable unusual driving events collection at very large scale, it is likely to help identify unusual driving events and develop test scenarios that will lead to more reliable automated driving systems.

3 SYSTEM OVERVIEW

The system enables scalability through aggressive in-vehicle filtering of sensor data, which reduces the volume of data that needs to be transferred over a network, ameliorates potential privacy concerns, and saves backend (human) analysis resources. The filtering removes all data that are not classified as unusual road situations. It detects unusual events using a two-pronged strategy that monitors (i) how a human driver, if present, reacts in terms of steering and braking and (ii) how different the camera inputs are from previously observed inputs.

In vehicles, the system assumes an on-board device with the sensing and computational capabilities of a high-end smartphone. Specifically, it requires a camera, accelerometer and gyroscope sensors, and processing capabilities to execute neural networks, and network connectivity to allow collection of data about unusual events and corner cases. Collected data can be stored and further analyzed in the backend infrastructure, as shown in figure 1.

The rationale for the first filtering approach, **Sudden Reaction Detection**, is that situations that surprise a human driver are more likely to also challenge an automated driving system than more standard driving situations. Since deployment on conventional human-driven vehicles would allow reaching the necessary scale much more quickly, the system can make use of detailed measurements of the human driver's sudden steering and braking reactions to road events.

However, one can also expect situation that does not elicit a reaction from an attentive human driver, but could confuse automated driving algorithms. This motivates the two-pronged approach where the system automatically seeks to identify unusual road imagery. This could be achieved through a set of classifiers that watch out for specific situations of interest such as a deer crossing or a stroller on the roadway. This would necessarily require an enumeration of expected unusual situations and not necessarily identify the unknown unusual road situations that are the motivation for this work. For this reason, the second filter, **Unfamiliar View Detection**, evaluates how different the image appears from previously observed road video. In order to efficiently compute this on an in-vehicle unit, the system employs an autoencoder for similarity detection. If computation is also available on cloud end, a fined-grained detection could be performed.

Note that these ideas could also extend to a more complete set of self-driving sensor inputs that include radar or lidar but that this design deliberately omits them to illustrate how such a system could be deployed on large numbers of vehicles without significant instrumentation cost.

4 SUDDEN REACTION DETECTION

Human drivers' reactions like hard braking or high speed swerving usually involve large accelerations and angular speed. Such features can be captured by the accelerometer and gyroscope of an Inertial Measurement Unit (IMU) available in many phones, cameras, and cars. The detector is triggered when the feature score exceeds a

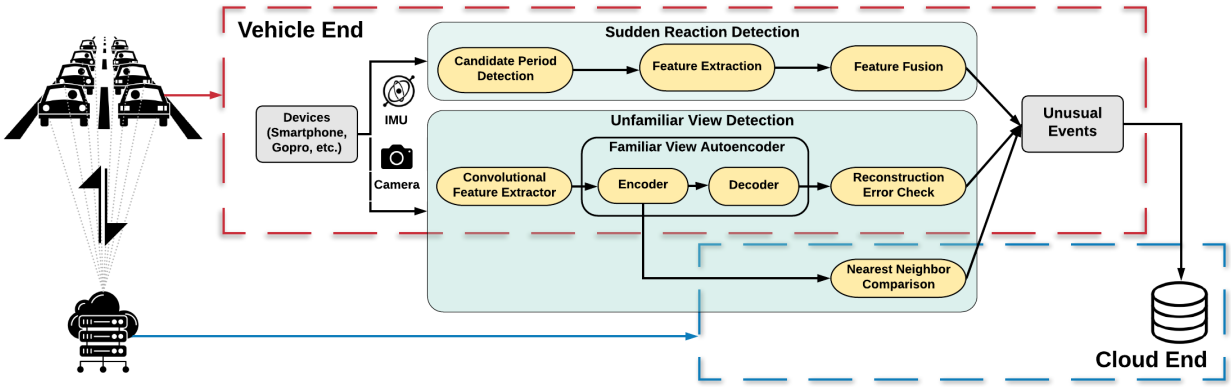
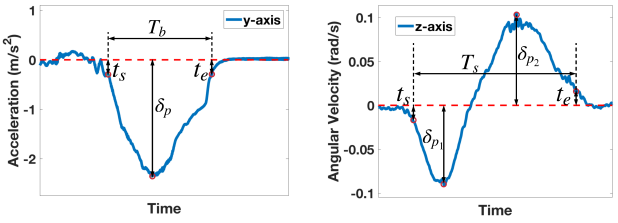


Figure 1: System overview of unusual events identification.



(a) Braking Event (b) Swerving Event

Figure 2: The accelerometer and gyroscope reading on sample braking and swerving event.

threshold, which can be chosen as a percentile of the feature value. Therefore, to identify such sudden reaction events, we propose a three-stage inertial sensing detection technique leveraging the IMU potentially available within vehicles. The three-stage detection mechanism includes *Candidate Period Detection*, *Feature Extraction* and *Feature Fusion*.

4.1 Candidate Period Detection

When unusual situations happen, human drivers may react with hard braking or swerving to avoid accidents. This motivates the first stage of our detection mechanism to identify candidate periods in terms of braking and swerving events, which show relatively high amplitudes on the accelerometer and the gyroscope.

Since the pose of the IMU within the vehicle is usually unknown, the system uses coordinate alignment algorithms to project the IMU's reading from its own coordinates system to the vehicle's coordinates system. As the vehicle will only sense gravity while stationary and will have a dominating acceleration component in the driving direction when accelerating, the vehicle's coordinate system can be determined from measurements during these situations [26]. Besides, we utilize a low pass filter to remove the noise from the raw IMU's reading caused by vehicle vibrations and bad road conditions.

Braking Event Detection. Generally speaking, when a driver brakes, a large acceleration can be observed in the opposite direction of the driving direction. Figure 2(a) shows the acceleration trace of a braking event, in which a negative spike can be observed due to braking. In order to more accurately capture the bumps that are actually caused by braking events, a peak detection method is applied first to find all the negative peaks of the accelerometer readings on the driving direction, whose value is defined as δ_p to quantify the amplitude of braking event as shown in figure 2(a). A threshold is used to remove noisy peaks such as the ones caused by road bump vibrations. Then the system searches forward and backward to find the starting point t_s and ending point t_e of this bump.

Swerving Events Detection. During swerving events, a driver usually first turns the steering wheel to one direction quickly and then turns back to the other direction. This action will result in two consecutive bumps in opposite directions of the gyroscope readings, as shown in Figure 2(b). Therefore, we capture such characteristics by thresholding peaks on gyroscope reading for peak detection, and then identifying swerving based on a short time interval between peaks in opposite directions. Similar to the brake detection, δ_p, t_s, t_e is defined as the amplitude of the higher peak, starting point, and ending point of a swerving event, respectively.

4.2 Feature Extraction

To identify the patterns of interest out of the candidate period set, we carefully select three feature extraction methods based on preliminary experiments and analysis. Based on the detected candidate periods, we first describe a **Strawman solution** using the *amplitude* of sensor readings as a feature to detect unusual events:

Amplitudes. Due to the large accelerations or gyroscope readings during unusual situations, using amplitudes of such reading as features seems to be an intuitive solution. Specifically, the sudden braking events and swerving events could be detected based on a threshold δ_p value.

However, this solution does not work well in practice because the majority of braking events with large acceleration and swerving events with large angular velocity are normal events like braking

when facing red traffic lights and swerving-like readings when changing lanes.

We therefore seek to emphasize more sudden events and propose **Derivative-based** as well as **Duration-based** features to further characterize detected events.

Derivatives. Since unusual events do not always come with high-amplitude readings, estimating the urgency or suddeness of a braking or swerving event is also helpful to determine unusual events. To this end, we calculate the derivative of acceleration to represent the urgency of a braking event. Similarly, calculating the derivatives of gyroscope reading during swerving events is used to identify urgent swerving events.

Duration. As a sudden braking or swerving event often happens in a short moment, we can also use the event duration to detect the urgency of unusual events. As shown in Figure 2, the duration of braking or swerving events are represented by T_b or T_s correspondingly, which is equal to the interval between t_s and t_e.

4.3 Feature Fusion

To effectively take advantage of all three features, we propose a feature fusion mechanism to combine the three extracted features (amplitude-based, duration-based and derivative-based) in order to increase the accuracy of identifying unusual events. This is motivated by preliminary results that showed that there is little overlap among the detected unusual events with any one of these features. We design an *accuracy driven weight assignment* method to assign weights to different features based on their detection accuracy. The principle underlying this method is illustrated in Equation 1.

$$f_{fusion} = \sum_i w_i f_i \qquad w_i = \frac{n_{f_i}}{\sum_i n_{f_i}} \tag{1}$$

The fused feature value (f_{fusion}) of a candidate period is equal to the sum of each feature value (f_i) multiplied by its weight (w_i). The value of each feature is normalized to adjust values measured on different scales to a notionally common scale. Specifically, We calculate the mean value and standard deviation of each feature, and use them to shift and scale each feature value. The weight of each feature (w_i) is calculated based on the detection accuracy. Specifically, for each feature, we extract the top 5% potential unusual events and calculate the detected unusual events for each feature (n_{f_i}). We divide n_{f_i} by the sum of detected unusual events for all features to calculate the corresponding weight (w_i). To design a general method that works for most real world scenarios, our fusion weights are generated from a large dataset which contains different scenarios including highways, local roads, night view roads, etc. With this method, the system assigns a higher weight for features with better detection accuracy, while assigning a lower weight to the one with worse detection accuracy.

5 UNFAMILIAR VIEW DETECTION

Although sudden reaction detection is able to identify events that surprise human drivers, one can still expect situations that do not elicit an attentive driver's reaction but confuse automated driving algorithms. This motivates identifying unusual road imagery by evaluating how different the image appears from previously observed self driving training images. Very different, distinct views may not have been sufficiently represented in training or test cases and lead to an increased risk of erroneous self-driving decisions. This can be intuitively implemented by calculating the Euclidean distances between new image samples and all samples used to train the automated driving system, but would require tremendous computational resources. To enable system scalability with limited in-vehicle computation, we propose two autoencoder-based unfamiliar view detectors, (i) a lightweight **in-vehicle detector** based on autoencoder's reconstruction error, and (ii) a **joint in-vehicle and cloud detector** based on the autoencoder's embedded vectors. Although autoencoders were applied to anomaly detection before [27, 28], to our knowledge, we propose the first design for driving video data with a novel architecture and loss function of the auto-encoder. This design of the autoencoder shows better performance than a naive auto-encoder application.

We develop this technique in the context of automated steering, since this is a key function of automated driving that usually heavily relies on camera data—data which can also be easily recorded from human driven vehicles [8]. Our unfamiliar view detectors will specifically identify corner cases for an end-to-end self steering systems. We expect though that the design of the detectors can also be extended to other self driving components that use camera data as input or where a comparable sensor readings can be collected from human-driven vehicles.

5.1 Autoencoder and Self-steering Background

Since our proposed unfamiliar view detectors are designed based on *autoencoder* for *end to end steering systems*, we will briefly introduce background on these two systems.

5.1.1 Autoencoder. An autoencoder is a neural network that is trained to encode the input in a set of low dimensional representations, which can be used to reconstruct an output that is nearly identical to its input. The groundtruth or labels of an autoencoder are just the input features themselves, therefore an autoencoder is also considered an unsupervised deep neural network. Internally, it has a hidden layer h that has a lower number of dimensions than the number of input dimensions, so that this hidden layer can be trained to describe an **embedded vector** used to represent the input. The network usually consists of two main parts, an encoder function $h = \phi(x)$ and a decoder that produce a reconstruction $r = \psi(h)$. An autoencoder is designed to simply learn the set $\psi(\phi(x)) = x$, but normally will not copy perfectly because the model size is usually restricted to allow them to copy only approximately, and to copy only input that resembles the training data. Because the model prioritizes the aspects of the input which should be copied, it often learns useful properties of the data.

$$L(x, x') = \|x - x'\|^2 = \|x - \psi(\phi(x))\|^2 \tag{2}$$

Autoencoders are usually trained to minimize **reconstruction errors**, the loss ($L(x, x')$), which is defined in equation 2. The reconstruction error is a metric to quantify the distance between the input and reconstructed input, and therefore can be considered as a difference indicator between the test samples and the training samples. Since autoencoders are trained to minimize the reconstruction errors when reconstructing training samples, a test sample which is similar to training set tends to have lower reconstruction error,

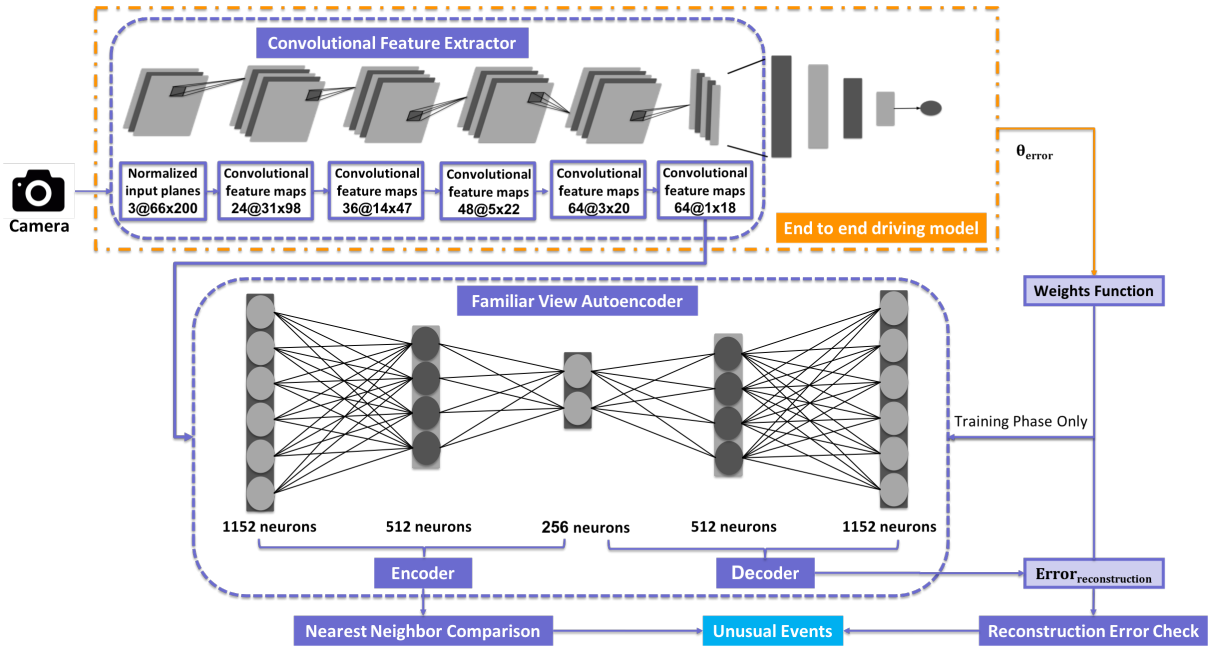


Figure 3: System overview of autoencoder based unusual event detection.

while a sample which is different from training samples will have higher reconstruction error.

5.1.2 End to end self steering system. An end to end self steering system maps raw pixels from a single front-facing camera directly to steering commands using a convolutional neural network. It is firstly proved to be powerful by Nvidia's demo [29] in lane keeping self driving tasks. Its neural network architecture [30] consists of 9 layers, including a normalization layer, 5 convolutional layers and 3 fully connected layers. The first layer of the network performs image normalization and the normalizer is hard-coded which is not adjusted in the learning process. Such normalization in the network will allow the normalization scheme to be altered with the network architecture and to be accelerated via GPU processing. The convolutional layers were designed based on different size of kernels and strides. Following five convolutional layers with three fully connected layers will lead to an output control value, which is the inverse turning radius. Based on this network architecture, a variety of end to end steering architectures are proposed. [31][32][33] utilize similar architecture with different layers and kernels to fit different input image dimensions. Other [34][35] bring in recurrent neural networks (RNN), and try to improve the performance by considering the sequential information from continuous frames. However, the majority of architectures share the common feature that starting with some convolutional layers as a feature extractor, and then use fully connected layers or RNNs to infer steering angles.

5.2 Design

To compare input images with previously observed training views, we propose a deep neural network architecture, which consists of a *convolutional feature extractor* to filter steering sensitive properties and a *familiar view autoencoder* to further learn the representation of well-trained samples in lower dimensionality. As shown in figure 3, both vehicle end and vehicle-cloud end unfamiliar view detectors are built based on this network architecture, but utilize different layer's output to achieve desired balance between accuracy and efficiency. **In-vehicle detector** leverages the outputs from the decoder of *familiar view autoencoder* to estimate the distance between an input sample and the whole training set by evaluating the reconstruction error. Since reconstruction error is calculated based on the learned distribution of the whole training set, in-vehicle detector will have relatively lower accuracy but much less computation cost as it only need perform one time pass of the neural network. While **joint in-vehicle and cloud detector** takes the outputs from the encoder of *familiar view autoencoder* to check the distance between the input sample and its nearest neighbours by encoding training samples into the same space. Such nearest neighbours comparison requires iterating pairwise distance evaluations on all the known samples, thus takes much more time especially with large amount training samples but can produce relatively higher accuracy.

As *convolutional feature extractor* and *familiar view autoencoder* are common modules of both detectors, we will introduce them in the first two subsections, and then discuss vehicle end and vehicle-cloud end unfamiliar view detectors' workflow respectively.

5.2.1 Convolutional Feature Extractor. Since front facing views include redundancy information that ends to end self steering system may not concern, we propose to apply a convolutional feature extractor to filter steering sensitive properties of the inputs before feeding inputs to the autoencoder. As the convolutional layers in deep neural networks usually serve as the feature extractor, we

implement an end to end driving network firstly and then utilize the convolutional layers as our convolutional feature extractor. As shown in figure 3, to obtain the convolutional feature extractor, we implement an end to end self steering deep neural network including 5 convolutional layers and 3 fully connected layers. Among the five convolutional layers, shallow layers which are close to input layers tend to keep more visual features, while depth layers will retain more abstract features for steering prediction. To further reduce the dimensionality and obtain abstract properties that may affect the final prediction, we choose to use all of the five convolutional layers as feature extractor. The end to end neural network is similar to Nvidia's architecture [30], whose strided convolutions are used in the first three convolutional layers with a 2×2 stride and a 5×5 kernel, and a non-strided convolution is performed with a 3×3 kernel size in the last two convolutional layers. But to get better performance on our dataset, we tuned our network to use inputs are from RGB space, take training labels with the steering angle in terms of radian, and trained only based on the center camera images. Based on our implementation, the input images will be cropped and resized to 66×200×3, and the output from convolutional feature extractor will be 64×18.

As introduced above, the convolutional feature extractor is obtained from the trained end to end driving neural network on the training set. Given a front-facing camera image X, it will be firstly processed by convolutional layers c, and then fed to fully connected layers f for steering angle prediction. Thus, if θ is the ground truth steering angle for current frame, the end to end steering prediction error is defined as equation 3, in which the overall end to end steering prediction process is $f(c(X))$ and predicted steering angle is $\hat{\theta}$.

$$\theta_{error} = \theta - \hat{\theta} = \theta - f(c(X)) \tag{3}$$

5.2.2 Familiar View Autoencoder. Convolutional feature extractor filters out steering sensitive properties, but the extracted feature vectors of training samples should not be treated equivalently. It is because the unusual road views that may lead to erroneous steering predictions, are usually unseen samples which are quite different from training samples or seen samples but with relatively large training error. Thus, if a sample is close to well-trained samples, which have lower steering prediction error θ_{error} among the training set, it will more likely to be handled well by the model and identified as a usual familiar view. Besides, extracted feature vectors are still in the order of thousand dimensions, so that a fewer dimensions encoding will be helpful for larger scale system employment and data collection. To address such challenges, we design and implement a familiar view autoencoder, which learns the representation of well-trained samples in lower dimensionality based on extracted feature vectors.

The familiar view autoencoder is designed to have four fully connected layers as shown in figure 3. The first two layers are trained as an encoder to map input from 1152 dimensions to 256 dimensions, while the last two layers are trained as decoder to restore the 1152 dimension vectors based on embedded 256 dimension vectors. Each fully connected layer includes a ReLU activation function to prevent overfitting. Note that, the hyperparameters of the autoencoder's architecture are set empirically to achieve a better performance on unusual views detection based on current data set scale.

Sample Weighted Loss Function. To distinguish the training samples with lower steering prediction errors from the ones with larger errors, we propose a sample weighted loss function to train the autoencoder. The sample weighted loss function is designed to assign more weights on well-trained samples, so that the autoencoder will focus more on the representation of such samples which have lower steering prediction error while learning. This motivates the encoding process of familiar view autoencoder to take more properties of well-trained samples into account. Therefore, we define the weight w for each training sample in equation 4.

$$w = \frac{1}{\log_{scalar_1}(\|\theta_{error}\| * scalar_2 + bias)} \tag{4}$$

The θ_{error} is the steering prediction error as defined in equation 3. $scalar_1$ and $scalar_2$ are the two scalars used to control the range and density of weights w. Both $scalar_1$ and $scalar_2$ are monotonically increasing with weight's value if other parameters are fixed. $scalar_1$ which is the base of the *log* function, decides the scale of the difference between small θ_{error} samples and large θ_{error} samples. The *bias* is the value we set to keep the value in the valid domain of *log* function, which is usually the same value of $scalar_1$. Therefore, the domain of w is (0,1], and the weight w value is monotonically decreasing with θ_{error} to guarantee larger θ_{error} will have less weight while smaller θ_{error} will have larger weight.

$$L(x, x') = w * \|c(x) - \psi(\phi(c(x)))\|^2 + L_2 \tag{5}$$

Based on the weights definition, the loss function of familiar view autoencoder is defined in equation 5. Compared with traditional autoencoder loss function as introduced in equation 2, different weights are applied on different samples according to a sample's θ_{error} during end to end steering prediction training. This will put penalties on the samples, which are not trained well on end to end driving systems, to make sure familiar view autoencoder's is trained learn more characteristics of well-trained samples. Besides, we also add a L_2 regularization term on the loss function, which is defined as the euclidean norm of all the trainable weights in the autoencoder. This regularization term will help prevent over fitting and force the weights to be sparse.

5.2.3 In-vehicle detector. To identify unusual imagery, in-vehicle detector performs unfamiliar view detection based on the reconstruction error. As introduced above, reconstruction error is a difference indicator between an input sample and training samples, so that unusual cases can be automatically identified by reconstruction error thresholding. Based on trained convolutional feature extractor and unfamiliar view autoencoder, reconstruction errors can be obtained through $\|c(x) - \psi(\phi(c(x)))\|^2$ by comparing the difference between an input sample and its corresponding reconstructed sample produced by the decoder. Since the calculation of reconstruction error is just one time pass of the neural network, in-vehicle detector is computational efficient and able to run on lost cost embedded devices like smartphones.

5.2.4 Joint in-vehicle and cloud detector. Although the in-vehicle detector can identify unfamiliar images efficiently, it is challenging to detect unusual cases which do not actually have similar cases

included in the training set but still relatively close to the majority of training samples. Such cases might produce lower reconstruction error but are hard to be handled by end to end driving systems. This motivates the joint in-vehicle and cloud detector to perform sample-wise distance comparison based on nearest neighbours instead of relying on the whole sample set distance evaluation according to reconstruction error. To enable the scalability for sample-wise comparison, joint in-vehicle and cloud detector takes outputs from the encoder of familiar view autoencoder in vehicle side and then performs k-nearest neighbours (kNN) checking over the cloud. Such scheme not only guarantees low computational cost in vehicle by running through part of the trained neural network, but also requires low network bandwidth since only encoded 256 dimensional floating vectors need to be transferred over the cloud. Therefore, to identify unusual imagery, an input image will be processed by convolutional feature extractor and the encoder of familiar view autoencoder to generate embedded vectors, then evaluated based on the mean distance of k nearest neighbors in the embedded space. The embedded vectors of the training samples will be pre-stored in the cloud to reduce the computation overhead.

6 EVALUATION

We evaluate our unusual event identification system with respect to (i) the accuracy of unusual driving event detection, (ii) the efficiency of the proposed unusual event detection system, and (iii) the usefulness of extracted unusual events.

6.1 Dataset Description

We use two different data sets to evaluate the performance of our unusual events detection system. The sudden reaction detection is evaluated with a 120-hour dataset collected in Los Angeles, CA. In this dataset, we use GoPros mounted at the bottom center under the windshield to record the full driver's front view videos with 1280×720 resolution at 30 *Fps*. The 200*Hz* accelerometer and 400*Hz* gyroscope readings are also recorded using the embedded inertial sensors in GoPros. The data collection is finished by ten different drivers under different road situations, including urban road, highway roads in both daytime and nighttime.

Since this dataset we collected does not have accurate driver's steering angle while driving, we use a dataset from Udacity end-to-end driving challenge [36] to train and evaluate our unfamiliar view detection method. Note that the driver's steering angle is necessary since it will be used twofold during the experiments. Firstly, it is used to train an end to end driving neural network, part of which will serve as the feature extractor. Then, it will also be used to evaluate whether the unusual views we identified will cause poor steering angle prediction performance in automated driving systems. The dataset contains 33,808 images with a resolution of 640×480 recorded by the center front facing camera including various driving conditions, such as different sunlight conditions, roads of different lanes, etc. The videos are recorded in 20*Fps*, and steering angles are logged in 50*Hz* and interpolated to be synchronized with the front view video frames.

6.2 Unusual Event Detection Accuracy

Our system can achieve high unusual event detection accuracy for both sudden reaction detection and unfamiliar view detection, as shown in Table 1. Setting 98 percentile of the fused feature value as the threshold, sudden reaction detection can achieve 53.16% and 63.16% accuracy for unusual braking events and swerving events respectively. Note that the detection accuracy is low because there are plenty of general sudden maneuvers performed by drivers, such as aggressive driving behaviors, sudden braking towards red traffic lights, etc., which were detected by our method but not labeled as unusual events in the evaluation. If the 98th percentile is also used for reconstruction error thresholding in unfamiliar view detection, the accuracy of in-vehicle detector is 71.43% and vehicle-cloud end is 80.30% [1]. A sample of detected unusual events are shown in figure 4. Figure 4(a) and 4(b) are detected by the driver's sudden reaction based on the IMU data, and figure 4(c) 4(d) 4(e) are captured by unfamiliar view detector since they are relatively unusual in the dataset. Detailed evaluation procedures are introduced in the subsections below.

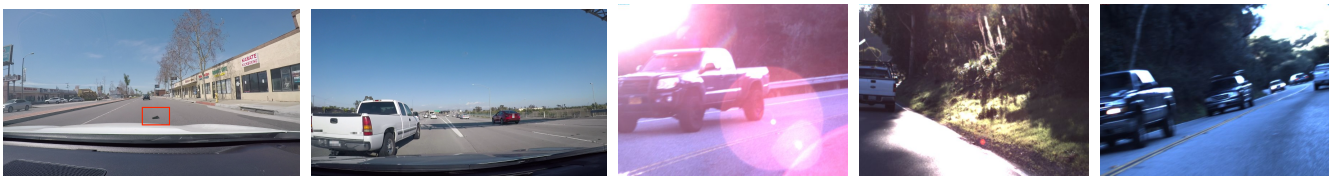
Methods	Proposed Method (%)	Baseline (%)
Sudden Reaction Detection for Braking Events	53.16	29.11
Sudden Reaction Detection for Swerving Events	63.16	31.58
Vehicle End Unfamiliar View Detection	71.43	64.53
Vehicle-Cloud End Unfamiliar View Detection	80.30	64.53

Table 1: Unusual event detection accuracy versus baseline/strawman solution accuracy for four methods.

6.2.1 Sudden Reaction Detection Evaluation. To demonstrate the performance of the sudden reaction detection method, we compare the proposed feature fusion detection method with the Strawman solution [2] as well as the approaches which filter unusual events based on the derivative and duration feature individually. Specifically, detected candidate periods are sorted in terms of the amplitude of accelerations, derivative of accelerations, duration of braking events, and the fused value of all the features respectively for braking event, according to Section 4. Then, unusual braking events are identified by thresholding the four metrics values to a percentile of threshold. Same process also applied for swerving detection based on gyroscope reading.

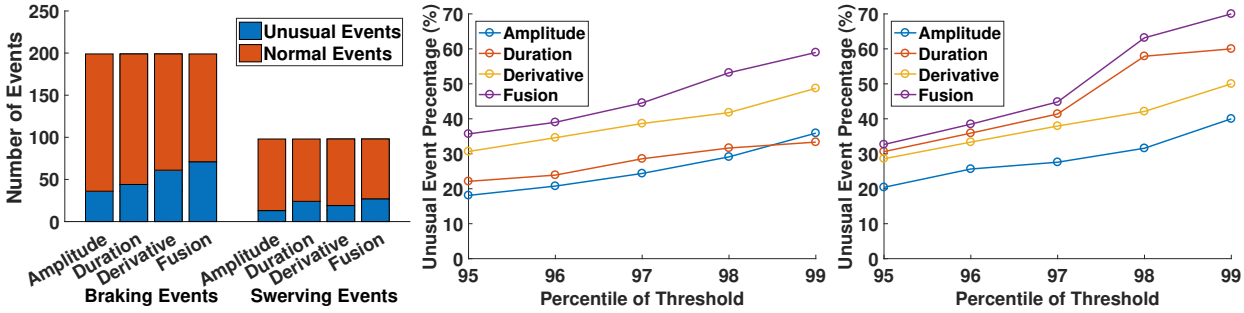
We find 3987 braking events and 981 swerving events in total over the 120 hours driving data. Thresholding the 95th percentile of braking feature values and the 90th percentile of swerving feature

[1] Unusual view detection will be determined as correct if its corresponding steering prediction error is larger than the median.
[2] The Strawman solution is used as baseline technique for comparison.



(a) Swerving to avoid a tire segment (b) Braking due to a cut-in vehicle (c) Special lightening condition (d) Complex roadside lightening condition (e) Unusual traffic and vehicle pose

Figure 4: Examples of detected unusual events.



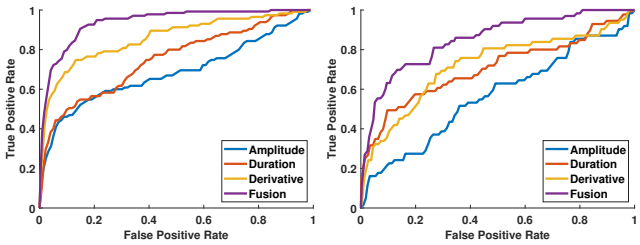
(a) Unusual Events Distribution (b) Braking Events Detection Accuracy (c) Swerving Events Detection Accuracy

Figure 5: Evaluation Results of Sudden Reaction Detection.

values [3] will filter out 199 braking events and 98 swerving events respectively, and the number of unusual events among them are shown in Figure 5(a). The unusual events are manually labeled in terms of whether the driver was surprised to perform sudden reactions for the unusual cases, but the usual sudden maneuvers such as aggressive driving behaviors, sudden braking towards red trafficlights, etc., are not included. Among 199 detected braking events, feature fusion approach extracts 71 unusual braking events, which surpasses the amplitude (36 detected), duration (44 detected) and derivative (61 detected) based approaches. Similarly, 27 sudden swerving events are detected out of 98 chosen swerving events with the feature fusion approach, which is better than other approaches (13, 24 and 19 sudden swerving events detected correspondingly). Since the length of unusual events captured by feature fusion is 0.25 hours out of 120 hours driving, our sudden reaction detection can largely save the bandwidth by only uploading detected unusual situations.

To further explore the relationship between sudden reaction detection accuracy and percentile of threshold value for each method, we plot the accuracy for braking events detection in Figure 5(b) and swerving events detection in Figure 5(c). We can observe that as the percentile of threshold increasing, the detection accuracy all of the four methods are getting higher. Among the four approaches, our proposed fusion method performs better than the other three methods. Since the dataset is too large to label ground truth for every event, we do not evaluate the recall of this approach in this paper.

[3]We chose 90 percentile as threshold for swerving events because smaller amounts of swerving events are included in the dataset.



(a) Braking Events ROC Curve (b) Swerving Events ROC Curve

Figure 6: Evaluation Results of Sudden Reaction Detection.

Besides the precision improvement, our system also achieves a high estimated recall compared to the baseline approaches. Due to the large amount of manual effort needed to label the large video dataset with ground truth, we limited labeling to 40% of events with at least one high feature value (a total of 1878 labeled events) and calculate recall over this dataset as an estimate for overall recall. Figure 6 shows the ROC curve of our braking events detection and sudden swerving events detection methods correspondingly. We can observe that the feature fusion approach achieves the highest area under the curve (AUC) value, which indicates the performance improvement compared to baseline approaches.

6.2.2 Unfamiliar View Detection Evaluation. In this section, we evaluate our unfamiliar view detection method by comparing the proposed two detectors (in-vehicle detector and joint in-vehicle and cloud detector) with a baseline approach. In particular, we randomly

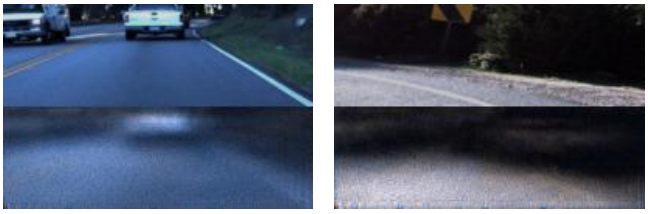


Figure 7: Sample input images and corresponding reconstructed images of baseline autoencoder.

sampled 80% of video frames from the Udacity dataset [36], and use them to train all of the three models. The rest of 20% of video frames are served as test set to evaluate the performance in terms of detection precision and recall.

Baseline Model. In order to show the advantage of our in-vehicle and joint in-vehicle and cloud detectors, we compare our system with a baseline model which uses the raw training images as input for the autoencoder. The baseline model also includes 4 layers: (i) a convolutional layer taking inputs images with 60×200×3 dimensions and apply a [5, 5] kernel, (ii) a fully connected layer further encode inputs to 256 dimensions. (iii) a fully connected layer to decode the inputs from 256 dimensions, and then (iv) a deconvolutional layer to reconstruct input images back to 60×200×3 dimensions also with a [5, 5] kernel. The baseline autoencoder is trained on the same training set as used for our proposed autoencoder. Figure 7 shows that the baseline autoencoder is able to reconstruct the input images to similar lossy images, since the embedded layer is much smaller than the original input. Thus, the reconstruction error of baseline autoecnoder can also indicate the distance between a test sample and the training set.

Precision. To evaluate unfamiliar view detectors, we define p in equation 6 as the detection accuracy, which represents the precision for binary classification task. Since unfamiliar view detection aims to identify the corner cases which are challenging for end to end self driving system, we determine a correct unusual detection if the sample's steering prediction error θ_{error} is larger than a threshold $thre_\theta$.

$$p = \frac{\text{number of detected events whose } \theta_{error} > thre_\theta}{\text{number of detected events}} \quad (6)$$

Figure 8 shows the detection accuracy p with respect to (i) different thresholds value to identify unusual events and (ii) different $thre_\theta$ to determine correct detections. As illustrated in the legend, the red, blue and yellow curves in the figure represent the detection accuracy p of three approaches respectively. x axis is defined as threshold value to identify unusual events, which is percentile of reconstruction error for baseline detector and in-vehicle detector, and the percentile of 20 nearest neighbour's distance for joint in-vehicle and cloud detector. The $thre_\theta$ is set to 50 percentile, 70 percentile, 90 percentile of the steering prediction error on test set to define different correct unusual events detections.

We can observe that all the curves have larger p while the threshold value of x axis increases. This verifies our motivation that driving views which have larger distances with previously observed views are more likely to get poor steering predictions. When x axis value is close to 0, the curves start from 0.5, 0.3, 0.1 respectively, which is basically random guess. However, as the distance threshold on x axis increasing to a larger value, such as 90 percentile, p boost to a very high accuracy. This illustrates that the events which have large distances namely very different from the training samples are more likely to confuse end to end driving system and get poor predictions.

For the baseline autoencoder, although it can also quantify the distances between test samples and training samples, the accuracy p is not as high as the two other approaches. It is because that the reconstruction error of baseline autoencoder's is estimated without discrimination, while our familiar view autoencoder uses the feature map from the convolutional feature extractor and focuses on the well-trained samples through weighted loss function. Thus, our approach drops the redundant information and emphasizes more on the information end to end driving model cares about. Besides, the droppings of p at the tails of baseline autoencoders curves also show that even though some images may visually different from training set, they still work for steering prediction model with similar feature vectors of training samples. For the comparison between in-vehicle detector and joint in-vehicle and cloud detector, the latter one is usually more accurate, since it performs fine-grained sample-wise calculation as discussed in section 5.2. Therefore, the experiments show that our unfamiliar view detectors can accurately detect unusual events which self driving models fail to perform good predictions.

Area under the curve. To further explore the performance of unfamiliar view detectors, we plot the Receiver Operating Characteristic (ROC) curves in figure 9 with 90 percentile steering prediction error to define unusual events. The area under the curve (AUC) of baseline, in-vehicle, joint in-vehicle and cloud detectors are 0.59, 0.64, 0.75 respectively. Although our proposed detectors work better than baseline detection, they still do not achieve a high AUC score. This is because that the trained end to end driving model could not work well on the views which are similar to previously seen training samples and causes a lower true positive rate. Therefore, it is important to consider the balance between a detector's ability to capture most of unusual events and the cost of network bandwidth for data transmission as high recall to extract more unusual events will bring more false positive detection. Under the context of large scale employment and data collection, the system will more likely to focus on extremely unusual cases with minimum bandwidth which prefers a higher precision.

6.3 Efficiency of Unusual Events Detection

As the unusual events identification system is built towards large scale employment based on off-the-shelf devices, we further explore the time and space efficiency of our method. Since sudden driver reaction detection through inertial data is computational inexpensive and does not require extra storage to perform the detection, we only focus on the efficiency evaluation on unfamiliar view detection in this section.

The space requirement of baseline autoencoder and in-vehicle detector are evaluated through the size of inference graph in tensorflow. To obtain the inference graph, we run through the freezing and optimizing scripts of tensorflow to organize the trained model with only inference required nodes. The size of inference graphs for

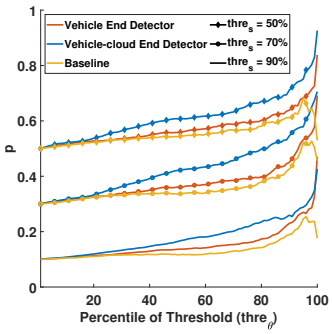


Figure 8: The relationship between detection precision and threshold value.

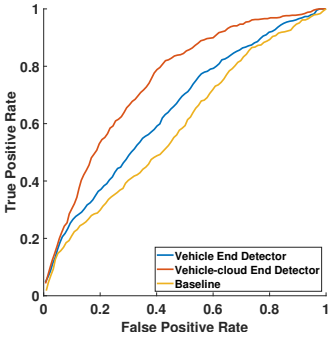


Figure 9: The ROC curve of baseline, in-vehicle, joint in-vehicle and cloud unfamiliar view detectors.

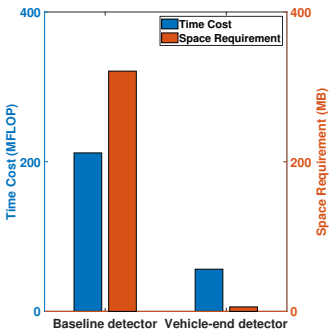


Figure 10: Efficiency comparison between baseline detector and in-vehicle detector.

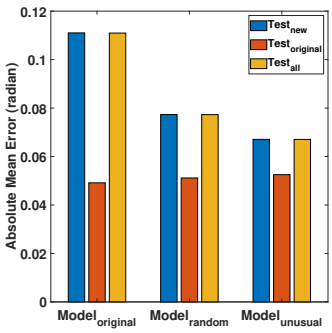


Figure 11: Model performance comparison on different training sets and test sets.

baseline autoencoder and in-vehicle detector are 321MB and 6MB respectively as shown in figure 10. We also evaluate the baseline autoencoder and in-vehicle detector based on the number of floating points operations (FLOP), which can be used to roughly estimate the time cost of models. The FLOP of both models are counted by the benchmark script provided by tensorflow. As shown in the figure 10, baseline autoencoder model include 211.74 MFLOPs, which is much higher than online in-vehicle detector's 56.39 MFLOPs. In addition, we also run the in-vehicle detector on a Nexus 5X Android smartphone, and each frame takes around 58ms to process, which showing a 17Hz inference capability. Therefore, the in-vehicle detector has higher efficiency in terms of both time cost and space requirement, and is applicable for real-time employment on off-the-shelf smartphones. Regarding joint in-vehicle and cloud detector, it will be more computational expensive since its complexity is proportional to the number of training samples, and requires significant space due to the storage of training sample vectors.

6.4 Usefulness of Unusual Events

In this section, we show that the unusual events extracted by our unfamiliar view detection method can increase the performance of end to end driving system for more complex situations. Specifically, we add the detected unfamiliar events into the training set of the end to end steering system and explore the performance on different test sets.

First, based on the dataset from Udacity as introduced in section 6.1, we use 80% of the samples to train an end to end driving model $\text{Model}_{\text{original}}$ and the rest of 20% samples as the test set $\text{Test}_{\text{original}}$. To get new training samples, we use another dataset also provided by Udacity[37], which is collected with the same experiment setup, but mostly driving on roads with heavier traffic. Among this dataset, first 10,000 samples are picked as a sample pool and the next 1000 samples are used as the new test set $Test_{new}$. The new training sets include original training samples as well as 1000 new samples, which are selected from the sample pool by two different strategies. One strategy randomly selects new samples from the sample pool, and the other one utilizes in-vehicle detector to pick 1000 unfamiliar views. We call the model trained with the first strategy $\text{Model}_{\text{random}}$, and the second model $\text{Model}_{\text{unusual}}$. To evaluate the overall performance on both datasets, we combine the samples from $\text{Test}_{\text{original}}$ and Test_{new} as Test_{all}.

All of the three different models are evaluated on three different test sets in terms of the absolute mean error of steering prediction, and the performance is shown in Figure 11. For the new test set Test_{new}, $\text{Model}_{\text{original}}$ performs worst since it had never trained on those roads. $\text{Model}_{\text{random}}$ shows better performance than $\text{Model}_{\text{original}}$, since the new training set contains randomly selected views with the same road condition. $Model_{unusual}$ further shows performance gain, which illustrates that the unfamiliar cases identified by our method is more representative of the new dataset and improve the model performance on the new route more efficiently. For the original test set $\text{Test}_{\text{original}}$, $\text{Model}_{\text{unusual}}$ and $\text{Model}_{\text{random}}$'s accuracy are slightly lower than $\text{Model}_{\text{original}}$, since they are trained to handle more cases, which create a neglectable performance compensation on $\text{Test}_{\text{original}}$. From the results based on the overall test set Test_{all}, $\text{Model}_{\text{unusual}}$ still shows the best performance than the other two, which shows the detected unusual events are able to increase model's overall performance and robustness on different roads and traffic conditions.

7 RELATED WORK

Detecting unusual events is of great importance to driving assistant system developments, since the analysis of such corner cases will be helpful to improve current systems [4, 30–35, 38]. There has been extensive research on vehicle sensing and unusual events detection based on inertial measurements. [9] utilizes the IMU of smartphones to detect and differentiate vehicle maneuvers, but only focusing on steering related maneuvers. [10] also take inertial reading from smartphones, and then perform aggressive driving style recognition based on dynamic time warping. However, such algorithm could only detect known aggressive driving patterns, while not diverse human driver reactions which are naturally performed during emergency periods. [11] is close to our vision that identifies unusual events based on inertial measurements, but the proposed solution heavily relies on large number of thresholds which are defined based on the amplitude of sensor readings. Besides, there

are some other driving sensing techniques built upon IMU, but towards different goals, such as [39] for drunk detection, [26, 40, 41] for driver determination, and [42, 43] for driver tracking, etc.

As visual imagery captures vehicle's surrounding condition and moving pattern, vision based techniques are also used to detect unusual driving situations. [12] [13] [14] leverage the spatial and temporal information from videos to detect high level anomaly patterns by comparing with previously observed views. While [15] and [16] specialized on concrete type of road emergency detection such as a collision based on vehicle tracking. However, these approaches rely on a fixed point of view, such as video feeding from surveillance camera, thus are not suitable for on-board imagery processing. [17] and [18] are able to utilize on-board cameras to capture unusual cases, but only cover on a subset of challenging situations such as abnormal pedestrian movements and unclear drivable roads.

Another body of work focuses on exploring the weakness and vulnerability of current self driving systems. [19][20][21] could be used to generate the synthetic cases which systems tend to have erroneous behaviors, but such situations can not extend the coverage of real unusual cases. [22] explored the robustness of traffic sign recognition model under physical world attacks. However, a detection mechanism for such attacks was not covered.

8 DISCUSSION

As more unusual driving events get collected by our proposed system, the previously obtained training set need to be extended to cover more diverse cases. This will require the familiar view autoencoder to be re-trained with more number of observations. Depending on the order of increased training samples, the familiar view autoencoder may have to incorporate more neurons and more layers to achieve equivalent detection performance. Such updates on the network architecture will increase the computational cost of the familiar view autoencoder with larger trained network size and longer processing time. To mitigate the computation requirement on the vehicle, the decoder and reconstruction error check module of the detector could be shifted to the cloud. Since the computational cost of the autoencoder is not proportional to the number of samples, the shifting is only necessary when the size of the training set is very large.

The low cost design of our unusual identification approach could enable large scale data collection potentially through crowd sourcing, but privacy could be a concern. When unusual events are recorded, drivers' privacy may be compromised by the imagery and inertial sensor readings, such as visited location which can be inferred from the front view camera, and driving decisions for an emergency event sensed from inertial sensor. Therefore, the vehicle end software may need additional functionality to buffer the detected unusual events and only upload the events if the driver confirms that there is no privacy concern. Besides, as the scale of crowd sourcing increases, the heterogeneity of mobile devices may become another challenge. For the inertial based sudden reaction events, there may be differences in sample frequency and sensitivity but we expect this to cause relatively low accuracy loss. It would be more challenging for the unfamiliar view detection to run on heterogeneous cameras with different resolutions and intrinsic parameters but this can be addressed in future work.

9 CONCLUSION

In this paper, we aim to automatically identify unusual driving events to allow scaling the collection of driving data and corner cases to a much larger fleet of human-driven vehicles without requiring upload or human review of all data. The proposed system is able to capture various unusual circumstances, including hazardous event like sudden braking and swerving events through a three-stage process involving inertial sensing and detecting outliers of current trained self-driving systems based on autoencoder deep neural network. The evaluation is based on more than 120 hours of real road driving data and shows that it outperforms baseline methods on unusual event with 82% accuracy improvement over baseline on sudden reaction detection and above 71% accuracy on unfamiliar views identification. The event identification process requires only inertial measurements and front view driving videos, allowing collection of data from smartphones or dashcams. The computational cost is only subject to the complexity of our pre-trained neural network. Thus, the light-weight design and minimal infrastructure requirement of this approach will allow large-scale unusual driving events identification and collection. We hope that an extensive dataset of driving corner cases collected with this approach would provide a better understanding of potential limitations of current systems and accelerate the development of robust automated driving technology.

ACKNOWLEDGMENTS

We sincerely thank anonymous reviewers for their valuable comments. This material is based in part upon work supported by the National Science Foundation under Grant Nos. CNS-1329939.

REFERENCES

[1] Tom Krisher and Joan Lowy. Tesla driver killed in crash while using car's 'autopilot'. *ASSOCIATED PRESS(June 30, 2016)*.
[2] Chris Urmson. Google Self-Driving Car Project. https://www.transportation.gov/sites/dot.gov/files/docs/AV%20policy%20guidance%20PDF.pdf. Talk at SXSW Interactive 2016.
[3] National Highway Traffic Safety Administration. Federal Automated Vehicles Policy. https://www.transportation.gov/sites/dot.gov/files/docs/AV%20policy%20guidance%20PDF.pdf.
[4] Tesla autopilot. https://en.wikipedia.org/wiki/Tesla_Autopilot.
[5] 2019 audi a8 level 3 autonomy first-drive: Chasing the perfect 'jam'. *Slash Gear (November 7, 2017)*.
[6] Justin Hughes. Waymo is already running self-driving cars with no one behind the wheel. *The Drive (November 7, 2017)*.
[7] National Highway Traffic Safety Administration. Fatality Analysis Reporting System. https://www.transportation.gov/sites/dot.gov/files/docs/AV%20policy%20guidance%20PDF.pdf. Talk at SXSW Interactive 2016.
[8] Luyang Liu, Hongyu Li, Jian Liu, Cagdas Karatas, Yan Wang, Marco Gruteser, Yingying Chen, and Richard P Martin. Bigroad: Scaling road data acquisition for dependable self-driving. In *Proceedings of MobiSys 2017*, pages 371–384. ACM, 2017.
[9] Dongyao Chen, Kyong-Tak Cho, Sihui Han, Zhizhuo Jin, and Kang G Shin. Invisible sensing of vehicle steering with smartphones. In *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*, pages 1–13. ACM, 2015.
[10] Derick A Johnson and Mohan M Trivedi. Driving style recognition using a smartphone as a sensor platform. In *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, pages 1609–1615. IEEE, 2011.
[11] Cheol Oh, Eunbi Jeong, Kyungpyo Kang, and Younsoo Kang. Hazardous driving event detection and analysis system in vehicular networks: Methodology and field

implementation. *Transportation Research Record: Journal of the Transportation Research Board*, (2381):9–19, 2013.

[12] Fan Jiang, Junsong Yuan, Sotirios A Tsaftaris, and Aggelos K Katsaggelos. Anomalous video event detection using spatiotemporal context. *Computer Vision and Image Understanding*, 115(3):323–333, 2011.

[13] Waqas Sultani and Jin Young Choi. Abnormal traffic detection using intelligent driver model. In *Pattern Recognition (ICPR), 2010 20th International Conference on*, pages 324–327. IEEE, 2010.

[14] Venkatesh Saligrama and Zhu Chen. Video anomaly detection based on local statistical aggregates. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2112–2119. IEEE, 2012.

[15] Kimin Yun, Hawook Jeong, Kwang Moo Yi, Soo Wan Kim, and Jin Young Choi. Motion interaction field for accident detection in traffic surveillance video. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 3062–3067. IEEE, 2014.

[16] Shunsuke Kamijo, Yasuyuki Matsushita, Katsushi Ikeuchi, and Masao Sakauchi. Traffic monitoring and accident detection at intersections. *IEEE transactions on Intelligent transportation systems*, 1(2):108–118, 2000.

[17] Dariu M Gavrila and Stefan Munder. Multi-cue pedestrian detection and tracking from a moving vehicle. *International journal of computer vision*, 73(1):41–59, 2007.

[18] Chunzhao Guo, Seiichi Mita, and David McAllester. Robust road detection and tracking in challenging scenarios based on markov random fields with unsupervised learning. *IEEE Transactions on intelligent transportation systems*, 13(3):1338–1354, 2012.

[19] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. *arXiv preprint arXiv:1708.08559*, 2017.

[20] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[21] Udacity. Udacity: A self-driving car simulator built with unity, Feb 2018.

[22] Ivan Evtimov, Kevin Eykholt, Earlence Fernandes, Tadayoshi Kohno, Bo Li, Atul Prakash, Amir Rahmati, and Dawn Song. Robust physical-world attacks on deep learning models. *arXiv preprint arXiv:1707.08945*, 1, 2017.

[23] Safety report – waymo. 2017.

[24] Uber's self-driving cars hit 2 million miles as program regains momentum. 2017.

[25] Self-driving uber vehicle strikes and kills pedestrian. 2018.

[26] Yan Wang, Jie Yang, Hongbo Liu, Yingying Chen, Marco Gruteser, and Richard P. Martin. Sensing vehicle dynamics for determining driver phone use. In *Proceeding of the 11th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '13, pages 41–54, New York, NY, USA, 2013. ACM.

[27] Dan Xu, Elisa Ricci, Yan Yan, Jingkuan Song, and Nicu Sebe. Learning deep representations of appearance and motion for anomalous event detection. *arXiv preprint arXiv:1510.01553*, 2015.

[28] Raghavendra Chalapathy, Aditya Krishna Menon, and Sanjay Chawla. Anomaly detection using one-class neural networks. *arXiv preprint arXiv:1802.06360*, 2018.

[29] Nvidia. Nvidia autonomous car. https://www.youtube.com/watch?v=qhUvQiKec2U, May 2016.

[30] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[31] Comma AI research. https://github.com/commaai/res, 2016.

[32] Cg23. https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/cg23, 2017.

[33] Rambo. https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/rambo, 2017.

[34] Chauffeur. https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/chauffeur, 2017.

[35] Komanda. https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/komanda, 2017.

[36] Udacity self driving car challenge dataset ch2_002. https://github.com/udacity/self-driving-car/tree/master/datasets/CH2.

[37] Udacity el camino training data for self driving car challenge. http://academictorrents.com/details/e9b47deb3391e33df794e5ec4399d38ef8767c07.

[38] Autumn. https://github.com/udacity/self-driving-car/tree/master/steering-models/community-models/autumn, 2017.

[39] Jiangpeng Dai, Jin Teng, Xiaole Bai, Zhaohui Shen, and Dong Xuan. Mobile phone based drunk driving detection. In *Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2010 4th International Conference on-NO PERMISSIONS*, pages 1–8. IEEE, 2010.

[40] Yan Wang, Yingying Jennifer Chen, Jie Yang, Marco Gruteser, Richard P Martin, Hongbo Liu, Luyang Liu, and Cagdas Karatas. Determining driver phone use by exploiting smartphone integrated sensors. *IEEE Transactions on Mobile Computing*, 15(8):1965–1981, 2016.

[41] Luyang Liu, Cagdas Karatas, Hongyu Li, Sheng Tan, Marco Gruteser, Jie Yang, Yingying Chen, and Richard P Martin. Toward detection of unsafe driving with wearables. In *Proceedings of the 2015 workshop on Wearable Systems and Applications*, pages 27–32. ACM, 2015.

[42] Cagdas Karatas, Luyang Liu, Hongyu Li, Jian Liu, Yan Wang, Sheng Tan, Jie Yang, Yingying Chen, Marco Gruteser, and Richard Martin. Leveraging wearables for steering and driver tracking. In *Computer Communications, IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on*, pages 1–9. IEEE, 2016.

[43] Cagdas Karatas, Luyang Liu, Marco Gruteser, and Richard Howard. Single-sensor motion and orientation tracking in a moving vehicle. In *2018 15th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. IEEE, 2018.