Global Scientific JOURNALS

GSJ: Volume 11, Issue 8, August 2023, Online: ISSN 2320-9186

www.globalscientificjournal.com

Autonomous vehicle fault Detection and Isolation Model Using Stochastic Gradient Descent

Okolai, B.D[1], Onyejegbu, L.N[2] and Onuodu, F.E[3]

[1]*Department of Computer Science, Niger Delta University, Amassoma, Bayelsa state, Nigeria*

[1,2,3]*Department of Computer Science, University of Port Harcourt, Choba, Nigeria*

ABSTRACT

It takes both time and resources to gather sizable autonomous vehicle dataset for stochastic gradient modeling. The capabilities of stochastic gradient descent (SGD) algorithm and pipeline in machine learning have been investigated in a number of researches. Several techniques used smaller samples of sensor data, which may have been adequate but wasn't always the best for employing ML pipelining classifier. The pipeline uses SGD ML classifier with an applicable parameterization to categorize training data samples in order to improve classification and fault detection accuracy. This study employed pipeline sensor data fusion architecture with stochastic gradient descent to address potential sensor signal failure types in autonomous vehicles. The suggested architecture uses a SGD classifier for sensor signal diagnosis, detection, and isolation in order to overcome the challenges in the environment of an autonomous vehicle. We are integrating our knowledge in stochastic modeling to create a reliable ML pipeline framework that categorizes various sorts of car sensor signal faults. The pipeline model was designed to validate input and output data of each stage, preventing sensor faults from emerging and boosting the pipeline model's dependability. We concentrated on defining a thorough fault isolation system model that handles a variety of fault types (normal, hard over, drift and spike). The model strength lies in its ability to unearth hidden patterns and establish a connection between car features with their historical sensor signal data. The SGD produced 95.27% while pipeline recorded 99.43% accuracy rate as the best. The pipeline framework enhanced the efficiency and reliability of the model development process while reducing the likelihood of faults and promoting high standards.

1.1 INTRODUNCTION

Autonomous vehicles, which are often known as self-driven cars, have driving capabilities that range from level 0 to level 5 and requiring minimal or no human interaction to operate (Jeong et al., 2019). Simple performance issues or security concerns could have significant impact because of their intelligence, leading to unanticipated failure with significant financial losses and casualties if not identified in time (Liu et al., 2021). Modern cities and streets have become home to autonomous vehicles since they utilize cutting-edge technologies including machine learning (ML)

algorithms and control systems. These techniques rely mainly on data gathered from numerous sensors. The importance of sensor fault surveillance and detection is highlighted by the possibility that vehicle sensor failure and the resulting error propagation could be detrimental to the dependability and safety of autonomous cars. The ultimate goal of a sensor health monitoring system is to anticipate sensor reliability and performance and make decisions in response to that prediction, rather than only detecting, isolating, and identifying faulty sensor signals (Fahim & Sillitti 2019). A suitable fault diagnostic method for autonomous vehicles with sophisticated electrical architecture is required. This strategy includes faulty sensor signal detection, which identifies systemic faults. Fault isolation identifies which sensor is defective, fault identification explains why the sensor malfunctioned, and sensor health prediction depicts the sensors' present and potential future states.

1.2 Types of signal faults in autonomous vehicles

The three categories of sensor signal failures proposed by Al-Zeyadi et al. (2020) are sensor fault, actuator fault, and process fault. The manufacturing facilities and post-sale services in the automotive sector both heavily utilize AI/ML technical advancements. The fault detecting system is one of the services provided by the automotive industry. Continuous and intermittent failures are two of the different types of sensor device fault patterns that Balaban et al. (2009) found. Continuous faults are ones that remain in the system, as opposed to intermittent faults, which happen only intermittently. Intermittent faults are categorized by Jan et al. (2017) as drift, hard-over erratic, and spike sensor faults. When a signal steadily deviates (linearly) from its true value, a drift fault occurs. When a sensor recovers a value that is outside of its measurable range and rises quickly to saturation, a hard-over fault happens. A sensor's data becomes significantly noisier, which causes an unpredictable fault. Over time, the amplitude of the signal variance can grow around the correct value. When the signal value briefly increases, spike faults happen. When the most recent cycle value of x(t) depending on the time provided, it is known as a delay time fault. According to Shen et al., (2020), a single DL approach might not be sufficient to detect and isolate vehicle sensor signal faults detection and isolation requirements. Min et al. (2023) identified spike, continuous, gradual drift, bias, and miss sensor signal faults types in autonomous vehicles. Many methods relied on smaller amounts of sensor data, which although sometimes sufficient for ML pipelining classifier with specific requirements and this is not always the ideal option. The collection of a significant autonomous vehicle dataset for stochastic gradient modeling requires both time and resources. However, little research has been done on their relative efficacy in regard to various heterogeneity dataset types. The pipeline employs an SGD ML classifier with the appropriate parameterization to categorize training data samples and increase classification and fault detection accuracy.

The aim is to build an efficient autonomous vehicle fault Detection and Isolation Model using stochastic gradient descent. The SGD and pipeline classifiers were used with the proper parameterization to help categorize training data samples. This improved classification and fault detection accuracy. We focused in developing a comprehensive fault isolation system model that can manage several failure types (hard over, drift, and spike) and defined a thorough fault isolation system model capable of handling variety of fault types (normal, hard over, drift and spike). The model strength lies in its ability to unearth hidden patterns and establish a connection between car features with their historical sensor signal data.

The paper is organized as follows: Section 1 provided an introduction; Section 2 offers a brief assessment of prior approaches related to the topic and the gap in studying the proposed model; Section 3 introduces the model's materials and methods; Section 4 covers the results and a thorough discussion of the results; and Section 5 provides the paper's conclusion.

2.0 LITERATURE REVIEW

Realpe, et al., (2015) employed a SVM architecture to diagnose and detect faulty signals in vehicle sensors using a suitable dataset. They verified that the suggested design was able to recognize both soft and hard errors from a specific sensor. Wang et al. (2022) adopted a sparse SGD optimal approach in order to boost the batch gradient updating methods. The network parameter was modified with the SGD parameters after the batch gradients were grouped using a distributed density-based clustering. Bello-Salau et al. (2020) presented a novel method for swiftly distinguishing potholes and bumps from noisy data recorded by an accelerometer. Their method works by filtering erratic readings obtained from an accelerometer fitted on the car. A wavelet transformation-based filter was used to separate the signals into several sizes. The suggested method detects and classifies road flaws with a high level of precision, accuracy, and a low occurrence of false alarms. Kim et al. (2015) offered a gateway structure based on the controller area network (CAN) for in-vehicle networks (IVNs). Security, multiple routing configuration, dynamic routing update, network management (NM), diagnostic routing, and parallel reprogramming are among the model functionalities made available by the gateway framework. Pradhan and Gupta (2017) proposed a failure detection technique for rolling element bearings and uses time-domain vibration signal analysis. They measured the severity of the defect using a fault scalar indicator whose value was extracted from the time domain signal by the increasing size. Davishi et al. (2022) used an MLPNN machine learning structures capable of recognizing and recreating normal sensor signal behavior in order to replace the faulty signals in the system. The purpose of this model's construction was to identify and isolate car sensor signal faults. Further investigation was required to utilize RNNs, CNNs, and reinforcement learning architectures with interpretable or XAI algorithms in order to improve explanations and gain trust from users. Ciaburro (2022) employed several ML models such as SVM, CNN, RNN and deep generative algorithm to detect popularly known sensor failure fault types in production and maintenance process. They first offered the background research needed to grasp the procedure, after which they examined a number of representative works that had produced the best outcomes for detecting faults in the industrial machine. Obodoeze et al. (2018), proposed ML technologies with effective digital fault tracking functionality to identify vehicle fault types. They suggested using computer programs for recording defects and enable users to transmit the report back to the expert through email or SMS for quick response. The suggested method systematically uncovered different fault types while minimizing the requirements for human work. It was useful in lowering the cost of vehicle maintenance and troubleshooting. Marzat et al. (2012) presented a sliding mode of observers and control model-based sensor failure detection and isolation technique. It was nearly impossible to accurately obtain some of the vehicle parameters given that the vehicle has a highly linked and complex nonlinear system and that the developed vehicle model was subjected to several situations of uncertainty. They recommended that, the challenge of signal isolation and detection could potentially be improved by using a handful of non-model-based strategies. Yurii

et al. (2017) used an efficient model-based fault isolation and detection technique was used by and it primarily depends on linear system modeling. These strategies could not produce the desired results for nonlinear systems. Nevertheless, due to their temporal complexity, model-based techniques actually possess a higher accuracy. Similar model-based strategies are still commonly used to deal with real technological problems. Chamseddine and Noura (2008) proposed vehicle fault detection and isolation system in order to identify sensor failures in automotive systems. They needed extra sensors, including displacement sensors, to replace the conventional sensor arrangements. Shen, et al. (2020) used a parity-check approach to identify issues with car sensor signals. Their model's low detection accuracy was caused by miss categorization errors, subpar interfaces and unpredictability. Glowacz (2018) developed an inventive method for identifying vehicle sensor faults using KNN-based model for car fault classification and recognition. The fault detection system that used acoustics had low operating costs and a high degree of dependability. These associated study results provided significant understandings into the area of diagnosing and isolating vehicles sensor signal faults. This model was created to only take into account one kind of fault at a time, ignoring other kinds of sensor faults. Gauerhof et al. (2018) produced angles of steering as responses using a CNN provided with a dataset of live camera photographs. The aforementioned learning framework brings up new issues which have to be addressed in relation to the produced results' dependability, safety, and interpretation by humans. The suggested hierarchical pipeline systems include strategies for independent localization, understanding, planning, and control. The data collected from unprocessed sensors is transferred along the software pipelines of a self-driving vehicle for recognizing objects and localization, after which it gets passed on for control and strategic planning.

3.0 MATHERIALS AND METHODS

This session discusses the potential of the suggested approach to identify and classify various sensor and communication failure types, as well as the assessment results of the SGD and pipeline classifier on the data from the test. The proposed SGD architecture has been refined to help determine the issues of pipeline approach, fault detection, and classification. The models and decisions made in order to implement the suggested architecture are examined below:

Datasets: The dataset used in this program is collected from the IEEE Data Port, which is accessed through the UCI Machine Learning Repository website at "https://ieee-dataport.org/documents/gps-data-collected-xinda-autonomous-vehicle-changan-university". It contains attributes like UID, ID, sensor signal value, brake hydraulic pressure, temperature, velocity, target, and others. The suggested collection offers a substantial dataset for training models, with around 2000 testing and 8000 training sets making up 10,000 dataset. The sensor signal faults in this 10-feature sets are divided into five target classes (Normal, hard over, drift and spike faults).

SGD optimization approaches have a non-deterministic process of learning. Such ML methods have a high degree of generalizability due to their random initialization of parameters for loss reduction. This property renders SGD predictions non-deterministic. Since ML is non-deterministic, the results are rather unclear for inputs that fall under the same set of target classes.

The proposed SGD and pipeline implementation for detecting sensor signal faults in autonomous vehicles is described below.

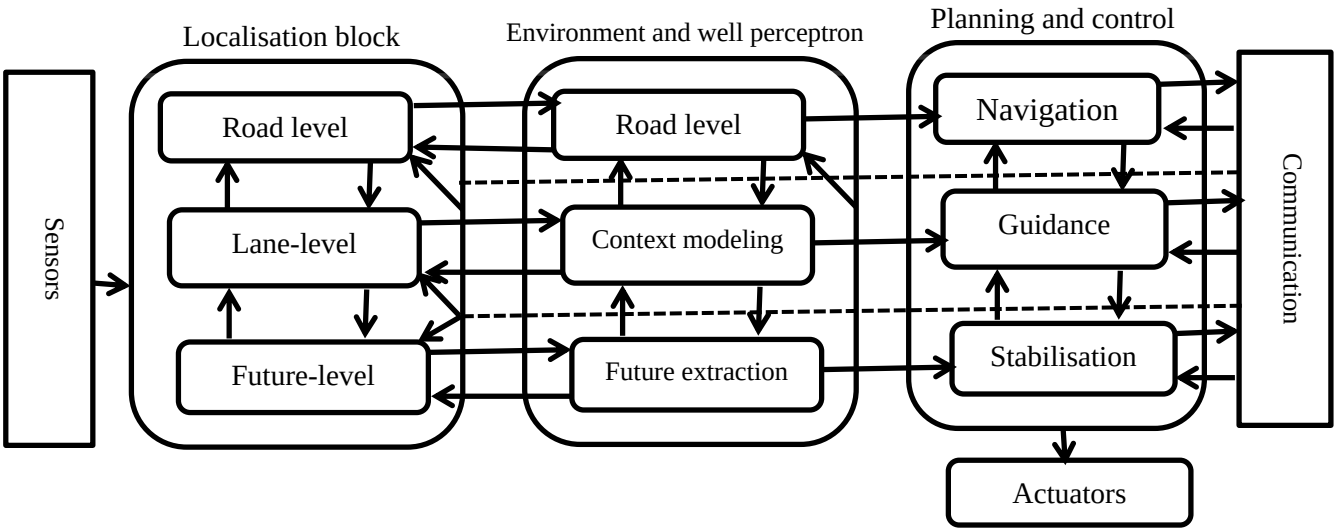

Figure 1: Study strategy for autonomous vehicle pipeline

Three basic abstractions make up the architecture: (a) localization; (b) environment and perception; (c) planning and control. The pipeline's localization and perception components provide input data to the planning and control block. Each of the smaller blocks represents the hierarchical structure of the data inputs/outputs for the task of driving.

Sensor: The key providers of data on the external and internal states of autonomous vehicles are the vehicle sensors (VS). The vehicle sensor conveys input data to the components responsible for localization, environment perception, and self-perception. The aforementioned blocks then process data to produce the output requirements for the corresponding planning and control block structures.

Localization block: maintains a current version of the original data by using the most recent information from the environment and perception block and the information from the localization sensor. The localization incorporates data from the lower levels to locate lanes, which are indicated by the upward-facing arrows. It assisted in building the architecture with updated features collected from the environment and perception block. The downward-pointing arrows illustrate how data acquired at higher levels can be utilized to connect the traits picked up at lower levels in a meaningful way.

(a). Road level: is the highest level among the localization blocks, which maintains and updates information on lane frameworks, topological structure, and other aspects of roads.

(b). Lane-level: creates the classification data for individual lanes that can be derived from the perception and environment blocks. Data concerning vehicle sensor signal specifics are dynamically updated in the localization and mapping of lane details at the lane level.

(c). **Feature level**: The dynamically tracked signals and supplementary data are maintained and updated by the precise feature data regarding localization and perception.

Environment and well perceptron: The core of the suggested autonomous vehicle pipeline design is the environment and one's own perception. It interfaces with each of the localization and planning components periodically with the goal of observing and updating localization block details. This generates sensor signal data for the planning block.

(a). **Feature extraction**: component analyzes the unprocessed environmental sensor signal data to recognize features, such as traffic signals, lanes on the road, and so on. The context modeling component, which is positioned above the feature extraction sub-block, receives the output from the corresponding block. Additionally, it supplies signals to the stabilization block of the planning and control unit and the nearby feature level of the localization block. The features of the perceived data about the environment are being updated using this output.

(b). Context modeling: The Planning and Control block's Guidance sub-block accepts the result of the context modeling. The localization and provision block details are updated using the context modeling's result.

(c). **Road level:** The localization block is generated and updated using the most recent data utilizing the sensor signal data obtained from the previous sub-blocks of the environmental and self-perception block. The navigation sub-block of the planning and control block makes additional use of the output produced road level as input.

Planning and control block: The final process in the ML pipeline, planning and control, can be seen as a vital component of the step behind it. The following list clearly identifies the many hierarchical levels of driving tasks:

(a). **Navigation**: The planning and control block's navigation level is where the communication component is fed and kept up to date. The navigational component (NC) relies on SGD optimization approaches and sends sensor signal like data to various blocks.

(b). Guidance: The Environment and Self-Perception block generates routing and environment information that is used by the Guidance sub-block in order to create mission-ready systems. The environment is updated and enhanced with more information in order to create a special circumstance for the vehicle sensor fault types on the availability of a trained algorithm. The aforementioned actions resulted in a number of different potential target sets required for the training of the SGD algorithm.

(c). **Stabilisation:** The data is generated in line with the trajectory for a particular operation specified by this block based on the available algorithms. The process monitoring revealed the deviation from the SGD pipeline model. The system controls and transmits this data as result to the actuators.

Communication: is the very last block in the ML pipeline, which interfaces with every other block and external communication architecture like human. This is often at the highest level of the human-involved planning and control block. The information from the communication can help improve the autonomous vehicles' accuracy.

The SGD model: An optimization strategy was adopted in choosing the mean value and assumed to follow a specific data distribution in order to carry out clustering on gradient descent. This is

helpful in preventing the task of initial aggregating by simple averaging and removal of noisy gradients. We computed the new values of the parameters as shown in equations 1 and 2 below after randomly shuffling the dataset and choosing the k needed samples.

$W=w-y \times \text{grad}(f(w))$ 1

Substituting the value of $y=\frac{\gamma}{k}\sum_{k=1}$ into equation1 will results into equation 2

$W=w-\frac{\gamma}{k}\sum_{k=1}\text{grad}(f(w))$ 2

Where k>0, grad(f(w)) >0 and this is fairly small for the pace of learning

The SGD model was developed to update variables only when all "n" training examples had been completed in order to enhance the number of updates per period.

Minimizing the overall cost of SGD is a simple task. Given that the data used for training the SGD is finite, the distribution of dP(z) will be discrete and can be represented as follows:

$dP(z) = \sum_{i=1}^{N}\frac{1}{N}\delta(x-x_i)$

$C(w) = \frac{1}{N}\sum_{i=1}^{N}J(x_i - w)$ 3

Each of the iteration in SGD algorithm that we are using will include adding the sums of the N samples x_i.

$W_{t+1} = w_t - \epsilon_t\frac{1}{N}\sum_{i=1}^{N}\nabla_w J(x_i, w_t)$ 4

We are using positive scaler as the gain ϵ_t. Each training cycle of the SGD demands a taxing computation of the average represented as $\nabla_w J(x,w)$ over the complete training set, which is necessary for the task of learning in equation 4 because we need sizeable data to provide precise information concerning the real phenomena.

The SGD approximates the gradient descent as opposed in computing the gradient for the entire data samples at each step, the algorithm only does so for one observation chosen at random.

We are introducing a linear function that we need to minimize the error rate in the loss function as given in equation

$L(w, b) = min_{w,b} + \sum_{i=1}^{n}(y_i - w^T x_i - b)^2$ 5

The gradient for our loss function(L) is been calculated with respect to the weights(w) and intercept(b). To compute the gradient the equation 6 is used.

$\frac{\partial L}{\partial W} = \sum_{i=1}^{n}(-2x_i)(y_i - w^T x_i - b)$ 6

$\frac{\partial L}{\partial b} = \sum_{i=1}^{n}(-2)(y_i - w^T x_i - b)$ 7

We keep changing the weights and intercept values after calculating the gradient to arrive at equation 8

$w_{i+1} = w_i - r\left(\frac{\partial L}{\partial w}\right)w_i$ 8

$$b_{i+1} = b_i - r\left(\frac{\partial L}{\partial w}\right) b_i \quad 9$$

Machine learning gradient descent uses derivatives and optimization techniques to solve optimization issues, such as determining whether to increase or decrease the weights.

We used the stochastic gradient descent algorithm with dataset(D), which included n training samples with features $X_j^{[i]}$ and targets or class labels $y^{[i]}$.

Where D $=\left(X^{[1]}, y^{[1]}\right), \left(X^{[2]}, y^{[2]}\right), \ldots, \left(X^{[n]}, y^{[n]}\right)) \, \epsilon(\mathbb{R}^m \times \{0,1\})^n$ 10

Metrics of Evaluation

It is necessary to use a standardized evaluation tools in order to compare the various results. There are various ways of visualizing correct and incorrect predictions for problems involving multi-classification task like the proposed system. Standard valuation tools like prediction accuracy, RMSE, confusion matrix, classification report and ROC curves are employed to evaluate the performance of SGD and pipeline classifier. The Classification accuracy is the ratio of correctly classified data points to the total number of points in the dataset.

The error rate can be measured with the general equation given by:-

Error rate $=\frac{\text{TP+TN}}{\text{TP+FP+TN+FN}}$ 11

The sensitivity or also called the True Positive Rate(TPR) is given by:

TPR $=\frac{\text{TP}}{\text{TP+FN}}$ 12

The False Positive Rate(FPR) or also called Fall-Out is gievn by:

FPR $=\frac{\text{FP}}{\text{FP+TN}}$ 13

Accuracy$=\frac{\text{Total number of correct classification}}{\text{Overall number of classes}} = \frac{TP+TN}{TP+TN+FP+FN}$ 14

Where TN represents true negative, FP is false positive, TP is true positive and FN is false negative cases.

Algorithm 1: proposed Stochastic gradient descent(SGD) classifier

Step	**Processes involved**
1	**Input:** Training_Dataset(T)
2	**Output**: Class of testing items
3	Initialize weights(w)= $0^{m-1}, b = 0$
4	For iteration t ϵ [1, . . . , T]:
	(a). Draw samples randomly with replacement: $\left(X^{[i]}, y^{[i]}\right) \epsilon$ D
	(b). Compute model predictions $\grave{y}^{[i]} => h(X^{[I]})$
	(c). Compute the loss value $L^{[i]} => L(\grave{y}^{[i]}, y^{[i]}$
	(d). Compute gradients
	(e).Update model parameters
5	Return

4.0 RESULTS AND DISCUSSION

The proposed model results are presented and discussed using the relevant SGD and ML Pipeline tools. The concept and its application have been improved in order to generate better and more accurate outcomes. We used standard ML and suitable diagnostic tools to display and discuss results of experiments.

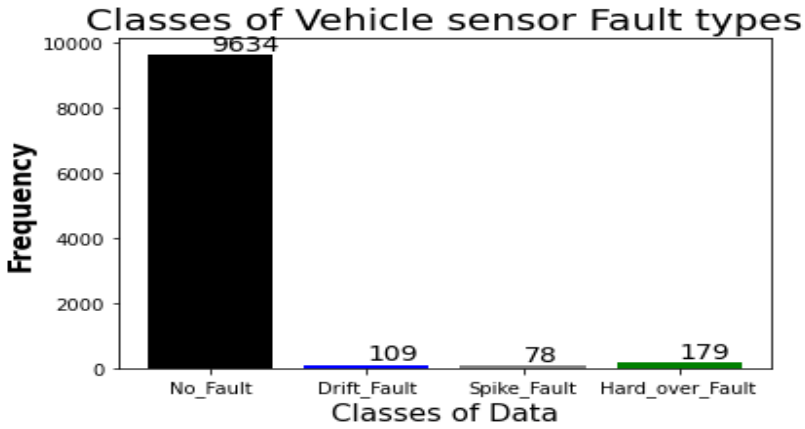

Figure 2: classes of vehicle sensor faults

Figure 2 depicts the classes of sensor fault types with values plotted on top bars to represent the uneven no_fault, drift, spiking, and hard over fault types in the target dataset. There are 9635 target classes which include: 9634 no faulty cases, 109 drift faults, 78 spike faults, and 197 hard over sensor fault types. Every other sensor fault case in the target set is surpassed by the no fault cases. The spike fault type occurs when a signal gradually deviates (linearly) from its actual value, hard-over fault occurs whenever a sensor recovers a value that is outside of its measurable range and rapidly increases to saturation and Spike faults occur when there are brief increases in the signal value. The signal's density of sudden increase faults can increase over time.

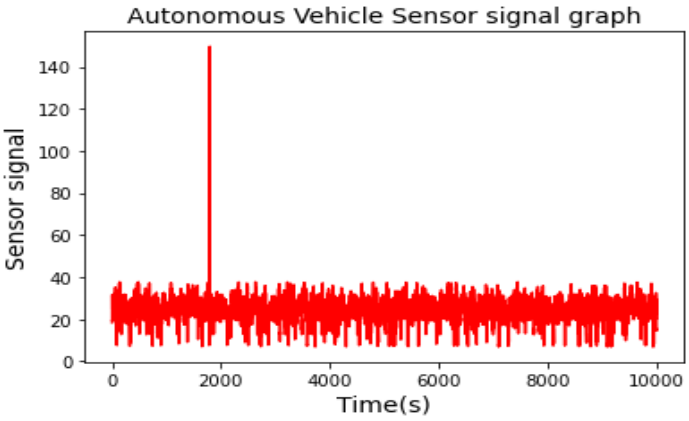

Figure 3: Autonomous vehicle sensor signal

Figure 3 depicts the chart of autonomous signal time plot demonstrating the significant rise of noise in the sensor data over time. The size of the signal variance also rises around the true value. There is a spike/rise in noisy data(signal) between 0 and 2000 seconds that falls below 20 and less above 30. There is a sharp signal spike that is detected over 140 just before 2000,

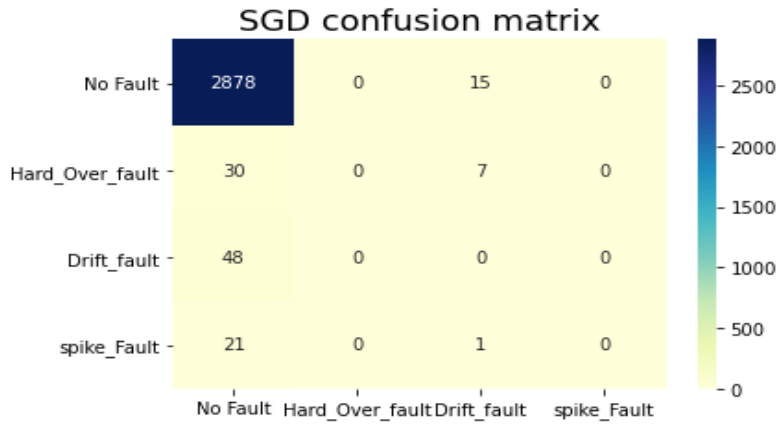

Figure 4: stochastic gradient desenct confusion matrix

Figure 4 shows the confusion matrix, which displays a table structure of the various SGD predicted results to aid the multi-classification task. Cell values above and below the main diagonal or off-diagonal elements shows the incorrectly predicted values and correctly predicted values that are equal to the actual or true values. According to the confusion matrix, hard over fault types had 30+7= 37 incorrectly predicted faults with no correct predictions. While spike fault provided 21+1= 22 incorrectly predicted values with zero true positive class prediction, drift fault recorded zero correct predictions with 48 incorrect misclassifications.

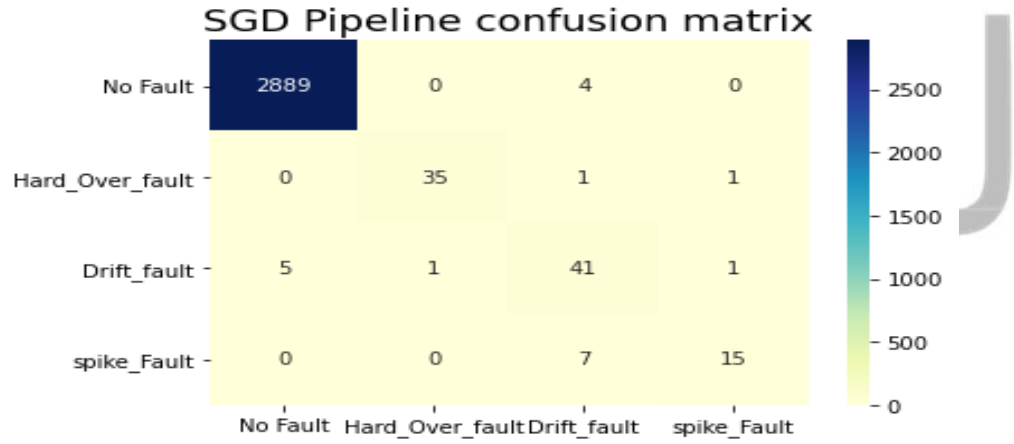

Figure 5: SGD Pipeline confusion matrix

Figure 5 depicts the SGD pipeline confusion matrix of 4 × 4 which is employed to assess how well a multiclassification report its performs, with 4 overall target classes. This is done to contrast the actual target values having 4-mutually exclusive possible outcomes of what the pipeline model predicted. The No_fault type recorded 2889 correctly predicted cases with 4 incorrectly classified classes. The hard over fault produced 35 correct predictions with 1+1=2 incorrectly classified data classes, the drift fault produced 41 correct predictions with 5+1+1=7 incorrect data classes, and the spike fault type produced 25 correct predictions with 7 incorrectly classified data classes. The pipeline confusion matrix is a type of table that aids in understanding how well a classification methodology works on a set of experimental data for which the true values are known.

Table 1: SGD classification report

	Precision	**recall**	**F1-score**	**support**
No Fault	0.97	1.00	0.98	2893

Hard_Over_fault	0.00	0.00	0.00	37
Drift_fault	0.00	0.00	0.00	48
Spike_fault	0.14	0.05	0.07	22
accuracy			0.96	3000
Macro avg	0.28	0.26	0.26	3000
Weight avg	0.93	0.96	0.95	3000

Table 1 depicts the SGD algorithm's classification report, including its precession, recall, and f1-score classification features. The Precision recorded 0.97 for cases of no fault, 0.00 for each type of fault: hard_over, drift, and 0.14 for all other fault categories. Recall produced values of 1.00, 0.00 for cases where there was no fault, 0.00 for each type of fault (spike, drift), and 0.05 for hard over fault signal. While the f1-score was 0.98 in cases where there was certainly no fault, it was 0.00 for spike and drift faults and 0.07 for hard over faults. The recall accuracy score was 1.00, which was the highest in circumstances where there was No_fault

Table 2: SGD Pipeline classification report

	Precision	**recall**	**F1-score**	**Support**
No Fault	1.00	1.00	1.00	2893
Har_Over_fault	0.97	0.95	0.96	37
Drift_fault	0.77	0.85	0.81	48
Spike_fault	0.88	0.68	0.77	22
accuracy			0.99	3000
Macro avg	0.91	0.87	0.88	3000
Weight avg	0,99	0.99	0.99	3000

Table 2 contains metrics for precession, recall, and f1-score accuracy for cases with no faults (NF), hard over faults (HOF), drift faults (DF), and spike faults (SF) in the SGD pipeline classification report. The classes NF (1.00), HOF (0.97), DF (0.77), and SF (0.88) were registered in terms of precision. Recall gave rise to the classes NF (1.00), HOF (0.95), DF (0.85), and SF (0.68), whereas the f1-score resulting in the classes NF (1.00), HOF (0.96), DF (0.81), and SF (0.77). The measurements for accuracy, recall, and f1-score Precision, recall, and f1-score values from No_fault were each 1.00.

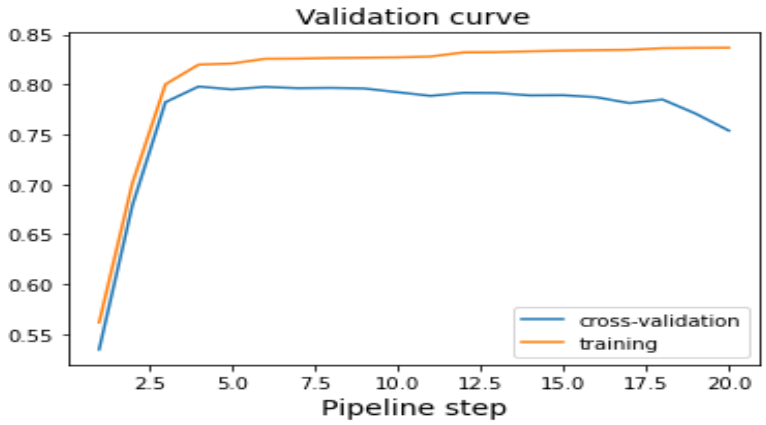

Figure 6: Validation curve

The learning curve of Catboot pipelining is shown in Figure 6 as a function of a number of training points, together with the training and validation score. Figure 6 illustrates the value of validation. The cross validation and training curve are increased by adding additional pipeline steps, while the training score is decreased from 5.0 to 20.0. The training and validation set as shown is low and the optimum tradeoff occurs when we choose d to be around 3 or 20.

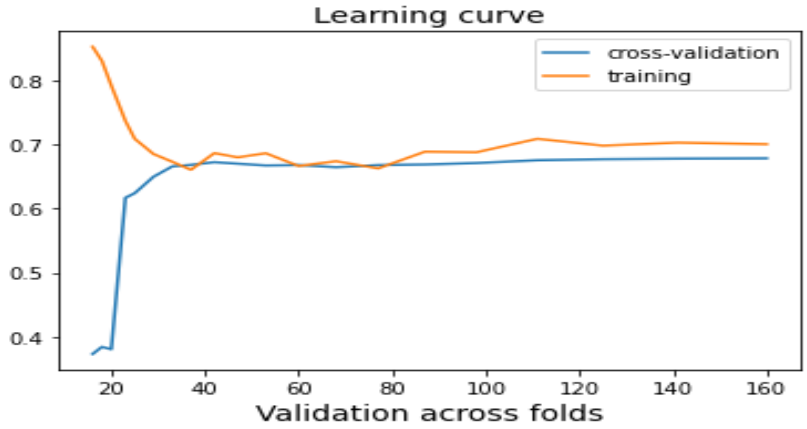

Figure 7: Stochastic pipeline learning curve

Figure 7 is the learning curve of Catboost pipeline model and it is shown that d=14 is the high-variance point of estimator which over-fits training data. The training score exceeds the validation score by a wide margin. The cross validation score will rise as more training set samples are added, but the training score will fall along various folds.

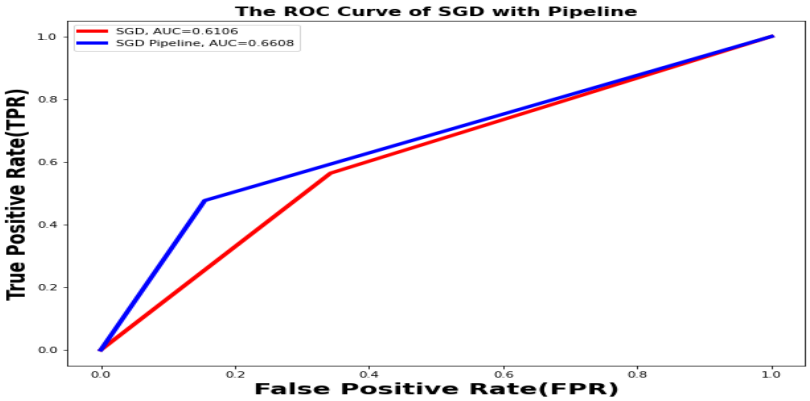

Figure 8: ROC plot of SGD with pipelining

Figure 8 depicts the ROC curve of SGD and pipeline classifier, which illustrates how well the suggested models work. It is one of the most crucial performance metrics used to assess the effectiveness of any classification algorithm. The pipeline model outperformed SGD with an AUC value that was closest to 1 (0.6608). The model with the AUC value (0.6106) closest to zero was the SGD model. The ROC plot made it very evident that the SGD's ROC curve is farther from 1 and located towards the upper left corner of the y-axis than the pipe line model.

Table 4: Detected and isolated fault types

S/N	UID	Sensor_Types	. .	Sensor_value	Fault_Type
1	4	LIDAR	. .	31.594618	Drift_fault
2	5	LIDAR	. .	26.382582	Hard_Over_Fault
3	8	LIDAR	. .	26.382582	Hard_Over_Fault
4	9	LIDAR	. .	31.594618	Drift_fault
5	15	LIDAR	. .	26.382582	Hard_Over_Fault
. . .	. . .	. . .	. .	. . .	. . .
9576	9577	LIDAR	. .	22.333044	Hard_Over_Fault
9613	9614	LIDAR	. .	28.698278	Drift_fault
9758	9759	LIDAR	. .	27.443794	Hard_Over_Fault
0764	9765	LIDAR	. .	21.851212	Drift_fault
9974	9975	LIDAR	. .	31.111763	Drift_fault

Table 4 shows the fault types (drift, hard_over, spike) for detected and isolated sensor signals which include attributes such as UID, Sensor_type, sensor value, fault categories, and more.

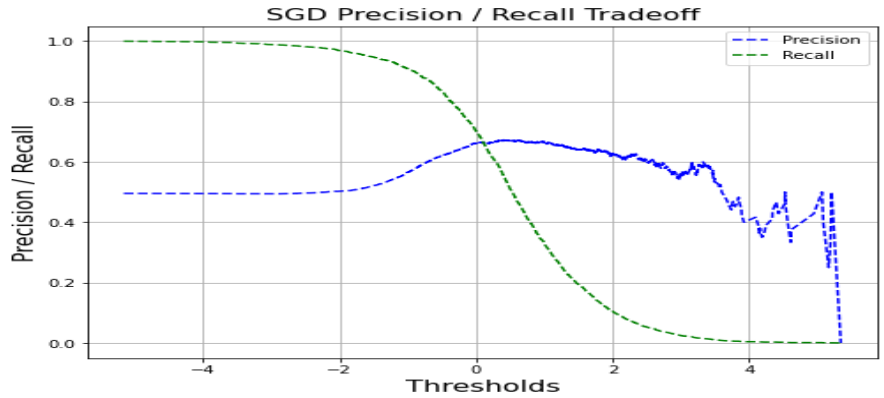

Figure 9: The SGD pipeline precision/recall tradeoff

Figure 9 provides an unambiguous representation of the pipeline model's performance given that if we choose the value of 65% and below, the recall is essentially measured to be more than zero, which is a suitable scenario for our proposed pipeline model. A recall value of almost 0% is not a good enough situation for our approach, and it was not evident in our plot.

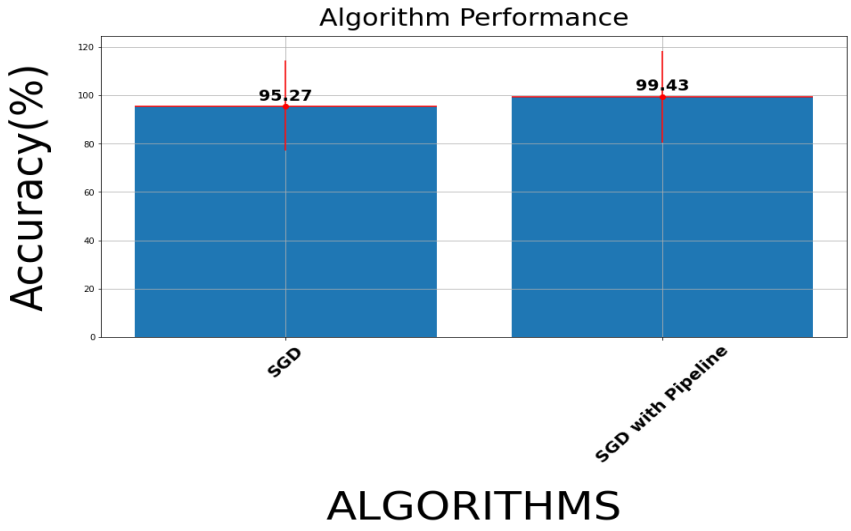

Figure 10: Performance accuracy

Figure 10 depicts the overall success of the proposed SGD and pipeline pipelining technique. The SGD had the lowest prediction accuracy and pipeline (99.43%). The SGD pipelines perform exceptionally well, with performance metrics increasing from 95.27% to 99.43% due to the added pipeline knowledge,

5.0 CONCLUSION

The SGD model performed quite well when utilized in conjunction with the pipelining concept to detect various forms of sensor signal faults in autonomous vehicles. This can serve as a baseline for other specialists and will help in the detection and isolation of any preselected sensor failure types in order to assure safety. The implementation process was made easier by a programming language called Python. It offers several deployable stochastic gradient learning libraries and classes that are accessed by just a few lines of code and have been optimized for speed. This will be helpful to autonomous vehicle manufacturers in accurately diagnosing different sorts of sensor problems by sorting through noisy signals and identifying relevant information. Its success in identifying sensor signal faults associated with autonomous vehicles has drawn attention to the importance of putting forth the suggested approach into practice.

FURTHER STUDIES

This system was trained using exist dataset, something that may become out-dated in the future, but it can also be used for future testing in order to pass new test cases. Future work could take this paper to a relatively high level in order to enhance the system's changes. The proposed SGD fault detection system can be enhanced with genetic algorithm and other deep learning frame works to help identify, locate and isolate semsor fault types with user feedback mechanism.

References

Al-Zeyadi, M., Andreu-Perez, J., Hagras, H., Roycey, C., Smithy, D., Rzonsowskiy, P., & Malik, A. (2020). Deep Learning towards Intelligent Vehicle Fault Diagnosis.1-7.

Balaban E., Saxena A., Bansal P., Goebel K.F. and Curran S.(2009) Modeling, detection, and disambiguation of sensor faults for aerospace applications. *IEEE Sens. J*, **9**, 1907–1917.

Bello-Salau, H., Aibinu, A. M., Onumanyi, A. J., Onwuka, E. N., Dukiya, J. J., & Ohize, H. (2020). New road anomaly detection and characterization algorithm for autonomous vehicles. *Applied Computing and Informatics*, *16*(1), 223–239. https://doi.org/10.1016/j.aci.2018.05.002.

Chamseddine, A., Noura, H.(2008) Control and Sensor Fault Tolerance of Vehicle Active Suspension. *IEEE Trans. Control Syst. Technol*, **16**, 416–433.

Chen, H., Jiang, B., & Huang, B. (2020). Data-driven fault diagnosis for traction systems in high-speed trains: A survey, challenges, and perspectives. *IEEE Trans. Intell. Transp. Syst*, 1–17.

Ciaburro, G.(2022), Machine fault detection methods based on machine learning algorithms: A review, Mathematical Biosciences and Engineering, 19(11), 11453-11490.

Davishi, H., Ciuonzo, D. and Rossi, P. S.(2022), A Machine-Learning Architecture for Sensor Fault Detection, Isolation and Accommodation in Digital Twins, IEEE SENSORS JOURNAL, xx(xx), 1-18

Fahim, M. and Sillitti, A.(2019)Anomaly detection, analysis and prediction techniques in IOT Environment: A systematic literature review. *IEEE Access*, *7*, 81664–81681

Gauerhof, L., Munk, P. and Burton, S,(2018) Structuring Validation Targets of a Machine Learning Function Applied to Automated Driving, in Computer Safety, Reliability, and Security, 11088, Cham: Springer International Publishing, 45–58.

Glowacz, A. (2018). Acoustic based fault diagnosis of three-phase induction motor. *Applied Acoustics*, *137*, 82–89.

Jeong, Y., Son, S. & Lee, B.K. (2019). The Lightweight Autonomous Vehicle Self-Diagnosis (LAVS) Using Machine Learning Based on Sensors and Multi-Protocol IoT Gateway. *Sensors*, *19*(2534), 1-24

Liu, C., Cichon, A., Królczyk, G., &· Li, Z. (2021). Technology development and commercial applications of industrial fault diagnosis system: a review. *The International Journal of Advanced Manufacturing Technology*, 1-34.

Marzat J., Piet-Lahanier H., Damongeot F.(2012) Control-based fault detection and isolation for autonomous aircraft. *Proc. Inst. Mech. Eng.***226**, 510–531.

Min, H., Fang, Y., Wu, X., Lei, X., Chen, S., Teixeira, R., Zhu, B., Zhao, X. M. and Xu, Z.(2023), A fault diagnosis framework for autonomous vehicles with sensor self-diagnosis, Expert Systems with Applications, 224, 34.

Obodoeze F.C., Okoye, F.A. and Ifeyinwa, O.N.(2018), Design and implementation of a vehicle fault detection system (FDS) with online and SMS fault reporting: Case study of ford motors, American Journal of Engineering Research, 7, 53-64.

Pradhan, M. K., & Gupta, P. (2017). Fault Detection using Vibration Signal Analysis of Rolling Element Bearing in Time Domain Using an Innovative Time Scale Indicator. *Int. J. Manufacturing Researc.* 12(3), 1-14.

Realpe, M., Vintimilla, B. and Vlacic, L.(2015), Sensor Fault Detection and Diagnosis for autonomous vehicles, 2015 the 4th International Conference on Material Science and Engineering Technology (ICMSET 2015), 30(3), 29-40.

Shen, Y. and Khorasani, K.(2020) Hybrid multi-mode machine learning-based fault diagnosis strategies with application to aircraft gas turbine engines. Neural Netw, 130, 126–142.

Wang, S., Zhang. Y., Zhang, B., Fei, Y., He, Y., Peng Li, P. and Xu, M.(2022), On the Sparse Gradient Denoising Optimization of Neural Network Models for Rolling Bearing Fault Diagnosis Illustrated by a Ship Propulsion System, International journal of marine science and engineering, 10(10), 23-30.

Yurii, K., & Liudmila, G. (2017). Application of Artificial Neural Networks in Vehicles' Design Self-Diagnostic Systems for Safety Reasons. *Transp. Res. Procedia*, *20*, 283–287.