Anomaly Detection in Time-Series Driving Data via Gaussian Mixture Models

Yisel Breton, Victoria Hong, and Shreya Suresh
Project SUPER Mentors: Prof. D. Pompili and PhD student Vidyasagar Sadhu
Cyber-Physical Systems Laboratory (CPS Lab), Electrical and Computer Engineering
Rutgers University–New Brunswick, NJ

***Abstract*— With Machine Learning and Artificial Intelligence becoming increasingly popular, many approaches have been developed in order to detect and classify anomalous data. In our project, we focused on the application of these anomaly detection approaches on autonomous vehicle technology. Autonomous-driving (AD) technology refers to the technology that allows vehicles to operate with little to no human intervention. Being a relatively new field, the main concern behind AD vehicles is safety. Thus, the purpose of this project is to train existing clustering algorithms and accurately detect anomalies that occur in AD vehicles, making AD technology more safe and efficient. Specifically, we want to test the effectiveness of using the Gaussian Mixture Models to detect anomalies in AD vehicles and compare them against existing methods: Multi-Class Long Short-Term(LSTM) auto encoder based approach and Multi-task Learning Model. The data used for this research is obtained from the Honda Research Center in California, USA.**

I. Introduction

Overview: Machine learning allows for computers to learn on their own without human intervention. Machine learning uses unsupervised learning techniques such as clustering and finding patterns in data sets without the need for previously labeled data. These techniques allow for computers to learn autonomously [1]. Furthermore, clustering is a collection of objects that are collected based on the basis of similarity and dissimilarity between them. Two popular clustering algorithms are K-Means and Gaussian Mixture Models. This report describes the application of the Gaussian Mixture Model using the Estimation-Maximization algorithm to train a model to detect anomalous behavior.

Motivation: Autonomous driving is a trend among car manufacturers that is becoming increasingly popular; this is where vehicles can drive on their own without the assistance of a human driver. Autonomous driving is made safe by the algorithms that effectively mimic human driving behavior without error [2]. To mass produce these self-driving cars, ensuring that the technology is error-free is vital and thus, the goal of this project is to use a Gaussian Mixture Model to detect anomalous data and make self-driving technology more safe and efficient.

Our Approach: An anomaly detection algorithm is imperative for an autonomous vehicle to have in order to effectively mimic human driving behavior. Our goal is to prove that the implementation of Gaussian Mixture Models in the algorithm is more effective and efficient than the two previously implemented methods: Multi-Class Long Short-Term (LSTM) Autoencoder and the Multi-task Learning Model. In this section, we will describe in detail the functionalities of the Multi-Class Long Short-Term Autoencoder and the Multi-task Learning Model. Both models function via an encoder and a decoder: the input data is mapped into a latent space via an encoder, and then the latent space is then remapped and to input space using a decoder.

Multi-Class Long Short-Term (LSTM) Autencoder: The Multi-Class LSTM Autoencoder is a model that works to capture long term temporal behavior. This is done by predicting current and future data from past data that is fed into the model. This approach, though, is only ideal for data that has similar patterns.[3] It also is not efficient enough because when the network is fed data that has a completely different pattern than the one used in training, a large reconstruction error is produced [3].

Multi-Task Learning Model: The Multi-Task Learning Model functions in that multiple learning tasks are solved simultaneously. The learning tasks in this case are the encoder and the decoder. Task A, the autoencoder, is a decoder but function as an encoder in that it performs operations in reverse order to reconstruct the input data [3]. Task B, the predictor, only tasks in forward cell states from the encoder and adopts a greedy encoder where the most probable symbol output of the previous cell is fed to the next cell [3]. The Multi-task Learning Model is a more effective model than the Multi-Class LSTM, however the focus of this paper is to showcase if the Gaussian Mixture Model approach is the most efficient model compared to existing methods.

Our Contributions: The key contributions of this paper are as follows.

- Applying a Gaussian Mixture Model to detect anomalous data obtained from autonomous vehicles at the Honda Research Center.
- Comparing and providing an overview of different types of anomaly detection methods.

Paper Outline: In Sect. II, the Related Work is presented. In Sect. III, the existing constraints, proposed methods, and approach to attain this project objective and its experimental scopes are discussed. In Sect. IV, the Performance Evaluation is described by comparing other models that have been used for anomaly detection on the same data. Finally in Sect. V, the Conclusion and Future Work is discussed.

II. Related Work

What is an Anomaly? Anomalies are patterns that does not match the expected behavior and appears infrequently in a given data set [4]. When applying machine learning to autonomous driving, it is important to have an anomaly detection algorithm.

A. Types of anomalies

Point Anomalies: When there is a single instance of data that can be considered as an anomaly, the instance is classified as a point anomaly. This is the simplest form of anomaly and is generally the focus of most anomaly detection research [5].

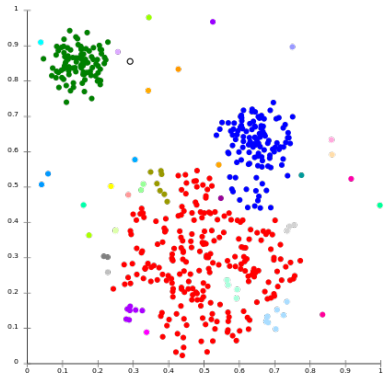

Fig. 1. In point anomaly, the three clusters are represented as green, red, and blue and the anomalies are represented as other colors. Graph taken from [6].

Contextual Anomalies: When a data instance is anomalous in a specific context, the instance is classified as a contextual anomaly (also referred to as conditional anomaly) [5].

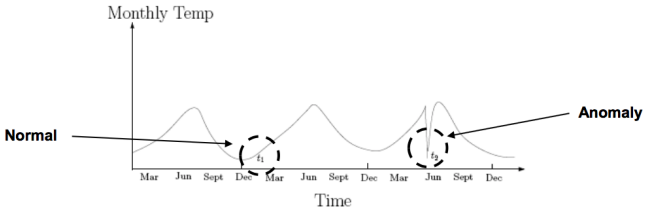

Fig. 2. In contextual anomaly, the anomalies are identified in March and June because there should not be low temperatures in those specific months. Graph taken from [7].

Collective Anomalies: When a collection of related data instances with respect to the entire data set is anomalous, the instance is classified as a collective anomaly. The individual data instances may not be anomalies by themselves but as a collective, this makes them anomalous [5].

B. Types of Anomaly Detection Methods

Nearest-Neighbor Based Methods: Nearest neighbor-based anomaly detection techniques classify data based on similarity. This method finds a predefined number of training samples closest in distance to a new point and predict the label. The distance can be measured with Euclidean distance, the most common choice. To determine an anomaly, it must be far from the nearest neighbor or compute the density of each data to obtain an anomaly score [5].

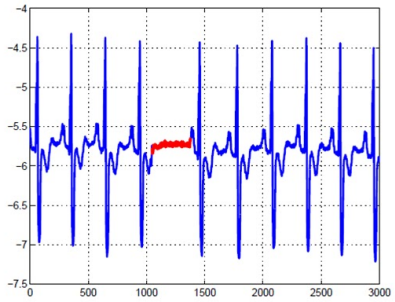

Fig. 3. In collective anomaly, the anomaly is in red because it does not follow the trend. Graph taken from [8].

Clustering Based Methods: Clustering is typically an unsupervised learning technique and is known as a collection of objects that are grouped based on the basis of similarity and dissimilarity [9]. K-Means clustering is a popular clustering method that searches for a predetermined number of clusters within an unlabeled data set. The target number (k) refers to the number of centroids, the center of a cluster, that are needed within the data set. Each point is assigned to the nearest cluster based on its euclidean distance to the centroid and once all of the points are added, the centroid is recomputed by taking the mean of all points that are assigned to the cluster. Following these steps, the algorithm classifies each data point [10].

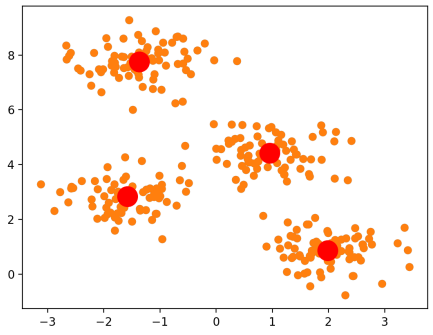

Fig. 4. K-Means Clustering uses a group of data to classify anomaly. Here, four clusters and their respective centers are visible.

Statistical Methods: Statistical anomaly detection techniques assume normal data instances locate at higher probability while anomalies occur at lower probability, thereby samples with lower scores are detected as anomalous. Both parametric (i.e., Gaussian and Regression Models) and non-parametric (i.e., Histogram-based) techniques use this model [5].

In this section, existing constraints are taken into consideration as we discuss about the Gaussian Mixture Model and EM algorithm. Details on the data used and the implemen-

tation of the Gaussian Mixture Model and EM algorithm will be introduced in detail in Section IIIA and Section IIIB respectively.

C. Our Approach:

Existing Constraints: As it was discussed before, anomaly detection is certainly not a new field but its application on autonomous driving is. Therefore, the constraint was not on our approach but rather, the application in technique. Comparing with existing models was beneficial and certainly helped with our research.

Proposed Method: For the purpose of this project, we wrote the code on a Python Integrated Development Environment called PyCharm, in Version 2019.3.3. The project was completed in two parts:

- First, test the code on a set of randomized Gaussian Multivariate Data that we generated.
- Second, relying on the success of the first step, apply the same algorithm to real data provided by the Honda Research Center from California.

We assumed that the data was a multivariate Gaussian distribution. In this section, we discuss what a Gaussian is and the Expectation-Maximization (E.M.) algorithm that we used to fit the data to the model.

D. Gaussian Distributions and Mixture Models:

Gaussian Mixture Model (GMM), as mentioned before, is another method for anomaly detection. This method assumes that there are a certain number of Gaussian distributions and each distributions represent a cluster. Therefore, GMM tends to group data points belonging to a single distribution together, using a probabilistic model and soft clustering approach [11]. The probability formula is given by:

$$f(x|\mu,\sigma^2) = \frac{1}{sqrt(2\Pi\sigma^2)} * e^{-}\frac{(x-\mu)^2}{2\sigma^2} \tag{1}$$

GMMs are used to find clusters or assume the number of clusters enclosed within a given data set when the location or shape is not known, making it the most effective anomaly detection because it takes into consideration all ellipsoidal shapes and circular clusters within a data set [4].

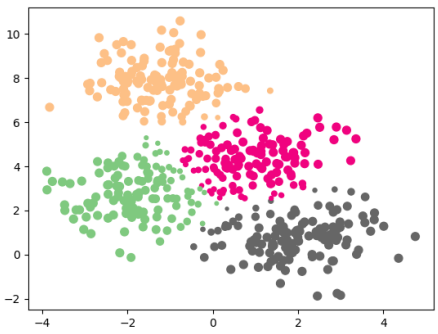

Fig. 5. GMM Clustering uses a probabilistic approach to classify a data point into a distribution. Here are four Gaussian clusters each with the same standard deviation of 1.

The covariance matrix allows this to happen because it is a measure of how much two variables vary together as a relationship [12]. The covariance matrix formula is given by:

$$C = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x}) * (x_i - \bar{x})^T \tag{2}$$

E. Expectation-Maximization Algorithm:

The Gaussian Mixture Model algorithm also comprises of an Expectation-Maximization step, or the EM-step. The EM-step is comprised of three steps: Initialization, Expectation, and Maximization. The data is already initialized since the car data sets are provided. For the E-step, the weights, mean, and covariance are used to calculate the probability that a data point belongs to a specific cluster.

$$r_{ic} = \frac{\pi_c \times N(x_i|\mu_c,\Sigma_c)}{\sum_{k=1}^{n}\pi_k \times N(x_i|\mu_k,\Sigma_k)} \tag{3}$$

Where the probability of

$$x_i \tag{4}$$

belonging to a cluster c, is divided by the sum of probability

$$x_i \tag{5}$$

belongs to either cluster

$$c_1, c_2, ...c_k \tag{6}$$

In the M-step, the probabilities found in the E-step are used to find the weight, mean, and covariance of each Gaussian cluster. In this step, the probability of a data point belonging to a cluster is identified.

$$\Sigma_c = \frac{1}{m_c} \times \Sigma_i \times r_{ic}(x_i - \mu_c)^T \times (x_i - \mu_c) \tag{7}$$

Based on the values generate in this step, the probabilities of each data point are updated iteratively. This process, repeated over and over again, maximizes the log-likelihood function of the program.

$$lnp(X|\pi,\mu,\Sigma) = \sum_{i=1}^{N} ln(\sum_{k=1}^{K}\pi_k N(x_i|\mu_k,\Sigma_k)) \tag{8}$$

III. Performance Evaluation

The data, as stated before, was provided by the Honda Research Center in California, USA.

The implementation can be divided into four sections:

- Confirming that the code written by a previous student worked by testing it with a randomly generated set of multivariate Gaussian data.
- Downloading and combining the Honda data so that it is ready to create the model.
- Reshaping the data to fit the Gaussian Mixture Model and finding the window scores.
- Displaying the results of the models by arranging the scores in ascending order and detecting the occurrence of anomalies.

A. Details about the data:

Key information about how the data was handled:

- The data utilized consisted of 266 hours of predominantly daytime raw driving data from February 2017 to March 2018, with a frequency of 100Hz that was down-sampled to 5Hz.
- The data consisted of 6 modalities: steer angle, steer speed, speed, yaw, pedal angle, and pedal pressure.
- The data utilized was a time-series data that was segmented into windows using a sliding window approach with 5s window size and stride length of 0.5s.
- 70% of the data was used for training, while the remaining 30% was used for testing.
- The U-turn was not used in the training process, but instead considered as anomalous for testing purposes.
- Using 11 maneuvers, the GMM classifier classified any window corresponding to a U-turn as an anomaly.

B. Implementing the algorithm:

Stage 1: First, we accessed the code that was written by a previous student who conducted this project on Jupyter Notebook in 2018. His code was written using the following steps:

- Create training data sets by making 3 data frames with 6 features and 25000 rows using the randn function from Python libraries. Assign them to arrays of multivariate normal type with previously assigned means and covariances. Follow the same procedure to create a data frame with a mean matrix that is significantly larger to the previous mean; This will differentiate between the training and test set. Normalize the data to make it stay in desired range.
- Split the 3 training data frames into parts of 25 windows for testing and 75 for training. Stack them vertically and reshape into size (-1,150) to create a training data point that has one row corresponding to one window.
- Train the model by selecting a number of components, i.e. 2, and the covariance type, i.e. 'full', and save it.
- Find the anomaly score for each window of the training and test set using a loop.
- Display the score and if they are different from one another, with one being larger, the model is working.

Stage 2: Now, we train the model on the experimental data we have collected.

- Download all the necessary data files provided by Honda and load them into the project. These includes twelve different maneuvers such as going straight, turning left and right, parking, etc. The data was in the form of different number of windows of size (50,6) and some were found more or less data than another. Using a loop, we deleted the last 25 rows of each window in order to make the windows to size (25,6). All of the windows were stacked vertically and shuffled. The 6 features represented different sensor readings such as acceleration, velocity, etc. making each window a representation of different maneuvers, in terms of sensor readings.
- There are 12 different maneuvers (228802 vertically stacked, shuffled windows of size (25,6)) in the training data set and the 'u-turn' data was not included. Therefore, we are using it as an anomaly to test with.

Stage 3: In this stage, we reshaped the data to have one window correspond to a row and train the model using the 228802 training data set windows with the 'mixture.GaussianMixture()'function of the 'sklearn' library. Once trained, we saved it using the 'pickle' library, enabling the program to run efficiently without having to load the data each time to test the code. Then, we load the test set into the development environment and pickled model and test data. The test data consists of windows from the previously seen 228801 and 'u-turn' windows.

Stage 4: This stage consists of displaying the results of our test. We used a function to arrange the scores in ascending order (along with their maneuver labels) and detected how many 'u-turns' occurred in the top 0.001, 0.01, 0.1 0.5 and 1st percentiles of the data. In this approach, Python libraries are used to identify the clusters and anomalies within the data set. Libraries are defined as a collection of functions and methods that allows a user to perform many actions without writing the algorithm from scratch. Throughout this entire project, we used the following Python libraries which can be installed using the 'pip' utility function:

- numpy
- scipy
- scikit-learn
- pickle
- utils
- sys

C. The Results:

Fig. 6 displays the results that were found, listing them in ascending order.

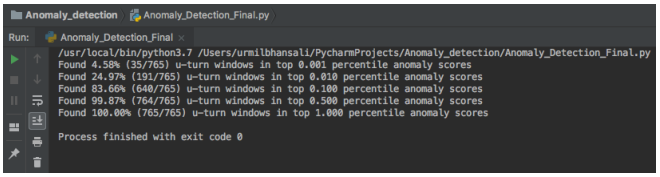

Fig. 6. The Results

The Gaussian Mixture Model found:

4.58% (35/765) u-turn windows in top 0.001 percentile anomaly scores.

24.97% (191/765) u-turn windows in top 0.010 percentile anomaly scores.

83.66% (640/765) u-turn windows in top 0.100 percentile anomaly scores.

99.87% (764/765) u-turn windows in top 0.500 percentile anomaly scores.

100.00% (765/765) u-turn windows in top 1.000 percentile anomaly scores.

Table 1. compares the results that we found with the results from [3]. The same data set was used but different anomaly

TABLE I
COMPARISON OF DETECTION METHODS' ANOMALY SCORES IN PERCENTAGES

Percentile Scores	Multi-Class LSTM Autoencoder	Gaussian Mixture Model	Multi-task Learning Model
0.001%	0.39% (3/765)	**4.58% (35/765)**	7.97% (61/765)
0.010%	1.96% (15/765)	**24.97% (191/765)**	29.02% (222/765)
0.100%	13.33% (102/765)	**83.66% (640/765)**	48.63% (646/765)
0.500%	73.64% (562/765)	**99.87% (764/765)**	84.44% (646/765)
1.000%	100.00% (765/765)	**100.00% (765/765)**	100.00% (765/765)

detection methods were applied. This includes a Multi-Class LSTM Autoencoder and a Multi-Learning Model, while we used the Gaussian Mixture Model (GMM).

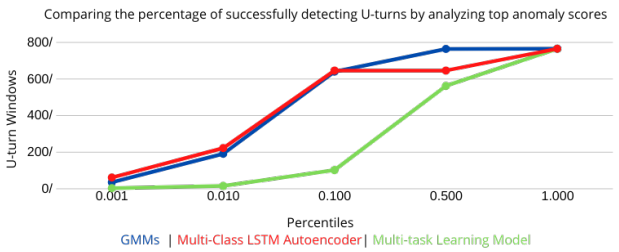

Fig. 7. This chart compares three different approaches for anomaly detection. GMM is represented in blue, the Multi-Class LSTM Autoencoder in red, and the Multi-task Learning Model in green.

The GMM and Multi-Class Long Short-Term autoencoder performed similarly, but GMM performed more efficiently overall for anomaly detection.

IV. CONCLUSION AND FUTURE WORK

In comparison to Multi-Class Long Short-Term autoencoders (LSTMs) and Multi-task Learning Models, our analysis concluded that Gaussian Mixture Models was the most efficient anomaly detection method. Despite the Multi-Learning Model being the most accurate in the first two percentiles, GMM proves to be the best at detecting the most anomalies in a given set overall.

In the future, we anticipate to incorporate the Generative Adversarial Network (GAN) concept into our work because it is a new concept in the field of machine learning that creates new data instances that resemble existing training data [13]. GAN allows the algorithm to become better and more precise at detecting real data from non-real data, GAN generated data, thereby making it useful for anomaly detection.

A GAN is composed of two parts: a generator and a discriminator. The generator captures the data distribution and learns to generate plausible data. The discriminator learns to distinguish the generator's fake data from real data. In anomaly detection, the task of using GANs is for modeling normal behavior by using the adversarial training process. In the adversarial training process, inputs are created by the generator to purposely get the algorithm to make a mistake in classifying the data point. Data points that are classified incorrectly, become 'loss' and this way, anomalies are detected by being assigned a measured 'anomaly score' [14].

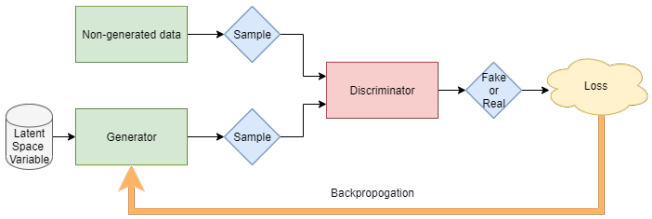

Fig. 8. The diagram above is the Generative Adversarial Network (GAN) Structure (with Backpropogration) and it describes how this network functions.

For the net to be trained and become better at making decisions, the step of backpropagation is introduced. This step begins at the output and flows back to the input where each weight of the data point is changed. By changing the data points weight, the changes (in the output) can be analyzed [14].

Acknowledgements We want to acknowledge and thank Dr. Dario Pompili and Vidyasagar Sadhu for advising us throughout this research experience.

REFERENCES

[1] S. Pokhrel, "Important topics in machine learning you need to know," 2019. [Online]. Available: https://towardsdatascience.com/important-topics-in-machine-learning-you-need-to-know-21ad02cc6be5
[2] A. Schroer, "Artificial intelligence in cars powers an ai revolution in the auto industry," 2019. [Online]. Available: https://builtin.com/artificial-intelligence/artificial-intelligence-automotive-industry
[3] V. Sadhu, T. Misu, and D. Pompili, "Deep multi-task learning for anomalous driving detection using can bus scalar sensor data," pp. 2038–2043, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8967753
[4] M. G. Gumbao, "Best clustering algorithms for anomaly detection," 2019. [Online]. Available: https://towardsdatascience.com/best-clustering-algorithms-for-anomaly-detection-d5b7412537c8
[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," no. 15, July 2009.
[6] S. Perera, "Introduction to anomaly detection: Concepts," 2018. [Online]. Available: https://iwringer.wordpress.com/2015/11/17/anomaly-detection-concepts-and-techniques/
[7] "Methods for anomaly detection of online social," 2019. [Online]. Available: https://ukdiss.com/examples/online-social-networks-anomaly-detection.php
[8] "Anomaly detection," 2019. [Online]. Available: https://www.commonlounge.com/discussion/2c2f22c8de0c40d4a4d5336bf2b1c06d
[9] G. Seif, "The 5 clustering algorithms data scientists need to know," 2019. [Online]. Available: https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68
[10] Y. P. Raykov, A. Boukouvalas, F. Baig, and M. A. Little, "What to do when k-means clustering fails: A simple yet principled alternative algorithm," 2017. [Online]. Available: https://doi.org/10.1371/journal.pone.0162259
[11] S. Kaushik, "Clustering introduction different methods of clustering," 2019. [Online]. Available: https://www.analyticsvidhya.com/blog/2016/11/an-introduction-to-clustering-and-different-methods-of-clustering/
[12] N. Janakiev, "Understanding the covariance matrix," 2018. [Online]. Available: https://datascienceplus.com/understanding-the-covariance-matrix/
[13] "Introduction — generative adversarial networks — google developers." [Online]. Available: https://developers.google.com/machine-learning/gan/
[14] F. Di Mattia, P. Galeone, M. De Simoni, and E. Ghelfi, "A survey on gans for anomaly detection," 2019. [Online]. Available: https://arxiv.org/abs/1906.11632/