A Holistic Evaluation of Vehicle Driver Stress Detection via Multi-View Learning

Arman Tabaddor
University of Michigan
armantab@umich.edu

Nathan Ozog
University of Michigan
ozog@umich.edu

Qian Qian
University of Michigan
qqbruce@umich.edu

Xinyuan Yu
University of Michigan
xyyu@umich.edu

Zachary Zipper
University of Michigan
zezip@umich.edu

Abstract

The rise of advanced and connected vehicle technologies such as Advanced Driver Assistance Systems (ADAS) is rapidly improving the safety and reliability of intelligent vehicles. Despite this, certain driving conditions may increase physiological responses in human drivers and passengers resulting in unintended, and sometimes fatal, consequences. In this paper, we address the challenge of detecting elevated levels of stress in individuals using a variety of biometric signals. Towards this end, we provide a holistic evaluation of supervised and unsupervised algorithms via a multi-view learning approach to better support driver-vehicle interactions. To supplement this, we also benchmark latency for these systems. We evaluate on a popular database containing real-world biometric signals.

1 Introduction

One of the most common causes of vehicle accidents is due to driver-induced physiological changes stemming from fatigue, poor decision making, and driver inattention [8]. More specifically, over $3,000$ individuals are involved in fatal vehicle accidents annually due mental or physiological changes or distractions to the driver [1]. As a result, several driver drowsiness detectors and stress detectors have been proposed in the hopes of mitigating the probability of a fatal driver-induced stress accident.

Vehicle driver stress detection has gained attention in the field of affective computing and driver vehicle interaction. More recently, several works have been proposed addressing the challenge of detecting elevated driver stress levels. Recent literature has seen several machine learning algorithms as suitable techniques for driver stress recognition, which we provide a brief survey of in section 2. In general, the features of interest are the driver's electrocardiogram (ECG) and electrodermal activity (EDA) signals. High levels of stress may cause irrational or unexpected decisions, thus posing a high risk to the driver, passengers, nearby vehicles, and pedestrians. However, even the simple notion of a binary classification of "high stress" or "low stress" is not a simple task to identify as biometric patterns differ between drivers. Thus, traditional machine learning procedures of training a model on a driver's physiological data and simply evaluating for that specific driver will fall short. Such an approach will fail to generalize to new drivers; each model will need to be trained on data for a specific driver, increasing deployment time and the feasibility of the system to broad applications.

In addition, a future of autonomous vehicles (AVs) where human driver error has been eliminated, passenger stress analysis will become a primary issue for AVs. As a result, generalizability is paramount when AV ride sharing becomes widespread. Granting AVs the ability to analyze passenger stress will reduce unintended consequences, such as passenger heart attacks, panic attacks, etc., and will enable a more pleasurable ride in general. In regards to generalizability, we take a multi-view learning approach to our evaluation. More precisely, multi-view learning considers a diversity of different source or features by defining one function to model a specific view, while jointly optimizing other views (generally representing the same semantic data) [22]. Multi-view learning has found success in different domain adaptations, such as image classification, image clustering, and more recently, one use case in driver stress recognition [13, 10]. In our work, we denote our approach as Leave-One-Driver-Out (LODO) to provide an evaluation of a machine learning model's ability to generalize to unseen physiological signal behavior.

To address the challenges highlighted above, we propose a holistic, breadth-based evaluation of machine learning methods, both supervised and unsupervised, to benchmark performance and feasibility via a multi-view learning approach for effective and efficient vehicle driver stress detection. Stress detection has traditionally been formulated as a binary classification problem (stress vs. no stress). We further expand on this to the multi-class classification problem (i.e, low stress, medium stress, and high stress) as well.

2 Related Work

Vehicle driver stress recognition is a well-studied problem. Data used to predict driver stress commonly contains physiological sensor information. MIT Media Lab's Driver Stress Dataset is used in foundational work in driver stress recognition. This dataset contains ECG information (which records electrical signals from the heart), EMG information (which records the electrical activity produced by skeletal muscles), chest cavity expansion, and skin conductance data obtained from "drivers in natural driving situations that included rest, city driving and highway driving tasks" [7].

Previous work has attempted to predict stress from varying combinations of ECG signals [12, 4], transformations of those signals [10, 21], other biological features including galvanic skin response (GSR) and respiration through chest cavity expansion (R) [6], and even photographed facial expressions [15].

Formulations of this problem have either mapped to binary classification, where the driver is stressed or not stressed [12, 4, 6, 10, 19, 21] or abstracted from a variety of other human emotions (anger, sadness, disgust, fear, etc.) which reduce to stressed or not stressed [15].

Past work has taken a variety of approaches to modeling this problem, including logistic regression [10], support vector machines (SVM) [12], multilayer perceptrons (MLP) [15], neurofuzzy min-max classifiers (MLPs that use fuzzy logic) [15], K nearest neighbors [4], shallow artificial neural networks [4], recurrent neural networks (RNN) using long short term memory (LSTM) cells [2], and multitask neural networks (MT-NN) [19].

Driver stress recognition has been successfully modeled, with the performance of simple models (SVMs, logistic regression, shallow MLPs) on the most general features (BR, HR) hovering from $70-90\%$ accuracy [12, 15, 10]. More complex models, including MT-NNs, and models using more complicated transformations of ECG inputs, including principal component analysis (PCA) and linear discriminant analysis (LDA), have achieved upwards of 90% accuracy [4].

Cutting edge research [9] has turned to advanced CNN models. They achieve a binary classification accuracy of 95.67% using the Stress Recognition in Automobile Drivers (SRAD) dataset from PhysioNet. However, this accuracy is achieved using a validation approach different from ours; they use different hyperparameters for each unseen individual instead of fixing the hyperparameters. This approach only works if the newly unseen drives have provided stress labels (which in general practice they would not).

3 Proposed Methodology

Though the works reviewed in the previous section report promising results, most fail to tackle feasibility of the models evaluated. We focus on holistically evaluating how machine learning (supervised and unsupervised) may be used for multi-view learning scenarios where a model is evaluated against a completely new, unseen individual. Towards this, to stay consistent in our evaluations across the different models, we employ the following methodology in Figure 1. Note that some models (i.e, RNN - LSTM) may not require explicit feature extraction as other models (i.e, SVMs). We evaluate our models using binary and multi-class classification. In contrast to most related work, multi-class classification may provide better granularity between stress levels.

Note that, in Figure 1, all drives but one are used in training, and that the one drive left out is used for evaluation. The finer details for LODO are discussed further in subsection 3.3.

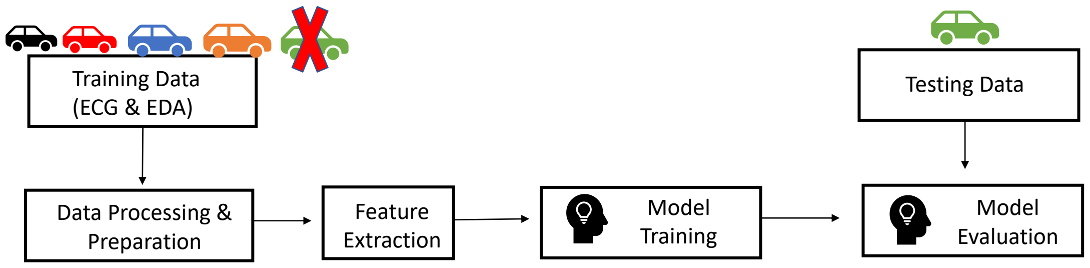

Figure 1: Proposed Methodology

3.1 Data Preprocessing & Preparation

We use the AffectiveROAD Database for our experimental evaluations. This database was compiled and published in 2018 [5]. It contains 13 drives of varying length and location. The drivers wear two Empatica E4 devices on the left and right wrists and a BioHarness. During the drive, an observer situated in the rear seats uses a slider to estimate the driver's stress between 0 and 1. Once complete, the driver reviews video footage of their drive and validates the observer's estimations. Compiling the data gives features outlined in Table 1.

Feature Label	**Description**
bioHR	BioHarness heart rate
bioBR	BioHarness breathing rate
bioPosture	BioHarness posture
bioActivity	BioHarness electrodermal activity
(l/r) xAcc/yAcc/xAcc	Left/Right wrist x/y/z axis acceleration
(l/r) BVP	Left/Right wrist blood volume pulse
(l/r) EDA	Left/Right wrist electrodermal activity
(l/r) HR	Left/Right wrist heart rate
(l/r) TEMP	Left/Right wrist body temperature
stressMetric	Subjective recorded stress metric

Table 1: AffectiveROAD Features

To prepare the datasets, we first perform data interpolation. The AffectiveROAD Dataset utilizes different sampling rates for all features. We treat the stress metric sample rate as our sample size and interpolate feature data to match its sampling rate. Signals sampled at lower rates are "scaled up" to match the stress metric sample rate by performing linear interpolation, adding mean-valued data points between existing data as needed. Likewise, signals that need to be "scaled down" are linearly interpolated by averaging nearby samples into a new data point. Care is taken to ensure the overall structure of the signal curve is preserved, maintaining the meaning of the signal at every instance in time. After preparation we are left with an $N \times D$ data matrix (N is the number of data points and D is the number of raw features) and the corresponding stress metric for each sample. Likewise, for EDA signals, we perform a low-pass filter of the data. Precisely, a digital low-pass filter was applied to eliminate high-frequency noise [10].

If the machine learning model requires normalization, we must normalize the data on a per-driver basis because we are using biometric data. For example, consider a driver with a naturally higher heart rate (possibly due to high blood pressure, etc.) having their data normalized with that of another driver with a lower average heart rate will lead our models astray. As a result, the model may assume the driver with a higher natural heart rate is stressed when they are not.

3.2 Feature Extraction

In order to improve model detection performance, we implement feature extraction over sliding windows of size $ws = 50$. To do so, for a sliding window with a 50% overlap, we construct feature vectors for mean, standard deviation, maximum, and minimum and concatenate all constructed feature vectors to form a new input matrix X_{new} of size $M \times 4D$. In addition to extracting features, we also need to assign new labels due to the reduced number of examples from N to M. First, we take the mean y_μ of a previously prepared label across a window. Then,

Algorithm 1 Feature extraction over sliding windows with 50% overlap

```
procedure FEATURE-EXTRACTION($X, y, ws$)
    $\mu, \sigma, \max, \min$,y$_{new}$ $\leftarrow \emptyset$
    idx $\leftarrow 0$
    for $x_i \in X$ do
        while $idx + ws < len(X)$ do
            $\mu_{x_i} \leftarrow \mu_{x_i} \cup$ X[idx:idx + ws].mean
            $\sigma_{x_i} \leftarrow \sigma_{x_i} \cup$ X[idx:idx + ws].std
            $\max_{x_i} \leftarrow \max_{x_i} \cup$ X[idx:idx + ws].max
            $\min_{x_i} \leftarrow \min_{x_i} \cup$ X[idx:idx + ws].min
            $y_{new} \leftarrow y_{new} \cup$ y[idx:idx + ws].mean
            idx $\leftarrow$ idx + $\frac{ws}{2}$
    X$_{new} \leftarrow \mu \cup \sigma \cup \max \cup \min$
return X$_{new}, y_{new}$
```

depending on the number of classes desired, we can label the extracted mean y_μ to assign a new discrete-valued output accordingly.

3.3 Leave-One-Driver-Out (LODO) Validation

Similar to Leave-One-Out Cross Validation, we propose the Leave-One-Driver-Out (LODO) Validation method. Prior works [19, 10, 3, 21] utilize k-fold (usually 5/10-fold) validation methods when evaluating models and usually lend to very high accuracy ($> 90\%$). K-fold is effective in evaluating overall performance of models in this space. However, it is not effective at demonstrating the generalizability of a model on an unknown passenger or driver. The number of samples in a given drive vary from drive-to-drive and dataset-to-dataset. As a result, creating validation and training sets in a traditional k-fold manner would mean drives may be split (leaving part of the drive in the training set and part of the drive in the validation set). Because we aim to demonstrate generalizability on an unknown subject, it is not effective to treat samples as independent. Instead, we have to treat drives as independent. Biometric data varies from person to person; the model should not be given any prior knowledge of an individual's characteristics. Imagine a scenario in which a drive has been split in the middle, leaving half in the training and half in the validation set. In this instance, the model could potentially learn characteristics of the individual that it will eventually be validated on: expected heart rate range, breathing rate, etc. Methods similar to ours are used by prior works [12, 4]. We extend upon this work by broadening the scope from single ML models to a holistic evaluation of many models.

3.4 Supervised Learning Algorithms

We evaluate our proposed methodology first on supervised learning algorithms, where the algorithm is given the respective labels for a training example. Namely, we evaluate k-Nearest Neighbor (kNN), Support Vector Machine (SVM), Neural Network (NN), and Recurrent Neural Network with Long-Short Term Memory (RNN - LSTM). In the subsequent sections, we provide a brief theoretical description of each algorithm and its respective set-up for our evaluation.

3.4.1 k-Nearest Neighbor

k-Nearest Neighbor is a non-parametric supervised learning algorithm. In kNN classification, a data point is classified by its neighbors' plurality vote, with the data point being assigned to the class most common among its k nearest neighbors [18]. We tuned the number of neighbors (k) to obtain the best LODO validation result. We found that the k that returns the best f1 score is different from the k that returns the best accuracy (Figure 4). In our results, we report the number of neighbors with the highest f1 score since this score is more important for real-world, safe application. The best result is obtained when k neighbors number is $k = 11$ for binary classification, and $k = 20$ for multi-class classification.

3.4.2 Support Vector Machine

Support Vector Machine is a supervised learning algorithm that tries to find the optimal hyperplane in a high-dimensional feature mapping to separate the data [14]. This gives us the objective function $\max C \sum_{n=1}^{N} \xi^{(n)} + \frac{1}{2}||w||^2$

in the form of a Lagrangian function such that we want to maximize the margin with respect to the following constraints: $y^{(n)}h(x^{(n)}) \geq 1 - \xi^{(n)}, \forall n$ and $\xi^{(n)} \geq 0, \forall n$.

The two hyperparameters we tune for SVMs are C and the desired kernel function. We select these hyperparameters by performing LODO experiments with varying values of C and linear and radial-basis kernel function. Consistent with literature in this domain, we use the Radial-Basis Kernel Function and set $C = 10$. In this context, C represents how much we want to avoid misclassifying a training example. Comparably, 10 is not generally considered a large value, and as a result, we may expect some misclassification. However, the model may still generalize soundly, as SVMs have performed historically strong in a variety of domains and classification tasks.

3.4.3 Neural Network

Traditional neural networks (NN) have proved versatile in many fields in machine learning. As a result, we explore their application in this domain as well. Performing a design space exploration to determine the best network architecture led us to use a fully connected two layer model with the ReLU activation function at each node. Each layer was chosen to be of size 32 (64 also performed well). Attempting to use smaller layer sizes led to underfitting while larger layer sizes led to overfitting for both binary and three-class classification.

3.4.4 Recurrent Neural Network with Long-Short Term Memory

Recurrent Neural Networks (RNNs) have proved useful when evaluating time-series data, such as language prediction, language recognition, and stock market analysis. Because driver stress biometric data is inherently time-series based, we expect RNN to perform well in this domain. We utilize a Long-Short Term Memory (LSTM) RNN to provide hidden state memory over time. Note, our final results utilize a single direction LSTM. Bidirectional LSTM was also explored but provided no benefit. Similar to the NN, we tune hyperparameters to achieve the best LODO validation accuracy. Our finalized model uses an LSTM with two hidden layers, each of size 16, and an output fully connected layer on the most recent time-series hidden-layer-output (many-to-one architecture). Our many-to-one architecture observes a sliding window of the previous 50 sample features and makes a prediction about the current driver state upon observing this window. Given the sampling rate of our preprocessed data, this equates to approximately the previous 1-2 seconds of real-time driver data.

3.5 Unsupervised Learning Algorithms

We also evaluate unsupervised learning algorithms, rather untouched in state of the art literature in driver stress detection. Though performance may not surpass, or match, that of supervised learning, clustering techniques may provide valuable insights about the data. Furthermore, because there is a lack of labeled driver stress data, this space will benefit from unsupervised algorithm evaluation.

3.5.1 Nearest Centroid

We employ Nearest Centroid (NC) classification on top of both K-Means and GMM. This strategy allows direct comparison from unsupervised to supervised algorithms, as the clusters generated by K-Means and GMM do not directly map to a classification problem. In NC classification, our data is first clustered with some unsupervised learning technique. Then, each centroid is computed and assigned the mode label of points in that cluster. To predict for a sample, we find the Euclidean distance between that sample and all centroids; we assign the label of the centroid that minimizes distance to the sample. NC classification is more robust to outliers than kNN but is very sensitive to the fit of its upstream unsupervised model [20]. It is also asymptotically at least as cheap as a 1-nearest-neighbor classification model.

3.5.2 K-Means

K-Means is an unsupervised learning algorithm that attempts to partition all samples into k clusters, minimizing the sum of the mean-squared error (MSE) of all clusters. Given S_i is the set of samples assigned to cluster i, the formal objective is to minimize $\Sigma_{i=1}^{k} \Sigma_{j \in S_i} ||j - \mu_i||^2$ where $\mu_i = \frac{1}{|S_i|} \Sigma_{j \in S_i} j$. We apply a mini-batched implementation of K-Means to drastically reduce computation time and achieve comparable results to a standard implementation [17]. We use batch sizes of 10,000 samples and run for 10,000 iterations or until convergence. We used the elbow method to determine the appropriate number of clusters for our dataset (as seen in Figure 2), providing us with the intuition that 7 clusters best describe our natural data. Finally, we apply NC classification as described in subsubsection 3.5.1.

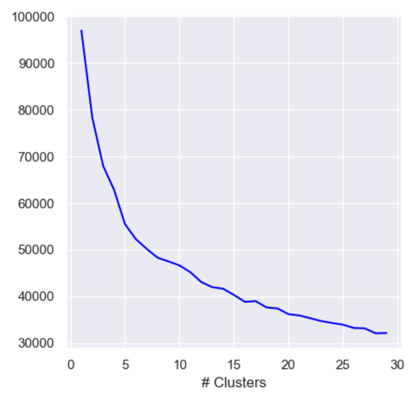

Figure 2: Vanilla elbow graph for K-Means loss (no PCA or feature extraction)

To test this classification strategy, we first set k to the number of clusters learned from the relevant elbow graph (depicted in Figure 5). We then set k to number of classes in the classification problem, where we hope to exploit a potential 1-1 cluster-label mapping (visualized in Figure 6). This is done for each combination of applying feature extraction and a PCA-based dimensionality reduction (reducing to two components). We apply LODO validation to evaluate each of these methods and report the best result for each task.

3.5.3 Gaussian Mixture Model

Gaussian Mixture Model (GMM) with NC is explored for binary classification and three-class classification. We perform experiments with the LODO technique. In particular, we modify the (preprocessed) raw data by PCA and feature extraction. After sufficient experiments, we found that performing PCA does not noticeably improve or degrade the validation accuracy for GMM, while it can degrade the pricision, recall, and f1 scores. However, PCA can reduce the dimensionality of the training dataset so that the standard GMM can save runtime and memory for clustering operations. In practice, we can set `number_of_PCA_components = 2` to reduce the dimensionality from 18 to 2, if one only pursues the highest accuracy. Besides, we also found that, for GMM, performing feature extraction does not saliently increase or decrease the accuracy but decreases the other three, too. A thorough discussion about different validation scores are provided in section 5.3. Our experiments also show that using all 18 features and LODO validation can achieve the best validation scores for GMM (with NC) and all related results are provided in Table 2.

GMM, as an unsupervised method, cannot be directly applied to solve such a classification problem; however, with the help of LODO and NC, we acquire decent validation results as we show in section 4.

3.6 Feasibility Benchmarks

In order to provide some level of feasibility for our machine learning models, we benchmark latency (i.e, detection time). This is important to report as it allows us to assess how fast our models can actually detect appropriate stress levels in a timely (i.e, within 100 ms) manner for support needed to the vehicle, driver, and/or passengers.

4 Experimental Evaluation

4.1 Experimental Setup

We conduct all experiments using Python 3 and Pytorch [16] on an Ubuntu 20.04 machine with eight Intel i5-8250U processors and four cores per socket. Note that our results in Table 2 report an average across all LODO folds for window sizes of 500. Our window size is meant to represent streaming data that is sent to our system in real-time, presumably every 500 samples obtained. Metrics include accuracy, precision, recall, f1 score, and latency. The same metrics are reported in Table 2 for multi-class classification (three classes). Accuracy, precision, recall, and f1 score are calculated as follows:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4}$$

4.2 Results

Our experimental evaluation across all models for binary classification is displayed in Table 2.

	Two Classes					Three Classes				
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**Latency (ms)**	**Accuracy**	**Precision**	**Recall**	**F1**	**Latency (ms)**
kNN	78.78	66.36	58.33	58.22	42.61	58.50	52.27	43.4	42.86	51.18
SVM	76.59	67.42	66.26	66.26	56.59	56.26	49.18	46.04	46.04	23.99
NN	**82.18**	62.73	73.70	**67.77**	20.61	59.13	46.87	50.73	**48.72**	19.98
RNN - LSTM	81.50	59.32	73.16	65.52	40.88	63.70	46.08	42.01	43.95	39.09
K-Means	74.13	66.92	57.51	54.69	**9.71**	58.99	32.98	41.33	33.77	**0.48**
GMM	79.48	62.46	50.45	53.85	20.05	**65.47**	44.91	33.90	36.97	36.75

Table 2: Model Performance Evaluation

We also utilize the neural network to provide relative feature importance rankings for our raw (non-feature extraction) data. This is important information if automakers need to consider the sensor-cost-to-safety ratio.

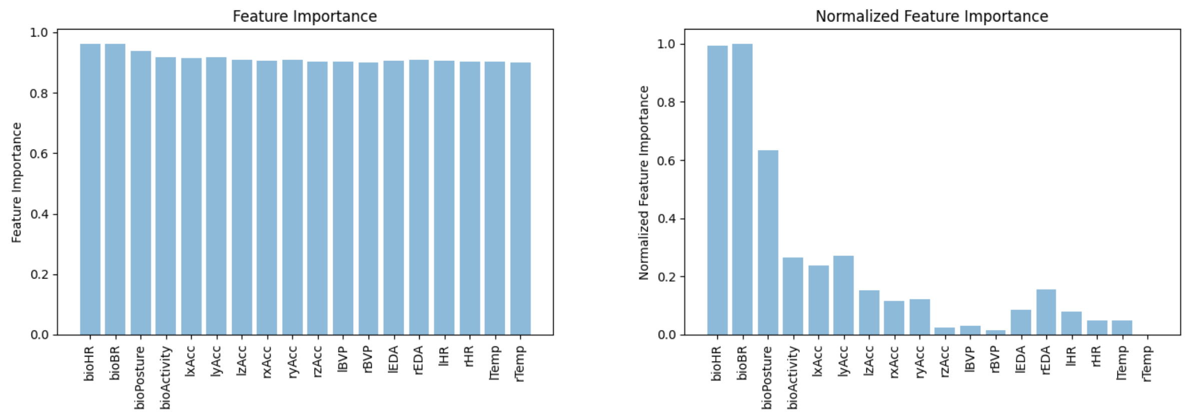

Figure 3: Feature Importance of Raw AffectiveROAD Data

5 Discussion

5.1 Comparison to Related Work

Driver stress recognition research is plagued by several issues that lend difficulty to model comparison: varied datasets between papers, using simulator data vs real-world data, and binary vs multiclass classification. For example, one of the works we studied utilized AffectiveROAD but dropped several features to combine with other datasets, making direct comparisons much less reasonable [10]. In addition, we argue simulator datasets are not sufficient for biometric data collection, as the driver may have different responses when the conditions involve real lives. Finally, as highlighted by our paper, the addition of one extra class impacted performance significantly for all models. This research space has several large issues to solve when it comes to model comparison and evaluation. Much like in the image processing space, this field should maintain standardized databases and testing procedures for all models to reference.

As we argue in subsection 3.3, LODO should become the preferred method of evaluation for models in this space. We compare our results to two papers that utilize similar validation methodology. [12] achieves a binary classification accuracy of 70%, less than several of our models – however, they utilize a concatenation of datasets with HR and BR. Similarly, [4] achieves 65% using HR, BR, and EDA through a combination of several datasets. The discrepancy in accuracies between these works and ours could be described by our use of 18 features and our feature extraction methods.

5.2 Binary Classification vs Multiclass Classification

A majority of the prior work we studied only utilized binary classification (stress vs. no stress) [12, 19, 10]. However, noting the results studied in our work, we can see a dramatic LODO validation accuracy loss upon addition of a third class. Giving applications of stress recognition higher granularity is crucial to appropriately remedy issues arising in the individual. If the driver is at max stress, the vehicle should always recommend pulling the car over. However, without this granularity we simply cannot determine if our driver is under maximum stress or slightly above average stress (i.e. 51% with binary classification). We note that the accuracy achieved by our breadth-based study shows that traditional models perform quite well for binary classification. However, they will not achieve sufficiently high accuracy for multiclass classification using the LODO validation methodology.

5.3 Supervised vs Unsupervised

Supervised learning models rely on labeled training data, while unsupervised learning models can learn without labeled training data. Labeling data is both time and resource consuming; unless a problem is well-enough-studied such that the process of data collection and labeling is streamlined, adequate labeled data is hard to come by.

Specifically, GMM achieves 79% accuracy for binary classification and 65% accuracy for three-class classification, which is the highest among all models we apply. Although GMM is designed to cluster all input data points based on the assumption that the data is generated from a mixture of Gaussian distributions, the results reveal the practicality of solving classification problems (or at least this problem) with the absence of true labels, especially when the data is generated and collected through time-series with potentially large differences in mean and covariances. The highest accuracy for three-class classification shows that the Gaussian distribution may be an appropriate approximation toward the dataset we use.

On the other hand, we also notice that GMM has the average-level precision and the lowest recall for both binary classification and three-class classification. Precision captures the performance of the model when the harm of False Positive (FP) is severe. In our problem, a FP refers to the case that the driver is predicted as stressed while he or she actually is not. In this case, only reasonably good precision is acceptable. However, based on Equation (3), recall best represents the performance of a model where the impact of False Negatives (FN) is severe. In our problem, a FN refers to the case that the driver is predicted as not stressed while he or she actually is; such misclassification can cause potentially fatal consequences. As a result, whether or not GMM, and similar unsupervised techniques, are an appropriate solution to this problem is still worth further study. While the accuracy supports the rationality, the problems related to the precision and recall motivate future research.

5.4 Improvements

Although we have shown that machine learning methods can serve as viable solutions for driver stress recognition, further research may investigate and consider features beyond biometric data (i.e, facial expression video stream) for a combination or fusion of machine and deep learning models (i.e, CNN + multimodal LSTM). On a similar note, unsupervised learning will benefit from further investigation, due to the current lack of adequate labeled data for supervised models. Additionally, we provided a baseline evaluation of standard, common machine learning models. There may exist more complex, state-of-the-art models that fit the context, such as an LSTM Encoder-Decoder, which may be used as an unsupervised method [11].

6 Conclusion

In this work, we have provided a holistic baseline evaluation of supervised and unsupervised learning algorithms via a multi-view learning approach to better support the driver-vehicle interaction for stress detection. It is expected that this work will advance productive discussion and future research on the trade-offs (i.e, detection performance, resource feasibility, labeled data, etc) of various machine learning algorithms for safety critical functionalities regarding driver stress detection and similar affective computing systems.

References

[1] N. H. T. S. Administration et al. 2015 motor vehicle crashes: overview. *Traffic safety facts: research note*, 2016:1–9, 2016.

[2] P. Agarwal and M. Alam. A lightweight deep learning model for human activity recognition on edge devices. *Procedia Computer Science*, 167:2364–2373, 2020.

[3] J. Azar, A. Makhoul, M. Barhamgi, and R. Couturier. An energy efficient iot data compression approach for edge machine learning. *Future Generation Computer Systems*, 96:168–175, 2019.

[4] S. Bianco, P. Napoletano, and R. Schettini. Multimodal car driver stress recognition. In *Proceedings of the 13th EAI International Conference on Pervasive Computing Technologies for Healthcare*, pages 302–307, 2019.

[5] N. E. Haouij, J.-M. Poggi, S. Sevestre-Ghalila, R. Ghozi, and M. Jaïdane. Affectiveroad system and database to assess driver's attention. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*, SAC '18, page 800–803, New York, NY, USA, 2018. Association for Computing Machinery.

[6] J. Healey and R. Picard. Smartcar: detecting driver stress. In *Proceedings 15th International Conference on Pattern Recognition. ICPR-2000*, volume 4, pages 218–221. IEEE, 2000.

[7] J. A. Healey and R. W. Picard. Detecting stress during real-world driving tasks using physiological sensors. *IEEE Transactions on intelligent transportation systems*, 6(2):156–166, 2005.

[8] D. L. Hendricks, M. Freedman, J. C. Fell, et al. The relative frequency of unsafe driving acts in serious traffic crashes. Technical report, United States. National Highway Traffic Safety Administration, 2001.

[9] J. Lee, H. Lee, and M. Shin. Driving stress detection using multimodal convolutional neural networks with nonlinear representation of short-term physiological signals. *Sensors (Basel, Switzerland)*, 21(7), March 2021.

[10] D. Lopez-Martinez, N. El-Haouij, and R. Picard. Detection of real-world driving-induced affective state using physiological signals and multi-view multi-task machine learning. In *2019 8th International Conference on Affective Computing and Intelligent Interaction Workshops and Demos (ACIIW)*, pages 356–361. IEEE, 2019.

[11] P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff. Lstm-based encoder-decoder for multi-sensor anomaly detection. *arXiv preprint arXiv:1607.00148*, 2016.

[12] P. Napoletano and S. Rossi. Combining heart and breathing rate for car driver stress recognition. In *2018 IEEE 8th International Conference on Consumer Electronics-Berlin (ICCE-Berlin)*, pages 1–5. IEEE, 2018.

[13] F. Nie, G. Cai, J. Li, and X. Li. Auto-weighted multi-view learning for image clustering and semi-supervised classification. *IEEE Transactions on Image Processing*, 27(3):1501–1511, 2017.

[14] W. S. Noble. What is a support vector machine? *Nature biotechnology*, 24(12):1565–1567, 2006.

[15] M. Paschero, G. Del Vescovo, L. Benucci, A. Rizzi, M. Santello, G. Fabbri, and F. F. Mascioli. A real time classifier for emotion and stress recognition in a vehicle driver. In *2012 IEEE international symposium on industrial electronics*, pages 1690–1695. IEEE, 2012.

[16] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *arXiv preprint arXiv:1912.01703*, 2019.

[17] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[18] L. E. Peterson. K-nearest neighbor. *Scholarpedia*, 4(2):1883, 2009.

[19] A. Saeed and S. Trajanovski. Personalized driver stress detection with multi-task neural networks using physiological signals. *arXiv preprint arXiv:1711.06116*, 2017.

[20] S. Tan. An improved centroid classifier for text categorization. *Expert Systems with Applications*, 35(1-2):279–285, 2008.

[21] J.-S. Wang, C.-W. Lin, and Y.-T. C. Yang. A k-nearest-neighbor classifier with heart rate variability feature-based transformation algorithm for driving stress recognition. *Neurocomputing*, 116:136–143, 2013.

[22] C. Xu, D. Tao, and C. Xu. A survey on multi-view learning. *arXiv preprint arXiv:1304.5634*, 2013.

7 Appendix

See Figure 4 for our experimental results to determine the optimal k for kNN. See Figure 5 for our loss plots of kMeans with PCA and feature extraction. Lastly, see Figure 6 for scatter plots of each task-feature extraction combination for our unsupervised learning methods.

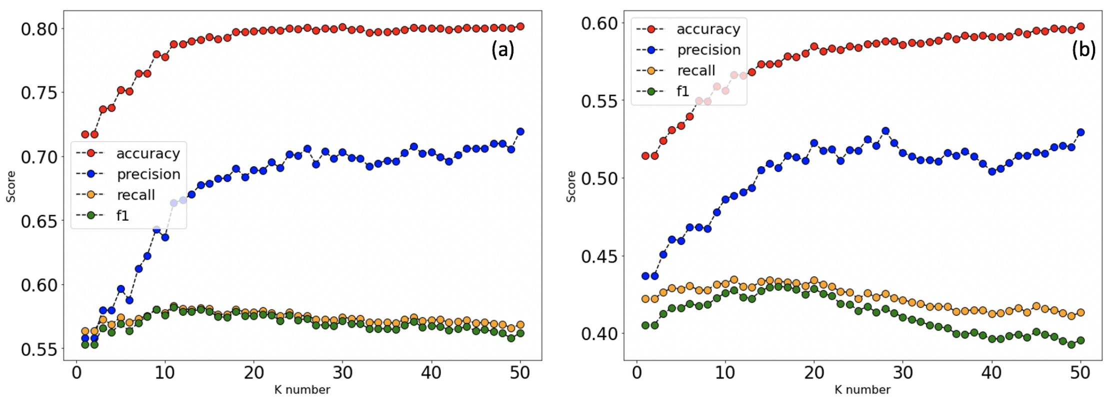

Figure 4: (a) Binary (2 classes), (b) Multi-Class (3 classes)

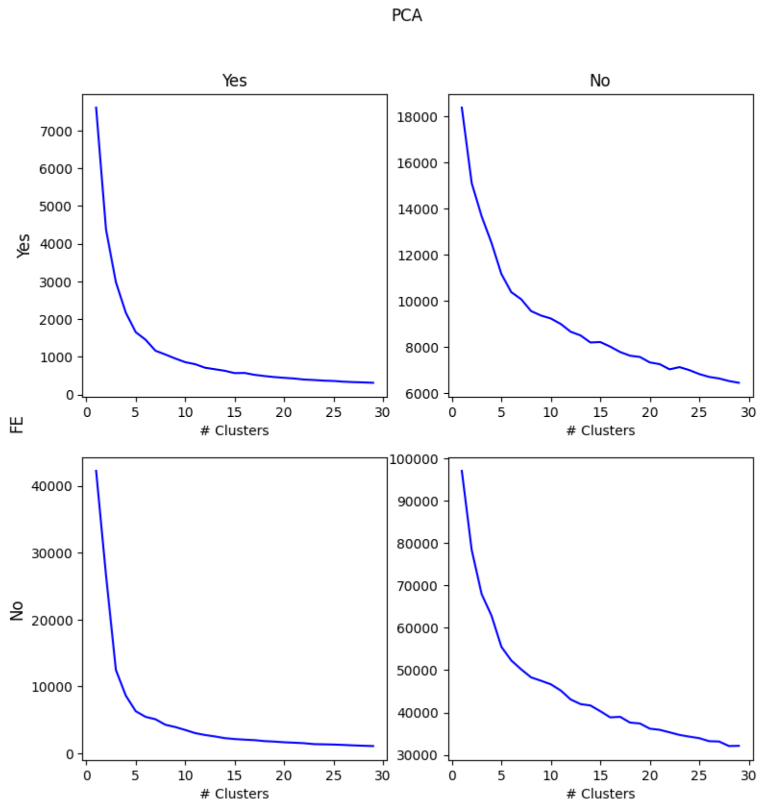

Figure 5: K-Means elbow graphs for each combination of PCA and feature extraction. The y axis measures MSE.

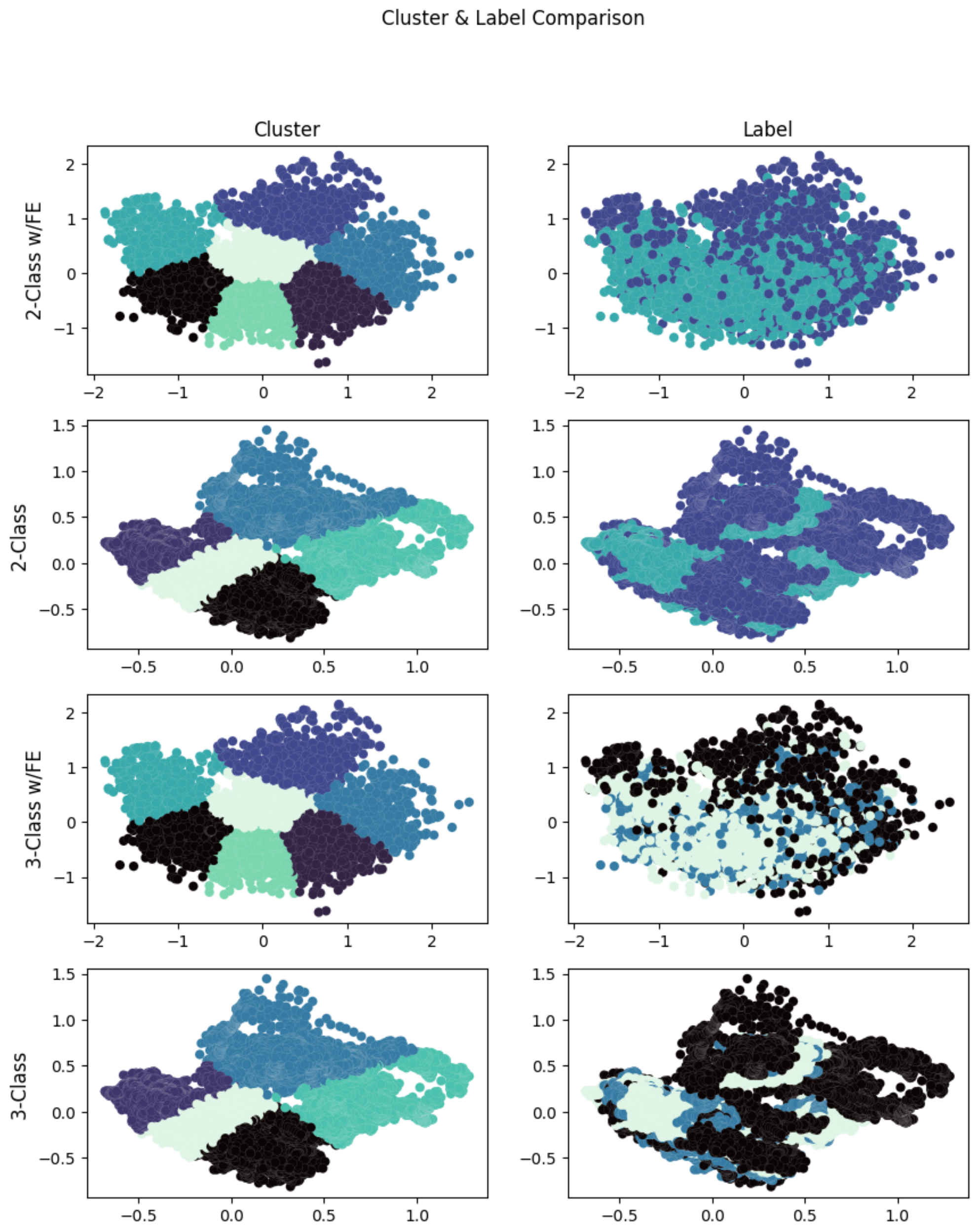

Figure 6: Scatter plots for each task-feature extraction combination (PCA is applied to generate the visualization in all cases). The number of clusters is learned from the elbow method. There are few, if any, unambiguous cluster-label mappings.