VAE-Guided Testing Framework for OpenPilot's Vision Model

CHRISTOPHER MORSE, University of Virginia, USA

The control of autonomous driving systems is largely based on their visual predictions. However, due to the uncertainty and complexity of real-world driving environments, these systems will need to safely handle unfamiliar inputs. Otherwise, a misinterpretation of these inputs can lead to costly real-world misbehaviors. We examine the vision model from *OpenPilot*, an open-source autonomous driving system that has been widely deployed in recent years. Since the majority of their training data has been collected from users on highways, we suspect that there are underrepresented features in the training set that have led to the insufficient training of their vision model. To detect these high-risk features, we introduce a VAE-guided approach for the extraction of rare features from OpenPilot's training set and a framework for the independent testing of their vision model. Our results suggest that there are rare features that cause uncertainty in OpenPilot's visual predictions for their Automated Lane Centering (ALC) system.

1 INTRODUCTION

Data scarcity and domain coverage are two of the most prominent limitations to the advancement of deep learning technologies [2]. Improperly trained models, particularly those in physical robot systems, pose serious risks that are highlighted by Sunderhauf et al. [11]. At its core, these authors demand for the distinction between *computer vision* and *robotic vision*, because visual inputs for robot systems are typically translated into physical actions, a property that is not true for all computer vision applications. Therefore, the misinterpretation of these inputs can lead to costly real-world misbehaviors.

Recently, these deep-learned vision models have become increasingly prevalent in autonomous driving systems. CommaAI's OpenPilot is an open-source autonomous driving system that utilizes a mounted camera and built-in radar sensors to provide *Level 2* driver assistance for limited autonomous control of the vehicle. Specifically, OpenPilot is comprised of adaptive cruise control (ACC) and automated lane-centering (ALC) systems [3]. While OpenPilot has noted that their users have collectively driven with their system for over 40 million miles, this is an incomplete metric to determine the quality of their vision model. The vast majority of their data, used to train and test their deep-learned components, have been collected from users driving on highways. This means that their dataset is naturally diluted with common road features, yet the OpenPilot team publicly stated that they *"believe this dataset to be a fairly complete representation of the driving problem"* [4]. Due to the unpredictability and complexity of the real world, this "driving problem" is not trivial. Therefore, to better assess their vision model's performance, we need to evaluate its ability to generalize across traffic features that it is unfamiliar with (i.e. *rare* features).

This work introduces a pipeline to automatically extract *rare* features from the OpenPilot training set to expose a subset of the domain for which the model was insufficiently trained. [1] Our approach uses a Variational Autoencoder (VAE) to learn the latent distribution of the OpenPilot training set. Afterward, we extract high-loss images and utilize K-Means Clustering on their latent representations to cluster images based on their rare features. To evaluate OpenPilot's vision model (named *Supercombo*), we construct a pipeline to independently test its lane confidence on videos containing the clustered rare features and examine the results.

It is worthy to note that many existing approaches that learn a distribution from traffic image datasets are inapplicable for extracting rare features from OpenPilot's dataset. One approach offers a trained VAE for traffic images, but the architecture was constructed for a dataset of much smaller images, from which many of the reconstructed features are

[1] For fairness, we respect OpenPilot's stated operational design domain in our evaluations.

ambiguous [12]. Other approaches capture lower-level rare features that are not helpful for testing the OpenPilot vision model [1]. In the following section, these approaches are described in greater detail.

This work highlights the importance of greater domain coverage in training sets for vision models to assist developers (particularly those at OpenPilot) in augmenting their training sets for future models. In the future, for autonomous robotics to operate safely in the real world, it is crucial for developers to identify features that their models may not have been properly trained on.

The primary contributions of this work are:

- An automated pipeline for the extraction of rare features from OpenPilot's training set.
- A framework for the independent testing of OpenPilot's vision model on videos.
- An evaluation and analysis of the strengths and limitations of OpenPilot's vision model.

2 RELATED WORK

We have split existing work into two categories: the modeling of image data distributions and the testing of OpenPilot's vision model.

2.1 Modeling of Image Data Distributions

The modeling of image data distributions is a popular problem that has been widely studied. The authors in [9] provide insight into techniques used for the deep-learned modeling of image data distributions. These approaches seek to learn a low-dimensional latent representation of the input data, and are typically based on the Variational Autoencoder (VAE) or Generative Adversarial Network (GAN) architectures. These are often used for deep generative modeling, which has been used to target the data distributions for semantic data augmentation [13], DNN verification and falsification [12], and latent semantics [10].

Other approaches have used these learned distribution models as anomaly detectors. The authors from [1] train a VAE to learn latent distributions from dashcam image datasets and characterize rare features for the purpose of balancing and de-biasing training sets. However, their approach is limited in that its learned features primarily focus on the low-level road topography (i.e. road curvature), rather than a broad set of rich, rare features. To the best of our knowledge, there is no existing work that automatically extracts rare features from OpenPilot's training set for the purpose of targeted testing.

2.2 OpenPilot Testing Pipelines

In recent years, two methods for the independent testing of OpenPilot's vision model on videos have been developed. The original pipeline [6] is functional, but depends on an old, Keras-based version of Openpilot's vision model. Therefore, it cannot be used with the newer ONNX-based versions of the model. A more recent pipeline was built upon the aforementioned approach, such that it can be applied to ONNX-based models [7]. However, their code is incomplete, in that it incorrectly parses the model's output. Furthermore, OpenPilot recently released a new version of their trained ONNX model, whose inputs are no longer compatible with these existing testing pipelines. While our pipeline is based on these approaches, it features a new parser and is compatible with the latest version of the vision model.

3 BACKGROUND

At a high level, a standard Variational Autoencoder (VAE) architecture is comprised of an encoder network and a decoder network. The goal of the encoder network is to convert the input image into a much smaller, latent representation. As it is trained, the encoder learns to distinguish between relevant and irrelevant information from an input image, and tries to preserve the most important information in the encoded vector. Due to the lower dimensionality of the latent space, the encoder will also need to decide which information from the input is likely to be irrelevant, and can be excluded from the latent representation. As the decoder network is trained, it learns to convert this smaller representation back to the original image. A standard loss function for a VAE consists of two parts: *reconstruction error* and *KL divergence*. Reconstruction error measures the dissimilarity between the original image and the reconstructed image, which is typically computed by Mean Squared Error (MSE) or L1 loss. KL divergence measures the distance between the learned latent distribution and the ground-truth distribution.

In contrast to traditional autoencoders, whose encoder networks output a single latent representation of the input, a VAE's encoder network outputs a mean vector and a standard deviation vector. Together, this pair of vectors describes the distribution of the latent space, from which single latent vectors can be sampled. Since this distribution is continuous, sampling near the means of each variable in the latent space generates new in-distribution images. However, due to the high dimensionality of these latent spaces, strategic exploration is difficult (e.g. targeting specific image features, or the "tails" of these distributions). The *manifold hypothesis*, which states that high-dimensional data tend to reside in a space of much fewer dimensions, has been considered to help ease these issues. Some approaches have tried to use this notion to interpret variables in the latent space [10] or perform targeted semantic transformations on images [13], but their approaches are limited by the complexity of the domain and their dependence on massive, annotated datasets.

4 APPROACH

Our approach is composed of three distinct components. First, we train a Variational Autoencoder (VAE) to learn to model the distribution of images in the *Supercombo* training set. Second, each image in the dataset is ranked based on its reconstruction loss, and the high-loss images are clustered by their features. Third, the *Supercombo* model is evaluated through our pipeline on videos that contain these features.

4.1 Modeling the Training Set Distribution

We use a VAE to model the distribution of the *Supercombo* training set. This data set contains over 220,000 road images of size (1164×1080) [5], which were converted to single-channel images of size (392×392) prior to training. To select our target VAE model, we trained 6 models of various latent dimensions and architectures, based on the code provided by Toledo et al. [12]. Each model was trained for 100 epochs with a batch size of 64. After training, we evaluated each model's average reconstruction loss over a test set of 15,000 images. These models and results are provided in Table 1. We selected *Model2* as our target model, since it had the lowest average reconstruction loss over the test set. The general architecture of our model's encoder and decoder networks are shown in Figure 1. While omitted from the figure for clarity, there are batch normalization and ELU layers between each of the convolutional layers.

To explore the latent features that were learned by our model, each of the 512 latent dimensions were explored individually. As shown in Figure 2, many of these latent variables correspond to specific features (e.g. vehicles, tree

Model	# Latent Dimensions	Encoder: # Conv / # Total Layers	Decoder: # Conv / # Total Layers	Average Reconstruction Loss
Model1	256	6 / 18	8 / 24	0.0197
Model2	512	6 / 18	8 / 24	**0.0184**
Model3	1024	6 / 18	8 / 24	0.0219
Model4	256	7 / 21	9 / 27	0.0322
Model5	512	7 / 21	9 / 27	0.0268
Model6	1024	7 / 21	9 / 27	0.0707

Table 1. Trained VAE Models.

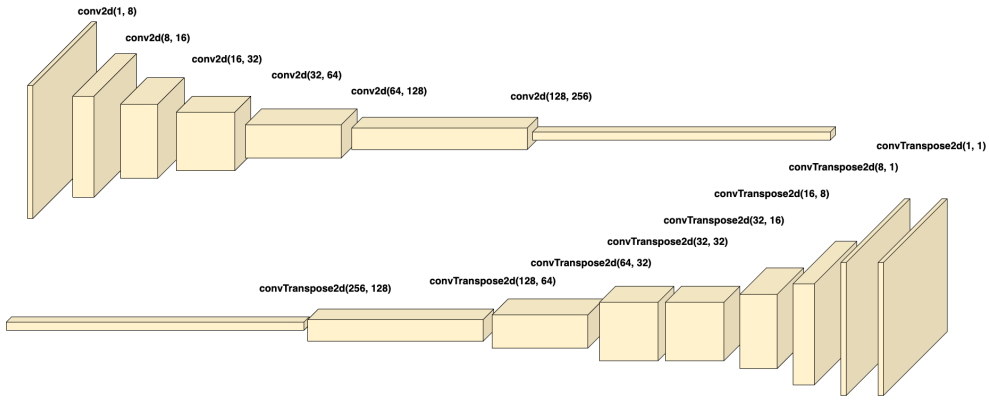

Fig. 1. Our VAE encoder and decoder architectures. Note that the batch normalization and ELU layers between each convolutional layer are omitted for clarity.

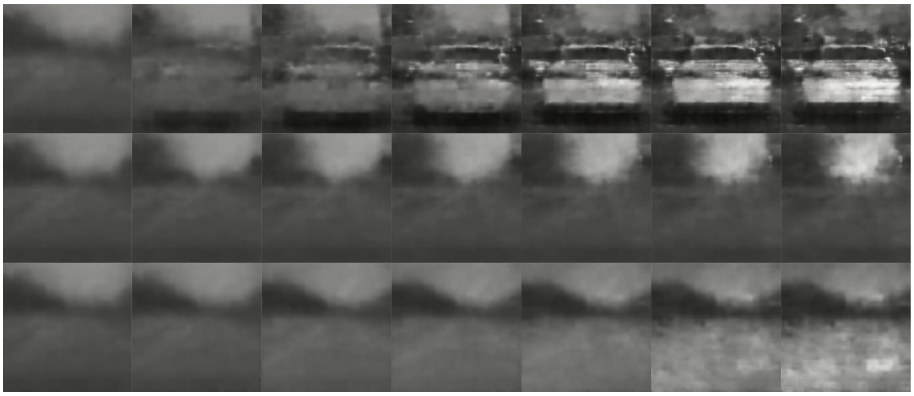

Fig. 2. Latent variable exploration. Features include car (top), tree canopy (middle), and road illumination (bottom).

coverage, road illumination). These qualitative results demonstrate that the VAE has learned a model of the image distribution, such that discernible road image features can be reconstructed from the sampled latent vectors.

4.2 Rare Feature Extraction

With a learned model of the image distribution, the next step in our approach is to extract rare features from the training set. The first step in this procedure is largely based on existing work [8], which uses MSE-based reconstruction losses to classify images with high reconstruction loss as *rare*. This technique operates under several assumptions. First, it assumes that the VAE is broadly worse at encoding and decoding *rare* image features than it is with common features. Second, it requires the image features to be large enough to have a substantial effect on the resulting MSE. Third, it assumes that the VAE is not significantly overfit to the training set. As noted by Richter et al. [8], this reconstruction-based approach for anomaly detection tends to work well for image sets of similar structure (e.g. MNIST), but poorly for those with highly dissimilar images (e.g. ImageNet). For our approach, we assume that the images in the Supercombo training set are adequately similar in structure.

Under these considerations, we ran the entire training set through the trained VAE and computed their reconstruction losses. We then arbitrarily extracted the top 0.6% of images with the highest reconstruction losses and formed a dataset containing the latent vectors for each of these rare images. Since we suspect that this set contains multiple rare features, we chose to perform K-Means clustering on the latent vectors. The K-Means algorithm is an iterative approach for locating optimal cluster centers, and assigning each data point a cluster based on its nearest cluster center. However, K-Means has two primary challenges regarding the optimization of its clustering: high dimensionality and the selection of k (the desired quantity of clusters). First, since the K-Means algorithm clusters data points based on euclidean distance, it will struggle to extract distinct clusters from high dimensional data. To help mitigate this issue, we examine two dimensionality reduction techniques – Principal Component Analysis (PCA) and Truncated Singular Value Decomposition (TruncatedSVD) – to extract the most important components from the data. Our goal with dimensionality reduction is to reduce the quantity of redundant variables in the latent vectors, thereby allowing for K-Means to generate more distinct clusters. The second challenge regarding the K-Means algorithm is that it requires the desired number of clusters (k) to be specified prior to execution. Popular methods for estimating the number of clusters include the *elbow* and *silhouette score* methods. These methods informed our selection for k to be 5 and 7 for the TruncatedSVD and PCA methods, respectively. More detailed definitions and information about the selection of these parameters are provided in Section 5.

Figure 3 shows clusters generated by K-Means from the PCA reduction (top row) and the TruncatedSVD reduction (bottom row). Each row contains the two clusters with the highest silhouette scores. Through manual inspection, we find that K-Means struggles to form meaningful clusters from the PCA reduction. However, the TruncatedSVD reduction resulted in clusters containing bridges (left) and light glare (right) as rare features from the OpenPilot training set. We then manually compiled videos with and without these rare features, to be evaluated by the vision model.[2]

4.3 Evaluation Pipeline

The OpenPilot system is comprised of several complex and interconnected components, so the high-level analysis of the system's inputs and outputs would fail to target specific component-level failures. For this reason, we developed a framework to independently test the Supercombo vision model on videos. Supercombo accepts two pairs of images, a desire state, a recurrent state vector, and traffic convention as input. Inspired by existing pipelines for older versions of

[2]It is worthy to note that this manual inspection has a human bias, and these features may be entirely independent from how the clusters were formed. For future work, we suggest to reduce this bias by learning the semantics of each latent variable.

Fig. 3. Samples from the two highest-scoring K-Means clusters from the PCA reduction approach (top) and the TruncatedSVD reduction approach (bottom)

the model [6][7], our approach starts by parsing the input video into image pairs, and initializes the remaining inputs with dummy values. For each sequential image pair in the video, the input is fed through the model and the output is parsed. In this output, the model provides information regarding the state of the ego vehicle, lane lines, road edges, and lead vehicles, along with their associated probabilities.

5 EVALUATION

This section provides an evaluation of our approach for rare feature extraction, along with our examination of OpenPilot's Supercombo model on these rare scenarios. Our evaluation targets the following research questions:

(1) Is our method for extracting distinguishable rare features in OpenPilot's training set effective?
(2) Will Supercombo produce low-confidence or incorrect predictions on scenarios containing these rare features, or does it generalize well across this unfamiliar data?

5.1 RQ1: Rare Feature Extraction

To assess the validity in our choices for rare feature extraction, we provide an analysis for each step. To determine the optimal number of components for PCA-based dimensionality reduction, we apply the PCA transformation to our dataset of latent vectors. We then compute the amount of variance from the dataset that each component captures, and form a curve based on the accumulation of these values across the first N components (See Figure 4). This curve suggests that approximately 87% of the variance in the dataset is captured by the first 9 PCA components. This quantity was selected as a trade off between dimensionality reduction and the preservation of information from the latent vectors.

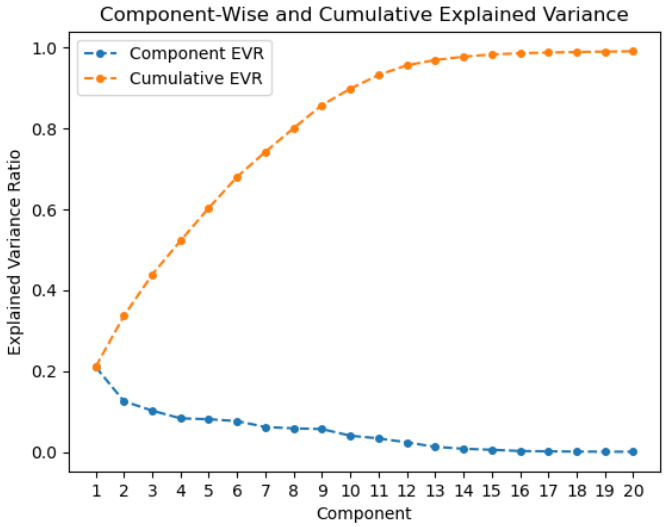

Fig. 4. Component-wise explained variance and cumulative explained variance.

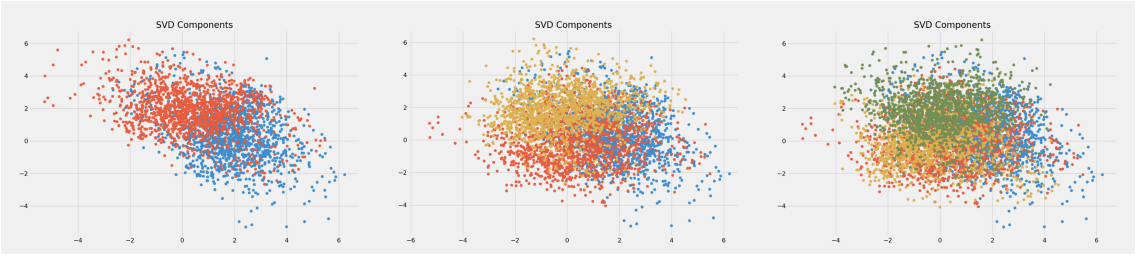

Fig. 5. Scatter plots of n TruncatedSVD components (left to right: n = 2, 3, 4)

Our method for selecting the optimal number of components for TruncatedSVM-based dimensionality reduction was largely based on qualitative analysis. In Figure 5, we provide scatter plots of TruncatedSVD components, for various quantities. Visually, we observe a relatively clear boundary between clusters for n = 2 components, but it captures less of the data. Conversely, for n = 4 components, the boundary between clusters is ambiguous. Therefore, we selected 3 components for our TruncatedSVD reduction.

To determine the optimal number of clusters to produce from K-Means, we gauge cluster quality with two popular methods: the *elbow method* and the *silhouette score*. The *elbow method* is a technique for analyzing the Within-Cluster Sum of Squares (WCSS) for a range of cluster quantities. To compute the WCSS of a set of clusters, we compute the mean distance between all points in a cluster and the cluster center, and average this value across all clusters. In essence, WCSS measures how "compact" the set of clusters are. For a set of cluster quantities (k), the associated WCSS scores are

provided in Figure 6 for both the TruncatedSVD and PCA approaches. Considering the bend in each of the curves, this method suggests that 5 and 7 clusters would be optimal for TruncatedSVD and PCA, respectively.

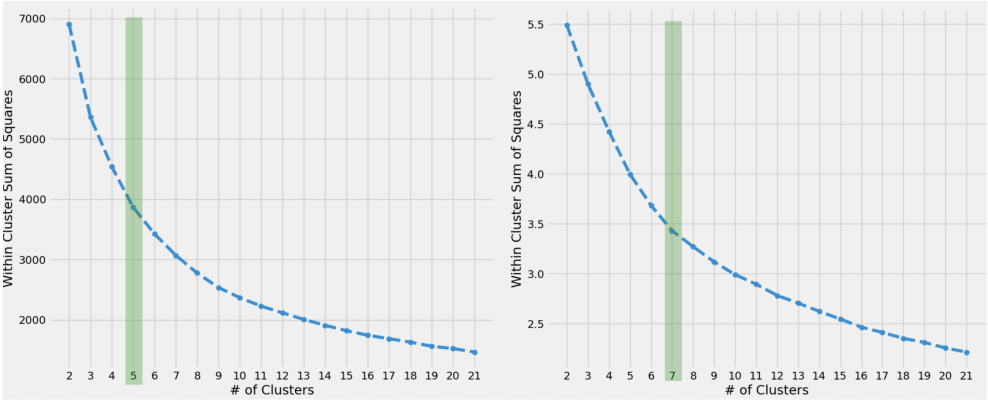

Fig. 6. Elbow plots for TruncatedSVD (left) and PCA (right). The highlighted region designates the optimal number of clusters for each method.

To give more credibility to this selection for k, we also evaluate the optimal number of clusters based on the average *silhouette score* across clusters. The *silhouette score* of a data point measures how well it fits into its assigned cluster. This is measured as:

$$\frac{\beta - \alpha}{max(\alpha, \beta)},$$

where α is a data point's average distance from all other points within its own cluster, and β is the same data point's average distance from all points in the next-nearest cluster. To compute the silhouette score for a cluster, we simply average the silhouette score for each of its data points. Low scores (i.e. near 0) signify that cluster membership is ambiguous, and high scores (i.e. near 1) signify that there are clear memberships to each cluster. These results are provided in Figure 7 for various cluster sizes. Each horizontal line corresponds to a single data point, and each colored block represents a cluster. The red vertical line marks the average silhouette score across all clusters. While the average scores are similar between each value of k, the silhouette methods suggests that the optimal number of clusters are 5 and 7 for the TruncatedSVD and PCA approaches, respectively. We also observe that the average silhouette score for TruncatedSVD is 0.261, while the average silhouette score for PCA is 0.159. This shows that there is slightly less ambiguity in the cluster membership for the data points from the TruncatedSVD method than for PCA.

Because both the elbow and silhouette methods suggest 5 and 7 clusters for TruncatedSVD and PCA approaches, respectively, these were our chosen k values for the K-Means clustering. Given by the results in Figure 3, we qualitatively determine that transforming the latent space with 3-component TruncatedSVD reduction and forming 5 clusters with K-Means is an adequate method for rare feature extraction in this domain.

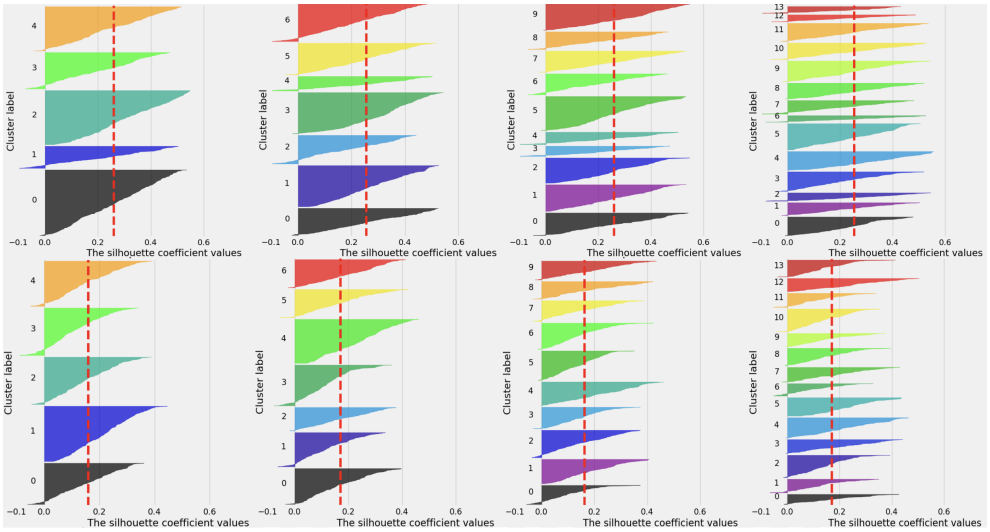

Fig. 7. Silhouette scores for TruncatedSVD (top row) and PCA (bottom row), while varying k (left to right: k = 5, 7, 10, 14).

5.2 RQ2: Supercombo Model Evaluation

Next, we evaluate the Supercombo model's ability to generalize well over features that were underrepresented in its training set. To do so, we use our testing pipeline with manually-collected videos and extract the model's lane confidence (i.e. the model's confidence that *any* lane line exists in the frame). We targeted test videos that the rare features found in the feature clusters (i.e. *bridge* and *glare*). We also evaluate the model on a scenario in which a car is making a turn along a single lane line. While this feature was not extracted by the clustering, we noticed that it was a prominent feature in our assessment for rarity. Table 2 shows the results for each scenario. Here, we manually partition each video into segments with and without the rare feature (See Figure 8), and compute the minimum, maximum, and average confidences within these partitions. These results show that the bridge and glare scenarios had a negligible effect on the lane confidence scores. However, the turn scenario resulted in a significant decrease in lane confidence. Overall, these results show that the Supercombo model generalizes well over some rare features, but not all. Therefore, further testing is necessary to determine the quality of its training.

6 CONCLUSION

This work introduces a pipeline to automatically extract *rare* features from the OpenPilot vision model's training set to expose insufficiencies in how it was trained. First, we trained a VAE to learn to model the distribution of the *Supercombo* training set. We then ranked each image in the dataset based on its reconstruction loss and extracted the "rarest" images. Afterward, we performed dimensionality reduction on the latent vectors and clustered these representations with K-Means, to better guide the manual selection of test videos. Finally, we evaluated the *Supercombo* model through our independent testing pipeline on these videos. We find that our approach has some success in clustering images based on their rare features, but these features have a negligible effect on the resulting lane confidence. Through

Scenario	Metric	Lane Confidence (No Feature)	Lane Confidence (Feature)	Delta
Bridge 1	Average	99.16%	99.10%	−0.06%
	Maximum	99.89%	99.95%	+0.06%
	Minimum	95.81%	97.01%	+1.20%
Bridge 2	Average	99.63%	99.13%	−0.50%
	Maximum	99.87%	99.84%	−0.03%
	Minimum	98.91%	97.96%	−0.95%
Bridge 3	Average	99.63%	99.14%	−0.49%
	Maximum	99.93%	99.94%	+0.01%
	Minimum	98.76%	97.05%	−1.71%
Glare 1	Average	97.04%	98.94%	+1.90%
	Maximum	99.84%	99.89%	+0.05%
	Minimum	91.87%	91.96%	+0.09%
Glare 2	Average	99.39%	99.23%	−0.16%
	Maximum	99.95%	99.86%	−0.09%
	Minimum	96.46%	95.21%	−1.25%
Glare 3	Average	99.60%	99.46%	−0.14%
	Maximum	99.96%	99.96%	−0.00%
	Minimum	95.99%	94.90%	−1.09%
Turn 1	Average	99.40%	92.85%	−6.55%
	Maximum	99.81%	97.72%	−2.09%
	Minimum	97.64%	83.25%	**−14.39%**
Turn 2	Average	99.14%	85.70%	**−13.44%**
	Maximum	99.85%	98.80%	−1.05%
	Minimum	92.79%	67.17%	**−25.62%**
Turn 3	Average	99.64%	94.08%	−5.56%
	Maximum	99.90%	98.78%	−1.12%
	Minimum	99.04%	87.10%	**−11.94%**

Table 2. Lane confidence results for the each of the scenarios: *Bridge*, *Glare*, and *Turn*.

manual clustering, we find that the system is vulnerable to moderate turns, and can cause as much as a 25.62% drop in confidence.

There are various limitations to our approach that call for future work. First, our evaluations do not independently test the presence and absence of these rare features. For this reason, there could be other variables that simultaneously hinder the model's confidence. Second, this work does not provide insight into how the vehicle will react to these detections. For future work, we would like to determine whether or not these sudden changes in confidence cause OpenPilot to disengage. Finally, we acknowledge that there are several assumptions and biases within our approach. Extended quantitative analyses and more advanced metrics for reconstruction could help to limit the effects of these limitations.

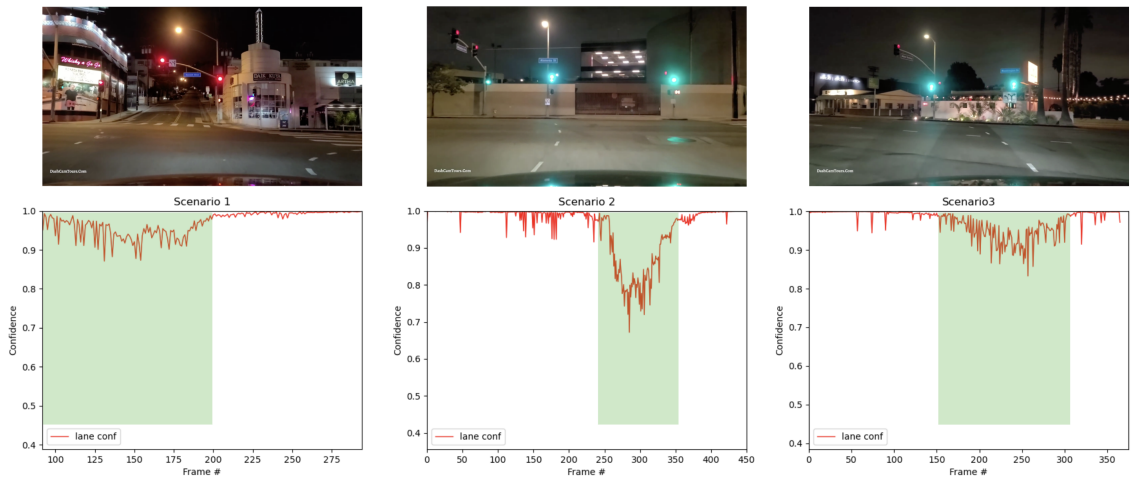

Fig. 8. Images from the *Turn scenario* (top) and corresponding lane confidences (bottom). The shaded regions contain the timesteps in which the vehicle was making the turn.

REFERENCES

[1] Alexander Amini, Wilko Schwarting, Guy Rosman, Brandon Araki, Sertac Karaman, and Daniela Rus. 2018. Variational Autoencoder for End-to-End Control of Autonomous Driving with Novelty Detection and Training De-biasing. *IEEE/RSJ International Conference on Intelligent Robots and Systems* (October 2018).

[2] Daniel Camilleri and Tony Prescott. 2017. Analysing the Limitations of Deep Learning for Developmental Robotics. In *Biomimetic and Biohybrid Systems*, Michael Mangan, Mark Cutkosky, Anna Mura, Paul F.M.J. Verschure, Tony Prescott, and Nathan Lepora (Eds.). Springer International Publishing, Cham, 86–94.

[3] comma.ai. 2016. OpenPilot. https://github.com/commaai/openpilot.

[4] comma.ai. 2020. Towards a Superhuman Driving Agent. https://comma-ai.medium.com/towards-a-superhuman-driving-agent-1f7391e2e8ec

[5] comma.ai. 2021. Comma2k19. https://github.com/commaai/comma2k19.

[6] littlemountainman. 2020. Self Driving Car Lane and Path Detection. https://github.com/littlemountainman/modeld.

[7] MTammvee. 2021. OpenPilot Supercombo Model Deployment. https://github.com/MTammvee/openpilot-supercombo-model.

[8] Charles Richter and Nicholas Roy. 2017. Safe Visual Navigation via Deep Learning and Novelty Detection. *Robots: Science and Systems* (July 2017).

[9] Lukas Ruff, Jacob R. Kauffmann, Robert A. Vandermeulen, Gregoire Montavon, Wojciech Samek, Marius Kloft, Thomas G. Dietterich, and Klaus-Robert Müller. 2021. A Unifying Review of Deep and Shallow Anomaly Detection. *Proc. IEEE* 109, 5 (February 2021), 1–40.

[10] Yujun Shen, Jinjin Gu, Xiaoou Tang, and Bolei Zhou. 2020. Interpreting the Latent Space of GANs for Semantic Face Editing. *Computer Vision and Pattern Recognition* (March 2020).

[11] Niko Sünderhauf, Oliver Brock, Walter Scheirer, Raia Hadsell, Dieter Fox, Jürgen Leitner, Ben Upcroft, Pieter Abbeel, Wolfram Burgard, Michael Milford, and Peter Corke. 2018. The Limits and Potentials of Deep Learning for Robotics. *International Journal of Robotics Research* 37, 4-5 (April 2018), 405–420.

[12] Felipe Toledo, David Shriver, Sebastian Elbaum, and Matthew B. Dwyer. 2021. Distribution Models for Falsification and Verification of DNNs. *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)* (2021), 317–329.

[13] Yulin Wang, Gao Huang, Shiji Song, Xuran Pan, Yitong Xia, and Cheng Wu. 2021. Regularizing Deep Networks with Semantic Data Augmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* (June 2021).