Importance Weighting of Diagnostic Trouble Codes for Anomaly Detection

Diogo R. Ferreira[1], Teresa Scholz[2], and Rune Prytz[2]

[1] IST, University of Lisbon, Portugal
diogo.ferreira@tecnico.ulisboa.pt
[2] Stratio Automotive, Lisbon, Portugal
rune@stratioautomotive.com

Abstract. Diagnostic Trouble Codes (DTCs) allow monitoring a wide range of fault conditions in heavy trucks. Ideally, a perfectly healthy vehicle should run without any active DTCs; in practice, vehicles often run with some active DTCs even though this does not pose a threat to their normal operation. When a DTC becomes active, it is therefore unclear whether it should be ignored or considered as a serious issue. Recent approaches in machine learning, such as training Variational Autoencoders (VAEs) for anomaly detection, do not help in this respect, for a number of reasons that we discuss based on actual experiments. In particular, a VAE tends to learn that a frequently active DTC is of no importance, when in fact it should not be dismissed completely; instead, such DTC should be assigned a relative weight that is smaller but still non-negligible when compared to other, more serious DTCs. In this work, we present an approach to measure the relative weights of active DTCs in a way that allows the analyst to prioritize them and focus on the most important ones. The approach is based on the concept of binary cross-entropy, and finds application not only in the analysis of DTCs from a single vehicle, but also in monitoring active DTCs across an entire fleet.

Keywords: Automotive Industry · Anomaly Detection · Variational Autoencoders · Binary Cross-Entropy

1 Introduction

In the automotive industry, Diagnostic Trouble Codes (DTCs) [1] are a standard way to monitor the health of a vehicle. This applies not only to automobiles that people use routinely for their transportation needs, but also to other classes of road vehicles, namely heavy-duty trucks.

Basically, a DTC is an alarm that is activated under certain operating conditions. For example, in a diesel engine it is important to maintain a certain fuel pressure. Since the fuel goes through several filters, pumps and injectors, a low fuel pressure may cause damage to these components and ultimately to the engine itself. Therefore, if the fuel pressure drops below a certain threshold, a special DTC for that purpose will become active.

However, the fact that a DTC is active does not indicate, by itself, whether the issue is serious or not. For example, the fuel pressure may be momentarily low due to the ambient temperature, such as a cold start in an early morning with freezing temperatures. On the other hand, it could be caused by a fuel leak, and this would be much more serious, requiring repair.

An assumption that is often made in practice is that the importance of an issue correlates inversely with its frequency. For a vehicle that starts every morning in cold temperatures, the low fuel pressure DTC may be frequently active without representing a serious issue, whereas for another vehicle a single, unusual activation of the same DTC may require immediate attention.

This makes it possible to train machine learning models – such as Variational Autoencoders (VAEs) [2] – for anomaly detection based on data from a single vehicle, but it is difficult to generalize the results to other vehicles. In particular, training a model on a vehicle and then applying it to detect anomalies on another vehicle will not yield good results.

This problem is further exacerbated by the fact that companies have fleets comprising heterogeneous vehicles (e.g. trucks from different brands, different models, different years, etc.). When monitoring such a fleet, it becomes very difficult to determine which DTCs should be the focus of attention.

In this work, we apply anomaly detection to prioritize DTCs according to their importance, where this importance must be learned from data. The method should be able to determine the importance of each DTC not only in the context of a single vehicle, but also across multiple vehicles in order to assist the analyst in identifying which vehicles and issues should be at the top of the list when it comes to potential anomalies. An initial attempt at achieving this with a VAE eventually leads us to define a proper metric for importance weighting of DTCs.

2 Analysis of a sample vehicle

Fig. 1 shows the DTCs for one particular vehicle in a fleet comprising about 40 trucks. The DTCs are being plotted over the period of one year, with a 60-second resolution. This particular vehicle had a relatively large number of DTCs that were active at least once during that period.

In Fig. 1 there are several different kinds of behaviors, from DTCs that appear to be randomly triggered on and off in quick succession, to DTCs that are active only once for a very brief period of time (short spikes). However, the most interesting DTCs are those that, being usually inactive, become active and stay that way for an extended period of time, such as several weeks or even a few months. This is what happens around May 2019, when the vehicle starts accumulating DTCs that were otherwise inactive and which, when considered altogether, point to one or more serious issues.

The repair history for this particular vehicle shows that it had a breakdown due to “overheating” (no further info provided) in mid-May. In Fig. 1, the fact that several active DTCs are being turned off (or reset) simultaneously in mid-May is consistent with a repair at that point in time.

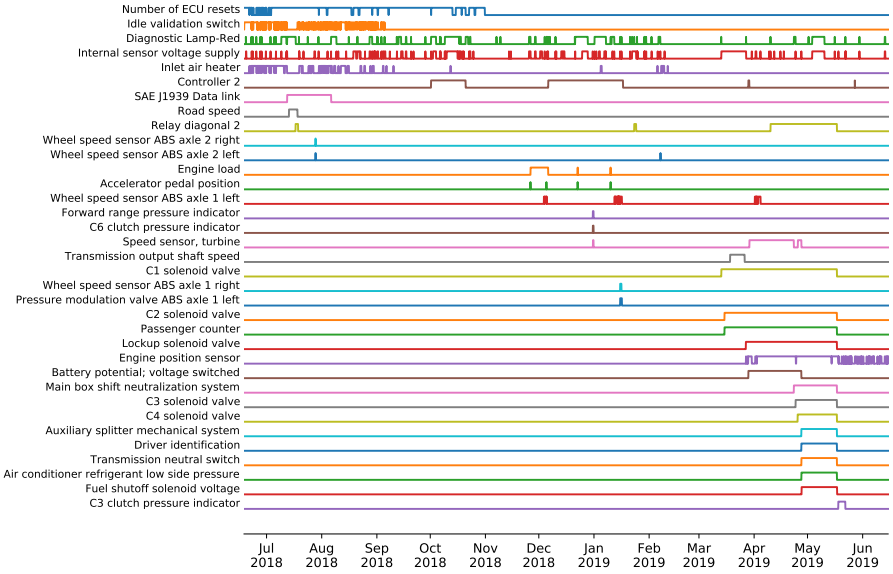

Fig. 1. DTCs for a sample vehicle over a period of one year.

An anomaly detection model trained on the first half of this one-year period should be able to pick up this anomaly in the second half. However, in the absence of labeled data, it might be that this anomalous behavior ends up in the training data for the model, which will then learn it as usual behavior and fail to recognize it as an anomaly in the future. The following section illustrates this point by means of an actual experiment.

3 Anomaly detection with VAEs

Variational Autoencoders (VAEs) have been extensively used for anomaly detection, usually in combination with other methods, such as recurrent neural networks, to capture the evolution of time series data, e.g. [3–7]. In its basic form, anomaly detection relies on the ability of a VAE to reconstruct a given input signal [8]. When the VAE struggles to reproduce the input signal, this is an indication that it has not seen such behavior before, and therefore such behavior is classified as an anomaly.

While any neural network that produces an output of the same type and shape as its input can considered to be an autoencoder, the variational autoencoder has the distinctive feature of encoding the input as a probability distribution, and then generating the output by sampling from such distribution. As a result, the output is a stochastic approximation of the input, but such approximation should be sufficiently good for the training data that the model has

already seen; it is only for data that the model has not seen before that the reconstruction error will be large.

Fig. 2 shows the general structure of a VAE, illustrating the fact that the encoder learns the parameters of a latent distribution (μ and σ of a bivariate normal distribution, in this case), while the decoder produces the output based on samples drawn from that distribution.

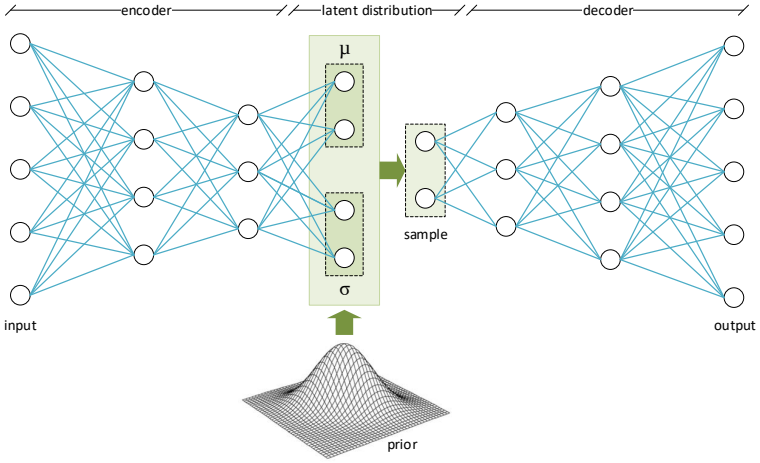

Fig. 2. General structure of a variational autoencoder.

Furthermore, the parameters of the latent distribution are not to be learned freely, as this could possibly lead to overfitting to the training data. Instead, a prior distribution is usually imposed (a standard bivariate normal, in this case), which provides some (tunable) degree of regularization.

The VAE is trained by minimizing both the reconstruction error and the Kullback–Leibler divergence of the latent distribution with respect to the prior, which is equivalent to maximizing the well-know evidence lower bound (ELBO) [9].

3.1 Formal background

Let x be the input that the VAE is trying to reconstruct, and let z be the latent representation of that input, which the encoder is trying to learn by adjusting its network weights θ. In effect, what the encoder is learning is to approximate the posterior distribution $p(z|x)$ with a variational approximation $q_\theta(z|x)$, where θ are the variational parameters of such approximation.

On the other hand, the decoder is learning to generate x from the latent representation z by adjusting its network weights ϕ. In other words, the decoder is learning the likelihood distribution of data $p(x|z)$ with a parametrized approximation that is usually denoted by $p_\phi(x|z)$.

With some mathematical manipulations, it is possible to show that:

$$\log p(x) = \underbrace{\int q_\theta(z|x) \log \frac{p(x,z)}{q_\theta(z|x)} dz}_{\text{ELBO}} + \underbrace{\int q_\theta(z|x) \log \frac{q_\theta(z|x)}{p(z|x)} dz}_{D_{\mathrm{KL}}(q_\theta(z|x) \| p(z|x))} \tag{1}$$

Since $\log p(x)$ is constant w.r.t. $q_\theta(z|x)$, and since the Kullback–Leibler divergence in the second term is non-negative, the first term on the right-hand side is effectively a lower bound for the log-evidence $\log p(x)$. Hence the designation of evidence lower bound (ELBO) for this term.

Furthermore, since $\log p(x)$ is constant, maximizing the ELBO is equivalent to minimizing the Kullback–Leibler divergence between the variational approximation $q_\theta(z|x)$ and the posterior distribution $p(z|x)$, which is exactly what we would like to achieve when training the VAE.

With some further manipulations, it can be shown that:

$$\mathrm{ELBO} = \underbrace{\int q_\theta(z|x) \log p(x|z) dz}_{\mathbb{E}_{q_\theta(z|x)}[\log p(x|z)]} - \underbrace{\int q_\theta(z|x) \log \frac{q_\theta(z|x)}{p(z)} dz}_{D_{\mathrm{KL}}(q_\theta(z|x) \| p(z))} \tag{2}$$

Therefore, maximizing the ELBO can be achieved by maximizing the first term and minimizing the second one. The first term is maximized by training the decoder to learn $p(x|z)$ in order to reconstruct x from z as good as possible, while the second term is minimized by training the encoder to keep the latent distribution as close as possible to a given prior $p(z)$.

In this context, the second term in Eq. (2) can be viewed as a regularization term that precludes the latent distribution from overfitting the training data. A simple choice for the prior distribution $p(z)$ is, for example, a multivariate standard normal with an identity covariance matrix.

3.2 Implementation

To detect anomalies in DTC data, we implemented a VAE to be applied to DTC signals such as those depicted in Fig. 1. For this sample vehicle, we found 62 DTCs that were active at least once over that one-year period. Therefore, we designed a VAE with layer dimensions that were commensurate with the dimensionality of the input data. A sequence of dense layers encodes the input data into a 4-dimensional latent distribution, and another sequence of dense layers decodes the output from a sample of that distribution.

An implementation model for the VAE is shown in Fig. 3, and it uses both deterministic and probabilistic layers. The deterministic layers (e.g. `Dense`) come from the base TensorFlow library,[3] while the probabilistic layers (such as `MultivariateNormalTriL`) come from TensorFlow Probability.[4] Both kinds of layers can be used seamlessly in the same model.

[3] https://github.com/tensorflow/tensorflow
[4] https://github.com/tensorflow/probability

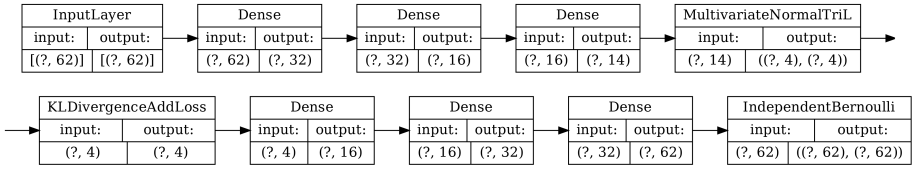

Fig. 3. Implementation of a VAE with deterministic and probabilistic layers.

In particular, `MultivariateNormalTriL` represents a multivariate normal distribution that is parametrized by a mean vector and the lower triangular part of the covariance matrix (since the matrix is symmetric, there is no need to specify the upper part). For an n-dimensional normal distribution, the total number of parameters is $n+n(n+1)/2=14$ for $n=4$.

An interesting feature of `MultivariateNormalTriL` (and this applies to other layers representing probability distributions) is that it receives as input a tensor with the distribution parameters and produces as ouput a sample drawn from a distribution with those parameters.

The same principle applies to the very last layer in Fig. 3, where we use `IndependentBernoulli` to generate a binary random variable for each DTC. Rather than outputting the probability of a DTC being active, we sample from a Bernoulli distribution in order to produce an actual value (0 or 1). However, the main advantage of using a probabilistic layer here is that we can train the VAE by maximizing the log probability of its output (or by minimizing the negative of such log probability, which is the same).

A third probabilistic layer, `KLDivergenceAddLoss`, is a pass-through layer that adds a Kullback–Leibler divergence penalty to the loss function. This penalty represents the second term in Eq. (2) and is computed with respect to a prior distribution which, in our case, is a 4-dimensional independent normal.

The remaining layers in Fig. 3 are deterministic dense layers to implement the compression done by the encoder (top row, from 62 to 14 units) and the decompression by the decoder (bottom row, from 4 back to 62 units). The question mark (?) is a placeholder for the batch size to be used during training.

3.3 Anomaly detection

To test the VAE in an anomaly detection setting, we split the data (Fig. 1) in two halves, where one half is used for training and the other is used for testing. In a first experiment, we train on Jul-Dec 2018 and test on Jan-Jun 2019. In a second experiment, we do the opposite by swapping the train and test sets.

To provide a single measure for anomaly detection, we use the binary cross-entropy (BCE) to assess the accuracy of the VAE in reconstructing the DTC signals at each point in time. Specifically, the BCE is given by:

$$\mathrm{BCE} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log p_i + (1-y_i)\log(1-p_i)\right] \tag{3}$$

where N is the number of DTCs, y_i is the actual value of a DTC (0 or 1), and p_i is the probability of such DTC being active ($0 < p_i < 1$) as predicted by the VAE. Since the last layer of the VAE is an independent Bernoulli distribution for each DTC, p_i is simply the parameter of that distribution.

Fig. 4 shows the results obtained in the two experiments, where we plot the BCE across both the training set and the test set. In the first plot, we see that the VAE has learned to reproduce the training data almost perfectly, regardless of what it contains. As for the test data, the VAE clearly recognizes the anomalous behavior around May 2019 (cf. Fig. 1).

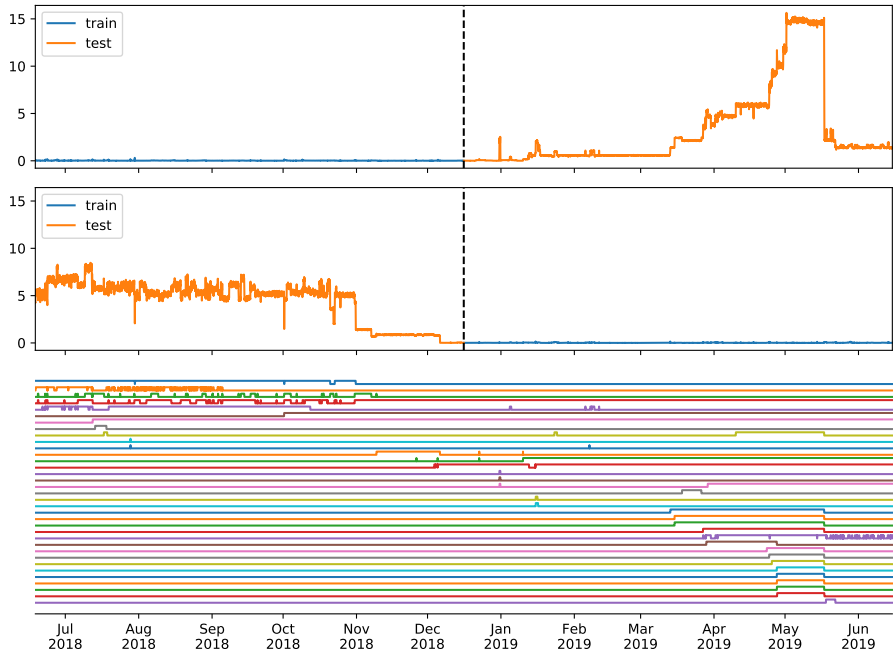

Fig. 4. Binary cross-entropy across train and test sets in two experiments.

In the second plot, where the training set is Jan-Jun 2019, again the VAE learns to reproduce the training data almost perfectly, regardless of the anomalous behavior that it contains. It is then in Jul-Dec 2018 that the VAE recognizes some unusual behavior, especially at the beginning of that period.

This points to a fundamental issue with this approach. Since the VAE learns to reproduce whatever training data it is given, any anomalous behavior must be labeled in order to split the training and test data in a way that the training set does not contain the kind of anomalies that need to be detected.

Furthermore, an analysis of multiple vehicles reveals that a model trained on one vehicle is not directly transferable to other vehicles, as they often display

significantly different behavior in terms of active DTCs, even for vehicles with the same specifications (e.g. same brand, model and year). This makes it difficult to apply anomaly detection across a heterogeneous fleet of vehicles.

4 Importance weighting of DTCs

In the previous section, we used binary cross-entropy (BCE) as an anomaly score, and with this scoring function we showed that a VAE can learn to reproduce whatever training data it is provided, even if the training data contains actual anomalies. In this section, we leave the VAE aside, but the concept of BCE is still useful to come up with an anomaly score that can be applied directly to DTC data, even without training a model.

According to Eq. (3), the BCE will be low either when $y_i = 1$ and $p_i \to 1$ (the second term under the summation will be zero) or when $y_i = 0$ and $p_i \to 0$ (the first term will be zero). Conversely, the BCE will be high either when $y_i = 1$ and $p_i \to 0$, or when $y_i = 0$ and $p_i \to 1$. In other words, the BCE will be low whenever y_i and p_i agree, and it will be high whenever y_i and p_i disagree.

However, when applied to DTC data, the two situations $y_i = 0$ and $y_i = 1$ should be treated differently. It is correct to say that when $y_i = 0$ (DTC is off) and $p_i \to 0$, the anomaly score should be low; as it is correct to say that when $y_i = 1$ (DTC is on) and $p_i \to 0$, the score should be high, because the DTC is active and this could point to an anomaly, regardless of the value of p_i.

Now, the BCE specifies that when $y_i = 1$ and $p_i \to 1$ the anomaly score is low, which should not happen, because an active DTC always points to a potential issue. Similarly, when $y_i = 0$ and $p_i \to 1$ the anomaly score is high, when it should not be, because the DTC is inactive, regardless of the value of p_i.

If a DTC is active ($y_i = 1$) this should always point to an anomaly, albeit a small anomaly if $p_i \to 1$, and a large anomaly if $p_i \to 0$. On the other hand, if a DTC is inactive ($y_i = 0$) this should indicate no anomaly, regardless of p_i.

Therefore, we introduce a different anomaly score, which is obtained by removing the second term in Eq. (3), and which we denote by loss L:

$$L = -\frac{1}{N} \sum_{i=1}^{N} y_i \log p_i \tag{4}$$

According to this formula, if a DTC is inactive ($y_i = 0$) then the corresponding term is zero, and the DTC does not contribute to the anomaly score. On the other hand, if a DTC is active ($y_i = 1$) then the corresponding term is non-zero, and the DTC contributes with a weight of $(-\log p_i)$ to the anomaly score.

If $p_i \to 0$ then the weight of the DTC can become very large, but in practice this does not occur, for two main reasons. First, in the extreme case when $p_i = 0$ the weight of a DTC would become infinite, but such DTC will never turn on, so $y_i = 0$. Second, a DTC that is active for only one minute in an entire year has a weight of $\left(-\log \frac{1}{365 \times 24 \times 60}\right) \approx 13.2$, which is of the same order of magnitude as the maximum values observed in the first plot of Fig. 4.

To avoid training a model, we take the probability p_i to be the fraction of time that a DTC was active during a certain period of time. For example, from the data in Fig. 1 it is possible to compute p_i for each DTC over the period of one year. We then plug these p_i values together with the actual data y_i into Eq. (4) to compute the anomaly score across the same time span.

Fig. 5 shows the results, where the anomaly score is plotted at the top with the DTC signals right below, for comparison.

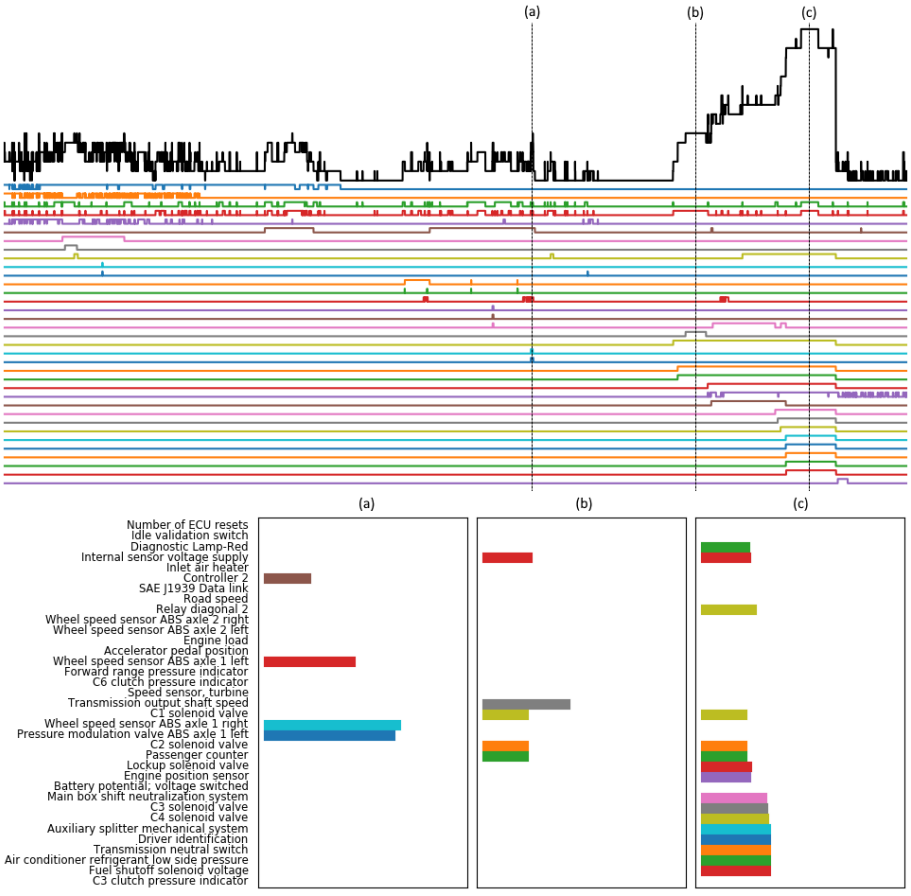

Fig. 5. Importance weights of DTCs at three points in time (a), (b) and (c).

In addition, by means of colored horizontal bars, Fig. 5 illustrates the contribution of each DTC to the anomaly score at three points in time: point (a) represents a brief spike in a few DTCs; point (b) represents the beginning of a long period of anomalous behavior; and point (c) represents the flat top where the anomaly score reaches its peak.

The weight $(-\log p_i)$ can therefore be regarded as a measure of the relative importance of each active DTC with respect to the remaining DTCs. In the midst of several active DTCs, the ones with a larger weight will point to more unusual events, as the examples in Fig. 5 illustrate.

This importance measure can be used not only across DTCs of a single vehicle, but also across DTCs from multiple vehicles simultaneously, provided that the p_i values for all DTCs are evaluated based the same period of time for all vehicles. This makes it possible to prioritize DTCs across an entire fleet and identify vehicles that possibly require immediate attention.

Being able to quickly identify vehicles with issues or issues within vehicles can significantly reduce the amount of data that an analyst must go through in order to find or confirm those issues. The approach can also be used as a first step towards building labeled datasets for training other models.

5 Related work

The analysis of DTC data is central to many anomaly detection approaches in the automotive industry. For example, in [10] the authors propose a framework where DTC data plays two different roles. First, they focus on the detection of anomalous repairs by training a model to learn the appropriate repairs for given issues, where issues are characterized by DTC data. This requires the availability of both repair data and DTC data for training purposes. Second, they use decision trees to infer the operating conditions under which DTCs are activated; this can be used to check if a DTC is being activated according to the correct preconditions.

In more recent works, the focus has been shifting towards the use of machine learning for predictive maintenance. For example, in [11] the authors use random forests to predict which components are about to fail based on DTC data. Again, not only the DTC data but also the repair data are needed in order to build a labeled dataset for training. An interesting aspect of this work is that the authors extract features from DTC data instead of working with DTCs directly. Most of the extracted features are based on aggregated counts within a certain observation window. The authors then go into a detailed study of how the size of this window affects the prediction results.

The use of unsupervised learning techniques (with at least part of the data being unlabeled) has also been gaining ground. For example, in [12] the author focuses on DTC activations which occur in rare operating conditions (called operating modes). Using a clustering algorithm (DBSCAN) and also with the aid of domain experts, the author identifies clusters of operating modes where the DTC activations predominantly occur. The outliers to those clusters are labeled as anomalies. Then, using classifiers based on support vector machines (SVMs), it becomes possible to determine whether a new DTC activation takes place under a typical or atypical operating mode. The underlying assumption is that if the DTC was activated in a rare operating mode, this may point to an uncommon fault which could take longer or be more expensive to repair.

A more fully unsupervised approach is adopted in [13], where the authors use different kinds of models, including autoencoders, for fault detection across a fleet of city buses. A separate autoencoder is used to reconstruct the raw signals from each vehicle. Also, the autoencoder is trained on a chunk of data (e.g. on a daily basis) so there are several trained models for the same vehicle at different points in time. All of these models from each and every vehicle are then compared by assessing their performance on a reference dataset which, ideally, should contain only typical behavior. The models that deviate the most, not only from the reference dataset but also from a "consensus" that is obtained from all vehicles, point to potential faults. In their experiments, the authors have shown that these deviations correlate well with the repair history.

The relation of this brief survey of selected works to the present paper can be described as follows:

- When using supervised learning for anomaly detection, as in [10], this requires the availability of labeled data, for example repair data that can be used to label the anomalies in DTC data. In some cases, such as [13], the repair history is available but is not used for training; rather, it is used for validating the results. This is the same approach that we adopted here.
- While it is possible to apply feature extraction over the training data, as in [11], deep learning models such as convolutional networks and autoencoders should be able to perform feature extraction themselves, and this was one of the reasons for using autoencoders alongside with linear models in [13]. This was also our motivation for experimenting with VAEs.
- Anomaly detection can be performed at different levels of abstraction, either from raw signals [13], or from DTC data [10], or from feature engineering over DTC data [11]. Here we have found DTC data to be a good compromise between overly detailed raw data and overly simplistic features, with the additional benefit of being able to work with binary distributions.
- Eventually, we introduced a simple anomaly score loosely inspired by the concept of binary cross-entropy, which provided similar results to a VAE but without having to train such model. This could be the first stage in a multi-step approach towards obtaining labeled data for training classifiers in a subsequent stage. Such multi-step approach would be somewhat similar to [12], where our anomaly score could be used to label the outliers.

6 Conclusion

In the automotive industry, there are large amounts of data that can be used to train different kinds of models. One of the major challenges is making use of unlabeled data for anomaly detection. While a VAE is certainly capable of capturing normal behavior and signaling anomalies, some care must be taken to avoid including anomalies in the training data. In practice, this is harder than it seems because even healthy vehicles often run with some issues.

In addition, the problem of performing anomaly detection across a heterogeneous fleet does not have a one-fits-all solution, since each vehicle requires a

separately trained model. In this work, we proposed an anomaly score that was based on our previous experiments with VAEs. As is often the case, a simpler solution turns out to be the more effective in practice. The anomaly score and importance weighting of DTCs are being implemented in a monitoring platform that allows an operator to prioritize issues across their fleet.

References

1. Walter, E., Walter, R.: Diagnostic Trouble Codes (DTCs). In: Data Acquisition from Light-Duty Vehicles Using OBD and CAN, pp. 97–108. SAE International (2018)
2. Kingma, D.P., Welling, M.: Auto-Encoding Variational Bayes. In: International Conference on Learning Representations (ICLR) (2014)
3. Suh, S., Chae, D.H., Kang, H., Choi, S.: Echo-state conditional variational autoencoder for anomaly detection. In: International Joint Conference on Neural Networks (IJCNN). pp. 1015–1022 (July 2016)
4. Guo, Y., Liao, W., Wang, Q., Yu, L., Ji, T., Li, P.: Multidimensional time series anomaly detection: A GRU-based gaussian mixture variational autoencoder approach. In: 10th Asian Conference on Machine Learning. PMLR, vol. 95, pp. 97–112 (2018)
5. Park, D., Hoshi, Y., Kemp, C.C.: A multimodal anomaly detector for robot-assisted feeding using an LSTM-based variational autoencoder. IEEE Robotics and Automation Letters **3**(3), 1544–1551 (2018)
6. Kawachi, Y., Koizumi, Y., Harada, N.: Complementary set variational autoencoder for supervised anomaly detection. In: IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). pp. 2366–2370 (2018)
7. Pereira, J., Silveira, M.: Unsupervised anomaly detection in energy time series data using variational recurrent autoencoders with attention. In: 17th IEEE International Conference on Machine Learning and Applications (ICMLA). pp. 1275–1282 (2018)
8. An, J., Cho, S.: Variational autoencoder based anomaly detection using reconstruction probability. Tech. rep., SNU Data Mining Center (2015)
9. Hoffman, M.D., Johnson, M.J.: ELBO surgery: Yet another way to carve up the variational evidence lower bound. In: Advances in Approximate Bayesian Inference (NIPS Workshop) (2016)
10. Singh, S., Pinion, C., Subramania, H.S.: Data-driven framework for detecting anomalies in field failure data. In: IEEE Aerospace Conference. pp. 1–14 (2011)
11. Pirasteh, P., Nowaczyk, S., Pashami, S., Löwenadler, M., Thunberg, K., Ydreskog, H., Berck, P.: Interactive feature extraction for diagnostic trouble codes in predictive maintenance: A case study from automotive domain. In: Proceedings of the Workshop on Interactive Data Mining (WIDM). ACM (2019)
12. Theissler, A.: Multi-class novelty detection in diagnostic trouble codes from repair shops. In: 15th IEEE International Conference on Industrial Informatics (INDIN). pp. 1043–1049 (2017)
13. Rögnvaldsson, T., Nowaczyk, S., Byttner, S., Prytz, R., Svensson, M.: Self-monitoring for maintenance of vehicle fleets. Data Mining and Knowledge Discovery **32**(2), 344–384 (2018)