A Supplementary Material: Anomaly Detection in Multi-Agent Trajectories for Automated Driving

A.1 Visualisation of the Proposed Approach

Our approach during training and in the inference phase is visualised in Figure 1 and Figure 2, respectively.

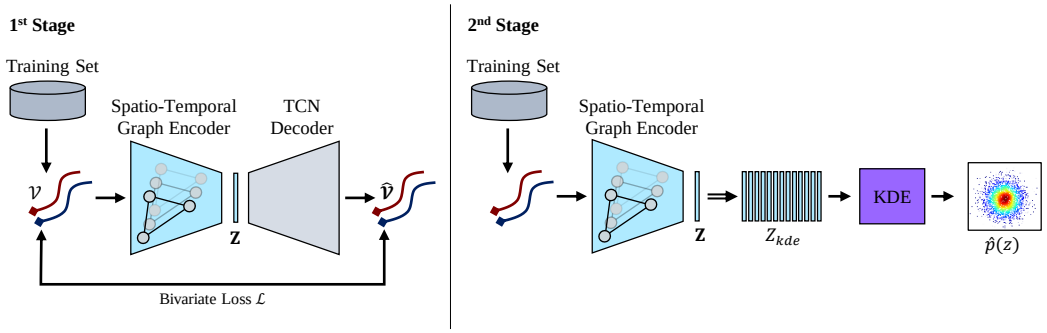


Figure 1: Visualisation of our approach during training. We follow a two-stage approach. **In the first stage**, we use the training set with the normal trajectories to train our spatio-temporal graph auto-encoder (STGAE). It consists of a spatio-temporal graph encoder and a temporal convolution network (TCN) decoder. The latent representation is denoted as $\mathbf{Z}$. The auto-encoder is trained using a bivariate loss $\mathcal{L}$ by comparing the input trajectories $\mathcal{V}$ with the predicted reconstruction $\hat{\boldsymbol{\mathcal{V}}}$. **In the second stage**, we use the trained encoder to compute the latent representation of each segment in the training set, we summarise as Z_{kde}. From the set of feature vectors a kernel density estimation (KDE) learns an approximation of the real density of the normal samples $\hat{p}(z)$.

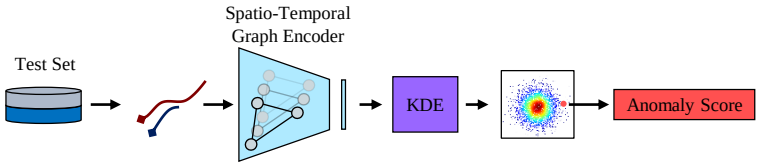
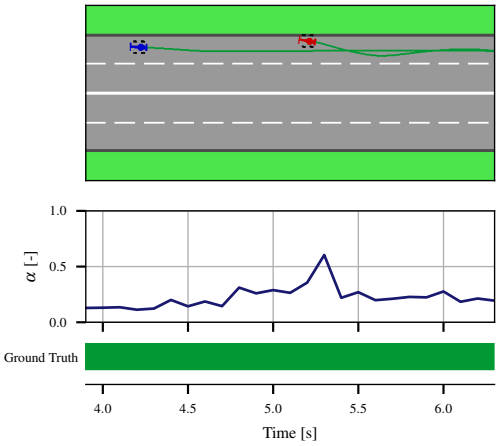


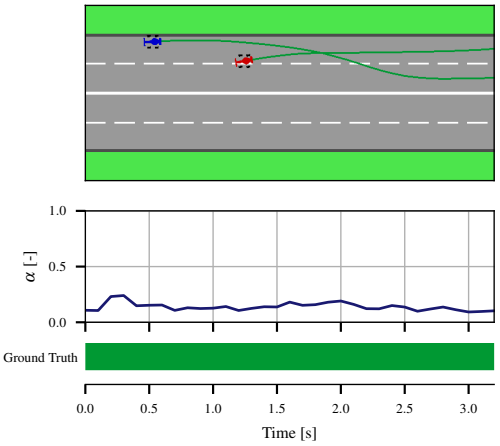
Figure 2: Visualisation of our approach during inference. The test set contains both normal and abnormal behaviour. A given segment from the test set is fed into the trained spatio-temporal graph encoder to predict a feature vector. The feature vector is mapped into the estimated density of the KDE. Given the location in the density function, the KDE computes an anomaly score for each frame and agent in the segment.

A.2 Qualitative Results

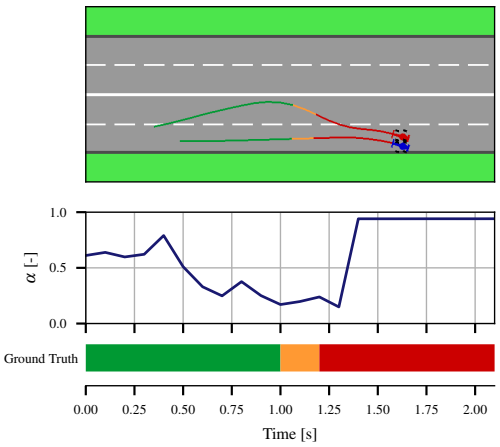
Figure 3 shows qualitative results of our approach on multiple scenes from the test set. In general, the normal scenarios in Figure 3a and 3b are rated with lower anomaly scores compared to the four abnormal scenes (Figure 3c, 3d, 3e and 3f). Interestingly, during the normal driving in Figure 3a the red vehicle is doing a staggering action which results in an increase of the anomaly score at around 5.3 seconds. As this manoeuvre is underrepresented in the training data it falls into a low density region of the KDE such that the anomaly score increases.
Our approach is able to recognise three of the four abnormal manoeuvres correctly. See *pushing aside*, *staggering* and *aggressive reeving* in Figure 3c, 3d and 3e, respectively. The *thwarting* manoeuvre of the blue car in Figure 3f is detected with a minor delay. Overall we can conclude, our proposed approach is able to identify most of the anomalies in multi-agent trajectories.



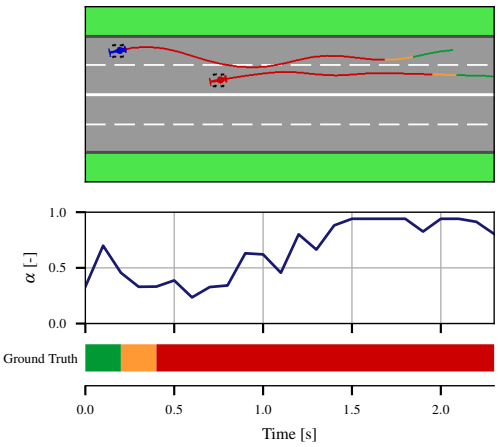
(a) Normal driving.



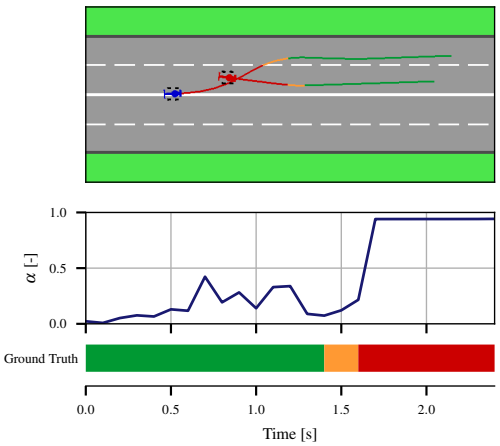
(b) Normal overtaking action.



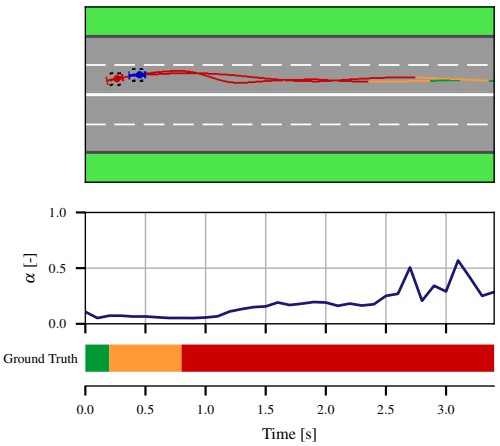
(c) Pushing aside manoeuvre.



(d) Staggering manoeuvre.



(e) Aggressive reeving manoeuvre.



(f) Thwarting manoeuvre.

Figure 3: Qualitative results on two normal and four abnormal scenes. For each scenario we show the frame-wise ground truth annotations (normal: green, transition: orange, abnormal: red), the normalised frame-wise anomaly scores α and the visual scene. The frames annotated as transition between normal and abnormal driving are considered as ignore regions in the evaluation.

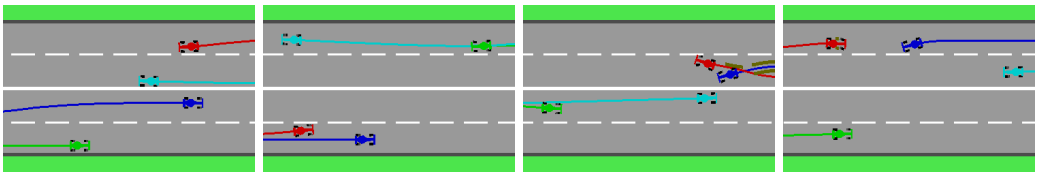


Figure 4: Example sequences of the test set with four agents. We add two additional vehicles to the original dataset, i.e. the green and turquoise cars. Note that we do not add additional anomalies but keep the original ones. In the first two scenarios, all vehicles behave normally following the driving rules. The third scene shows abnormal behavior, where the blue vehicle is required to initiate an evasive manoeuvre after the red car rapidly changes into the left lane. In the last scene, the red car is a wrong-way driver and is even about to collide with the oncoming blue vehicle.

Table 1: Inference time over train set size. The inference time of the STGAE is independent of the amount of training samples. For the KDE we create training sets with $M \in \{100, 1000, 10k, 100k, 1M\}$ samples from the original train set. The inference time of the KDE increases linearly with the number of training samples. We use two different units, milliseconds (ms) and microseconds (μs), for better visibility and average over three runs.

	100	**1000**	**10k**	**100k**	**1M**
STGAE			0.7 ($\pm$0.1) ms		
KDE	7.1 ($\pm$0.4) μs	67.7 ($\pm$6.9) μs	568.0 ($\pm$3.2) μs	6.9 ($\pm$0.1) ms	111.5 ($\pm$0.3) ms

A.3 Test Set for Ablation Study: Scalability Beyond Pairs of Agents

As mentioned in the paper we evaluate the recognition performance for a highway with higher traffic density and to this end create a test set for the ablation study with four agents. To measure the influence of additional agents on the recognition performance, we add extra agents to a subset of the original MAAD test set. We randomly select 11 abnormal and 11 normal sequences from the test set and drive new trajectories with a third and fourth agent. We provide some examples in Fig. 4. We assure that each of the 11 anomaly types is represented in the test set. The additional green and turquoise agents are passive such that the original anomalies are preserved. In total, the test set for the ablation study includes 22 sequences with four agents in each sequence. We perform the scalability experiment beyond pairs of agents as described in the paper.

A.4 Data Scalability and Computational Complexity

This section studies the computational complexity of the model and the scalability on the amount of training data.

Computational Complexity. The presented two-stage approach consists of the joint STGAE trajectory encoding and the KDE for anomaly detection. In comparison to single-agent models, which process agent trajectories sequentially, our STGAE computes the latent representations of all agents in the scene simultaneously. This saves computational resources and at the same time enables the modelling of interactions. During inference, the STGAE is independent of the amount of training data. In the second stage, the KDE computes a probabilistic density of a given agent feature vector with

$$\hat{p}(z) = \frac{1}{|Z_{kde}|h} \sum_{i=1}^{|Z_{kde}|} \kappa_{kde}\left(\frac{\mathbf{z} - \mathbf{z}_i}{h}\right), \tag{1}$$

as defined in the method section. In this case, each feature vector is processed sequentially. The algorithm computes the sum of the kernel similarities between the given feature vector z and all feature vectors z_i in the training set. Therefore, the KDE time complexity scales linearly, $\mathcal{O}(|Z_{kde}|)$, with the number of training samples $|Z_{kde}|$.

Scalability. We study the inference time and the anomaly detection performance of the proposed approach for a different amount of training samples. We randomly draw $M \in \{10^2, 10^3, 10^4, 10^5, 10^6\}$ samples from the original training set, i.e. from 1.91×10^5 samples. In the case of upsampling, we add a small Gaussian noise $e \sim \mathcal{N}(0, 0.1)$ to each repeated feature vector to avoid copies of the samples. Table 1 shows the inference time of both stages over the train set size. While the inference time of the STGAE is independent of the training data, the KDE scales linearly. Assuming a data frequency of 20 Hz in an automotive setting, we reach real-time performance for training datasets with less than 700k samples. To process larger datasets in real-time, sub-sampling should be used, which we address in the next paragraph.

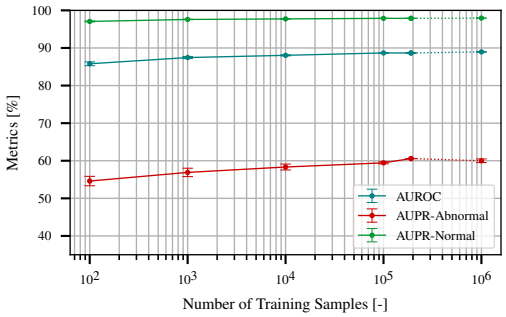


Figure 5: The AUROC, AUPR-Abnormal and AUPR-Normal metrics for the KDE trained on different numbers of training samples $M \in \{10^2, 10^3, 10^4, 10^5, 1.91 \times 10^5, 10^6\}$. Interpolation is denoted with solid (-) and extrapolation with dashed (- -) lines. Averaged over three seeds.

Figure 5 shows the anomaly detection performance of our model with the KDE trained on smaller train sets. The STGAE uses the entire train set, as the inference is independent of the train set size. All metrics decrease as the number of samples decrease, but only by a small amount compared to full-size training. For small training sets, e.g. $M = 10^2$, the results vary over multiple sampling seeds. Additionally, we provide the results for the upsampled training set, i.e. $M = 10^6$. Upsampling has a mild positive effect on AUROC and AUPR-Normal but slightly decreases the AUPR-Abnormal score.

To sum up, the inference time of the proposed approach scales linearly with the number of training samples because of the KDE. For larger training sets, sub-sampling can be used as an efficient technique to reduce the inference time and keep the performance stable in a large range.