EagleSemble at the ICST 2026 Tool Competition – Self-Driving Car Testing Track

Cristian Aquilino
Department of Mathematics, Computer Science and Physics
University of Udine
Udine, Italy
aquilino.cristian@spes.uniud.it

Ettore Ritacco
Department of Mathematics, Computer Science and Physics
University of Udine
Udine, Italy
ettore.ritacco@uniud.it

Vincenzo Riccio
Department of Mathematics, Computer Science and Physics
University of Udine
Udine, Italy
vincenzo.riccio@uniud.it

***Abstract*—This paper presents *EagleSemble*, our submission to the SDC Tool Competition at ICST 2026. EagleSemble is a transformer-based test case prioritization tool for self-driving car testing that ranks test roads according to their predicted likelihood of revealing failures, using geometric segment features and an ensemble of classifiers.**

***Index Terms*—autonomous driving, test prioritization**

I. Introduction

As self-driving cars (SDCs) become increasingly relevant in modern transportation, ensuring their reliability is of paramount importance. A central component of this effort is the systematic testing of autonomous driving agents in simulation-based environments [1]. While simulation enables scalable and controlled experimentation, it is still computationally expensive, as each test case may require several minutes to execute. Moreover, only a limited number of generated test cases is failure-inducing. Thus, considerable testing effort may be spent on non-informative scenarios.

Test case prioritization aims to mitigate this issue by ordering test cases so that the most informative ones are executed earlier. In SDC testing, this translates into prioritizing roads that are more likely to expose failures, enabling faster feedback and more effective use of computational resources.

In this paper, we present *EagleSemble* [2], a transformer-based test case prioritization tool for autonomous driving systems. EagleSemble encodes each road through a sequence of geometric segment features and applies an ensemble of transformer classifiers to predict its likelihood of inducing a failure. These predictions are then aggregated into a final ranking score, producing a prioritized execution order that promotes earlier failure triggering.

II. The EagleSemble Tool

A. Road Representation

In this competition, a road is represented as an ordered sequence of points (x, y) in the Cartesian plane. After interpolation, these points define the road centerline, from which the lane boundaries can be derived, as shown in Figure 1.

We do not use absolute coordinates of road points directly, as they mostly capture position in space but are not ideal for learning. In fact, two roads with very similar shapes may have very different coordinates if one is translated or rotated.

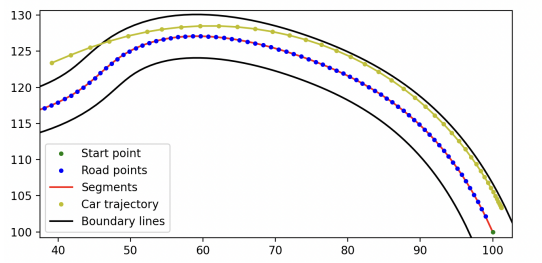

Fig. 1. A failure-inducing test case from Sensodat

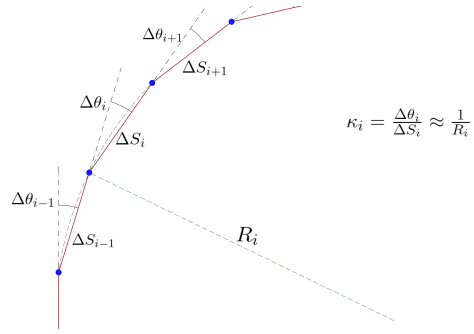

Fig. 2. Visual representation of the features we extracted

Instead, EagleSemble adopts a shape-oriented representation, leveraging a segment-based encoding similar to ITS4SDC [3]. For each pair of consecutive points, we derive three relevant geometric features (depicted in Figure 2) that are more likely to influence driving difficulty:

- *Segment length* (ΔS), computed as the Euclidean distance between two consecutive points;
- *Angular displacement* ($\Delta\theta$), computed as the change in direction with respect to the previous segment, expressed in radians. For the first segment, this value is set to 0;
- *Curvature* (κ), which captures how sharply the road bends. We approximate curvature as

$$\kappa \approx \frac{\Delta\theta}{\Delta S}.$$

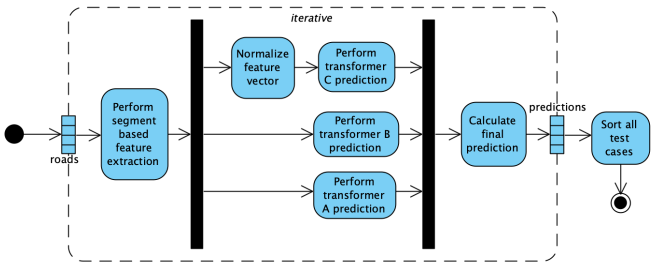

Fig. 3. Overview of the EagleSemble prioritization workflow.

Starting from an input road:

$$\{(x_1, y_1), \ldots, (x_N, y_N)\},$$

our feature extractor produces the sequence of feature values:

$$\{(\Delta S_1, \Delta\theta_1, \kappa_1), \ldots, (\Delta S_{N-1}, \Delta\theta_{N-1}, \kappa_{N-1})\}.$$

Together, our features describe key properties of road segments that may influence the driving agent: how far the vehicle must travel, how sharply it must change direction, and how strongly it must steer.

B. Architecture

Given the sequential nature of the input, we opted for a transformer-based architecture [4]. The considered models consist of 3 encoder layers [5], with embedding dimension 64 and 4 attention heads, followed by a feed-forward classifier with a final sigmoid output. The sigmoid value represents the predicted probability that the input road will induce a failure. EagleSemble combines 3 of these transformers (A, B, and C) into an ensemble. Models A and B are trained on raw features, while model C is trained on features normalized with a standard scaler. The ensemble design improves robustness by reducing the influence of errors or instability of any single model, and it improves predictive performance by combining complementary patterns learned from raw and normalized feature representations. We use 90% of the Sensodat dataset [6] for training and the remainder for testing.

C. Prioritization Workflow

The activity diagram representing the adopted prioritization workflow is shown in Figure 3 For each candidate road, EagleSemble first extracts the sequence of features described in Section II-A. The raw feature sequence is provided to models A and B, while the normalized sequence is provided to model C. Each model outputs a failure probability for the road. The final score assigned to the road is the arithmetic mean of the three model predictions. Once all candidate roads have been scored, EagleSemble ranks them in descending order of predicted failure probability. Roads with higher scores are therefore executed earlier during testing.

TABLE I
EXPERIMENTAL COMPARISON. BEST METRIC VALUES IN BOLD.

Tool	TTFF	APFD	APFDC
EagleSemble	**7.6273**	**0.7944**	**0.8096**
Random Baseline	9.0467	0.5015	0.5005

III. PRELIMINARY ASSESSMENT

We compared our final trained tool with the baseline provided by the organizers of the tool competition [7]. We used a random subset of 6 675 test cases of the available dataset and evaluated the prioritized test suites using three metrics adopted in the competition:

- *Time to First Fault (TTFF)*: cumulative simulation time required to detect the first failure. Lower values indicate that failure-inducing test cases are executed earlier.
- *Average Percentage of Faults Detected (APFD)*: a widely used metric that measures how early failure-inducing test cases appear in the prioritized order. Higher values indicate better prioritization performance.
- *Average Percentage of Faults Detected Cost-Aware (APFDC)*: a cost-aware variant of APFD that also accounts for the execution cost of each test case, i.e., its simulation time. Higher values indicate that faults are revealed earlier while considering execution cost.

Table I reports the results of our evaluation and shows that EagleSemble clearly outperforms the baseline for all the considered metrics.

IV. CONCLUSIONS AND FUTURE WORK

In this paper, we presented EagleSemble, a transformer-based test case prioritization tool for self-driving car testing. Preliminary results suggest that the proposed ensemble effectively ranks failure-inducing test cases. Future work will focus on assessing the approach across different driving agents and on enriching the input representation with additional features.

REFERENCES

[1] M. Fazzini, A. Gambi, V. Riccio, A. Panichella, and S. Klikovits, "Devops testing for cyber-physical systems," in *Roadmap for DevOps in Cyber-Physical Systems: Challenges and Future Directions*. Springer, 2026.
[2] EagleSemble Tool. [Accessed: 2026-03-12]. [Online]. Available: https://github.com/ConteMascetti22/sdc-testing-competition/tree/main/tools/prioritizers/eagle_semble
[3] A. Güllü, F. A. Shah, and D. Pfahl, "ITS4SDC at the ICST 2025 Tool Competition - Self-Driving Car Testing Track," in *2025 IEEE Conference on Software Testing, Verification and Validation (ICST)*, 2025.
[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is All you Need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.
[5] PyTorch Transformer Encoder. [Accessed: 2026-03-12]. [Online]. Available: https://docs.pytorch.org/docs/stable/generated/torch.nn.TransformerEncoder.html
[6] C. Birchler, C. Rohrbach, T. Kehrer, and S. Panichella, "SensoDat: Simulation-based Sensor Dataset of Self-driving Cars," in *IEEE/ACM International Conference on Mining Software Repositories*, 2024.
[7] P. Aryan, T. Fulcini, L. Starace, C. Birchler, and S. Panichella, "ICST Tool Competition 2026 – SDC Testing Track," in *International Conference on Software Testing, Verification and Validation (ICST)*, 2026.