



MASTER THESIS

**A machine learning-based framework for effectively
analyzing execution results of the back-to-back test
method during real-time validation of automotive
software systems**

Shaoqing Guo
Matriculation Number: 529572

MSc Informatik

1st Examiner: Prof. Dr. Andreas Rausch
2nd Examiner: apl. Prof. Christoph Knieke
Supervisor: Dr. Mohammad Abboush

Institute for Software and System Engineering, Clausthal University of Technology,
Germany

March 27, 2025

Declaration of Authorship

I have read and understood the guidelines of the Technical University of Clausthal and those published in the ITE bulletin, including those regarding the use of literature and other sources. I confirm that I have prepared this thesis independently by myself. Any information taken from other sources and being reproduced in this thesis is clearly referenced.

In terms of the general examination regulations, this work has not yet been submitted to any other examination division.

I hereby agree that my master thesis may be exhibited in the institute's library and kept for inspection. Note: is not applicable if the thesis is rated as confidential in the companies agreement!

Clausthal-Zellerfeld, 27th March 2025

Location, Date

First name and family name

Abstract

With the rapid development of the automotive industry towards electrification, intelligence, and connectivity, automotive software systems have shown unprecedented growth in complexity. Modern high-end vehicles contain up to 100 million lines of code, with complexity comparable to advanced aviation systems. This complexity introduces significant verification challenges for back-to-back (B2B) testing in line with the ISO 26262 standard, including ensuring real-time responsiveness, accurately handling multi-variable interactions, and dynamically adapting fault detection thresholds under varying operational scenarios. This research proposes an intelligent analysis framework based on deep learning and clustering analysis to improve the efficiency and accuracy of back-to-back testing for automotive software systems. The framework integrates a CNN-LSTM deep autoencoder for feature extraction and anomaly detection based on the fault-free behavior, along with K-means clustering algorithm for fault pattern classification. Our goal is to address the limitations of traditional fixed-threshold methods by providing a more adaptive, accurate, and interpretable approach to detecting and classifying faults in complex automotive systems. Through systematic investigation and extensive experimentation, we designed and validated a two-stage processing approach where potential anomalies are first identified through reconstruction error analysis before applying clustering for fault pattern differentiation. This approach not only improved computational efficiency but also enhanced fault identification clarity. The framework was evaluated using unseen testing data from both Simulink Model-in-the-Loop environment (MIL) and SCALEXIO real-time Hardware-in-the-Loop platform (HIL) across various driving scenarios, demonstrating its robustness and generalizability. The main contributions of this research include: an innovative CNN-LSTM hybrid architecture that demonstrated superior performance in automotive signal fault detection with 90.54% accuracy and a 0.926 F1 score; optimized K-means clustering with fault-based K value selection that improved the Davies-Bouldin Index by 38.6% in multi-fault scenarios, providing more interpretable fault pattern differentiation while reducing processing time by 36.4%; strong cross-scenario generalization capability, allowing a single model trained on mixed scenario data to be deployed across various operational conditions; and a comprehensive intelligent B2B testing framework that provides engineers with automated feature extraction, statistically optimized threshold selection, and clear visualization and interpretation of

fault patterns. Experimental results demonstrate that the proposed CNN-LSTM-DAE model with optimized K-means clustering framework exhibits strong robustness in high-noise environments, successfully identifies and clusters both single and multiple fault scenarios, maintains highly stable performance across different driving scenarios, and keeps all processing times within the computational costs requirements. This research provides a novel intelligent approach to automotive software back-to-back testing, overcoming the limitations of traditional methods and offering a more efficient and accurate solution for ensuring system safety and reliability.

Zusammenfassung

Mit der rasanten Entwicklung der Automobilindustrie in Richtung Elektrifizierung, Intelligenz und Konnektivität haben Automobilsoftwaresysteme ein beispielloses Wachstum an Komplexität gezeigt. Moderne Fahrzeuge der Oberklasse enthalten inzwischen bis zu 100 Millionen Codezeilen, deren Komplexität mit fortschrittlichen Luftfahrtsystemen vergleichbar ist. Diese Komplexität führt zu erheblichen Herausforderungen bei der Verifizierung von Back-to-Back (B2B) Tests gemäß der ISO-26262-Norm, insbesondere hinsichtlich der Sicherstellung von Echtzeitreaktionen, der präzisen Handhabung mehrerer gleichzeitig wechselwirkender Variablen sowie der dynamischen Anpassung von Fehlererkennungsschwellen unter variierenden Betriebsbedingungen.

In dieser Arbeit wird ein intelligentes Analyse-Framework basierend auf Deep Learning und Clusteranalyse vorgeschlagen, um die Effizienz und Genauigkeit von Back-to-Back Tests für Automobilsoftwaresysteme zu verbessern. Das Framework integriert einen CNN-LSTM Deep Autoencoder zur Merkmalsextraktion und Anomalieerkennung basierend auf dem fehlerfreien Verhalten sowie einen K-Means-Clustering-Algorithmus zur Klassifizierung von Fehlermustern. Ziel ist es, die Grenzen traditioneller Methoden mit festgelegten Schwellenwerten zu überwinden und eine adaptive, genauere und interpretierbarere Methode zur Erkennung und Klassifizierung von Fehlern in komplexen Automobilsystemen bereitzustellen.

Durch systematische Untersuchung und umfangreiche Experimente wurde ein zweistufiger Verarbeitungsansatz entwickelt und validiert. Dabei werden potenzielle Anomalien zunächst mittels Rekonstruktionsfehleranalyse identifiziert, bevor zur Differenzierung von Fehlermustern das Clustering-Verfahren angewendet wird. Dieser Ansatz verbessert nicht nur die Recheneffizienz, sondern erhöht auch die Klarheit bei der Fehlererkennung. Das Framework wurde mit ungesiehenen Testdaten sowohl aus der Simulink Model-in-the-Loop-Umgebung (MIL) als auch von der SCALEXIO Hardware-in-the-Loop-Plattform (HIL) über verschiedene Fahrszenarien hinweg evaluiert, wobei es seine Robustheit und Generalisierbarkeit unter Beweis stellte.

Die wichtigsten Beiträge dieser Forschung umfassen: eine innovative CNN-LSTM Hybridarchitektur, welche bei der Erkennung von Automobilsignalfehlern eine überlegene Leistung mit einer Genauigkeit von 90,54% und einem F1-Score von 0,926 erzielte; ein optimiertes K-Means-Clustering mit fehlerbasierter Auswahl des K-Wertes, das den Davies-

Bouldin-Index bei Mehrfehlerszenarien um 38,6% verbesserte, eine interpretierbarere Differenzierung der Fehlermuster ermöglichte und die Verarbeitungszeit um 36,4% reduzierte; eine starke szenarienübergreifende Generalisierungsfähigkeit, die es erlaubt, ein einziges, auf gemischten Szenariendaten trainiertes Modell unter verschiedenen Betriebsbedingungen einzusetzen; sowie ein umfassendes, intelligentes B2B-Testframework, das Ingenieuren automatisierte Merkmalsextraktion, statistisch optimierte Schwellenauswahl sowie eine klare Visualisierung und Interpretation von Fehlermustern bietet.

Experimentelle Ergebnisse zeigen, dass das vorgeschlagene CNN-LSTM-DAE-Modell mit optimiertem K-Means-Clustering-Framework eine ausgeprägte Robustheit in stark verrauschten Umgebungen aufweist, sowohl einzelne als auch multiple Fehlerszenarien zuverlässig gruppiert, eine stabile Leistung über verschiedene Fahrszenarien hinweg gewährleistet und dabei sämtliche Verarbeitungszeiten innerhalb der Anforderungen an die Rechenkosten hält. Diese Forschung stellt somit einen neuartigen intelligenten Ansatz für den Back-to-Back-Test von Automobilsoftware dar, welcher die Grenzen herkömmlicher Methoden überwindet und eine effizientere und genauere Lösung zur Sicherstellung der Systemsicherheit und -zuverlässigkeit bietet.

Contents

1	Introduction	1
1.1	Research Background and Significance	1
1.2	Related Work	2
1.2.1	Back-to-Back Testing and Automotive Functional Safety	3
1.2.2	Applications of Deep Learning in Automotive System Analysis	4
1.2.3	Applications of Cluster Analysis in Fault Diagnosis	5
1.3	Research Objectives	6
1.4	Thesis Structure	7
2	Theoretical Background	8
2.1	ISO 26262 Functional Safety Standard	8
2.1.1	Standard Overview	8
2.1.2	V-Model Development Process	8
2.1.3	Software Testing Requirements	9
2.2	Back-to-Back Testing (B2B)	10
2.2.1	Definition and Principles	10
2.2.2	Testing Methods during Model-Driven Development	10
2.2.3	Status in ISO 26262	11
2.2.4	Traditional B2B Test Result Analysis Methods and Their Limitations	11
2.2.5	Threshold Selection Theory for Anomaly Detection	12
2.3	Deep Learning Techniques	13
2.3.1	Convolutional Neural Networks (CNN)	13
2.3.2	Long Short-Term Memory Networks (LSTM)	14
2.3.3	Deep Autoencoders	15
2.3.4	Hybrid Deep Learning Architectures	16
2.3.5	Other Machine Learning Models	17
2.4	Clustering Analysis Algorithms	18
2.4.1	K-means Clustering	18
2.4.2	Density-Based Clustering Algorithms (DBSCAN)	19
2.4.3	Gaussian Mixture Models (GMM)	20

2.4.4	Fuzzy C-means Clustering (FCM)	21
2.4.5	Cluster Validity Assessment	21
2.5	Hyperparameter Optimization Methods	23
2.5.1	Hyperparameter Definition and Importance	23
2.5.2	Optuna Framework	23
2.6	Automotive System Development and Testing Environments	24
2.6.1	MATLAB/Simulink Development Environment	24
3	Proposed Method	27
3.1	Research Framework Overview	27
3.2	Data Collection Strategy	28
3.3	Data Preprocessing	29
3.4	CNN-LSTM Deep Autoencoder Model Design	30
3.4.1	Architecture Components and Mathematical Formulation	31
3.4.2	Model Training and Optimization	32
3.5	Reconstruction Error-Based Anomaly Detection	33
3.5.1	MSE Threshold Selection Method	33
3.5.2	Anomaly Detection Workflow	34
3.5.3	Error Analysis Methods	35
3.6	Cluster Analysis Methods	35
3.6.1	Feature Extraction for Clustering	35
3.6.2	Clustering Algorithms - K-means Mathematical Formulation	36
3.6.3	Clustering Algorithms Comparison	37
3.6.4	Fault-based K Value Selection Process	37
3.6.5	Clustering Evaluation and Visualization	38
4	Experimental Implementation and Case Studies	40
4.1	Case Study: Gasoline Engine and Vehicle Dynamics	40
4.2	Data Collection	43
4.3	Data Preprocessing	46
4.4	Model Selection and Implementation	48
4.4.1	CNN-LSTM-DAE Model Hyperparameter Optimization Strategy	49
4.4.2	CNN-LSTM-DAE Model Architecture Design	51
4.4.3	Model Training Process	52
4.5	Implementation Details of Threshold Selection	56
4.6	Cluster Analysis Method Selection	58

5 Results and Discussion	63
5.1 Cross-Scenario Performance Analysis	63
5.1.1 Loss Convergence and Training Consistency Across Scenarios	63
5.1.2 Fault Detection Results in Individual Driving Scenarios	64
5.2 B2B Testing Framework Clustering Results	65
5.2.1 Fault Detection and Clustering Pipeline	65
5.2.2 Original Clustering Approach	66
5.2.3 Optimized K-means Clustering Approach	71
5.2.4 Computational Analysis	75
5.3 Results Summary	75
6 Conclusions and Future Work	77
6.1 Research Summary	77
6.2 Key Contributions	77
6.2.1 Innovative CNN-LSTM Hybrid Architecture	77
6.2.2 Optimized K-means Clustering with Fault-based K Value Selection	78
6.2.3 Strong Cross-Scenario Generalization	78
6.2.4 Computational costs Optimization	79
6.2.5 Comprehensive Framework for Intelligent B2B Testing	79
6.3 Limitations	80
6.4 Future Research Directions	80
6.4.1 Hybrid Learning Architectures	80
6.4.2 Model Compression and Optimization	81
6.4.3 Enhanced Interpretability Methods	81
6.4.4 Incremental Learning Approaches	81
6.4.5 Integration with Complementary Verification Technologies	82
6.5 Concluding Remarks	82
References	83

List of Figures

2.1	V-model development process ^[35]	9
3.1	Framework Overview	27
3.2	CNN-LSTM-DAE Architecture Overview	30
4.1	Internal structure of the ASM Gasoline Engine Model	41
4.2	Real-time virtual test drives environment	42
4.3	Vehicle Speed Comparison Across Different Driving Scenarios	47
4.4	Analysis of hyperparameter relationships and their influence on model performance	54
4.5	Hyperparameter optimization results with training and validation accuracy under different noise levels	55
4.6	Validation loss comparison at different noise levels (3%, 6%, 8%, and 10%), showing how all models eventually converge despite initial differences in loss values	56
5.1	Performance comparison of the model across different dataset configurations	64
5.2	Framework output showing interval-based fault detection with healthy ratio assessment	66
5.3	K-means clustering results for RPM gain fault scenario [25,40] with K=2 (DBI=1.5908, Silhouette=0.2984)	67
5.4	K-means clustering results for multi-fault scenario [40,55] with K=3 (DBI=0.8212, Silhouette=0.8011)	68
5.5	K-means clustering results for complex concurrent fault scenario [55,70] with K=4 (DBI=1.3450, Silhouette=0.3225)	69
5.6	K-means clustering results for multi-fault scenario [70,80] with K=3 (DBI=0.4352, Silhouette=0.9549)	70
5.7	K-means clustering results for RPM noise fault scenario [80,90] with K=2 (DBI=0.3915, Silhouette=0.6971)	71
5.8	Comparison of clustering results for [40,55] interval using original (K=3) and optimized (K=2) approaches	72
5.9	Comparison of clustering results for [55,70] interval using original (K=4) and optimized (K=3) approaches	73

5.10 Comparison of clustering results for [70,80] interval using original (K=3) and optimized (K=2) approaches	73
---	----

List of Tables

1.1 Overview of the related works	6
4.1 Collected Signals and Their Units	44
4.2 Modeling Data (Simulink) Driving Scenario Statistics	45
4.3 Vehicle Dynamics Model Data Collection	46
4.4 Healthy Operating Data (Scalexio) Statistics	47
4.5 Fault Data (Scalexio) Statistics	47
4.6 Model Performance Comparison	48
4.7 Python Environment Dependencies and Versions	53
4.8 CNN-LSTM-DAE Model Hyperparameters	53
4.9 Model Performance Under Different Noise Levels	54
4.10 Percentile-Based Threshold Values	57
4.11 Performance Comparison Across Different Threshold Levels	57
4.12 Clustering Performance Comparison Across Different Methods	60
4.13 Detailed Clustering Performance for Each Fault Interval	61
4.14 Processing Time Comparison for Different Clustering Methods	62
5.1 Cross-scenario fault detection performance for RPM_Gain fault	65
5.2 Performance Comparison Between Original and Optimized K-means Clustering	74
5.3 Computational Performance Metrics for Original and Optimized Ap- proaches	75