UFPE

UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

PAULO JOSÉ NUNES BATISTA DOS PRAZERES

ADS TESTING WITH ATTENTION

Recife
2023

PAULO JOSÉ NUNES BATISTA DOS PRAZERES

ADS TESTING WITH ATTENTION

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Área de Concentração: Engenharia de Software e Linguagens de Programação
Orientador: Marcelo d'Amorim
Coorientador: Andrea Stocco

Recife
2023

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

P921a Prazeres, Paulo José Nunes Batista dos
ADS testing with attention / Paulo José Nunes Batista dos Prazeres. – 2023.
79 f.: il., fig., tab.

Orientador: Marcelo d'Amorim.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2023.
Inclui referências.

1. Engenharia de software. 2. Segmentação de imagens. 3. Inteligência artificial. I. d'Amorim, Marcelo (orientador). II. Título.

005.1 CDD (23. ed.) UFPE - CCEN 2023-176

Paulo José Nunes Batista dos Prazeres

“ADS Testing with Attention”

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Engenharia de Software e Linguagens de Programação.

Aprovado em: 28 de agosto de 2023.

BANCA EXAMINADORA

_____

Prof. Dr Tsang Ing Ren
Centro de Informática / UFPE

_____

Prof. Dr. Vincenzo Riccio
Dipartimento di Scienze Matematiche, Informatiche e Fisiche/University of Udine

_____

Prof. Dr. Marcelo Bezerra D'Amorim
Centro de Informática / UFPE
(**Orientador**)

ACKNOWLEDGMENTS

First and foremost, I would like to thank God, whose grace and guidance have been a constant source of strength and wisdom in my life.

I would like to express my deepest gratitude to my father, Paulo Batista dos Prazeres, for his unwavering support and tireless encouragement in the realm of education. You are the solid foundation upon which I have built my dreams.

I would also like to honor the memory of my beloved mother, Ana Claudia Nunes da Silva, who is no longer with us. Her love, support, and values continue to guide me, and I am forever grateful for the time we shared. In her memory, I dedicate this achievement.

To my beloved wife, Luana, thank you from the bottom of my heart for the companionship and constant support. Having you by my side has made this journey not just possible, but also joyful and fulfilling. Your presence is a source of strength and inspiration.

I would like to extend my sincere thanks to my advisor, Marcelo D'Amorim, for his invaluable guidance. His expertise, wisdom, and support were pivotal in the development of this work. I am grateful to have had the opportunity to learn and grow under his tutelage.

My gratitude also goes to my co-advisor, Andrea Stocco, for generously sharing technical knowledge and for all the guidance provided. Your expertise was instrumental and your attentive approach greatly facilitated the process.

A special thank you to my friend Vincenzo Riccio, for enriching this work with your technical knowledge and for being always a colleague willing to collaborate and share ideas.

Finally, I am immensely grateful to my boss, Srinath Gopinath, for his understanding and flexibility in allowing me to balance my commitments with the company and the master's program. His encouraging approach to self-development, both professional and personal, is something I will carry with me throughout my career.

To all of you, my sincerest thanks.

AGRADECIMENTOS

Primeiramente, gostaria de agradecer a Deus, cuja graça e orientação têm sido uma constante fonte de força e sabedoria em minha vida.

Gostaria de expressar minha mais profunda gratidão ao meu pai, Paulo Batista dos Prazeres, por seu apoio inabalável e encorajamento incansável no âmbito da educação. Você é a base sólida sobre a qual construí meus sonhos.

Também gostaria de homenagear a memória de minha amada mãe, Ana Claudia Nunes da Silva, que não está mais entre nós. Seu amor, apoio e valores continuam a me guiar, e sou eternamente grato pelo tempo que compartilhamos. Dedico a ela esta conquista.

À minha amada esposa, Luana, agradeço do fundo do meu coração pelo companheirismo e apoio constante. Ter você ao meu lado tornou esta jornada não apenas possível, mas também alegre e gratificante. Sua presença é uma fonte de força e inspiração.

Gostaria de estender meus sinceros agradecimentos ao meu orientador, Marcelo D'Amorim, por sua orientação inestimável. Sua experiência, sabedoria e apoio foram cruciais no desenvolvimento deste trabalho. Sou grato por ter tido a oportunidade de aprender e crescer sob sua tutela.

Minha gratidão também vai para o meu coorientador, Andrea Stocco, por generosamente compartilhar conhecimento técnico e por toda a orientação fornecida. Sua experiência foi fundamental e sua abordagem atenciosa facilitou enormemente o processo.

Um agradecimento especial ao meu amigo Vincenzo Riccio, por enriquecer este trabalho com seu conhecimento técnico e por ser sempre um colega disposto a colaborar e compartilhar ideias.

Finalmente, sou imensamente grato ao meu chefe, Srinath Gopinath, por sua compreensão e flexibilidade em permitir que eu equilibrasse meus compromissos com a empresa e o mestrado. Sua abordagem encorajadora para o autodesenvolvimento, tanto profissional quanto pessoal, é algo que levarei comigo ao longo da minha carreira.

A todos vocês, meus mais sinceros agradecimentos.

ABSTRACT

As autonomous vehicles become more common worldwide, there is growing interest in the safety of their control systems. These systems generally include Neural Networks that use road images to guide the vehicle's steering direction. Developing effective strategies to monitor and test these Neural Networks is thus critical. This dissertation proposes two distinct but complementary strategies to address this challenge. The first strategy employs an innovative technique based on the attention maps computed by explainable artificial intelligence. This approach actively monitors the Neural Network's operations within the vehicle, identifying anomalous instances where the Neural Network may behave unexpectedly, thereby mitigating potential accident risks. This method was empirically validated using the Virtual Driving Simulator developed by Udacity. The second part of the dissertation presents a preliminary study of a testing strategy applicable during the Neural Networks' testing phase. This strategy involves generating a variety of extreme driving scenarios to expose and understand the Neural Network's limitations and weaknesses. A Neural Network trained on the MNIST dataset to classify digits was employed in this study, serving as a proof of concept for the effectiveness of attention maps in guiding the generation of digit variations and identifying corner cases. The analogy between the shape of a digit and the layout of a road formed the basis for using digit classification in this preliminary study. The goal is to demonstrate that the efficiency gains achieved with the application of attention maps would hold promising results if replicated in the automatic generation of simulated road scenarios for driving simulations. The results suggest that our approach can substantially improve the safety and reliability of autonomous vehicles.

Keywords: image segmentation; autonomous vehicles safety; explainable artificial intelligence; automated tests generation.

RESUMO

À medida que veículos autônomos se tornam mais comuns em todo o mundo, há um crescente interesse na segurança de seus sistemas de controle. Esses sistemas geralmente incluem Redes Neurais que usam imagens da estrada para orientar a direção do volante do veículo, desempenhando um papel fundamental nesses veículos. Desenvolver estratégias eficazes para monitorar e testar essas Redes Neurais é, portanto, crítico. Esta dissertação propõe duas estratégias distintas, mas complementares, para enfrentar esse desafio. A primeira estratégia emprega uma técnica inovadora baseada nos mapas de atenção calculados pela *explainable artificial intelligence*. Esta abordagem monitora ativamente as operações da Rede Neural dentro do veículo, identificando instâncias anômalas onde a Rede Neural pode se comportar de maneira inesperada, mitigando assim potenciais riscos de acidentes. Este método foi validado empiricamente usando o Simulador de Direção Virtual desenvolvido pela Udacity. A segunda parte da dissertação apresenta um estudo preliminar de uma estratégia de teste aplicável durante a fase de teste das Redes Neurais. Esta estratégia envolve gerar uma variedade de cenários de condução para expor e entender as limitações e fraquezas da Rede Neural. Uma Rede Neural treinada no conjunto de dados MNIST para classificar dígitos foi empregada neste estudo, servindo como uma prova de conceito para a eficácia dos mapas de atenção em orientar a geração de variações de dígitos e identificar *corner cases*. A analogia entre a forma de um dígito e o layout de uma estrada formou a base para usar a classificação de dígitos neste estudo preliminar. O objetivo é demonstrar que os ganhos de eficiência alcançados com a aplicação de mapas de atenção podem ser replicados na geração automática de cenários de estrada simulados para simulações de condução. Os resultados sugerem que nossa abordagem pode melhorar substancialmente a segurança e a confiabilidade dos veículos autônomos.

Palavras-chave: segmentação de imagens; segurança de veiculos autônomos; inteligência artificial explicável; geração de testes automatizados.

LIST OF FIGURES

LIST OF TABLES

LIST OF ALGORITHMS

CONTENTS

1 INTRODUCTION

The development and implementation of autonomous driving systems (ADS) is a complex and multifaceted endeavor with considerable implications for public safety. Ensuring that these systems can operate safely and efficiently, even in unknown and uncertain situations, has stimulated a substantial amount of research into autonomous vehicles, machine learning, and, most notably, the usage of explainable artificial intelligence (XAI).

This dissertation delves into two key research projects in this domain, both centralizing the role of XAI, particularly focusing on the utilization of attention maps during the ADS testing and runtime monitoring phase. Each project is encapsulated in a standalone chapter, which is divided into main sections: Introduction, Background, Methodology, Results, and Conclusion.

In the first chapter, we delve into the detailed examination of a runtime monitoring technique known as ThirdEye. This technique originated a paper titled "ThirdEye: Attention Maps for Safe Autonomous Driving Systems"(STOCCO et al., 2022). The foundation and content of this chapter are principally rooted in the findings and discussions presented in the aforementioned paper. This tool uses attention maps computed by XAI techniques to differentiate between safe and unsafe driving behaviors. We explore the hypothesis that uncommon attention maps may indicate unexpected runtime conditions that could potentially lead to the misbehavior of the autonomous system. In essence, the purpose of this chapter of the study is to present a comprehensive analysis of ThirdEye's theoretical framework, implementation, and empirical evaluation in predicting simulation-based failures.

The second chapter of the dissertation draws focus on a preliminary study, called MUTANT XAI GENERATOR, centered on a testing strategy employed during the Neural Networks' testing phase. This strategy encompasses generating various driving scenarios to expose and understand the Neural Network's limitations and weaknesses. The chapter starts by providing the necessary background for understanding Neural Network's usage in autonomous vehicles. The intent is to illustrate how efficiency gains achieved with attention maps can be mirrored in the automatic generation of simulated road scenarios for driving simulations.

2 THIRDEYE- A XAI RUNTIME MONITORING APPROACH FOR AUTONOMOUS DRIVING SYSTEMS

2.1 INTRODUCTION

The advancements in autonomous driving systems (ADS) have necessitated the development of sophisticated runtime monitoring techniques to ensure safety even under unexpected conditions. This chapter presents an in-depth exploration of ThirdEye, a novel tool designed to address this pressing need.

ThirdEye is an innovative application of explainable artificial intelligence (XAI) techniques in the realm of ADS, specifically focusing on the use of attention maps. Attention maps in this context are computational representations that highlight areas in an input data that a machine learning model, particularly a neural network, pays most attention to while making decisions. ThirdEye leverages these attention maps to discern between safe and unsafe driving behaviors by associating uncommon attention maps with unexpected runtime conditions.

The primary objective of this chapter is to present a comprehensive examination of ThirdEye, covering its theoretical underpinnings, practical implementation, and empirical evaluation. In the process, we delve into the realms of XAI and attention maps, as well as their importance in autonomous driving scenarios.

The chapter begins by introducing the concept of ThirdEye, followed by a review of relevant background literature on ADS, XAI, and attention maps. We then move on to the methodology section, where we describe the design, implementation, and testing of ThirdEye, including details on how attention maps are computed and utilized in the system.

In the Results section, we present findings from our empirical study where ThirdEye was evaluated on its ability to predict simulation-based failures, induced by both unknown conditions (adverse weather and lighting) and unsafe/uncertain conditions created with mutation testing. We outline how attention maps were used to generate confidence scores that differentiated between safe and unsafe driving behaviors.

Finally, in the Conclusion section, we draw upon the results to reflect on the effectiveness of ThirdEye and its potential implications for the safety of ADS. We also discuss potential areas for future research, inspired by the findings of this study.

Overall, the aim of this chapter is to elucidate the potential of XAI in ensuring the safety of ADS, exemplified through our study of the ThirdEye tool. By conducting this research, we aspire to contribute to the broader understanding of autonomous driving systems and the pivotal role of XAI therein.

2.2 MOTIVATION

2.2.1 ADS System Monitoring

Autonomous driving systems (ADS) consist of an integration of established systems of adaptive cruise control, parking assistance, and autopilots into a unified functional unit (YURTSEVER et al., 2020). Modern ADS are developed with increasing capabilities to act autonomously with little to no human input, using a perception-plan-execution strategy (YURTSEVER et al., 2020). The perception part is typically delegated to deep neural networks (DNNs) which are capable of learning driving actions from labeled input-output samples (GRIGORESCU et al., 2020). For ADS, typical inputs consist of driving images, whereas the outputs are driving commands predicted by the DNN, such as the angle that the car must steer at to drive safely. The input space of ADS (i.e., all possible driving images) is huge and hard to cover adequately, even with automated testing techniques (GAMBI; MUELLER; FRASER, 2019; Ben Abdessalem et al., 2018; ABDESSALEM et al., 2018; Ben Abdessalem et al., 2016; RICCIO; TONELLA, 2020a). Consequently, one of the main challenges associated with deploying trustworthy ADS on public roads consists in their need to operate safely even in partially unknown and uncertain environments, which can result in unpredictable and hazardous situations. On the other hand, increased acceptance of such driverless vehicles requires a high degree of robustness also in the presence of non-modelled phenomena, uncertainties, as well as errors or inaccuracies at the sensor level (SCULLEY et al., 2015).

Existing works have proposed DNN supervisors to build a safety envelope over a DNN to assess its level of dependability in operation (HENRIKSSON et al., 2019; KIM; FELDT; YOO, 2019; XIAO et al., 2021; ZHANG et al., 2018b; WANG et al., 2020; STOCCO et al., 2020; HUSSAIN; ALI; HONG, 2022; HELL et al., 2021). Generic solutions consist of measuring the distance of a given data point from the distribution of the training dataset (KIM; FELDT; YOO, 2019), or through input validation frameworks based on internal inconsistencies (XIAO et al., 2021), or prediction snapshots (WANG et al., 2020). For ADS, specific runtime monitoring solutions have been proposed to mitigate system-level failures. Frameworks such as SelfOracle (STOCCO et al., 2020), DeepRoad (ZHANG et al., 2018b), or DeepGuard (HUSSAIN; ALI; HONG, 2022) monitor the ADS as a black box and examine its behaviour in response to changeable environmental conditions, essentially only by considering the input images processed by the system.

2.3 PROBLEM

The main limitations of these approaches are twofold. First, black-box solutions can only handle data-driven failures induced by significant changes (e.g., corruptions) in the input image that makes it fall beyond the distribution of the inputs on which the ADS has been

trained (out-of-distribution, or OOD) (DOLA; DWYER; SOFFA, 2021). Thus, they suffer from the inability to capture failures caused by an inadequate training of the DNN model or by bugs at the model level (HUMBATOVA et al., 2020). Second, black-box solutions are prone to false positives/negatives as their functioning is extraneous to the internal state of the system, which can lead to a discrepancy between the system being monitored and the monitor. Indeed, if the ADS and the corresponding monitor have different generalization capabilities, this can cause false alarms to be reported, or, worse, safety-critical failures to be missed, as also noticed in the original papers (ZHANG et al., 2018b; STOCCO et al., 2020; HUSSAIN; ALI; HONG, 2022).

2.4 WORK OBJECTIVES

2.4.1 General Objective

This work investigates the problem of building a white-box ADS failure predictor. Although there are many methods to investigate the internal functioning of a DNN (WANG et al., 2020; KIM; FELDT; YOO, 2019; GAL; GHAHRAMANI, 2016), this work focuses on the attention maps produced by explainable artificial intelligence techniques (XAI). Attention maps (JETLEY et al., 2018; SAMEK et al., 2019; SAMEK; WIEGAND; MÜLLER, 2017) are post-training approaches that highlight the input pixels that influence the output predictions the most. While primarily used for comprehension and debugging of DNNs, Tjoa et al. (TJOA et al., 2022) have provided empirical evidence of the informative content of heatmaps. In this work we leverage attention maps for failure prediction to maintain the reliability of the ADS within a safety net.

Our technique, implemented in a tool called ThirdEye, consists of a self-attention monitor for ADS that turns attention maps into XAI-driven scores used as a white-box confidence estimator of the system. More specifically, ThirdEye performs online monitoring capturing visual snapshots during the execution of ADS and leverages the visual information extracted from the attention maps to automatically identify conditions in which the system is unconfident. We show that attention snapshots offer clues about the reliability of the ADS; ThirdEye synthesizes such snapshots into a confidence score using different summarization strategies (i.e., average, average derivative over time, reconstruction loss). Our technique works in an unsupervised fashion: failure prediction is performed by setting a threshold over the nominal XAI-confidence scores using probability distribution fitting. Anomalous driving conditions are detected when the confidence scores decrease within a detection window that precedes the failure.

We have evaluated the effectiveness of ThirdEye on the Udacity simulator for self-driving cars (Udacity, 2017), using ADS available from the literature and a diverse set of failures induced by adverse operational scenes and mutation testing-simulated malfunctions. In our experiments on +70 simulations accounting for more than 350 failures,

ThirdEye was able to safely anticipate up to 98% of them, up to 3 seconds in advance, a 30% increase with respect to SelfOracle (stocco et al., 2020), a state-of-the-art black-box strategy from the literature. The improvement is particularly evident for failures induced by mutation testing: on average, ThirdEye anticipated 85% more failures caused by mutated driving models. ThirdEye also achieves a better trade-off between prediction of misbehaviours and false alarms, with an F_3 improvement up to 49%.

2.4.2 Contributions

Our work makes the following contributions:

Technique. A self-attention monitoring technique for ADS failure prediction based on attention maps produced by XAI techniques. Our approach is implemented in the publicly available tool ThirdEye (stocco et al., 2022). To the best of our knowledge, this is the first solution that uses XAI techniques to estimate the confidence of a DNN-based ADS and to anticipate system-level failures.

For the preliminary study, an in-depth exploration of the influence of Explainable AI on the generation of corner cases for Neural Networks. The established technique is not only applicable to the present study but can also be leveraged in the creation of simulated road driving scenarios. This has substantial potential to enhance the testing methodology for Autonomous Driving Systems (ADS) Neural Networks, facilitating their robustness and reliability.

Evaluation. An empirical study showing that the XAI-based confidence scores used by ThirdEye are a promising white-box confidence metric for failure prediction, outperforming the black-box approach of SelfOracle (stocco et al., 2020).

Dataset. A dataset of more than 350 out-of-distribution and mutation-testing-induced ADS failures, based on the Udacity simulator for self-driving cars. This dataset can be used to evaluate the performance of failure prediction systems for ADS.

2.5 BACKGROUND

In this section, relevant concepts will be presented about the subjects related to lane-keeping Autonomous Driving Systems (ADS) and failure prediction.

2.5.1 Lane-keeping ADS

ADS benefit from data gathered by sensors, cameras, and GPS to perceive the environment and predict the vehicle's controls (i.e., steer, brake, acceleration) through advanced DNNs.

This work focuses on ADS that perform *behavioural cloning*, i.e., the vehicle learns the *lane keeping* functionality from humanly-labeled driving samples. The lane-keeping component is vital for the safe deployment of DNN-based ADS. The U.S. Department of Transportation, National Highway Traffic Safety Administration (NHTSA) reported that off road failures are second in frequency and first in cost (+15B USD) (TRANSPORTATION, 2007).

Models such as NVIDIA's DAVE-2 (BOJARSKI et al., 2016b) learn how to drive by discovering latent patterns within a training set of images collected when the driver is an expert human pilot, and by predicting the corresponding driving commands imitating the driving behaviour of the human. In its most simplified form, a lane-keeping ADS such as DAVE-2 can be seen as a function $f : R^d \rightarrow [-25°, +25°]$ where d is the dimension of the input image $\boldsymbol{x} \in R^d$ (e.g., for a 140×320 image, $d = 44800$ pixels) and the output is a vector y of length 1, e.g., a real number representing a (predicted) steering angle in the range $\left[-25°, +25°\right]$, where $-25°$ indicates steering full left, $+25°$ indicates steering full right, and 0 means no steering applied.[1]

2.5.2 Failure Conditions for lane-keeping ADS

The ISO/PAS 21448 Safety of the Intended Function (SOTIF) standard (STANDARDIZATION, 2019) mandates risk mitigation strategy to be implemented within ADS to reduce risks and hazards associated with malfunctioning behaviour. At NHTSA Level 4 (High Automation), a system monitor checks for emerging functional insufficiencies with the aim to keep a high functional quality also in extreme conditions (WANG et al., 2020; ZHANG et al., 2018b; HENRIKSSON et al., 2019). The ADS should disengage if the monitor regards the current conditions as unsafe, requiring the human driver to take control of the vehicle.

Among the root causes for ADS failures (e.g., off road driving) SOTIF recalls external unknown and internal uncertain conditions (STANDARDIZATION, 2019). External unknown conditions consist of "abnormal" inputs representing rare, unexpected, and possibly unsupported environmental events, for which no prior knowledge was available during the training of the ADS (e.g., a specific road type, or weather/lighting condition). The DNNs used within ADS are not invariant to severe data distribution changes and this can cause

[1] these values reflect the steering capability of an ADS in a driving simulator (Udacity, 2017).

system-level failures. Internal uncertain conditions correspond instead to misbehaviors of the perception component caused by the bugs inherent to the DNN model, introduced during its development. Instances of such bugs include inadequate training data and suboptimal choice of the model's architecture or of the training hyper-parameters (HUMBATOVA et al., 2020).

2.5.3 Black-box Unsupervised Failure Prediction

Despite standards such as SOTIF (STANDARDIZATION, 2019), in practice, enumerating all possible hazardous conditions for an ADS in a written requirement specification is a challenging, if not infeasible, endeavour. As a consequence, research has considered failure prediction models that can be trained with no supervision (i.e., no knowledge of the anomalies), and, to make them more applicable, with no need to access information of the main system (black-box) (HENRIKSSON et al., 2019; STOCCO et al., 2020; HUSSAIN; ALI; HONG, 2022; HELL et al., 2021).

A black-box unsupervised failure predictor analyzes inputs and assigns a suspiciousness score to them, which should be low (below a threshold) if the inputs are supported, or high (above a threshold) otherwise. Notable examples are one-class SVM (SCHÖLKOPF et al., 1999), clustering (EVERITT; LANDAU; LEESE, 2009), self-organizing maps (KOHONEN, 2001), and autoencoders (AEs) (CHANDOLA; BANERJEE; KUMAR, 2009).

The variational autoencoder (VAE) is the most popular AE architecture (CHANDOLA; BANERJEE; KUMAR, 2009) as it is able to efficiently learn the probability distribution of a large amount of complex data (such as images) using variational inference (AN; CHO, 2015). The VAE is trained to minimize the distance between the original data and its low-dimensional reconstruction with metrics such as the Mean Squared Error (MSE). A low MSE indicates that the input has characteristics similar to those of the training set, whereas a high MSE indicates potentially an OOD sample. As such, VAEs are used in anomaly detection tasks (HUSSAIN; ALI; HONG, 2022; STOCCO et al., 2020; HENRIKSSON et al., 2019), as well as automated validity checkers for DNNs (DOLA; DWYER; SOFFA, 2021).

The main limitations of black-box approaches, including VAEs, consist in their zero knowledge of the system's internal behaviour, thus they are designed to react only for failures induced either by the corruption of inputs, or by a large degree of out-of-distributioness. Indeed, for unknown inputs, the ADS is likely to make a sequence of inaccurate predictions that may ultimately lead to a system failure, because of the prediction errors accumulated over time (HAQ et al., 2020).

2.5.4 Deep Neural Networks Explanation

Explaining the predictions of DNNs has been largely studied using several interpretation methods (TJOA; GUAN, 2019; SAMEK; WIEGAND; MÜLLER, 2017; SAMEK et al., 2019; LAGE

et al., 2019; LEI; BARZILAY; JAAKKOLA, 2016; SPRINGENBERG et al., 2014; ZEILER; FERGUS, 2013). The survey by Tjoa and Guan (TJOA; GUAN, 2019) distinguishes three main categories, namely *verbal methods*, *signal methods*, and *saliency methods*.

Verbal methods such as decision sets (LAGE et al., 2019) or encoder-decoder frameworks (LEI; BARZILAY; JAAKKOLA, 2016) have been adopted in NLP problems since they produce lexical statements that humans can interpret naturally.

Signal methods target the stimulation of individual neurons or collections of neurons in a DNN to reconstruct an image similar to the input, based on the partial information stored in the neurons. However, feature maps produced with methods such as guided back-propagation (SPRINGENBERG et al., 2014), or deconvolutional networks (ZEILER; FERGUS, 2013), are known for producing sparse heatmaps.

Saliency methods explain DNN predictions by attributing a negative or positive value to each input feature according to how much it influenced the prediction. For instance, LIME (RIBEIRO; SINGH; GUESTRIN, 2016) is a black-box technique that understands classification networks' decisions by assessing how the predictions change in response to local perturbations of the input data. Other methods use decomposition of signals propagated by their algorithms and selectively re-arrange them to provide interpretable information. For example, GradCam (SELVARAJU et al., 2016) uses gradient back-propagation up to the last convolutional layer to explain classifiers. Differently, LRP (BINDER et al., 2016) uses relevance scores that are decomposed such that the sum of the scores in each layer of the DNN will be equal to the output. LRP has the drawbacks of generating noisy explanations as well as very similar outputs for samples pertaining to different classes (JUNG; HAN; CHOI, 2021).

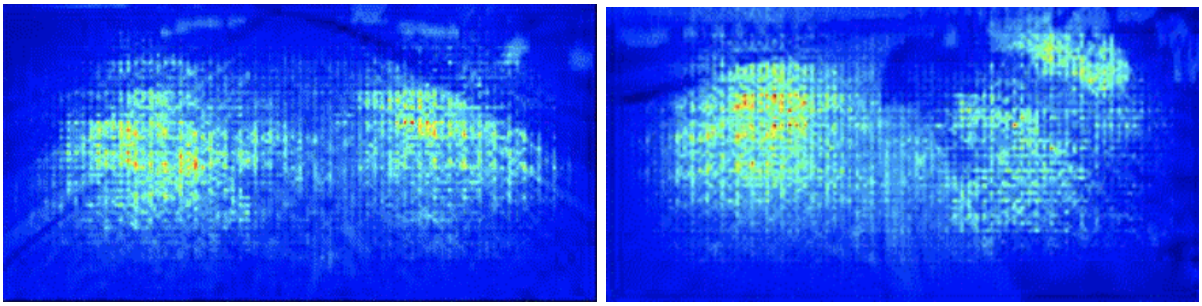
In this work we consider the attention maps produced by the SmoothGrad algorithm (SMILKOV et al., 2017). Unlike LRP, SmoothGrad makes gradient-based explanations sharper by adding noise and averaging over these artificially created noisy gradients. Like GradCAM, attention maps consider the gradient of the output prediction with respect to the input pixels (JETLEY et al., 2018). Unlike GradCAM, SmoothGrad also works with regression DNNs such as those of ADS.

2.6 MOTIVATING EXAMPLE

Attention maps are images where relevant locations correspond to hot color intensities (e.g., red/yellow), whereas irrelevant locations correspond to cold color intensities (e.g., blue).

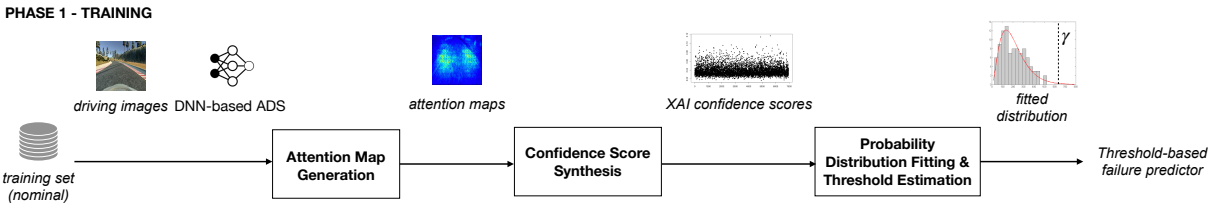
Figure 1 shows an example in which attention maps—obtained with SmoothGrad (SMILKOV et al., 2017)—are indicative of an upcoming failure of the ADS. In nominal conditions (left), the ADS focuses on foreground features that characterize the road. In this case, the attention map portrays two main clusters of attention, corresponding to the road's lanes. When driving in unsupported conditions (right), prior to a failure, the ADS is more

Figure 1 – Attention map of an ADS during nominal driving (left), and a few seconds before an off-road failure (right).



Source: (STOCCO et al., 2022).

Figure 2 – Training of ThirdEye.



Source: (STOCCO et al., 2022).

uncertain. This is reflected in the attention map as only part of the attention still focuses on the road, while substantial attention is also paid to features in the background.

In the next section, we will describe our proposal for using attention maps by SmoothGrad to anticipate failures in conditions that cause the ADS to fail. Our technique aims to capture a *drop of DNN confidence by means of metrics derived from the attention maps*, considering single maps, consecutive maps, or map reconstruction based on nominal maps.

2.7 RELATED WORK

Identifying unexpected driving scenarios is the number one need during ADS testing, according to the survey with developers and domain experts by Lou et al. (LOU et al., 2021). The problem has been tackled by researchers either by (1) generating test cases for ADS, (2) proposing anomaly detection tools. We also provide an overview of (3) generic OOD detectors and (4) the main XAI methods used for ADS testing.

2.7.1 Test Generation for Autonomous Driving

Test generation techniques mostly use search-based techniques to automatically construct test cases for DNN-based ADS (MOGHADAM et al., 2022; RICCIO; TONELLA, 2020a; ABDESSALEM et al., 2018; Ben Abdessalem et al., 2016; Ben Abdessalem et al., 2018; PEI et al., 2017a; TIAN et al., 2018; ZHANG et al., 2018b; KIM; FELDT; YOO, 2019). Test cases are real-

world images of driving scenes, or road abstractions that are rendered within a driving simulator. Abdessalem et al. (ABDESSALEM et al., 2018; Ben Abdessalem et al., 2016; Ben Abdessalem et al., 2018) combine genetic algorithms and machine learning to test a pedestrian detection system. Mullins et al. (MULLINS et al., 2018) use Gaussian processes to drive the search-based test generation towards yet unexplored regions of the input space, whereas Gambi et al. (GAMBI; MUELLER; FRASER, 2019) propose search-based test generation for ADS based on procedural content generation.

Test generators aim to maximize the number of failures, whereas our goal is to predict failures in online mode before they happen. Nevertheless, test generators can be used in conjunction with ThirdEye, to generate conditions for our approach to predict.

2.7.2 Anomaly Detection in Autonomous Driving

We already discussed SelfOracle (STOCCO et al., 2020), for which we performed an explicit empirical comparison in this work. DeepGuard (HUSSAIN; ALI; HONG, 2022) uses the reconstruction error by VAEs to prevent collisions of vehicles with the roadside. DeepRoad (ZHANG et al., 2018b) validates single driving images based on the distance to the training set, using embeddings rooted in the features extracted by VGGNet. In other works (STOCCO; TONELLA, 2021; STOCCO; TONELLA, 2020), continual learning is used to minimize the false positives of a black-box failure predictor. Hell et al. (HELL et al., 2021) evaluate three different OOD detection methods, namely VAEs, Likelihood Regret and the generative modelling SSD, for ADS testing on the CARLA simulator. Henriksson et al. (HENRIKSSON et al., 2019) use the the negative of the log likelihood as a black-box anomaly score. Borg et al. (BORG et al., 2022) propose to pair OOD detection with VAEs with object detection for an automated emergency braking system. Strickland et al. (STRICKLAND; FAINEKOS; Ben Amor, 2018) use an LSTM solution with multiple metrics to predict collisions with vehicles at crossroads.

Our approach differs from the aforementioned black-box approaches because it uses a white-box confidence score of the system synthesized from the attention maps given by an XAI algorithm. For a broad overview of anomaly detection techniques in autonomous driving, we refer the reader to the survey by Bogdoll et. al (BOGDOLL; NITSCHE; ZöLLNER, 2022).

2.7.3 Generic OOD Detectors

AutoTrainer (ZHANG et al., 2021a) monitors the training process of a DNN and automatically repairs it when the metrics used during training degrade. In contrast, ThirdEye operates at testing time, in production, to recognize unexpected execution conditions, while AutoTrainer operates at training time to fix common training faults.

Zhang et al. (ZHANG et al., 2021b) introduce the notion of relative activation and deactivation to interpret the decision behavior of a DNN and propose an algorithm for

automatic detection of OOD inputs. The abstraction relies on classifying neurons into different states based on stronger relative selectivity, which is a quantitative method of measuring the impact of a particular neuron, or a subset of them, on the inference of the model as a whole.

The use of this technique raises some challenges, such as which and how many layers should be selected, and how the different layers should be aggregated. SelfChecker (XIAO et al., 2021) is a tool that helps answer these questions, but the evaluation of the DNN prediction is performed for individual inputs. THIRDEYE uses attention map analysis and does not require to dissect the model layers or the activation states of the neurons, but just to retrieve the gradients during a normal feedforward pass, making it computationally more efficient and easier to integrate into the ADS development process.

Finally, Suraksha (ZHAO et al., 2021) is an automated ADS safety evaluation framework that quantitatively analyzes the safety sensitivity of different versions of an ADS. THIRDEYE can be used as one of the safety quantification metrics of Suraksha and help improve the safety parameters of ADS.

2.7.4 XAI for Autonomous Driving Testing

With the increasing application of DNNs to safety-critical domains such as autonomous during, XAI algorithms represent one of the standard choices to debug DNN's predictions and failures (e.g., during an incident). Moreover, XAI is also being adopted to build novel testing solutions to test DNN-based systems, including, but not limited to, ADS. In this section we focus on the main related propositions, i.e., techniques that use attention maps for ADS testing, and techniques that use XAI as a building block for DNN testing. For a complete overview of the state of the art on XAI for ADS, we refer the reader to the survey by Atakishiyev et al. (ATAKISHIYEV et al., 2021).

VisualBackProp (BOJARSKI et al., 2016a) was created to visualize which group of pixels of the input image contributes more to the predictions of a convolutional neural network (CNN). Kim and Canny (KIM; CANNY, 2017) explore the use of attention maps for explaining the CNN behaviour in a ADS. Lateef et al. (LATEEF; KAS; RUICHEK, 2021) uses generative adversarial networks to train a predictive model that generates attention maps from road scenes and gives more prominence to the objects in the scene that are most important to the driver's decision making (e.g., other cars, pedestrians, and traffic lights/signs). Xu et al. (XU et al., 2020) investigated the use of XAI techniques to detect action-inducing objects, i.e., objects that have a relevant effect on a driving decision, and jointly predict actions and their respective explanations. Mohseni et al. (MOHSENI; JAGADEESH; WANG, 2019) train the DAVE-2 model to predict a steering angle given the attention maps by VisualBackProp (BOJARSKI et al., 2016a). Similarly, THIRDEYE also leverages XAI to increase the level of reliability of an ADS. Differently to the aforementioned works, THIRDEYE focuses on failure prediction of lane-keeping based ADS during

external unknown and internal uncertain driving conditions.

Fahmy et al. (FAHMY et al., 2020) apply clustering to LRP heatmaps capturing the relevance of the DNN predictions to automatically support the identification of failure-inducing inputs. Such data is used for the retraining of a gaze detection system that uses DNNs to determine the gaze direction of the driver. The authors present an extension of the previous work (FAHMY; PASTORE; BRIAND, 2022) in which inputs identified by the heatmap-based mechanism are given in input to a search-based test generator. In contrast, in this work we use attention maps from SmoothGrad to support the prompt detection of low-confidence scenarios of a lane-keeping DNN-enabled ADS. Zohdinasab et al. (ZOHDINASAB et al., 2021) use illumination search to cover a feature map of external behaviours of an ADS. These feature maps are used as an adequacy criteria of the inputs generated by an ADS test generator, whereas we use attention maps from the XAI domain to validate the inputs processed by the ADS.

2.8 METHODOLOGY/APPROACH

Our approach ThirdEye consists of two main phases, namely Training and Usage. In the first phase (Training, see Figure 2), ThirdEye automatically generates the attention maps for nominal driving instances of the ADS (see subsubsection 2.8.1.1). Such attention maps are a visual snapshot of the ADS performance during nominal driving behaviour. ThirdEye is based on two intuitions: (1) attention maps derived during the processing of the inputs by the ADS are indicative of the confidence of the system (TJOA et al., 2022), and (2) nominal and failure-inducing attention maps exhibit differences that can be captured by an anomaly detector.

ThirdEye turns the attention maps into XAI-driven confidence scores of the ADS using different summarization methods (subsubsection 2.8.1.2). We consider a realistic setting, in which instances of failing driving behaviour cannot be sampled in any representative way, since failure conditions are potentially very diversified and partly unexpected. Hence, ThirdEye fits a probability distribution using only nominal scores. Then, it automatically estimates a threshold from such probability distribution (subsubsection 2.8.1.3). This threshold is derived from the user defined permissiveness of the failure predictor to accept false alarms – i.e., from the tolerable false positive rate, a tunable parameter of our approach.

In the second phase (Usage, see Figure 3), ThirdEye is used along with the main ADS system to automatically predict whether the driving conditions are safe or unsafe, according to the attention maps retrieved during driving, and the threshold estimated during the Training phase. If a driving condition is deemed as unsafe, ThirdEye warns the main driving component of the ADS (or the human driver). In the next sections, we describe each step of each phase in more detail.

2.8.1 Training of ThirdEye

2.8.1.1 Attention Map Generation

ThirdEye assumes having access to the training set $T = \{x_1, x_2, x_3, ..., x_n\}$ used to train the ADS, and to the trained ADS f. Our approach, however, does not need to modify the ADS model's architecture, nor to retrain it.

ThirdEye uses the XAI algorithm SmoothGrad (SMILKOV et al., 2017) to retrieve an attention map for each driving image $x \in T$ used to train f. In particular, SmoothGrad produces an attention map $\boldsymbol{h} = \{h_p\}$ assigning each pixel p of x a value $\{h_p\} = \mathcal{H}(x, f, p)$ according to some function $\mathcal{H}$ derived from f. In SmoothGrad, the function $\mathcal{H}$ constructs $\boldsymbol{h}(x)$ by differentiating f with respect to the input x:

$$\boldsymbol{h}(x) = \partial f(x)/\partial x$$

The attention map $\boldsymbol{h}$ has the same dimensionality as $\boldsymbol{x}$ (i.e., width W, height H, and C channels) and represents how much difference a small change in each pixel of x would make to the prediction score of f. Since the derivative of the function f may fluctuate sharply at small scales (SMILKOV et al., 2017), SmoothGrad uses a stochastic approximation by taking random samples in the neighbourhood of the input x, and averaging the resulting attention maps. Mathematically,

$$\hat{\boldsymbol{h}}(x) = \frac{1}{n} \sum \boldsymbol{h}(x + \mathcal{N}(\mu, \sigma^2))$$

To summarize, SmoothGrad (1) generates n versions of the image of interest by adding Gaussian noise to it, (2) it creates pixel attribution maps for all n versions of the image, and (3) it averages the pixel attribution maps. Averaging over multiple maps "smooths out" the derivative fluctuations. The output of the attention map generation step is a set $H = \{h_1, h_2, ..., h_n\}$ of attention maps for each image of the training set T.

The use of Gaussian noise in the SmoothGrad algorithm is aimed at mitigating the sharp fluctuations commonly observed when calculating derivatives in complex neural network models. These derivatives measure the sensitivity of the model to changes in individual pixel values within an image. Gaussian noise introduces randomness to the input data, enabling the generation of multiple attention map iterations. Averaging these iterations results in a smoother and more stable attention map, offering a reliable representation of the model's sensitivity to pixel-level changes in the input image.

2.8.1.2 Confidence Score Synthesis

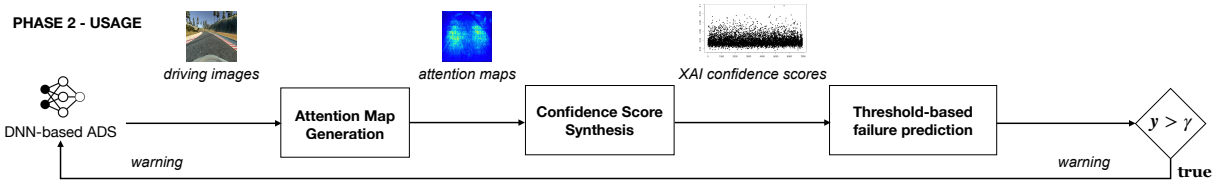
THIRDEYE uses three summarization functions to turn raw attention maps into XAI-driven confidence scores, namely average, derivative, and reconstruction loss.

Heatmap Average Function (HA). The first summarization function is based on the intuition that an attention map captured during nominal driving will have high relevance values (i.e., pixel intensities) focused on specific regions of interest (e.g., the lanes, see Figure 1), whereas attention will be more scattered and with lower pixel intensities during bad driving behaviour. As such, the HA function computes the average pixel intensity of an attention map.

Assuming attention maps have width W, height H and C channels (usually, RGB channels for colour images), notationally, for each attention map $\boldsymbol{h} \in H$, the average attention map score $\overline{\boldsymbol{h}}$ is computed as follows:

$$\overline{\boldsymbol{h}} = \frac{1}{WHC} \sum_{i=1,j=1,c=1}^{W,H,C} h_{[i][j][c]}$$

Figure 3 – Usage of ThirdEye.



Source: (STOCCO et al., 2022).

When applied to the whole training set T, the HA function returns the set of attention map average scores of each individual attention map in T, $HA = \{\overline{h_1}, \overline{h_2}, \overline{h_3}, ..., \overline{h_n}\}$.

Heatmap Derivative Function (HD). The second summarization function is based on the intuition that attention maps that do change frequently during driving could signal a poorly confident ADS. Thus, the HD function computes the average of the derivative of attention maps over time. Notationally, for two consecutive attention maps $\boldsymbol{h_{t-1}}, \boldsymbol{h_t} \in H$, the average of the derivative of attention map $\nabla \boldsymbol{h}$ is computed as follows:

$$\overline{\nabla \boldsymbol{h_t}} = \frac{1}{WHC} \sum_{i=1,j=1,c=1}^{W,H,C} h_{t[i][j][c]} - h_{t-1[i][j][c]}$$

The HD function returns the set of attention map average derivatives of each individual attention map in T (but the first), $HD = \{\overline{h_1}, \overline{\nabla h_2}, ..., \overline{\nabla h_n}\}$.

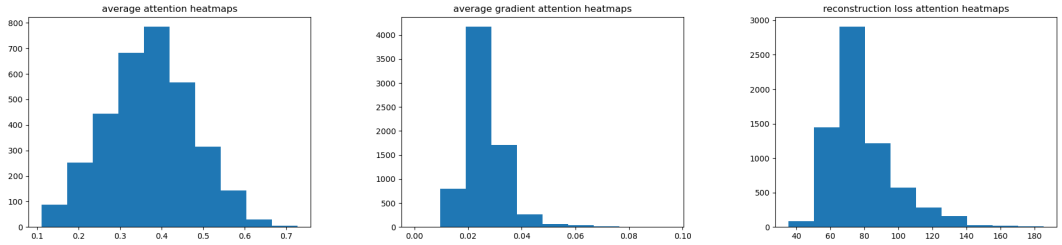
Heatmap Reconstruction Loss Function (HRL). The third summarization function uses reconstruction loss, i.e., it is based on learning a reconstruction function of latent features in the attention maps captured during nominal driving. A failure to reconstruct data during testing of the ADS can signal the presence of potentially failure inducing driving conditions.

Reconstructor: The first step involves obtaining a model of normality from the training driving scenarios. In the training set, we capture the heatmap images under nominal situations. Then, we train our driving scenario reconstructor with such "normal" instances of heatmpas. Let us consider a training set $X = \{x_1, x_2, \ldots, x_n\}$ of n heatmap image frames, where the index $i \in [1 : n]$ of $x_i \in X$ represents the discrete time t.

Depending on the considered architecture, a reconstructor can be single-image or sequence-based. For single-image reconstructors, only one image frame is considered at a time. When the discrete time is $t = i$, x_i is the input, and the reconstructor recreates it into x'_i.

For sequence-based reconstructors, assuming k image frames preceding x_i are used to reconstruct x_i, the sequence $\langle x_{i-k}, \ldots, x_{i-1} \rangle$ is the input used to output x'_i, a prediction of the actual current heatmap frame x_i. For instance, for $k = 3$ and $i = 4$, the reconstructor

Figure 4 – Examples of distributions of HA (left), HD (center), and HRL (right) XAI confidence scores.



Source: (STOCCO et al., 2022).

considers the sequence $\langle x_1, x_2, x_3 \rangle$ to predict the current heatmap frame x_4.

At the end of this task, each reconstruction error $e_i = d(x_i, x_i')$ can be computed, where d is a proper distance function (e.g., Euclidean distance). This results in the set of reconstruction errors $E = \{e_1, e_2, \ldots, e_n\}$, available for all elements in the training set $X = \{x_1, x_2, \ldots, x_n\}$. Following the same guidelines as SelfOracle (STOCCO et al., 2020), but this time, we are now applying them to attention map images.

The function HRL computes the reconstruction errors of the attention maps according to a variational autoencoder $\mathcal{V}$. Autoencoder (AE) is a DNN designed to reconstruct its own input. It consists of two sequentially connected components (an encoder, and a decoder) that are arranged symmetrically. The simplest form of autoencoder (SAE) is a three-layer DNN: the input layer, the hidden layer, and the output layer (STOCCO et al., 2020). The encoder of $\mathcal{V}$ encodes a given input $\boldsymbol{x} \in R^d$ to a compressed representation $\boldsymbol{z} \in R^z$ using a function $enc(\boldsymbol{x}) = \boldsymbol{z}$. The decoder of $\mathcal{V}$ decodes the encoded input with a reconstruction function $dec(\boldsymbol{z}) = \boldsymbol{x'}$, where $\boldsymbol{x'}$ is the reconstructed input $\boldsymbol{x}$. $\mathcal{V}$ minimizes a loss function $\mathcal{L}(\boldsymbol{x}, dec(enc(\boldsymbol{x})))$, which measures the distance between the original data and its low-dimensional reconstruction. Following existing guidelines (STOCCO; TONELLA, 2021), we set the dimension of the encoded representation z to 2 and used the mean squared error (MSE) as a loss function.

Notationally, for each attention map $\boldsymbol{h} \in H$, the reconstruction error $\boldsymbol{h_e}$ is computed as follows:

$$\boldsymbol{h_e} = \mathcal{L}(\boldsymbol{h}, dec(enc(\boldsymbol{h})))$$

The HRL function returns the set of attention maps' reconstruction errors of each individual attention map in T, $HRL = \{h_{e1}, h_{e2}, h_{e3}, ..., h_{en}\}$.

Windowing of Confidence Scores. To mitigate the effect of individual single frame outliers, which are not expected to have a big impact on the performance of the ADS, THIRDEYE applies a window function on non-overlapping, fixed length, sequences of scores. Two simple window functions are considered, one that computes the *maximum*

score within a window, and a second that computes the *arithmetic mean* of the scores within a window. Window functions are applied to each of the proposed XAI confidence scores.

2.8.1.3 Probability Distribution Fitting & Threshold Estimation

The sets of (windowed) XAI confidence scores HA/HD/HRL represent a model of normality collected in nominal driving conditions using different synthesis methods from the attention maps.

To determine a threshold γ that sets the expected false alarm rate in nominal conditions below some configurable level, we use probability distribution fitting to obtain a statistical model of the XAI confidence scores.

In particular, γ is computed by (1) estimating the shape κ and scale θ parameters of a fitted Gamma distribution of the XAI confidence scores and (2) by selecting an acceptable false alarm rate (STOCCO et al., 2020). We get a gamma instead of a χ^2 distribution because pixel-wise errors have different (channel/pixel dependent) and non-unitary variances. **Definition of Gamma Distribution:** Gamma is a probability model for a continuous variable on $[0, \infty)$ which is widely used in engineering, science, and business, to model continuous variables that are always positive and have skewed distributions. The probability density function of a random variable $x \sim \Gamma(\alpha, \beta)$ is:

$$f(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}, \quad x > 0;\ \alpha, \beta > 0 \quad (3)$$

where α is the shape parameter (which affects the shape of the distribution), β is the rate parameter (or inverse scale, which stretches/shrinks the distribution), and Γ is the gamma function. When α is large, the gamma distribution closely approximates a normal distribution with the advantage that the gamma distribution has non-zero density only for positive real numbers. The gamma function $\Gamma(z)$ can be seen as a solution to the interpolation problem of finding a smooth curve that connects the points (n, m) with $m = (n-1)!$ at any positive integer value for n. Such a definition was extended to all complex numbers with a positive real part by Bernoulli, as a solution to the following integral:

$$\Gamma(z) = \int_0^\infty x^{z-1} e^{-x} dx, \quad \text{for } \Re(z) > 0;$$

(STOCCO et al., 2020)

In this work, we set γ to the 95% percentile (i.e., we deem 5% as an acceptable false positive rate), in line with previous works (STOCCO et al., 2020). For example, for HRL, $\gamma_{0.95} = p_{0.95}(\mathcal{L}(\boldsymbol{h}, g(f(\boldsymbol{h}))) | \boldsymbol{h} \in W(H))$, where $W(H)$ represents the confidence scores after windowing.

2.8.2 Usage of ThirdEye

Figure 3 shows the second phase of our approach, Usage, in which ThirdEye is used as a runtime monitoring technique during the runtime execution of the ADS.

The ADS generates driving data that are processed by our approach. ThirdEye analyzes the incoming stream of driving images and attention maps are retrieved (subsubsection 2.8.1.1). Next, confidence scores are synthesized from the attention maps (either HA/HD/HRL, see subsubsection 2.8.1.2). When sufficient data samples are collected (e.g., matching the window size chosen during training, see subsubsection 2.8.1.2), ThirdEye applies the window function to the stream (either max or mean). Each resulting score is compared against the threshold $\gamma_{0.95}$, which determines whether the windowed sequence of XAI confidence scores is to be regarded as anomalous. In such a case, a warning is sent to the ADS (or to the human driver); otherwise, ThirdEye keeps monitoring the next incoming driving frames.

2.8.3 Implementation

We implemented our approach in a Python tool called ThirdEye, which is available (Stocco, 2022). The tool supports ADS models written in Tensorflow/Keras, and it is integrated in the Udacity simulator for self-driving cars (Udacity, 2017). For computing the attention maps, ThirdEye leverages the SmoothGrad (SMILKOV et al., 2017) implementation available in the toolkit `tf-keras-vis` (KUBOTA, 2021).

2.9 EMPIRICAL EVALUATION

2.9.1 Research Questions

We consider the following research questions:

RQ_1 (effectiveness): How effective is ThirdEye at predicting failures of ADS? What is the best configuration?

RQ_2 (prediction over time): How does the prediction power of ThirdEye change when considering different detection periods?

RQ_3 (comparison): How does ThirdEye compare with SelfOracle (stocco et al., 2020), a failure predictor from the literature?

The first research question (RQ_1) aims to assess whether our approach is able to attain a high failure prediction rate, and which configuration (i.e., XAI confidence scores and window functions) yields the best prediction rate score. Ideally, failure prediction is only useful if it helps to anticipate a failure, which is studied in the second research question (RQ_2). Lastly, to assess the effectiveness of our approach over existing solutions, the final research question (RQ_3) compares ThirdEye with the state-of-the-art failure predictor for ADS (stocco et al., 2020).

2.9.2 Testbed

We tested ThirdEye through simulation-based testing. The usage of simulation platforms is standard for testing ADS as simulator-generated data yield comparable conditions as the ones experienced in real world (haq et al., 2020; lou et al., 2021; stocco; pulfer; tonella, 2022). Moreover, driving simulators allow testing an ADS at the system level (online testing) because the DNN is embedded within the operational ecosystem in which the ADS is designed to operate. Testing the ADS only from the DNN model perspective (offline testing), disconnected from the ADS system, is not useful to expose the safety-critical failures that occur during in-field testing, such as the ones considered in this work (see subsection 2.5.2).

As simulation platform, we used the Udacity simulator for self-driving cars (Udacity, 2017), a cross-platform driving simulator developed with Unity3D (unity, 2021), used in the ADS testing literature (stocco et al., 2020; riccio et al., 2020; jahangirova; stocco; tonella, 2021; hussain; ali; hong, 2022; stocco; tonella, 2021; stocco; tonella, 2020). The simulator supports various closed-loop tracks for testing behavioral cloning ADS models, including the ability to generate changeable driving scenarios (e.g., weather effects), which is useful to test an ADS on both nominal and unseen conditions.

In this work, we chose the default sunny weather condition as the reference nominal scenario for our ADS models. Other choices of nominal condition are of course possible (e.g., snow). For unsupervised learning techniques such as ours (i.e., techniques that do not assume the availability of a representative set of anomalous conditions when training

Figure 5 – Examples of conditions from our evaluation set. Left: nominal (sunny). Center: OOD extreme (snow). Right: OOD moderate (snow).

Source: (STOCCO et al., 2022).

the failure predictor), the only requirement is that the chosen supported conditions are the same that are known at training time by the ADS.

2.9.3 Object of Study

To implement DNN-based ADS, we use Nvidia's DAVE-2 model (BOJARSKI et al., 2016b), a reference model widely used as object of study in prior related work (PEI et al., 2017a; TIAN et al., 2018; ZHANG et al., 2018b; STOCCO et al., 2020; RICCIO et al., 2020; JAHANGIROVA; STOCCO; TONELLA, 2021; HUMBATOVA; JAHANGIROVA; TONELLA, 2021). (ZHANG et al., 2018b) DAVE-2 consists of 9 layers, including a normalization layer, 5 convolutional layers, and 3 fully connected layers (CommaAI, 2016).

We obtained the trained DAVE-2 model from the replication package of our baseline (STOCCO et al., 2020), to make sure to test the failure predictors using the same ADS used in previous work.

2.9.4 Procedure

2.9.4.1 Evaluation Set

We simulate the ADS testing practices customary of industry, where testers use a closed-loop track in a virtual environment, prior to on-road testing on public roads (CERF, 2018; BGR Media, LLC, 2018; Waymo Driver, 2021; Waymo Secret Testing, 2017). We consider two kinds of scenarios for testing our failure predictor.

External unknown scenario. The first kind of testing scenario deals with failures induced by *out-of-distribution conditions* (OOD), exposing an ADS that has been trained on some given nominal conditions and environment to different instances of that environment. We use two OOD benchmarks in our study.

The first benchmark contains simulations provided by the replication package of the SelfOracle paper (STOCCO et al., 2020). We refer to this benchmark as $\text{OOD}_{\text{extreme}}$ because it is characterized by severe illumination conditions w.r.t. the nominal sunny scenario (see Figure 5). It accounts for 21 simulations with different degrees of extreme OOD conditions:

day/night, rain, snow, fog, day/night + rain, day/night + snow, day/night + fog. These conditions were created by (i) altering the environment's skybox (invisible ceiling object located at the boundary of the map) from sunny to adverse weather luminosity and by (ii) adding weather particles (snow or rain) rendered at runtime along the track.

We also consider a second benchmark of milder OOD conditions, called $\text{OOD}_{\text{moderate}}$, This second benchmark evaluates our approach considering weather effects only, i.e., without interferences due to adverse weather luminosity (see Figure 5). We deactivated the adverse weather luminosity skybox, while retaining a single unexpected weather condition at a time, namely *rain*, *fog*, or *snow*. We executed the DAVE-2 ADS varying the intensity of each condition in the range $\left[10\%, \ldots, 100\%\right]$, thus 10 times for each weather condition (day/night was discarded because non tunable).

Overall, concerning external unknown scenarios, a total of 51 OOD one-lap simulations were collected: 21 for $\text{OOD}_{\text{extreme}}$ and 30 for $\text{OOD}_{\text{moderate}}$ (10 × rain, 10 × fog, 10 × snow).

Internal uncertain scenario. The second kind of testing scenarios deals with *faulty ADS models* produced by automated mutation testing (HUMBATOVA; JAHANGIROVA; TONELLA, 2021) that drive on the simulator under nominal conditions (sunny). Intuitively, these scenarios simulate the development process of an ADS model that has not been yet trained adequately. The third benchmark—referred to as Mutants—represent these scenarios. We obtained instances of mutated DAVE-2 models from the replication package of the DeepCrime mutation testing tool (HUMBATOVA; JAHANGIROVA; TONELLA, 2021). DeepCrime automatically mutates a DNN model using mutation operators designed to mimic real fault types occurring when developing DNNs, considering both data-level faults (e.g., wrongly labelled training data) and model-level faults (e.g., a suboptimal learning rate or dropout rate).

We executed all DAVE-2 mutants in the Udacity simulator and discarded those that were consistently failing (e.g., the corruption induced by a particular mutation operator caused severe malfunctions to the ADS driving from the very beginning of the simulation). A total of 20 one-lap simulations were retained that we confirmed to create internal uncertain scenarios (more details about the selected mutation operators are in our replication package (Stocco, 2022).

Summary. Overall, our evaluation set comprises 349 failures that our approach is expected to predict and anticipate. Mutation testing caused most of the failures (66%), which is expected from a technique that systematically injects faults, whereas out-of-distribution conditions induced less failures (34%) as they were applied with different, increasing, levels of severity. Both scenarios are of interest for a failure predictor, which should be agnostic about the conditions that cause the failures (i.e., unknown inputs or DNN model bugs). To estimate the threshold used by ThirdEye, we finalized the evaluation data collection

by performing three one-lap simulations under nominal sunny weather conditions (one for each of three benchmarks $\text{OOD}_{\text{extreme}}$, $\text{OOD}_{\text{moderate}}$, and Mutants) using the robust, unmutated, DAVE-2 model.

2.9.4.2 Detection Windows in Evaluation Set

The Udacity simulator automatically labels individual failing frames as either nominal or failing using a boolean flag, according to whether the ADS was on track or off-track, respectively. Since the goal of our framework is on predicting misbehaviors before they occur, we focus on the part of the simulation *preceding each failure*, whereas the frames labeled as failing are not considered.

Each simulation can exhibit multiple failures: in our evaluation strategy we assessed each failure individually. For each of them, we consider a detection window corresponding to one second of simulation in the Udacity simulator. We move the detection window from 1 to 3 seconds prior to the failures (time to failure, TTF for short). Studies on pre-crash automated seat belt systems (ZHAO et al., 2017; LEE; LEE, 2013) indicate a range between 3 seconds to half a second as adequate TTF values for the activation of automated seat belt tightening. Also, according to previous studies in the Udacity simulator (STOCCO; TONELLA, 2021), a TTF of 3 seconds is deemed sufficient to avoid failures at 30 mph, which is the constant cruising speed of the ADS in the simulator.

2.9.4.3 ThirdEye's Configurations

We evaluate six configurations of ThirdEye. For SmoothGrad (SMILKOV et al., 2017), we use the same hyper-parameters suggested in the original paper, specifically a noise level of 20% and $n = 20$ samples for noise attenuation. Regarding the XAI confidence scores synthesis strategy, we assess all three alternatives, namely heatmap average (HA), heatmap derivative (HD), and heatmap reconstruction loss (HRL). We also vary the windowing method strategy in the detection window, using mean or max. On the detection sequences, if the mean/max score is higher than the automatically estimated threshold γ_{95}, an alarm is triggered (see subsubsection 2.8.1.3).

We executed ThirdEye to capture the attention maps for all simulations of our evaluation set using the studied ADS as input. For the external unknown scenarios ($\text{OOD}_{\text{extreme}}$ and $\text{OOD}_{\text{moderate}}$), we used the DAVE-2 model from the replication package of our baseline (STOCCO et al., 2020). For the internal uncertain scenarios (Mutants), we used all the selected 20 mutants.

2.9.4.4 Baseline

We use SelfOracle (STOCCO et al., 2020), a black-box misbehaviour predictor, as baseline for ThirdEye. We chose SelfOracle for the following reasons: (1) it is designed for the

Figure 6 – Results for all failure predictors. Scores are computed for γ_{95}. Average F_3 scores are highlighted in bold, best F_3 scores are highlighted in grey.

Benchmark	TTF (s)	ThirdEye										SelfOracle			ThirdEye								SelfOracle		
		Windowing: max													Windowing: mean										
		HA			HD			HRL			Rec. Loss				HA			HD			HRL			Rec. Loss		
		Pr	Re	F_3	Pr	Re	F_3	Pr	Re	F_3	Pr	Re	F_3	Pr	Re	F_3	Pr	Re	F_3	Pr	Re	F_3	Pr	Re	F_3	Pr	Re	F_3
OOD$_{\text{extreme}}$	1	40	90	79	30	100	80	27	86	76	24	90	33	26	46	36	19	58	54	10	25	25	20	79	33
	2	37	86	81	29	100	80	26	86	79	24	92	32	32	60	55	21	64	55	14	38	32	16	68	32
	3	41	93	82	30	100	80	29	93	82	20	77	33	38	71	65	24	71	62	21	51	47	16	68	33
	avg	39	90	**81**	30	100	**80**	27	89	**79**	23	87	**33**	32	59	**52**	21	64	**57**	15	38	**35**	17	71	**33**
OOD$_{\text{moderate}}$	1	49	92	84	35	99	83	35	100	83	24	100	75	8	17	15	27	65	56	26	79	65	24	100	75
	2	41	72	66	33	96	80	34	100	83	23	100	72	25	41	39	22	46	40	24	76	62	22	92	65
	3	41	71	66	33	94	78	34	95	79	23	93	59	24	36	34	25	49	44	24	72	59	18	72	45
	avg	44	78	**72**	34	96	**80**	34	98	**82**	23	98	**69**	19	31	**29**	25	53	**47**	25	76	**62**	21	88	**62**
Mutants	1	82	100	98	70	100	96	70	100	96	57	97	90	66	61	61	75	99	95	61	88	84	41	59	56
	2	82	100	98	70	99	95	70	100	96	47	70	66	66	56	57	70	80	78	50	65	62	6	9	9
	3	81	100	98	69	100	96	69	100	96	40	54	52	68	59	59	64	67	66	41	42	42	1	3	2
	avg	82	100	**98**	69	100	**96**	69	100	**96**	48	74	**69**	67	58	**59**	70	82	**80**	51	65	**63**	16	24	**22**
Average (All)	1	57	94	87	45	100	86	44	95	85	35	96	66	33	41	37	40	74	68	32	64	58	28	79	55
	2	53	86	82	44	99	85	43	95	86	31	87	57	41	52	50	37	63	58	29	60	52	15	56	35
	3	54	88	82	44	98	85	44	96	85	27	75	48	43	55	53	37	63	58	29	55	49	12	48	27
	avg	55	89	**83**	44	99	**85**	44	96	**85**	31	86	**57**	39	50	**47**	38	67	**61**	30	60	**53**	18	61	**39**

Source: (STOCCO et al., 2022).

task of failure prediction of ADS; (2) it is a competitive approach; results show that it outperforms the input validation strategy of DeepRoad (ZHANG et al., 2018b); (3) it was developed, integrated, and experimented on the Udacity simulator, which mitigates the threats to the internal validity that are possible when experimenting a tool in a simulation environment different from the one in which it was implemented.

We use the best configuration of SelfOracle presented in the original paper, i.e., a variational autoencoder (VAE) that reconstructs *driving images* and uses the reconstruction loss as a measure of confidence. The VAE has a latent size of 2 and it was trained to minimize the MSE (see subsection 2.5.3) between the original and reconstructed nominal images (sunny). In the original SelfOracle paper, only the arithmetic mean of the detection window was evaluated (STOCCO et al., 2020). In our study, we evaluate two configurations of SelfOracle, using both the mean and the max computed on the detection window, the latter being a new experimental contribution of this work.

2.9.4.5 Metrics used for Analysis

We compute the true positives as the number of correct failure predictions within the predefined TTF (see Section 2.9.4.2) and the false negatives as the number of missed failure predictions when our framework does not trigger an alarm in a detection window. We remember that the false positives and true negatives are measured using nominal simulations.

Our primary goal is to achieve high Recall (Re), or true positive rate, defined as Re=TP/(TP+FN)). Recall measures the fraction of safety-critical failures detected by a technique. It is also important to achieve high precision (Pr), defined as Pr=TP/(TP+FP). Precision measures the fraction of correct warnings that a technique reports. We also consider the F_{beta} score (BLAIR, 1979), with $\beta = 3.0$, as a weighted balance between precision and recall ($F_3 = \frac{10 \cdot \text{Precision} \times \text{Recall}}{9 \cdot \text{Precision} + \text{Recall}}$). We are interested in an F-measure that weights recall higher compared to precision, because the cost associated with false negatives is very high in the safety-critical domain (BLAIR, 1979) as it means a missed failure detection. In contrast, in our setting, the cost associated with false positives is relatively lower compared to false negatives. A false alarm causes annoyance to the human driver (or to the ADS) when there is no actual hazard; thus their number should be kept low.

2.9.5 Results

2.9.5.1 Effectiveness (RQ_1)

Figure 6 presents the effectiveness results for all configurations of ThirdEye (HA, HD, HRL) and SelfOracle (Rec. Loss), divided by windowing function (max, mean). Results are averaged across conditions, split between external unknown conditions (OOD_{extreme} and OOD_{moderate}) and internal uncertain conditions (Mutants). The effectiveness metrics

consider a confidence threshold γ_{95} (see Section 2.9.4.3), i.e., the threshold associated with an expected 5% false positive rate. Precision (Pr) is measured in anomalous conditions, which explains why it is lower than the expected value associated with γ_{95}, which is 95% in nominal conditions (see subsubsection 2.8.1.3).

Due to space constraints, in this section, we only comment the average F_3 scores over all benchmarks. On average, in terms of F_3, ThirdEye with windowing=max has a 77%/39%/60% average improvement over ThirdEye with windowing=mean, for HA/HD/HRL, respectively. However, this does not impact negatively the false alarm rate, which remains low (average Pr values for windowing=max are higher than those for windowing=mean). HD/HRL scores are slightly higher in terms of F_3 than HA (+2%).

We assessed the statistical significance of these differences using the non-parametric Mann-Whitney U test (WILCOXON, 1945) (with $\alpha = 0.05$) and the magnitude of the differences using the Cohen's d effect size (COHEN, 1988). The difference in F_3 score between HA and HD/HRL were found to be statistically significant (p-value < 0.05) even if with a negligible and small effect sizes. As expected by looking at the average F_3 scores of Figure 6, there is no statistically significant difference between HD and HRL (p-value ≥ 0.05).

RQ$_1$: *The configuration of ThirdEye using heatmap derivative function (HD) and reconstruction loss function (HRL), configured with windowing=max, achieve the highest failure prediction rate ($F_3 = 85\%$) over all conditions.*

2.9.5.2 Prediction Over Time (RQ$_2$)

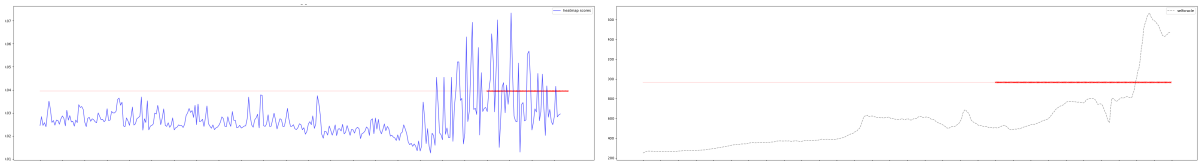
Figure 6 reports the effectiveness considering different TTF (Column 2). In principle, failure prediction should get more challenging as we move farther from the failure instant. This is true for all configuration of ThirdEye, except for HA (windowing=mean), in which the average prediction power (F_3) is higher for {2, 3} seconds before the failures that 1 second before them, on average (+30%). For ThirdEye HRL/HD/HA with windowing=max, the F_3 scores remain stable over time. On average, the prediction power decreases only by -6%/-1%/-1% as we move away from the failures.

RQ$_2$: *On average, the effectiveness of the best configurations of ThirdEye (HRL/HD windowing=max) remains high up to 3 seconds before the failures (-1% average F_3 decrease).*

2.9.5.3 Comparison (RQ$_3$)

Considering the average F_3 scores across benchmarks, all configurations of ThirdEye are superior to SelfOracle at predicting misbehaviours.

Figure 7 – Failure (red stroke) induced by mutation testing; threshold γ is represented by the red line. Left: ThirdEye predicts the failure a few seconds ahead (see spikes above the threshold). Right: SelfOracle reacts only when the failure has happened, when the ADS is off-road, and the input image deviates substantially from the nominal, on-road, driving frames.



Source: (STOCCO et al., 2022).

On the $\text{OOD}_{\text{extreme}}$ benchmark, ThirdEye scores a +142% increase in F_3 w.r.t. SelfOracle (the benchmark used in that work). For $\text{OOD}_{\text{moderate}}$ conditions, average F_3 scores raise to 62%, for ThirdEye's windowing=max, whereas the failure detection rate by ThirdEye (HD) is +16% higher (80%). For Mutants, our results show a remarkable difference of effectiveness between ThirdEye over SelfOracle. The configuration HD (windowing=max) predicts *all failures* induced by internal uncertain conditions (Re=100%), a +35% increase w.r.t SelfOracle, whereas for F_3 the increment is +39%.

Overall, average results for F_3 show significant improvements of ThirdEye over SelfOracle, regardless of the configuration being used and the reaction period considered. The best configurations HD/HRL (windowing=max) from RQ_1 achieve +49% failure prediction scores (F_3). We assess the statistical significance of the differences between ThirdEye HD and SelfOracle using the non-parametric Mann-Whitney U test (Wilcoxon, 1945) (with $\alpha = 0.05$), the magnitude of the differences using the Cohen's d effect size (Cohen, 1988). Statistical tests tell that the difference in F3 score between HD are statistically significant (p-value < 0.05) with a large effect size.

RQ_3: *ThirdEye outperforms SelfOracle in terms of failure prediction and minimization of false alarms (see F_3), with statistical significance.*

We analyzed qualitatively some of the failures, to understand the reasons behind the disagreements between ThirdEye and SelfOracle. Figure 7 reports a meaningful example from our experiments concerning a failure induced by mutation testing. In the example (not show in Figure 7 for space reasons), a mutated version of DAVE-2 is driving on nominal scenarios and fails in proximity of a bend on the right, missing the bend and proceeding straight off-road.

From the plot on the left, we can see that the white-box approach by ThirdEye (HD windowing=max) is able to anticipate the failure as the XAI confidence score spikes a few seconds before the off-road episode. On the other hand, SelfOracle misses the failure. The reconstruction errors (plot on the right) of the driving frames raise above the threshold

only when the ADS is already off-track and the image captured by the camera deviates substantially from the nominal or nearly-failing on-road images.

2.9.6 Threats to Validity

2.9.6.1 Internal validity

We compared all variants of ThirdEye and SelfOracle under identical experimental settings and on the same evaluation set (subsubsection 2.9.4.1). The main threat to internal validity concerns our implementation of the testing scripts to evaluate the failure prediction scores, which we tested thoroughly. Concerning the training of ADS model, we used artifacts publicly available in the replication packages of the SelfOracle (STOCCO et al., 2020) and DeepCrime (HUMBATOVA; JAHANGIROVA; TONELLA, 2021) papers. Regarding the simulation platform, we used the Udacity simulator adopted in analogous failure prediction studies (STOCCO et al., 2020; HUSSAIN; ALI; HONG, 2022).

2.9.6.2 External validity

The limited number of self-driving systems in our evaluation poses a threat in terms of generalizability of our results to other ADS. Moreover, results may not generalize, or generalize differently, when considering other simulation platforms than Udacity. For the attention maps, we considered only attention maps produced by SmoothGrad (SMILKOV et al., 2017), and the effectiveness of our tool may change when considering different XAI algorithms.

2.9.6.3 Reproducibility

All our results, the source code of ThirdEye, the simulator, and all subjects are available (Stocco, 2022).

2.10 DISCUSSION

2.10.1 XAI for Failure Prediction

Our study highlights the complexity and the variety of failure scenarios that runtime monitoring techniques should aim to handle. Attention maps are typically used qualitatively by humans to understand how a DNN processes its inputs. In this work, we used them quantitatively, under the assumption that they contain information that can potentially be used to assess the behaviour of DNNs (SAMEK; WIEGAND; MÜLLER, 2017; TJOA et al., 2022) and, by extension, of the ADS that use them.

Our approach depends on the capability of attention maps to constitute a reference model of normal driving behaviour. Well-trained DNNs better capture the relevant structures in an image, thus produce more meaningful attention maps than poorly trained

DNNs, which rather rely on global image statistics. Apart from this requirement, attention maps offer a more transparent and effective assessment of the ADS behaviour than a black-box technique because they indicate the degree of attention (or lack thereof) of the ADS in response to an input. Our results confirm that they are generally more effective than a competing black-box technique on both unknown (out-of-distribution) and uncertain (mutation testing) scenarios.

2.10.2 Discussing ThirdEye's Configurations

All configurations of ThirdEye are stable in terms of prediction power and we observed no big drop as we move to a longer duration between prediction and failure. For example, for ADS models produced by automated mutation testing, this can be explained by the fact that these self-driving cars are always characterized by a relatively high proportion of uncertainty internal to the system, which ultimately causes a failure that ThirdEye is able to detect because its predictions are made based on information that reflects the (buggy) internal state of the system.

We evaluated two windowing alternatives, *max* vs *mean*. Each has pros and cons: max is more reactive than mean, as it is enough to observe a spike in the window to trigger an alarm, which may potentially lead to a higher recall. At the same time, usage of max during threshold estimation makes γ_{95} higher, because a higher threshold must be chosen to ensure as few as 5% false positives in nominal conditions. A higher γ_{95} leads naturally to a lower recall. The combination of the two factors, higher reactivity and higher γ_{95} threshold, may either lead to better or to worse performance of max vs mean. Hence, the choice can only be made empirically and it is quite interesting that our empirical results show very clearly and neatly the superiority of max over mean.

2.10.3 Comparison with Other Approaches

Attention maps by XAI are not the only way to analyze the internal functioning of an ADS. For instance, other white-box approaches have been proposed in the literature, such as solutions based on activation traces (KIM; FELDT; YOO, 2019), cross-layer dissection (WANG et al., 2020), or uncertainty quantification measures (MICHELMORE et al., 2020). These methods have two main drawbacks that hinder their applicability as online failure predictors. First, in general, these methods are known for being computationally very expensive, thus they may not be real-time viable solutions. Second, these techniques must be integrated into the development process from the very beginning, as they require a white-box access to the model's architecture and to the training data, because the ADS must be retrained or modified to enable the computation of white-box confidence scores. Unlike these methods, we experiment with attention maps because they do offer a white-box view of the DNN internals without requiring access to the training data, nor the need to modify or retrain the ADS model.

2.11 CONCLUSION AND FUTURE WORKS

In this work, we describe and evaluate a white-box failure predictor that estimates the confidence of a DNN-based ADS in response to unpredictable execution contexts. Our tool ThirdEye performs confidence estimation by turning attention maps derived from the XAI domain into confidence scores of the driving ADS. Our approach is able to anticipate many potentially safety-critical failures by several seconds, with a low false alarm rate in anomalous conditions, and a fixed 5% expected false alarm rate in nominal conditions, outperforming a black-box predictor from the literature.

Future work includes extending the comparison to other white-box confidence estimators. At the same time, alternative confidence score synthesis methods based on the semantic of the input image will be investigated, as well as other XAI algorithms. Moreover, we also plan to extend the detection of finer-grained driving quality degradations (e.g., erratic driving behaviour) and to study self-healing mechanisms within the simulator.

3 MUTANT XAI GENERATOR- A XAI TEST CASES INPUT GENERATORS FOR NEURAL NETWORKS

3.1 INTRODUCTION

This chapter delves into a critical aspect of autonomous driving systems (ADS): the identification and understanding of Neural Networks' limitations and weaknesses. Building upon the foundational knowledge provided in Chapter 2, this chapter introduces a preliminary study of a strategy called XAI Mutant Generator, which is aimed at producing an array of driving scenarios. In this chapter, we explore a connection between the abstract form of digits and real-world driving scenarios, using a Neural Network trained on the MNIST dataset to classify digits as an analogy for road layouts.

It is important to underline that the XAI Mutant Generator is not intended to operate in isolation. Its purpose is to synergize with the ThirdEye technique described in the preceding chapter. By combining XAI Mutant Generator's scenario creation capability with ThirdEye's explainable AI-driven behavior prediction, we strive to enhance our testing capabilities and increase safety measures in autonomous driving systems. This synergy allows us to generate high-quality test inputs for the Udacity simulator, harnessing the strengths of both techniques for a more robust, thorough evaluation.

The chapter begins with an introduction that sets the context for the research, explaining the importance and need for an effective testing strategy during the Neural Networks' testing phase. Following this, the background section delves into a review of relevant literature on Neural Networks, XAI, attention maps, and their usage in ADS.

In the Methodology section, we outline the steps taken to implement the XAI Mutant Generator, detailing how attention maps were employed to guide the generation of digit variations, and the process for identifying corner cases. The comparison of digits and road layouts forms an important aspect of this explanation.

The Results section presents our findings, offering insights into how the XAI Mutant Generator exposed the limitations of the Neural Network. Through systematic analysis, we illustrate how the application of attention maps could enhance the generation of simulated road scenarios for driving simulations, thereby contributing to the safety and reliability of ADS.

The Conclusion section provides a summary of the research, reflects on the implications of the findings, and suggests potential avenues for further research. We discuss how the XAI Mutant Generator represents a novel and efficient strategy to improve the testing phase of Neural Networks in ADS, offering a robust mechanism for identifying and rectifying weaknesses.

In sum, this chapter aims to contribute to the growing body of knowledge around the efficacy of XAI techniques in ADS, specifically focusing on the implementation and

outcomes of the XAI Mutant Generator as an innovative approach to Neural Networks' testing.

3.2 MOTIVATION

In the context of Neural Networks for autonomous driving, the process of testing scenario generation assumes critical significance. This process involves designing intricate driving scenarios to discover corner cases or unusual situations that could potentially trigger erroneous behavior from the Neural Network (PEI et al., 2017b; ZHANG et al., 2018a). Identifying these corner cases allows for the retraining of the network, which can lead to improved resilience and performance, a technique that is commonly referred to as adversarial training (GOODFELLOW; SHLENS; SZEGEDY, 2014). This approach assists in improving the model's capability to cope with unexpected, real-world scenarios, consequently making the autonomous driving systems safer and more reliable (KOROSEC, 2019). Furthermore, the generation of diverse and representative scenarios is also critical to ensure the robustness of the Neural Network against sensor inaccuracies and uncertainties (SIMON et al., 2020). Therefore, investing in advanced testing scenario generation methods can substantially contribute to the development of robust and trustworthy ADS (LI et al., 2020).

3.3 WORK OBJECTIVES

3.3.1 General Objective

The MUTANT XAI GENERATOR encompasses a preliminary study that investigates the application of Explainable AI (XAI) in the creation of road test scenarios. The primary objective is to determine if XAI can effectively guide the generation of road layouts that will potentially cause failures in Neural Networks (NNs) embedded within Autonomous Driving Systems (ADS). To this end, a NN trained on the MNIST dataset - a collection of digit images - was used as the study's subject, facilitating the evaluation of the effectiveness of the corner case generation process guided by XAI. The underlying concept was to view a digit layout as a road layout, and by altering the digit layout, induce the NN, trained on the MNIST dataset, to misclassify its label.

This evaluation consisted of a comparative analysis between a random generation method and a method guided by Explainable AI. The generation process can be thought of as a mutation operation wherein the digit image undergoes modifications. Particularly, the Explainable AI-guided method focuses on mutating the portions of the digit image that are given higher attention scores by the XAI techniques.

For comparison, we examined various parameters, including the number of mutations required to lead to a misclassification and the total number of misclassifications identified over a series of controlled experiments. By conducting these evaluations, we seek to demonstrate that the use of attention maps derived from XAI can be an effective means

for automatic generation of diverse test scenarios. Such methods, if proven effective, could provide significant enhancements to the safety and reliability of autonomous vehicles by providing better guidance on potential areas of weakness during the simulation of driving scenarios.

3.3.2 Contributions

Our work makes the following contributions:

Technique. The core contribution of this work is a novel, high-efficiency methodology, leveraging Explainable AI to guide test case generation for driving simulators. This approach significantly reduces misclassification detection time and increases the number of corner cases generated, thereby enhancing the robustness and safety of Autonomous Driving Systems.

Evaluation. An in-depth exploration of the influence of Explainable AI on the generation of corner cases for Neural Networks. The established technique is not only applicable to the present study but can also be leveraged in the creation of simulated road driving scenarios. This has substantial potential to enhance the testing methodology for Autonomous Driving Systems (ADS) Neural Networks, facilitating their robustness and reliability.

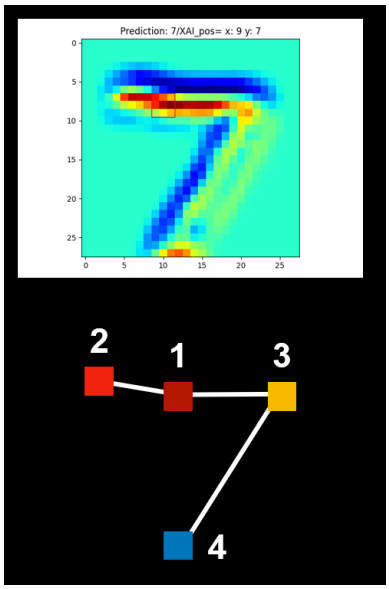
Dataset. Mutant XAI Generator culminated in the creation of a dataset comprising more than 100 uniquely mutated images that led to misclassifications in an MNIST-trained Neural Network. This curated dataset is a tangible contribution to the research community, serving as a useful tool for other researchers to evaluate and improve their own networks or to replicate and extend the present study.

3.4 BACKGROUND

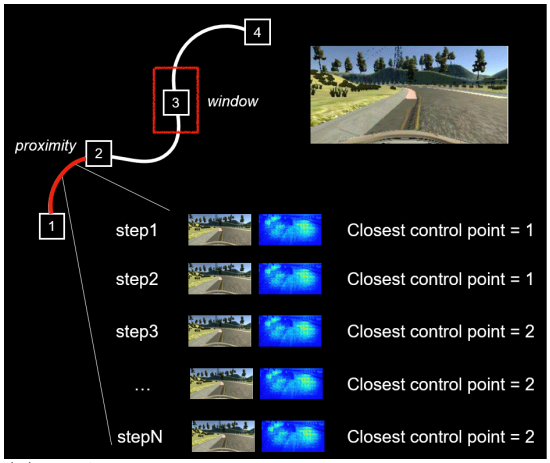
In this chapter, relevant concepts will be presented about the subjects related to test cases input generations for Neural Netowrk testing.

3.4.1 Road Representation using the MNIST Dataset

Figure 8 – Visual analogy illustrating the similar role of attention maps in selecting control points for both digits and roads.



(a) DIGIT control points representation. Derived from digit's attention maps.



(b) ROAD control points representation. Derived from car's front camera attention maps.

Source: The Author (2022).

The MNIST dataset (DENG, 2012), comprising images of handwritten digits, is a ubiquitous tool in machine learning tasks. This study employs this dataset to represent road layouts, where different digits and their variations can model a variety of road configurations. The process of mutating these digit-images allows us to generate an extensive array of road scenarios for testing ADS in virtual simulators.

In Figure 8a, a visual representation of the control point selection process for a digit, derived from attention maps generated by the Explainable AI Gradcam++ technique (CHATTOPADHAY et al., 2018), is presented. Correspondingly, Figure 8b on the right-hand side showcases the selection of control points grounded in the road's layout, incorporating attention maps extracted from the front-facing camera images of the vehicle.

In their work, Riccio and Tonella (2021) explore the representation of roads through the mutation of digit-images from the MNIST dataset. Their technique served as an early attempt to build a bridge between the representation of road scenarios and digit datasets commonly used in machine learning. This pioneering concept of using digit-image mutation to create diverse and unrealistic road configurations set the foundation

for the methodology employed in this preliminary study. By expanding on their work, the objective of this preliminary study is to develop a more efficient approach, aiming to minimize the quantity of iterations necessary for the identification of misclassified instances, allowing a more thorough and diverse testing of ADS in virtual simulators (Riccio and Tonella, 2021).

3.4.2 Guiding Test Samples Generation with Explainable AI

The key to an efficient mutation process lies in generating meaningful, diverse, and challenging road scenarios. Explainable AI, with its ability to interpret and explain AI model decisions, can direct the mutation of digit-images in a manner that maximizes these outcomes. By focusing on key areas in the digit-images identified through XAI, the mutation process can effectively generate a wide range of common and corner case road scenarios.

3.4.3 Unexplored Synergy of MNIST and XAI in ADS Testing

This research explores the under-investigated intersection of MNIST-based road representation and XAI-guided mutation. This innovative preliminary study holds significant potential for enhancing the ADS testing process in virtual simulators.

By leveraging Explainable AI techniques, we aim to guide the mutation process strategically, thus speeding up the process of finding corner cases in Neural Networks. This optimization is critical in the context of autonomous driving simulation, where each iteration can consume substantial time, ranging from several minutes to hours. By reducing the number of iterations needed to identify corner cases, this approach could lead to significant time savings in ADS testing.

Moreover, by generating a diverse array of road scenarios, our method is designed to expose potential weaknesses in ADS. This capability not only helps improve their overall performance but also enhances their safety measures. Ultimately, the combined use of MNIST-based road representation and XAI-guided mutation could significantly boost the effectiveness and efficiency of ADS testing.

3.5 RESEARCH QUESTION

With this background in mind, our study seeks to answer the following central research question:

Can Explainable AI, when compared to traditional mutation methods, serve as a more efficient technique for generating corner cases of Neural Networks?

3.6 RELATED WORK

The domain of testing and ivestigating Neural Networks, especially in the context of autonomous driving systems (ADS), has garnered considerable attention over the years. Several works have investigated various dimensions of this domain, offering critical insights and methodologies that have shaped our understanding of Neural Networks' vulnerabilities and the measures required to improve their robustness.

One of the primary inspirations behind the XAI Mutant Generator has been the paper "Model-based exploration of the frontier of behaviours for deep learning system testing" by Riccio and Tonella (2020) (RICCIO; TONELLA, 2020b). This pioneering work is one of the first to use digits from the MNIST dataset as an analogy to road layouts, which formed the basis for our methodology.

One remarkable study that significantly contributes to the field is the work of Pei et al. (2017) titled "DeepXplore: Automated Whitebox Testing of Deep Learning Systems" (PEI et al., 2017b). They introduced a rigorous approach for testing deep learning systems through differential testing, inspiring further research in the field. The philosophy of systematic testing is echoed in our work on the XAI Mutant Generator, albeit with a twist, by leveraging explainable AI to comprehend Neural Networks' limitations and weaknesses.

Zhang et al.'s (2018) work "DeepRoad: GAN-Based Metamorphic Autonomous Driving System Testing" (ZHANG et al., 2018a) also used a novel approach to generate realistic and challenging driving scenarios for testing using Generative Adversarial Networks (GANs). Their approach bears similarity with our work where we generate various driving scenarios. However, we harness the power of attention maps and digit variations from the MNIST dataset to simulate different scenarios.

Finally, the relevance and applicability of ADS in the real world have been demonstrated by Waymo's robotaxi pilot, which served more than 6,200 riders in its first month in California (KOROSEC, 2019), and Li et al.'s (2020) discussion on the utilization of deep reinforcement learning for multi-objective optimization (LI et al., 2020). These underscore the practical implications of improving the robustness of Neural Networks in ADS.

In conclusion, the XAI Mutant Generator is situated within a rich and growing body of research seeking to enhance the safety, reliability, and efficacy of Neural Networks in ADS. By using explainable AI, our work offers a unique approach to test and comprehend these networks, contributing to this evolving landscape.

3.7 METHODOLOGY/APPROACH

The mutation of a digit, whether through a random or natural process, fundamentally involves vectorizing the digit's image, extracting the control points that define its layout, randomly selecting one of these points, and applying a random displacement to its geometric coordinates (x, y). When the mutation process is guided by attention methods, the procedure remains largely similar, but with a critical distinction: the selection of the control point for mutation is not random. Instead, it relies on the application of attention map techniques to identify the control point nearest to the area of the image with the highest attention score.

Consequently, we can break down the Attention Maps guided digit mutation process into four main steps:

(1) Extraction of Attention Maps: The attention map is derived from the original digit image using the SmoothGrad method, as referenced in subsection 2.5.4.

(2) Digit Vectorization: The original digit image undergoes vectorization to extract the control points - a step analogous to the one employed in random mutation. Equipped with both the attention map, which indicates the regions of the digit image with the highest attention scores, and the positions of all control points, we proceed to the final step.

(3) Control Point Selection: Appropriate algorithms are employed to choose the optimal point for mutation. This selection is based on the location with the highest attention score in the image. Once this point has been identified, a displacement is applied to its coordinates. The image is subsequently reconverted to its rasterized format. This rasterized image then serves as a new input for the neural network, enabling the measurement of prediction feedback. This study introduces two novel algorithms designed to optimize control point selection based on attention maps: Scan-Square Distance-Based Point Selection (SS-DBPS) and Control-Point-Scan Weight-Based Selection (CPS-WBS). SS-DBPS selects the control point based on its proximity to the area with the highest attention score within the attention map image. Conversely, CPS-WBS operates by assigning a score to each control point according to the attention score within its vicinity. This score, termed "weight", is subsequently utilized in a weighted selection process. Detailed explanations and operational mechanisms of both these algorithms will be presented in the ensuing sections.

(4) Control Point Displacement: The final step in our methodology involves displacing the selected control point. We use two strategies: a fixed displacement method and an adaptive displacement method. The latter adjusts the displacement value based on the feedback from the neural network to increase the efficiency of finding misclassified cases. Our study reveals that this adaptive displacement method, applied in both random and attention-guided mutations, significantly enhances the efficiency in identifying misclassification instances in neural networks. The two methods will be explained in the

subsection 3.7.4.

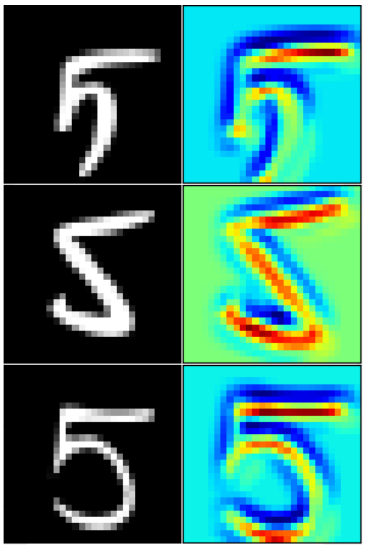
It is important to note that this entire process constitutes a single iteration. The research presented in this study demonstrates that utilizing attention maps to guide the digit mutation process significantly reduces the total number of iterations necessary to find a mutated digit that results in a misclassification by the neural network when compared to a random guided mutation method.

3.7.1 Extraction of Attention Maps

The initial phase of our process involves extracting the attention maps. This involves utilizing the Gradient-weighted Class Activation Mapping (GradCAM++) technique. Originating as an improvement over the earlier GradCAM method, GradCAM++ provides an advanced and effective means of visualizing attention in Convolutional Neural Networks. It works by attributing an 'attention score' to each pixel in the input image, a score which quantifies the pixel's contribution to the neural network's final prediction.

In practical terms, this translates to the generation of a new image—commonly referred to as a 'heatmap' or 'attention map'—where each pixel's intensity corresponds to its attention score. In this way, areas of the image that significantly influence the final prediction of the Neural Network are made visually identifiable, facilitating targeted and informed mutations in subsequent stages of the process.

Figure 9 – Examples of different attention score distributions of the digit "5" based on its shape.



Source: The Author (2022).

As depicted in Figure 9, attention score distribution varies significantly depending on the specific configuration of the digit 5. Some instances demonstrate a higher concentration of attention toward the upper segment of the digit. Conversely, in other cases, attention is distributed more uniformly across the entire digit. Additionally, certain configurations

yield a more compartmentalized attention distribution, with attention being allocated to distinct regions within the digit. Such variations underline the impact of the digit's shape on the distribution of attention scores, reflecting the complexity and variability inherent in the interpretative processes of neural networks.

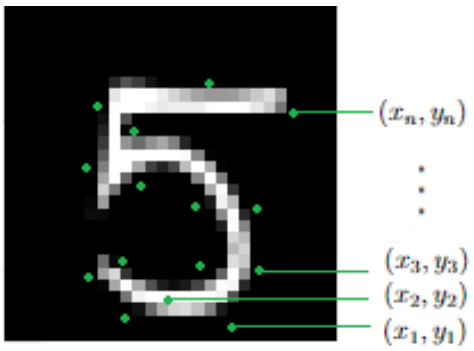
3.7.2 Digit Vectorization

The initial form of the digit images from the MNIST dataset is bitmap; thus, the second stage of the mutation process necessitates the conversion of these bitmaps into vector representations. This conversion, otherwise referred to as vectorization, facilitates the extraction of control points that delineate the digit's shape. Through these control points, we can either regenerate the original digit bitmap or create a variation thereof through the alteration of one or more control point coordinates - a process termed 'Mutation'. The result of this step is a compilation of control points that are capable of reconstructing the original digit bitmap.

$$\mathbf{CP} = \langle (x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n) \rangle$$

The Figure 10 shows an illustrative example of the control points for the number 5.

Figure 10 – Ilustrative example of control points of the digit 5.



Source: The Author (2022).

The method by which we select the control point for displacement, and thus create the mutated digit, will be elaborated upon in the ensuing section on Control Point Selection methods.

3.7.3 Control Point Selection and Mutation

The process of mutation is crucial to the generation of corner cases, and the choice of the control point to be displaced for this mutation is particularly significant. However, instead of randomly choosing control points, this study uses a more informed method for control point selection, based on the insights gained from the attention maps. This section

will dive into the detailed methodology used for Control Point Selection and Mutation, highlighting how the attention maps can be utilized for more effective mutation in our context.

3.7.3.1 Scan-Square Distance-Based Point Selection (SS-DBPS)

This algorithm operates on the principle of analyzing square areas of the attention map image and identifying the one that has the highest score of attention, in other words, the area that stimulates the highest response on the Neural Network.

The algorithm accepts three parameters: an explanation image (xai_image), and the x and y dimensions of a square (x_sqr_size, y_sqr_size) that scans across the explanation image. The xai_image is the output of the Gradcam++ explainability method. This grayscale image carries pixel values ranging between 0 and 255, each of which corresponds to the neural network's 'attention' score. The dimensions of the square determine the region within the xai_image where the attention score is computed.

The process begins by defining the x and y dimensions of the xai_image. Subsequently, the algorithm initiates a nested loop, shifting the defined square region across the entire xai_image. For every position of the square, the sum of pixel values (sum_xai) within that square region is computed. This sum acts as the 'attention score' for that particular region. The algorithm keeps track of the region with the highest sum (greater_value_sum_xai), updating it whenever a region with a larger sum is encountered. The final output of this algorithm is the x and y position of the square (x_final_pos, y_final_pos) that correspond to the highest attention score.

After identifying the region with the highest attention score, a control point from the digit's vectorized form is selected based on proximity. More specifically, the control point closest to the identified region is chosen.

The Figure 11 shows the Attention-Square Point Selection (SS-DBPS) Method. The image represents a digit from the MNIST dataset, overlaid with the xai_image output of the Gradcam++ explainability method. Pixel intensities in the xai_image signify the attention scores as assigned by the neural network. The square box highlights a particular region of the image under evaluation. The attention score of this region, the sum of pixel intensities within the square, is calculated and compared with other regions as the square slides across the entire image.

3.7.3.2 Control-Point-Scan Weight-Based Selection (CPS-WBS)

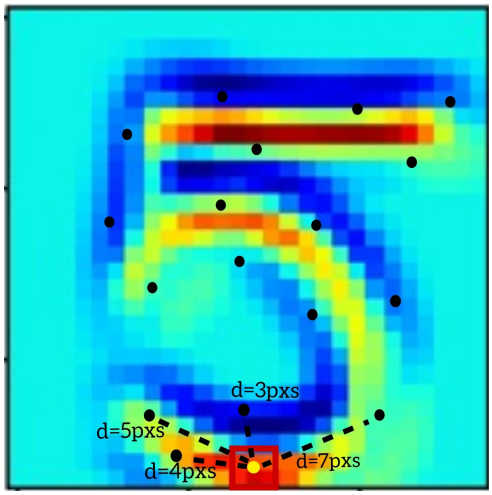
The algorithm 2 operates by identifying regions in the attention map and calculating their probability of selection, which is based on the region's attention score. This algorithm, known as the *Control-Point-Scan Weight-Based Selection (CPS-WBS) method*, processes each control point, considers the neighboring region in the attention map, computes the attention score for the region, and then uses this score to compute the selection probability.

Algorithm 1: Scan-Square Distance-Based Point Selection (SS-DBPS)

Data: xai_image, x_sqr_size, y_sqr_size
Result: greater_value_sum_xai, x_final_pos, y_final_pos

```
x_dim ← shape(xai_image)[0];
y_dim ← shape(xai_image)[1];
greater_value_sum_xai ← 0;
x_final_pos ← 0;
y_final_pos ← 0;
for y_sqr_pos in range(0, y_dim − y_sqr_size) do
    for x_sqr_pos in range(0, x_dim − x_sqr_size) do
        sum_xai ← 0;
        for y_in_sqr in range(0, y_sqr_size) do
            y_pixel_pos ← y_sqr_pos + y_in_sqr;
            for x_in_sqr in range(0, x_sqr_size) do
                x_pixel_pos ← x_sqr_pos + x_in_sqr;
                sum_xai ← sum_xai + xai_image[y_pixel_pos][x_pixel_pos];
        if sum_xai > greater_value_sum_xai then
            greater_value_sum_xai ← sum_xai;
            x_final_pos ← x_sqr_pos;
            y_final_pos ← y_sqr_pos;
```

Figure 11 – The Attention-Square Point Selection (SS-DBPS) method applied on a digit from the MNIST dataset with the Gradcam++ output.



Source: The Author (2022).

This algorithm accepts three parameters: the explanation image (xai_image), a list of control points (control_points_list), and a square size (square_size). The square size is related to the size of the neighborhood considered around each control point for calculating the attention score. Depending on the square size, different border variables are set, which determine the range for scanning the neighborhood of a control point.

The algorithm begins by calculating the dimensions of the xai_image. After this, it starts a loop over each control point in the control_points_list. For each control point, it computes the sum of pixel values within the corresponding square region in the xai_image. This sum represents the attention score of the control point's neighborhood.

After calculating the attention scores for all control points, it sums these scores to get a total sum (sum_xai_list). Then, the algorithm computes a list of weights by exponentiating the ratio of the individual attention score to the total sum and multiplying it by 100. These weights represent the 'importance' of each control point. The weights are then normalized by dividing them by their total sum to generate the list of probabilities (list_of_probabilities).

The output of this algorithm is the list of weights and the list of probabilities, both of which represent the likelihood of selection of the control points based on their neighborhood's attention scores. In the context of the MNIST digits, these probabilities can guide the selection of the control points for mutation, aiming to generate corner cases for the neural network.

The Figure 13 illustrates the Control-Point Weight-Based Selection (CP-WBS) method in action. Control points, marked on the attention map of an MNIST digit, are each surrounded by a square region. Each region's attention score, calculated from the pixel intensities within, is displayed next to it.

3.7.4 Control Point Displacement

Following the identification of the optimal control point for mutation, a displacement is applied to its position. The displacement method utilized within this study takes two principal forms. The first approach involves a random selection of direction (either x or y) and the application of a fixed displacement value. The second method, termed 'Adaptive Displacement,' also involves a random selection of direction for displacement but differs in the calculation of displacement magnitude. In the Adaptive Displacement method, the displacement value is adjusted according to the Neural Network's feedback. Specifically, for each iterative mutation, the network's response to the mutated candidate is evaluated. If the response does not exhibit significant change, the displacement factor is incrementally increased until a substantial variation in the Neural Network response is observed. At this point, the displacement factor is reset to its initial value, and the process continues. The Adaptive Displacement method was employed in both mutation guidance methods—random and attention-based—in this study. The results indicate that the Adaptive Displacement method provides greater efficiency in identifying misclassification cases compared to methods without adaptive displacement.

The 'Apply Displacement to Mutant' algorithm is a key component of the point mutation process, specifically designed to apply a random displacement to a selected control point. The algorithm begins by randomly selecting one of the two coordinates of the con-

Algorithm 2: Control-Point-Scan Weight-Based Selection (CPS-WBS)

```
Data: xai_image, control_points_list, square_size
Result: list_of_weights, list_of_probabilities
x_dim ← shape(xai_image)[0];
y_dim ← shape(xai_image)[1];
if square_size equals 3 then
    y_border_up ← −1;
    y_border_bottom ← 1;
    x_border_right ← 1;
    x_border_left ← −1;
else
    y_border_up ← −2;
    y_border_bottom ← 2;
    x_border_right ← 2;
    x_border_left ← −2;
xai_list ← [];
for pos in control_points_list do
    x_sqr_pos ← int(pos[0]);
    y_sqr_pos ← int(pos[1]);
    sum_xai ← 0;
    for y_in_sqr in range(y_border_up, y_border_bottom + 1) do
        y_pixel_pos ← y_sqr_pos + y_in_sqr;
        if y_pixel_pos ≥ 0 and y_pixel_pos ≤ y_dim − 1 then
            for x_in_sqr in range(x_border_left, x_border_right + 1) do
                x_pixel_pos ← x_sqr_pos + x_in_sqr;
                if x_pixel_pos ≥ 0 and x_pixel_pos ≤ x_dim − 1 then
                    sum_xai ←
                      sum_xai + xai_image[y_pixel_pos][x_pixel_pos];
    Append sum_xai to xai_list;
sum_xai_list ← sum(xai_list);
list_of_weights ← [];
list_of_probabilities ← [];
for sum_value, pos in zip(xai_list, control_points_list) do
    Append exp((sum_value/sum_xai_list) ∗ 100) to list_of_weights;
sum_weights_list ← sum(list_of_weights);
for weight in list_of_weights do
    Append weight/sum_weights_list to list_of_probabilities;
return list_of_weights, list_of_probabilities;
```

trol point using the random choice function. This function randomly picks an item from a list, in this case, either the x or y coordinate. Following this selection, it operates by assigning a 'displacement' value that is calculated as a random proportion (ranging from 0.01 to 0.6) of a predefined extent.

The algorithm then makes a decision to either add or subtract this displacement

Algorithm 3: Mutant Displacement Application

Data: Control point coordinates (control_point_coordinates), Extent of displacement (extent)
Result: Coordinate of the displaced mutant (result)

```
Define the lower bound of mutation (MUT_LOWER_BOUND = 0.01);
Define the upper bound of mutation (MUT_UPPER_BOUND = 0.6);
Define the mutation offset probability (MUT_OF_PROB = 0.5);
Choose a random coordinate from the control point (coordinate_chosen ← random.choice(control_point_coordinates));
Calculate the displacement (displacement ← uniform(MUT_LOWER_BOUND, MUT_UPPER_BOUND) * extent);
if random.uniform(0, 1) ≥ MUT_OF_PROB then
    Apply positive displacement (result ← coordinate_chosen + displacement);
else
    Apply negative displacement (result ← coordinate_chosen - displacement);
```

from the original control point's value. This decision is determined by generating another random value and comparing it with a predefined probability threshold (0.5 in this case). If the generated random value is greater or equal to this threshold, the displacement is added; otherwise, it is subtracted.

The algorithm operates under the assumption that the control point's value and the displacement extent are floating-point numbers. It also leverages Python's uniform function to generate random values with a uniform probability distribution within a specified range, ensuring a fair, unbiased selection process.

The resultant value, computed by applying the displacement to the original control point's value, is converted to its string representation for final output. This approach ensures a certain level of randomness in the control point mutation process, which in turn contributes to the overall robustness of the method.

3.7.4.1 Adaptive Extent

As briefly introduced in the preceding subsection, this preliminary study introduces an innovative approach to the displacement algorithm, incorporating the concept of Adaptive Displacement. As detailed in Algorithm algorithm 3, the calculation of the displacement value is dependent on a random selection of a number between the predefined MUT_LOWER_BOUND and MUT_UPPER_BOUND constants. This number is then multiplied by an extent factor, leading to the fact that a higher extent factor results in a more significant displacement applied to the digit's coordinate.

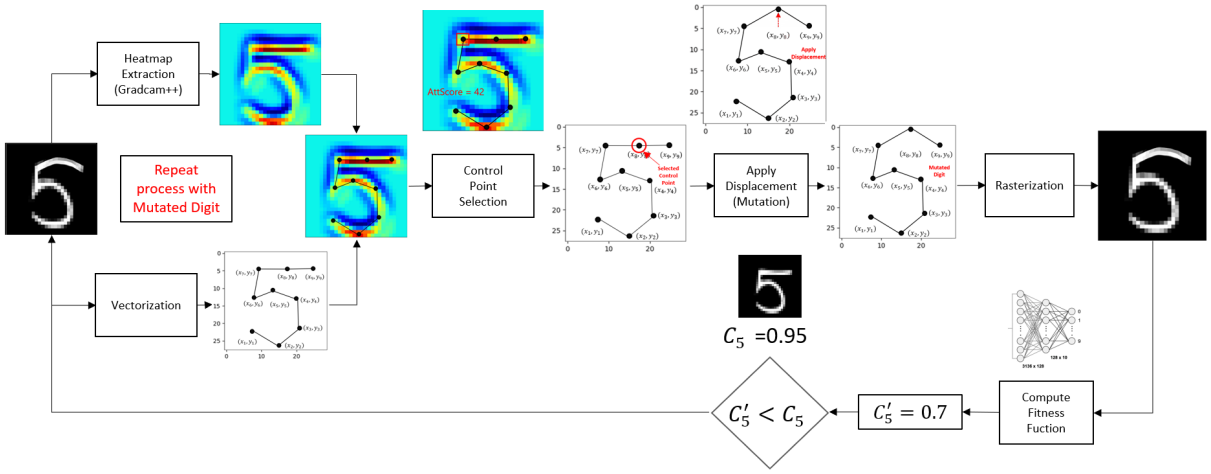
The strategy used to escalate the extent is based on the response of the neural network to a sequence of generated mutations. If the neural network's response shows no significant variation in the prediction (e.g., the generated digits are not challenging enough to be failure inducing) after a set number of iterations (n), an increment factor is applied to

increase the extent. This process continues until the neural network's response reveals a positive variation exceeding a predefined threshold. At this point, the extent value resets to its initial value, and the cycle repeats.

It is critical to note that the initial extent value requires careful calibration. Coupled with the extent resetting strategy, it helps mitigate the risk of generating highly deformed digits due to excessive displacement. This balance ensures the production of meaningful and valid mutated digits for the testing process.

3.7.4.2 Main Mutation Loop

Figure 12 – Workflow representing all the steps necessary to complete one iteration of the mutation process.



Source: The Author (2022).

In the primary mutation algorithm, the objective is to iteratively mutate the original digit image until it can be misclassified by the neural network model, or until the maximum number of iterations has been reached. The algorithm works as follows:

The process starts with the initialization of parameters such as the extent (used to control the displacement magnitude for mutation), the selection_method that determines the approach for control point selection (distance-based or weight-based), and the number_of_iterations_to_allow_adaptive, which dictates when adaptive extent modifications can be enabled.

The main mutation workflow's process can be observed in Figure 12. It encompasses all the steps executed during each iteration until reaching the maximum iteration limit.

1. The class of the original digit is predicted using the neural network model.

2. The attention map of the original digit image is extracted using the Gradcam++ explainability method.

3. Control points from the original digit image are identified.

4. A control point for mutation is chosen based on the selected method (distance or weight-based) and the attention map.

5. A displacement, governed by the extent, is applied to the selected control point, resulting in a set of mutated_control_points.

6. A new digit image (mutated_digit_image) is created by rasterizing the mutated control points.

7. The class of the mutated digit image is predicted using the neural network model.

If the predicted class of the mutated digit image is different from the original, it signifies that a misclassification has been achieved, leading to the termination of the mutation process.

If no misclassification is observed, the fitness function value for the original and mutated digit images are calculated based on the neural network predictions for these images. If the fitness function of the mutated image shows an improvement over the original, indicating that the mutation is leading the neural network toward misclassification, the original image is replaced with the mutated image.

However, if the mutated image does not show an improvement in the fitness function value, the extent is adaptively updated (if permitted by the iteration count), ensuring that the displacement magnitude for subsequent mutations can be adjusted in response to previous unsuccessful mutation attempts.

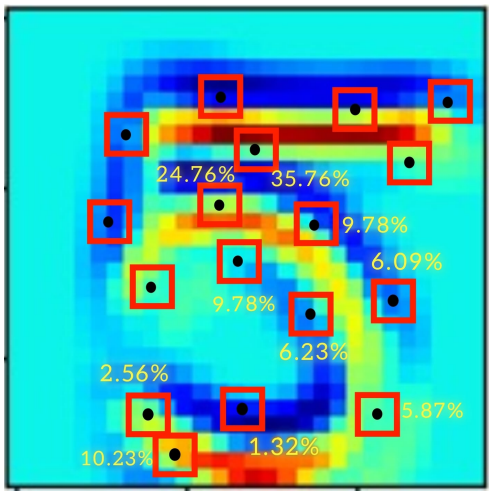
Through this iterative mutation process, the algorithm aims to progressively guide the evolution of the digit image toward a state that the neural network model is likely to misclassify, thereby identifying potential vulnerabilities in the model's prediction capability.

In Figure 13, each control point is encompassed by a red square box. The attention score of each region is determined by summing the attention scores of all the individual pixels enclosed within the respective red square box. Subsequently, this aggregated score is transformed into a probability value that directly corresponds to the magnitude of the attention score. This transformation is accomplished via a straightforward weighted calculation.

$$\text{Point Probability Value} = \frac{\text{PAS} \times 100}{\text{TAS}} \tag{3.1}$$

Where PAS is the sum of the attention score inside the square area for the specific point (PAS - Point Attention Sum) and the TAS is the sum of the attention score inside the square area for all the points (TAS - Total Attention Sum).

Figure 13 – Control points on an MNIST digit's attention map, each surrounded by a square region, with the region's attention score displayed for guiding mutation operations.



Source: The Author (2022).

Algorithm 4: Main Mutation Algorithm

Input: MNIST_dataset, model, Max_number_of_iterations

Output: Missclassification_found, original_digit_image

```
extent ← 0.2;
original_digit_image ← get_digit_from_dataset(MNIST_dataset);
original_NN_prediction ← model.predict(original_digit_image);
selection_method ← "distances";
number_of_iterations_to_allow_adaptive ← 10;
iteration ← 0;
while iteration ≤ Max_number_of_iterations do
    iteration ← iteration + 1;
    original_NN_prediction_class ← model.predict_classes(original_digit_image);
    attention_map_image ← extract_attention_gradcam++(original_digit_image);
    original_control_points_list ← extract_control_points(original_digit_image);
    control_point_for_mutation ← select_control_point(attention_map_image,
     original_control_points_list , selection_method);
    mutated_control_points_list ←
     apply_displacement(control_point_for_mutation, extent);
    mutated_digit_image ← rasterize_digit(original_control_points_list);
    mutated_NN_prediction_class ← model.predict_classes(mutated_digit_image);
    if mutated_NN_prediction_class != original_NN_prediction_class then
        Missclassification_found ← True;
        break;
    end
    original_NN_prediction ← model.predict(original_digit_image);
    original_fitness_function_value ←
     evaluate_fitness_function(original_NN_prediction);
    mutated_NN_prediction ← model.predict(mutated_digit_image);
    mutated_fitness_function_value ←
     evaluate_fitness_function(mutated_NN_prediction);
    if original_fitness_function_value - mutated_fitness_function_value >
     threshold_fitness_function then
        original_digit_image ← mutated_digit_image;
    end
    else
        extent ← adaptive_check_and_update(iteration,
         number_of_iterations_to_allow_adaptive, extent);
    end
end
```

3.8 RESULTS

3.8.1 Research Questions

We consider the following research questions:
RQ_1 (efficiency): How efficient is the MUTANT XAI GENERATOR attention-based mutation operator in exposing failures? What is the best attention guide method for that purpose?
RQ_2 (effectiveness): How many misclassifications MUTANT XAI GENERATOR is capable to generate considering the number of mutations iterations?
RQ_3 (comparison): How does MUTANT XAI GENERATOR compare with the random mutation methods.

The innovative strategies proposed in this study – guided mutation using attention maps, Scan-Square Distance-Based Point Selection (SS-DBPS) and Control-Point-Scan Weight-Based Selection (CPS-WBS) algorithms for control point selection, and Adaptive Displacement method for displacement calculation – have been thoroughly tested and validated. Their performance in reducing the number of iterations necessary to find misclassifications in the Neural Network, as compared to conventional random mutation and displacement methods, underscores their effectiveness and practical value in advancing the field of Autonomous Driving Systems (ADS) testing.

In the course of our experimentation, several patterns and insights have surfaced that elucidate the behavior and response of neural networks under the impact of the proposed methodologies. These findings not only validate the premise of the study but also open up further avenues for exploration and refinement of the techniques.

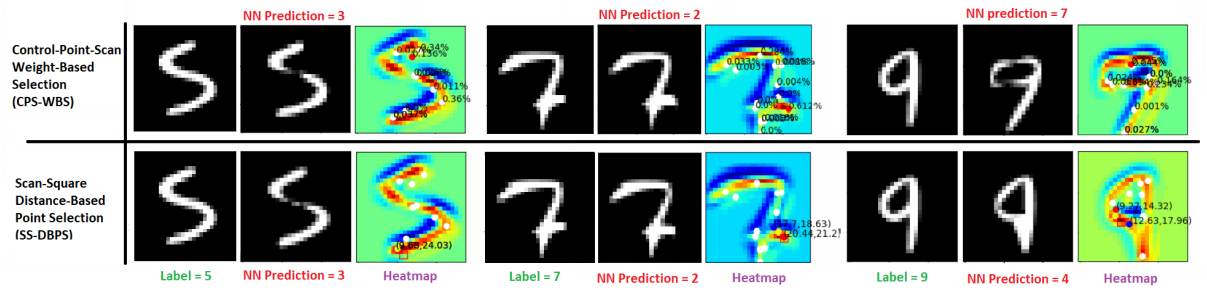
The investigation's findings have been organized into two primary categories: qualitative and quantitative results. The qualitative analysis focuses on the visual quality and recognizable nature of the mutated digits that cause misclassification in the neural network, whereas the quantitative analysis emphasizes the computational efficiency of the proposed mutation strategies and their capacity to generate misclassifications.

3.8.2 Qualitative Analysis

An important aspect of our research was the qualitative examination of the mutated digits. We observed the nature of these mutations and how they affect the Neural Network's misclassification. Emphasis was placed on whether the mutations were still recognizable as digits, despite the changes applied. A series of digit mutations that resulted in misclassifications are provided for examination, serving as effective corner cases for the study.

The Figure 14 presents a series of mutation digit samples generated using our proposed strategies. For each sample, the first column exhibits the original digit image and its corresponding label. The second column illustrates the mutated digit generated by each method, followed by the respective prediction made by the Neural Network, which notably

Figure 14 – Comparison of original, mutated digits and their respective Neural Network prediction heatmaps.



Source: The Author (2022).

differs from the original labels. The third column displays the heatmap of the mutated digit, highlighting areas that significantly influenced the Neural Network's prediction.

Upon comparing the mutated digits produced by the Control-Point-Scan Weight-Based Selection (CPS-WBS) and Scan-Square Distance-Based Point Selection (SS-DBPS) methods, there is not a substantial difference in the overall layout of the digits. Any minor discrepancies arise from the repetitive mutations concentrated in specific areas when using the distance-based method. In contrast, the weight-based selection method tends to distribute the control points selected for mutation more evenly across the digit, leading to a more uniform mutation distribution.

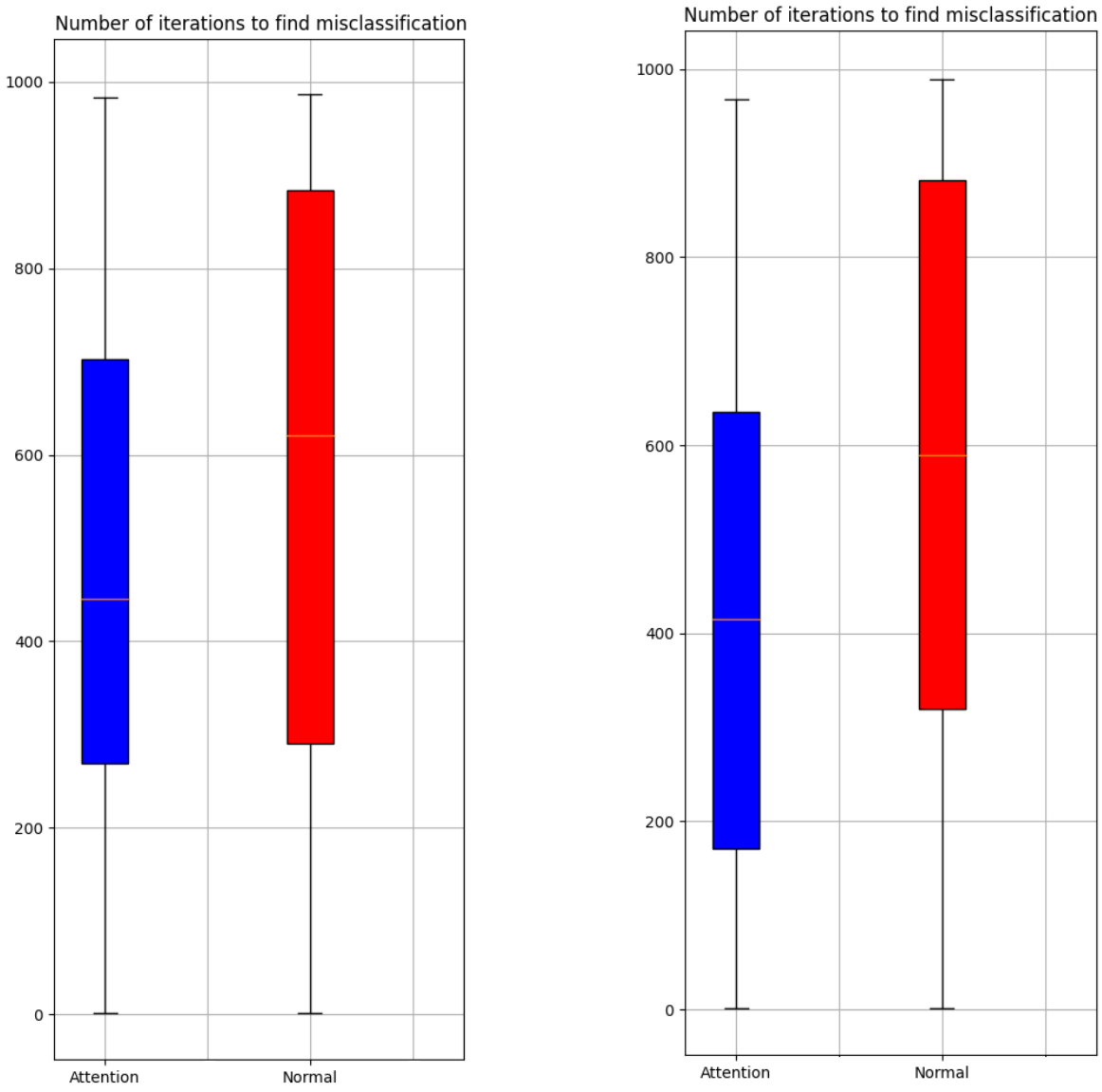
3.8.3 Quantitative Analysis

In our quantitative analysis, we focused on two primary metrics: the number of iterations required to discover a misclassification and the number of misclassifications generated by each method. This subsection will delineate the efficiency gains obtained from incorporating an attention strategy as compared to a purely random method. Furthermore, we will present the significant impact of the adaptive extent on both strategies, namely the attention-guided and random-guided methods. The results elucidate the considerable advantages provided by the adaptive extent, amplifying the performance of both strategies.

3.8.3.1 Number of iterations to find missclassifciations

The efficacy of the mutation process was evaluated based on the average number of iterations needed to generate a misclassified digit. Boxplots charts were utilized to represent these results, allowing for a comprehensive comparative analysis between the performance of attention-guided methods and the random mutation strategy. Furthermore, the influence of the adaptive extent on this process was also examined, showcasing its critical role in the efficiency of the mutation process.

Figure 15 – Candlestick chart comparing average iterations for misclassification across different strategies.



(a) Attention Weight Based (CPS-WBS)

(b) Attention Distance Based (SS-DBPS)

Source: The Author (2022).

Figure 15 illustrates the outcomes of an experiment involving 250 digits from the MNIST dataset, manipulated using the two attention methods and the conventional random one. For clarity, the results are presented through two candlestick charts, each comparing an attention method with the random approach. In both experiment iterations, the same 250 digits were used, with the reproducibility of the random method being disabled in the second round.

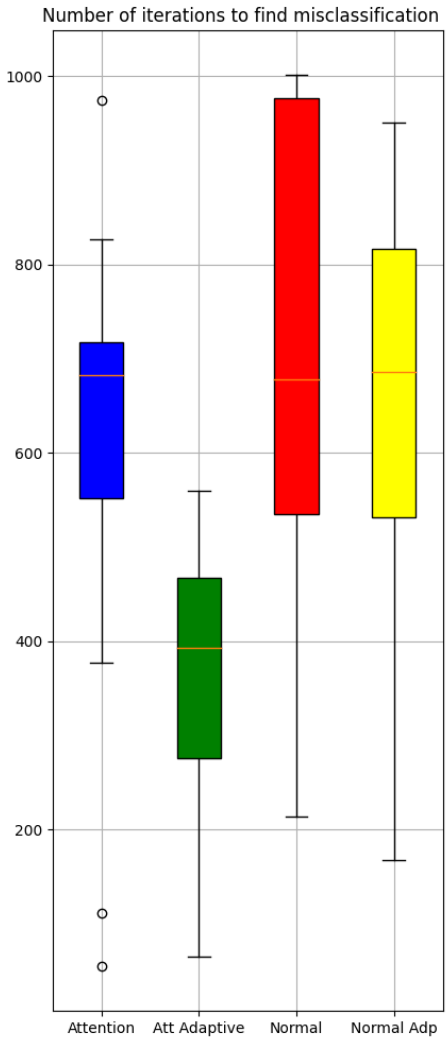
In Figure 15a, we observe the average number of iterations required to discover a misclassification using the probability weight-based strategy (CPS-WBS). The attention method averaged around 425 iterations to generate a misclassification, while the con-

ventional random approach required approximately 600 iterations, indicating that the attention-based method was more efficient by around 175 iterations.

Conversely, Figure 15b showcases the performance of the distance-based strategy (SS-DBPS) in comparison with the conventional random method. Here, the attention distance method averaged around 410 iterations to generate a misclassification, implying a gain of approximately 190 iterations over the conventional method, and a modest improvement of about 15 iterations over the probability-based method. The repetition of these patterns in various experiment runs highlights the consistent superiority of the distance-based method over both the probability-based and random strategies.

The candlestick chart in Figure 16 presents the results of another experiment iteration, this time incorporating the adaptive extent strategy into both the attention-guided and conventional random control point selection methods. This experiment also employed 250 distinct MNIST dataset digits from the previous iterations.

Figure 16 – Boxplots chart comparing average iterations for misclassification across different strategies.



Source: The Author (2022).

The chart indicates that the implementation of the adaptive technique profoundly affected the performance of the Attention strategy, with a less pronounced impact on the conventional random strategies. The conventional method maintained a consistent average, but the adaptive technique resulted in a reduced upper limit, implying that misclassifications were achieved with fewer iterations than when the method was executed without the adaptive feature.

Based on the preliminary analysis presented in this section, it can be stated that the use of Explainable AI techniques in mutation processes demonstrates promising results. Both the distance-based (SS-DBPS) and the probability-based (CPS-WBS) attention-guided methods showed a significant reduction in the average number of iterations necessary to generate a misclassified digit when compared to traditional random mutation methods. This improvement in efficiency, particularly in the case of the distance-based method, provides initial positive evidence towards answering our research question: "Can Explainable AI, when compared to traditional mutation methods, serve as a more efficient technique for generating corner cases of Neural Networks?"

However, to fully answer this question, we must not only consider the speed of generating misclassified digits but also evaluate the quantity of misclassifications found by each method. Therefore, in the upcoming section "Number of Misclassifications Found," we will further investigate these strategies' effectiveness by examining the volume of misclassifications they each were able to generate, adding another dimension to our analysis.

3.8.3.2 Number of Misclassifications Identified

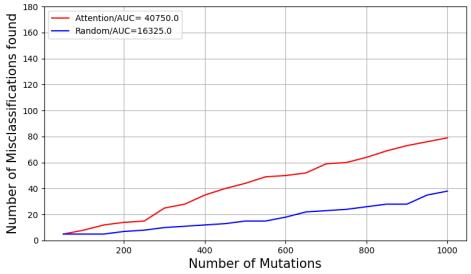
Given the need to quantify the misclassifications generated by each method, coupled with the average number of iterations required to locate a misclassification as discussed in the preliminary analysis, we identified another crucial parameter for our investigation: the maximum number of iterations set for the experiment.

Reflecting on this, a further experiment was designed with a maximum number of iterations significantly greater than the upper limit represented by the boxplots in our preceding analysis. This configuration was intended to maximize the possible number of misclassifications discovered. To maintain a consistent approach, another set of 250 digits were extracted from the MNIST dataset for this new experiment run.

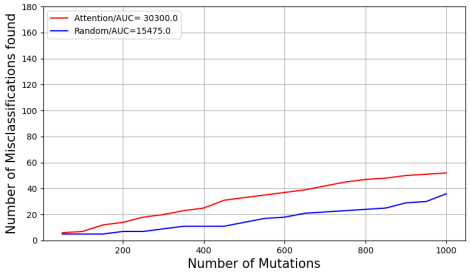
After conducting the aforementioned experiment and meticulously processing the resulting metadata, we were able to simulate a variety of scenarios by assigning different values to the maximum number of iterations. This process allowed us to quantify the number of misclassifications identified within each scenario, and subsequently, to construct a chart demonstrating the behavior of different strategies as a function of the maximum number of iterations. The results are as follows:

In Figure 17, we provide a comparative analysis of the performance of attention strategies versus the standard random method. The primary criterion employed to evaluate the

Figure 17 – Performance comparison of strategies based on the maximum number of iterations and resultant misclassifications.



(a) Comparative: Attention Probability-Based strategy (CPS-WBS) vs normal method.



(b) Comparative: Attention Probability-Based strategy (SS-DBPS) vs normal method.

Source: The Author (2022).

strategies is the Area Under the Curve (AUC). Figure 17a (on the left) illustrates the performance of the Attention Probability-Based strategy against the standard random method. Notably, the results suggest that the attention method outperforms the random method across the chosen range of maximum number of iterations for the experiment (max = 1000). Specifically, the AUC for the attention technique indicates a value approximately 2.5 times greater than that of the standard method.

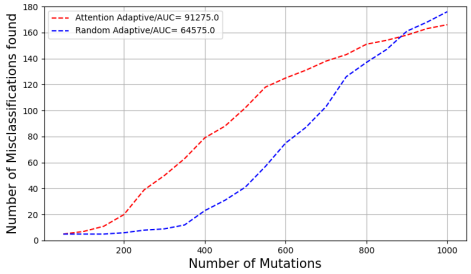
On the right in Figure 17b, we present the results for the Attention Distance-Based strategy. Similar to its probability-based counterpart, this strategy outperforms the standard random method across the entire range of maximum iterations. It is only when the maximum number of iterations approaches 1000 that the random method begins to generate a similar amount of misclassifications as the attention method. However, despite this narrowing gap, the overall AUC of the attention technique still surpasses that of the random method.

An additional series of experiments were conducted to investigate the impact of incorporating the adaptive extent strategy into the methods under examination. To ensure consistency and facilitate comparative analysis, the methodology used to generate the previous results was replicated precisely, with the same bunch of 250 digits selected from the dataset. The outcomes of these iterative tests are presented in Figure 18.

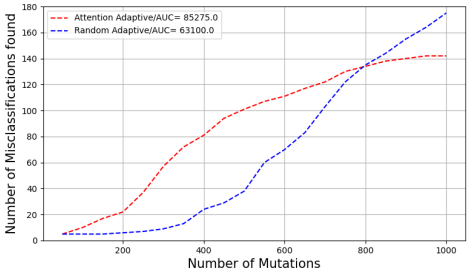
The results underscored the significant influence that the addition of the adaptive extent strategy has on the mutation process. Evidently, the integration of this strategy augmented the quantity of mutations generated by both methodologies across the entire range of maximum mutations. A particularly noteworthy observation is that for a segment of the range of maximum mutations - more precisely, proximate to the maximum value on the X-axis - the random method seemingly outperforms the attention-based one.

For a more comprehensive comparison and to emphasize the disparities among the methods, the data from each technique was incorporated into a single chart. As depicted in

Figure 18 – Performance comparison of strategies based on the maximum number of iterations and resultant misclassifications using adaptive extent strategy.



(a) Comparative: Attention Probability-Based Adaptive strategy (CPS-WBS) vs Normal Adaptive method.

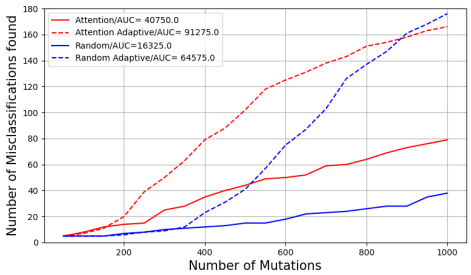


(b) Comparative: Attention Distance-Based Adaptive strategy (SS-DBPS) vs Normal Adaptive method.

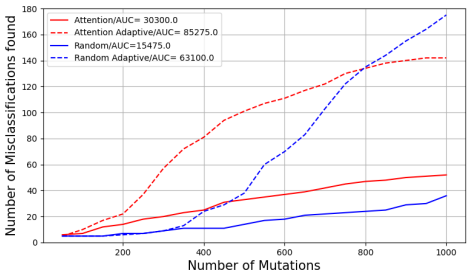
Source: The Author (2022).

figure Figure 19, the metrics of all the techniques are consolidated for ease of comparison. In Figure 19a, the left-hand chart represents the techniques in relation to the Probability Based Strategy (CPS-WBS), whereas Figure 19b on the right embodies the techniques in respect to the Distance Based Strategy (SS-DBPS). In both visualizations, the area under the curve (AUC) of the attention techniques surpasses that of the normal random method. When considering the adaptive extent strategy, it is apparent that techniques with this strategy implemented achieve a higher AUC relative to the others.

Figure 19 – Performance comparison of strategies based on the maximum number of iterations and resultant misclassifications using adaptive extent strategy.



(a) Comparison of CPS-WBS mutation strategies: attention, normal, attention adaptive, and normal adaptive.



(b) Comparison of SS-DBPS mutation strategies: attention, normal, attention adaptive, and normal adaptive.

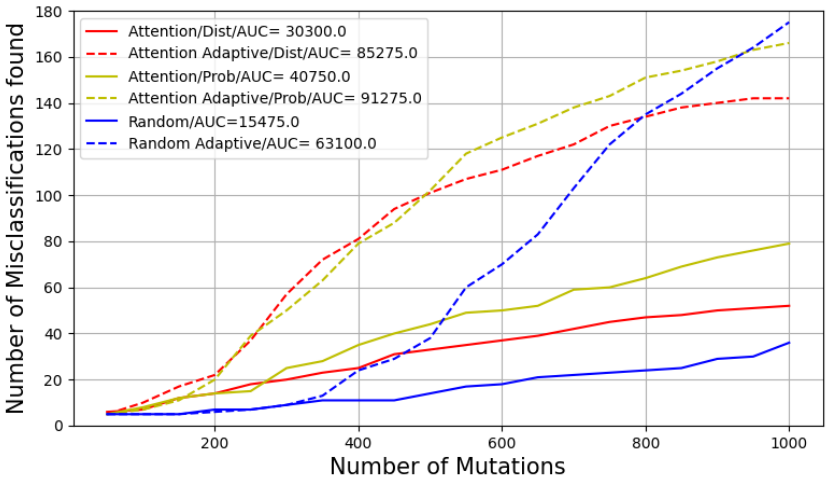
Source: The Author (2022).

The comprehensive comparative chart in Figure 20 encapsulates six distinct plots that represent three methods: Attention Distance, Attention Probability, and Normal Random, both with and without the incorporation of the adaptive extent. In the case of Attention Distance and Probability methods without the adaptive extent, they demonstrate superior performance over the Normal Random method across the entire range of the x-axis, which

represents the maximum number of mutations.

However, when inspecting the plots demonstrating the three techniques that utilize the adaptive extent, a more nuanced pattern is observed. For approximately the initial 80% of the range on the x-axis, both the Attention Distance and Attention Probability methods prevail over the Normal Random method. Remarkably, in the concluding 20% of the range, the Normal Adaptive method begins to surpass the performance of the Attention Adaptive methods. This intriguing transition emphasizes the critical role of adaptive extent in the mutation process and further elucidates the comparative dynamics of these strategies.

Figure 20 – Comprehensive comparison of Attention Distance, Attention Probability, and Normal Random methods, with and without the Adaptive Extent, demonstrating shifting performance dynamics



Source: The Author (2022).

3.8.3.3 Confusion Matrix of Misclassifications

The following tables present the confusion matrix for the number of misclassifications generated by each method. The set-up for this experiment paralleled the previous one, taking into account 250 digits from the MNIST dataset. Notably, the maximum number of mutations per digit was fixed at a threshold of 1000 mutations.

Probability Method

Table 1 – Without Adaptive Extent (CPS-WBS)

Att vs Rnd	Att Yes	Att No
Rnd Yes	31	7
Rnd No	48	164

Table 3 – With Adaptive Extent (CPS-WBS)

Att vs Rnd	Att Yes	Att No
Rnd Yes	133	43
Rnd No	33	41

Distance Method

Table 2 – Without Adaptive Extent (SS-DBPS)

Att vs Rnd	Att Yes	Att No
Rnd Yes	21	15
Rnd No	31	183

Table 4 – With Adaptive Extent (SS-DBPS)

Att vs Rnd	Att Yes	Att No
Rnd Yes	112	63
Rnd No	30	45

Source: The Author.

On the left-hand side, two tables display the results for the Probability-based method. The upper table (Table 1) presents the outcome using the attention-guided strategy as opposed to the normal one, without employing adaptive extent. The lower table (Table 3), meanwhile, reveals the results when the adaptive extent is applied.

On the right-hand side, the outcomes for the Distance-based method are shown. The top table (Table 2) represents the results without using the adaptive extent, while the bottom table (Table 4) shows the outcome when the adaptive extent is utilized.

As per the analysis of these results, we can corroborate the superior performance of the attention-guided strategy in generating a greater number of misclassifications across both strategies—distances and probabilities. After integrating the adaptive extent into both strategies, a significant improvement in the results generated by the normal method is evident in contrast to the attention method. In fact, as demonstrated in Table 4, the normal method surpassed the attention method by generating 33 additional misclassifications. This intriguing trend can initially be elucidated by our previous analysis. Given that the number of mutations in this experiment was capped at 1000, the normal adaptive method has been observed to outperform the Attention Adaptive method.

3.9 CONCLUSION AND FUTURE WORK

Throughout the course of the preliminary study, we embarked on the exploration of the central question: "Can Explainable AI, when compared to traditional mutation methods, serve as a more efficient technique for generating corner cases of Neural Networks?". This inquiry took us through various experimental designs and rigorous testing, with the aim of establishing a solid comparative foundation between traditional mutation methods and Explainable AI techniques.

Our findings, derived from extensive experimentation, indicate a clear superiority of Attention guided methods over traditional random ones in generating corner cases. The edge was prominent and consistent across different experimental setups, underscoring the robustness of the Explainable AI approach.

Moreover, our research journey led us to an innovative addition to the arsenal of AI strategies: the Adaptive Extent strategy. Incorporating this into the existing techniques, we observed a significant enhancement in performance across the board. Both Attention guided methods and traditional ones benefited from the Adaptive Extent strategy, making it a versatile tool that augments the overall efficiency of corner case generation.

Looking beyond the confines of this research, the promising results from the Attention Guided strategies open up avenues for further study. One of the most exciting prospects is the integration of this strategy in the realm of driving simulation. The explainable AI images obtained from vehicle front cameras and the superior performance of the Attention guided strategies in finding misclassifications imply an increased potential for the discovery of challenging road layouts during the testing of Autonomous Driving Systems (ADS). This enhanced ability to uncover corner cases that lead to system failure strengthens the testing process, leading to more reliable and safe ADS.

This research stands as a testament to the transformative potential of Explainable AI in creating efficient techniques for corner case generation. By augmenting traditional methods with AI-guided strategies, we can greatly improve the performance and versatility of our systems. However, it is essential to bear in mind that our work is but one step in a larger journey. As the field continues to evolve, it is incumbent upon future research to further explore and refine these strategies, ensuring their optimal application across diverse domains.

REFERENCES

ABDESSALEM, R. B.; PANICHELLA, A.; NEJATI, S.; BRIAND, L. C.; STIFTER, T. Testing autonomous cars for feature interaction failures using many-objective search. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. [S.l.]: ACM, 2018. (ASE 2018).

AN, J.; CHO, S. Variational autoencoder based anomaly detection using reconstruction probability. In: . [S.l.: s.n.], 2015.

ATAKISHIYEV, S.; SALAMEH, M.; YAO, H.; GOEBEL, R. Explainable artificial intelligence for autonomous driving: An overview and guide for future research directions. arXiv, 2021. Available at: <https://arxiv.org/abs/2112.11561>.

Ben Abdessalem, R.; Nejati, S.; Briand, L. C.; Stifter, T. Testing advanced driver assistance systems using multi-objective search and neural networks. In: 2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE). [S.l.: s.n.], 2016.

Ben Abdessalem, R.; Nejati, S.; C. Briand, L.; Stifter, T. Testing vision-based control systems using learnable evolutionary algorithms. In: 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE). [S.l.: s.n.], 2018.

BGR Media, LLC. Waymo's self-driving cars hit 10 million miles. 2018. <https://techcrunch.com/2018/10/10/waymos-self-driving-cars-hit-10-million-miles>. Online; accessed 18 August 2019.

BINDER, A.; MONTAVON, G.; BACH, S.; MÜLLER, K.; SAMEK, W. Layer-wise relevance propagation for neural networks with local renormalization layers. CoRR, abs/1604.00825, 2016. Available at: <http://arxiv.org/abs/1604.00825>.

BLAIR, D. C. Information retrieval, 2nd ed. c.j. van rijsbergen. london: Butterworths; 1979: 208 pp. price: $32.50. Journal of the American Society for Information Science, v. 30, n. 6, p. 374–375, 1979. Available at: <https://asistdl.onlinelibrary.wiley.com/doi/abs/10.1002/asi.4630300621>.

BOGDOLL, D.; NITSCHE, M.; ZöLLNER, J. M. Anomaly Detection in Autonomous Driving: A Survey. arXiv, 2022. Available at: <https://arxiv.org/abs/2204.07974>.

BOJARSKI, M.; CHOROMANSKA, A.; CHOROMANSKI, K.; FIRNER, B.; JACKEL, L.; MULLER, U.; ZIEBA, K. VisualBackProp: efficient visualization of CNNs. arXiv, 2016. Available at: <https://arxiv.org/abs/1611.05418>.

BOJARSKI, M.; TESTA, D. D.; DWORAKOWSKI, D.; FIRNER, B.; FLEPP, B.; GOYAL, P.; JACKEL, L. D.; MONFORT, M.; MULLER, U.; ZHANG, J.; ZHANG, X.; ZHAO, J.; ZIEBA, K. End to end learning for self-driving cars. CoRR, abs/1604.07316, 2016.

BORG, M.; HENRIKSSON, J.; SOCHA, K.; LENNARTSSON, O.; LöNEGREN, E. S.; BUI, T.; TOMASZEWSKI, P.; SATHYAMOORTHY, S. R.; BRINK, S.; MOGHADAM, M. H. Ergo, SMIRK is Safe: A Safety Case for a Machine Learning Component

in a Pedestrian Automatic Emergency Brake System. arXiv, 2022. Available at: <https://arxiv.org/abs/2204.07874>.

CERF, V. G. A comprehensive self-driving car test. Commun. ACM, ACM, v. 61, n. 2, Jan. 2018.

CHANDOLA, V.; BANERJEE, A.; KUMAR, V. Anomaly detection: A survey. ACM Comput. Surv., ACM, v. 41, n. 3, Jul. 2009.

CHATTOPADHAY, A.; SARKAR, A.; HOWLADER, P.; BALASUBRAMANIAN, V. N. Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks. In: 2018 IEEE Winter Conference on Applications of Computer Vision (WACV). [S.l.: s.n.], 2018. p. 839–847.

COHEN, J. Statistical power analysis for the behavioral sciences. Hillsdale, N.J: L. Erlbaum Associates, 1988. ISBN 978-1-134-74270-7.

CommaAI. End to end learning for self-driving cars. CoRR, abs/1604.07316, 2016.

DENG, L. The mnist database of handwritten digit images for machine learning research. IEEE Signal Processing Magazine, IEEE, v. 29, n. 6, p. 141–142, 2012.

DOLA, S.; DWYER, M. B.; SOFFA, M. L. Distribution-aware testing of neural networks using generative models. In: IEEE. 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE). [S.l.], 2021. p. 226–237.

EVERITT, B. S.; LANDAU, S.; LEESE, M. Cluster Analysis. 4th. ed. [S.l.]: Wiley Publishing, 2009.

FAHMY, H.; PASTORE, F.; BRIAND, L. Simulator-based explanation and debugging of hazard-triggering events in DNN-based safety-critical systems. arXiv, 2022. Available at: <https://arxiv.org/abs/2204.00480>.

FAHMY, H. M.; BAGHERZADEH, M.; PASTORE, F.; BRIAND, L. C. Supporting DNN safety analysis and retraining through heatmap-based unsupervised learning. CoRR, abs/2002.00863, 2020. Available at: <https://arxiv.org/abs/2002.00863>.

GAL, Y.; GHAHRAMANI, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48. [S.l.]: JMLR.org, 2016. (ICML'16).

GAMBI, A.; MUELLER, M.; FRASER, G. Automatically testing self-driving cars with search-based procedural content generation. In: Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis. [S.l.]: ACM, 2019. (ISSTA 2019).

GOODFELLOW, I. J.; SHLENS, J.; SZEGEDY, C. Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572, 2014.

GRIGORESCU, S.; TRASNEA, B.; COCIAS, T.; MACESANU, G. A survey of deep learning techniques for autonomous driving. Journal of Field Robotics, Wiley Online Library, v. 37, n. 3, p. 362–386, 2020.

HAQ, F. U.; SHIN, D.; NEJATI, S.; BRIAND, L. Comparing offline and online testing of deep neural networks: An autonomous car case study. In: Proceedings of 13th IEEE International Conference on Software Testing, Verification and Validation. [S.l.]: IEEE, 2020. (ICST '20).

HELL, F.; HINZ, G.; LIU, F.; GOYAL, S.; PEI, K.; LYTVYNENKO, T.; KNOLL, A.; YIQIANG, C. Monitoring perception reliability in autonomous driving: Distributional shift detection for estimating the impact of input data on prediction accuracy. In: Computer Science in Cars Symposium. New York, NY, USA: Association for Computing Machinery, 2021. (CSCS '21). ISBN 9781450391399. Available at: <https://doi.org/10.1145/3488904.3493382>.

HENRIKSSON, J.; BERGER, C.; BORG, M.; TORNBERG, L.; ENGLUND, C.; SATHYAMOORTHY, S. R.; URSING, S. Towards structured evaluation of deep neural network supervisors. In: 2019 IEEE International Conference On Artificial Intelligence Testing (AITest). [S.l.]: IEEE, 2019.

HUMBATOVA, N.; JAHANGIROVA, G.; BAVOTA, G.; RICCIO, V.; STOCCO, A.; TONELLA, P. Taxonomy of real faults in deep learning systems. In: Proceedings of 42nd International Conference on Software Engineering. [S.l.]: ACM, 2020. (ICSE '20), p. 12 pages.

HUMBATOVA, N.; JAHANGIROVA, G.; TONELLA, P. Deepcrime: Mutation testing of deep learning systems based on real faults. In: Proceedings of the 30th ACM SIGSOFT International Symposium on Software Testing and Analysis. New York, NY, USA: Association for Computing Machinery, 2021. (ISSTA 2021), p. 67–78. ISBN 9781450384599. Available at: <https://doi.org/10.1145/3460319.3464825>.

HUSSAIN, M.; ALI, N.; HONG, J.-E. Deepguard: A framework for safeguarding autonomous driving systems from inconsistent behaviour. Automated Software Engg., Kluwer Academic Publishers, USA, v. 29, n. 1, may 2022. ISSN 0928-8910. Available at: <https://doi.org/10.1007/s10515-021-00310-0>.

JAHANGIROVA, G.; STOCCO, A.; TONELLA, P. Quality metrics and oracles for autonomous vehicles testing. In: Proceedings of 14th IEEE International Conference on Software Testing, Verification and Validation. [S.l.]: IEEE, 2021. (ICST '21).

JETLEY, S.; LORD, N. A.; LEE, N.; TORR, P. H. Learn to pay attention. arXiv preprint arXiv:1804.02391, 2018.

JUNG, Y.-J.; HAN, S.-H.; CHOI, H.-J. Explaining cnn and rnn using selective layer-wise relevance propagation. IEEE Access, v. 9, p. 18670–18681, 2021.

KIM, J.; CANNY, J. Interpretable Learning for Self-Driving Cars by Visualizing Causal Attention. arXiv, 2017. Available at: <https://arxiv.org/abs/1703.10631>.

KIM, J.; FELDT, R.; YOO, S. Guiding deep learning system testing using surprise adequacy. In: Proceedings of the 41st International Conference on Software Engineering. [S.l.]: IEEE Press, 2019. (ICSE '19).

KOHONEN, T. Self-Organizing Maps, Third Edition. [S.l.]: Springer, 2001. (Springer Series in Information Sciences).

KOROSEC, K. Waymo's robotaxi pilot surpassed 6,200 riders in its first month in california. TechCrunch, 2019. Available at: <https://techcrunch.com/2019/10/30/waymos-robotaxi-pilot-surpassed-6200-riders-in-its-first-month-in-california/>.

KUBOTA, Y. tf-keras-vis. 2021. Available at: <https://keisen.github.io/tf-keras-vis-docs/>.

LAGE, I.; CHEN, E.; HE, J.; NARAYANAN, M.; KIM, B.; GERSHMAN, S.; DOSHI-VELEZ, F. An evaluation of the human-interpretability of explanation. CoRR, abs/1902.00006, 2019. Available at: <http://arxiv.org/abs/1902.00006>.

LATEEF, F.; KAS, M.; RUICHEK, Y. Saliency heat-map as visual attention for autonomous driving using generative adversarial network (gan). IEEE Transactions on Intelligent Transportation Systems, p. 1–14, 2021.

LEE, J. K.; LEE, K. W. Study on effectiveness of pre-crash active seatbelt using real time controlled simulation. In: . [S.l.: s.n.], 2013.

LEI, T.; BARZILAY, R.; JAAKKOLA, T. S. Rationalizing neural predictions. CoRR, abs/1606.04155, 2016. Available at: <http://arxiv.org/abs/1606.04155>.

LI, D.; WANG, T.; LI, Y.; HU, S.; XING, Z.; BAO, T.; ROTHERMEL, G. Deepmutation++: A mutation testing framework for deep learning systems. 2020 35th IEEE/ACM International Conference on Automated Software Engineering (ASE), p. 353–365, 2020.

LOU, G.; DENG, Y.; ZHENG, X.; ZHANG, M.; ZHANG, T. Investigation into the state-of-the-practice autonomous driving testing. arXiv, 2021. Available at: <https://arxiv.org/abs/2106.12233>.

MICHELMORE, R.; WICKER, M.; LAURENTI, L.; CARDELLI, L.; GAL, Y.; KWIATKOWSKA, M. Uncertainty quantification with statistical guarantees in end-to-end autonomous driving control. In: 2020 IEEE International Conference on Robotics and Automation, ICRA 2020, Paris, France, May 31 - August 31, 2020. IEEE, 2020. p. 7344–7350. Available at: <https://doi.org/10.1109/ICRA40945.2020.9196844>.

MOGHADAM, M. H.; BORG, M.; SAADATMAND, M.; MOUSAVIRAD, S. J.; BOHLIN, M.; LISPER, B. Machine Learning Testing in an ADAS Case Study Using Simulation-Integrated Bio-Inspired Search-Based Testing. arXiv, 2022. Available at: <https://arxiv.org/abs/2203.12026>.

MOHSENI, S.; JAGADEESH, A.; WANG, Z. Predicting model failure using saliency maps in autonomous driving systems. CoRR, abs/1905.07679, 2019. Available at: <http://arxiv.org/abs/1905.07679>.

MULLINS, G. E.; STANKIEWICZ, P. G.; HAWTHORNE, R. C.; GUPTA, S. K. Adaptive generation of challenging scenarios for testing and evaluation of autonomous vehicles. Journal of Systems and Software, v. 137, 2018.

PEI, K.; CAO, Y.; YANG, J.; JANA, S. Deepxplore: Automated whitebox testing of deep learning systems. In: Proceedings of the 26th Symposium on Operating Systems Principles. [S.l.]: ACM, 2017. (SOSP '17).

PEI, K.; CAO, Y.; YANG, J.; JANA, S. Deepxplore: Automated whitebox testing of deep learning systems. SOSP '17, p. 1–18, 2017.

RIBEIRO, M. T.; SINGH, S.; GUESTRIN, C. "why should I trust you?": Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016. [S.l.: s.n.], 2016. p. 1135–1144.

RICCIO, V.; JAHANGIROVA, G.; STOCCO, A.; HUMBATOVA, N.; WEISS, M.; TONELLA, P. Testing Machine Learning based Systems: A Systematic Mapping. Empirical Software Engineering, Springer, 2020.

RICCIO, V.; TONELLA, P. Model-Based Exploration of the Frontier of Behaviours for Deep Learning System Testing. In: Proceedings of ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering. [S.l.: s.n.], 2020. (ESEC/FSE '20).

RICCIO, V.; TONELLA, P. Model-based exploration of the frontier of behaviours for deep learning system testing. In: Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering. [S.l.: s.n.], 2020. p. 876–888.

SAMEK, W.; MONTAVON, G.; VEDALDI, A.; HANSEN, L. K.; MÜLLER, K.-R. Explainable AI: interpreting, explaining and visualizing deep learning. [S.l.]: Springer Nature, 2019.

SAMEK, W.; WIEGAND, T.; MÜLLER, K.-R. Explainable artificial intelligence: Understanding, visualizing and interpreting deep learning models. arXiv preprint arXiv:1708.08296, 2017.

SCHÖLKOPF, B.; WILLIAMSON, R. C.; SMOLA, A. J.; SHAWE-TAYLOR, J.; PLATT, J. C. Support vector method for novelty detection. In: Advances in Neural Information Processing Systems 12, (NIPS). [S.l.: s.n.], 1999.

SCULLEY, D.; HOLT, G.; GOLOVIN, D.; DAVYDOV, E.; PHILLIPS, T.; EBNER, D.; CHAUDHARY, V.; YOUNG, M.; CRESPO, J.-F.; DENNISON, D. Hidden technical debt in machine learning systems. In: CORTES, C.; LAWRENCE, N.; LEE, D.; SUGIYAMA, M.; GARNETT, R. (Ed.). Advances in Neural Information Processing Systems. [S.l.]: Curran Associates, Inc., 2015. v. 28.

SELVARAJU, R. R.; DAS, A.; VEDANTAM, R.; COGSWELL, M.; PARIKH, D.; BATRA, D. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. CoRR, abs/1610.02391, 2016. Available at: <http://arxiv.org/abs/1610.02391>.

SIMON, D.; KAUFFMANN, J.; MOSEBACH, J.; KUHNT, F.; DIETMAYER, K. Adversarial attacks against deep learning systems for ugv and uav autonomous navigation. In: 2020 IEEE Intelligent Vehicles Symposium (IV). [S.l.: s.n.], 2020. p. 2131–2137.

SMILKOV, D.; THORAT, N.; KIM, B.; VIÉGAS, F. B.; WATTENBERG, M. Smoothgrad: removing noise by adding noise. CoRR, abs/1706.03825, 2017. Available at: <http://arxiv.org/abs/1706.03825>.

SPRINGENBERG, J. T.; DOSOVITSKIY, A.; BROX, T.; RIEDMILLER, M. Striving for Simplicity: The All Convolutional Net. In: . [S.l.: s.n.], 2014.

STANDARDIZATION, T. R. I. . International Organization for. Road Vehicles - Safety of the Intended Functionality. 2019.

Stocco. Replication Package. 2022. <https://github.com/tsigalko18/ase22>.

STOCCO, A.; NUNES, P. J.; D'AMORIM, M.; TONELLA, P. ThirdEye: Attention maps for safe autonomous driving systems. In: Proceedings of 37th IEEE/ACM International Conference on Automated Software Engineering. [S.l.]: IEEE/ACM, 2022. (ASE '22).

STOCCO, A.; PULFER, B.; TONELLA, P. Mind the Gap! A Study on the Transferability of Virtual vs Physical-world Testing of Autonomous Driving Systems. IEEE Transactions on Software Engineering, IEEE, 2022.

STOCCO, A.; TONELLA, P. Towards anomaly detectors that learn continuously. In: Proceedings of 31st International Symposium on Software Reliability Engineering Workshops. [S.l.]: IEEE, 2020. (ISSREW 2020).

STOCCO, A.; TONELLA, P. Confidence-driven weighted retraining for predicting safety-critical failures in autonomous driving systems. Journal of Software: Evolution and Process, John Wiley Sons, 2021. Available at: <https://doi.org/10.1002/smr.2386>.

STOCCO, A.; WEISS, M.; CALZANA, M.; TONELLA, P. Misbehaviour prediction for autonomous driving systems. In: Proceedings of 42nd International Conference on Software Engineering. [S.l.]: ACM, 2020. (ICSE '20).

STRICKLAND, M.; FAINEKOS, G.; Ben Amor, H. Deep predictive models for collision risk assessment in autonomous driving. In: 2018 IEEE International Conference on Robotics and Automation, ICRA 2018. [S.l.]: Institute of Electrical and Electronics Engineers Inc., 2018. (Proceedings - IEEE International Conference on Robotics and Automation).

TIAN, Y.; PEI, K.; JANA, S.; RAY, B. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In: Proceedings of the 40th International Conference on Software Engineering. [S.l.]: ACM, 2018. (ICSE '18).

TJOA, E.; GUAN, C. A survey on explainable artificial intelligence (XAI): towards medical XAI. CoRR, abs/1907.07374, 2019. Available at: <http://arxiv.org/abs/1907.07374>.

TJOA, E.; KHOK, H. J.; CHOUHAN, T.; GUAN, C. Improving deep neural network classification confidence using heatmap-based explainable AI. CoRR, abs/2201.00009, 2022. Available at: <https://arxiv.org/abs/2201.00009>.

TRANSPORTATION, N. H. T. S. A. U.S. Department of. Pre-Crash Scenario Typology for Crash Avoidance Research. 2007.

Udacity. A self-driving car simulator built with Unity. 2017. <https://github.com/udacity/self-driving-car-sim>. Online; accessed 18 August 2019.

UNITY. Unity3D. 2021. <https://unity.com>.

WANG, H.; XU, J.; XU, C.; MA, X.; LU, J. Dissector: Input validation for deep learning applications by crossing-layer dissection. In: 2020 IEEE/ACM 42nd International Conference on Software Engineering (ICSE). [S.l.: s.n.], 2020. p. 727–738.

Waymo Driver. Waymo Driver. 2021. <https://waymo.com/waymo-driver/>.

Waymo Secret Testing. Waymo Secret Testing. 2017. <https://www.theatlantic.com/technology/archive/2017/08/inside-waymos-secret-testing-and-simulation-facilities/537648/>.

WILCOXON, F. Individual comparisons by ranking methods. Biometrics Bulletin, JSTOR, v. 1, n. 6, p. 80, Dec. 1945. Available at: <https://doi.org/10.2307/3001968>.

XIAO, Y.; BESCHASTNIKH, I.; ROSENBLUM, D. S.; SUN, C.; ELBAUM, S.; LIN, Y.; DONG, J. S. Self-checking deep neural networks in deployment. In: 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE). [S.l.: s.n.], 2021. p. 372–384.

XU, Y.; YANG, X.; GONG, L.; LIN, H.-C.; WU, T.-Y.; LI, Y.; VASCONCELOS, N. Explainable Object-induced Action Decision for Autonomous Vehicles. arXiv, 2020. Available at: <https://arxiv.org/abs/2003.09405>.

YURTSEVER, E.; LAMBERT, J.; CARBALLO, A.; TAKEDA, K. A survey of autonomous driving: Common practices and emerging technologies. IEEE access, IEEE, v. 8, p. 58443–58469, 2020.

ZEILER, M. D.; FERGUS, R. Visualizing and understanding convolutional networks. CoRR, abs/1311.2901, 2013. Available at: <http://arxiv.org/abs/1311.2901>.

ZHANG, J.; LI, J.; SHU, P.; GU, Y.; YANG, J. Deeproad: Gan-based metamorphic autonomous driving system testing. In: 2018 IEEE 29th International Symposium on Software Reliability Engineering (ISSRE). [S.l.: s.n.], 2018. p. 50–61.

ZHANG, M.; ZHANG, Y.; ZHANG, L.; LIU, C.; KHURSHID, S. Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems. In: Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering. [S.l.]: ACM, 2018. (ASE 2018).

ZHANG, X.; ZHAI, J.; MA, S.; SHEN, C. Autotrainer: An automatic dnn training problem detection and repair system. In: 2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE). [S.l.: s.n.], 2021. p. 359–371.

ZHANG, Z.; WU, P.; CHEN, Y.; SU, J. Out-of-distribution detection through relative activation-deactivation abstractions. In: 2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE). [S.l.: s.n.], 2021. p. 150–161.

ZHAO, H.; HARI, S. K. S.; TSAI, T.; SULLIVAN, M. B.; KECKLER, S. W.; ZHAO, J. Suraksha: A framework to analyze the safety implications of perception design choices in avs. In: 2021 IEEE 32nd International Symposium on Software Reliability Engineering (ISSRE). [S.l.: s.n.], 2021. p. 434–445.

ZHAO, Z.; ZHOU, L.; ZHU, Q.; LUO, Y.; LI, K. A review of essential technologies for collision avoidance assistance systems. Advances in Mechanical Engineering, v. 9, n. 10, 2017.

ZOHDINASAB, T.; RICCIO, V.; GAMBI, A.; TONELLA, P. Deephyperion: Exploring the feature space of deep learning-based systems through illumination search. In: Proceedings of the ACM SIGSOFT International Symposium on Software Testing and Analysis. [S.l.]: Association for Computing Machinery, 2021. (ISSTA '21).