Software Engineering for Artificial Intelligence (SE for AI)

One-Slide Summary

- **Why are AI systems so important?** AI systems are crucial today because they significantly enhance our ability to process and analyze vast amounts of data, leading to more informed decision-making and automation of complex tasks.
- **PAC:** Probably Approximately Correct (PAC) is a framework in computational learning theory for analyzing AI systems.
- **SE for AI:** Applying software engineering methods is crucial in developing and using AI systems because it ensures the creation of reliable, scalable, and maintainable AI applications.

Learning Objectives: by the end of today's lecture, you should be able to…

1. (*Knowledge*) describe the primary activities in Artificial Intelligence (AI) using software engineering (SE)

2. (*Value*) understand why the applications of SE in AI are important

3. (*Skill*) Review some recent papers

Overview

- Background
- What is Software Engineering (SE) for Artificial Intelligence (AI)?
- What are the applications of SE in AI?
  - White-Box Testing of DNNs
  - Black-Box Testing of DNNs
  - Formal Verification of DNNs
  - How can we evaluate or write tests for LLMs

Background

How widespread are AI systems and applications?

- AI systems and applications are now deeply integrated into various aspects of modern life and industries.
- According to recent reports, around 34% of companies currently use AI, with an additional 42% exploring its potential.
- AI technologies are prevalent in healthcare, finance, retail, manufacturing, and smart vehicles, where they enhance efficiency, accuracy, and decision-making processes.

Advantages of Artificial Intelligence (AI) Systems

- Artificial Intelligence (AI) increases efficiency and productivity by automating repetitive tasks, which allows humans to focus on more complex and creative activities.
- AI systems enhance decision-making by quickly analyzing vast amounts of data, providing valuable insights across various fields like finance, healthcare, and marketing.
- Additionally, AI enables personalized experiences, reduces operational costs through automation, and improves accuracy in tasks such as medical diagnostics.

AI Prospects

- In healthcare, AI is set to revolutionize diagnostics and treatment personalization.
- The global AI market is projected to grow significantly, reaching an estimated $1,339 billion by 2030.
- Autonomous systems, such as self-driving cars and drones, are expected to enhance transportation safety and logistics.
- AI's role in developing smart cities can lead to optimized energy use and improved public safety.
- AI can contribute to environmental sustainability by optimizing resource use and predicting natural disasters.
- The future of AI is bright, with its applications expanding and evolving to significantly impact various aspects of our lives and society.

Data Quality, Privacy, and Security Challenges

- Ensuring high-quality and sufficient data is difficult for accurate results.
- Data privacy and security are major concerns, requiring strict compliance with regulations and robust protection measures.
- Selecting the right algorithms and models demands significant expertise and computational resources.

Integration, Ethics, and Evaluation Challenges

- Integrating AI with existing systems can be complex, often necessitating substantial changes.
- Ethical considerations, such as preventing bias and ensuring transparency, are critical.
- Additionally, measuring AI performance can be difficult, and managing changes in workflows and processes is essential for successful implementation.

How can we tackle these AI challenges?

- We will show how software engineering techniques, like testing and formal verification, can help with each of the AI challenges mentioned before.
- In essence, we try to answer the question: How do we evaluate or write tests for large AI systems such as ChatGPT?”

Probably Approximately Correct (PAC)

- Probably Approximately Correct (PAC) is a framework in computational learning theory for analyzing machine learning algorithms.
- It was introduced by Leslie Valiant in 1984.

PAC Learning

- In PAC learning, the goal is to find a hypothesis (a generalization function) that is "probably approximately correct."
- This means that with high probability, the hypothesis will be close to the true function, within a specified error margin.

PAC Framework

- The PAC framework provides a way to understand the relationship between the number of training samples, the error rate, and the confidence level that the hypothesis is correct.

PAC's Key Components

- Probably: The hypothesis is correct with a high probability (1 - δ), where (δ) is a small probability of failure.
- Approximately: The hypothesis is approximately correct with an error rate less than a specified threshold (ϵ).
- Correct: The hypothesis correctly classifies new samples drawn from the same distribution as the training data.

Who is Leslie Valiant?

- Leslie Gabriel Valiant is a renowned British-American computer scientist and computational theorist.
- Born on March 28, 1949, in Budapest, Hungary, he has made significant contributions to computer science, particularly in computational learning theory and complexity theory.

Valiant’s other achievements

- Valiant has received numerous accolades for his work, including the prestigious Turing Award in 2010, often referred to as the "Nobel Prize of Computing”.
- He is currently the T. Jefferson Coolidge Professor of Computer Science and Applied Mathematics at Harvard University.

What are Algorithms?

- An algorithm is simply any well-defined procedure.
- It is derived from the Latinized transliteration Algoritmi of the name of the mathematician Al-Khwārizmī, who worked in the House of Wisdom in Baghdad in the ninth century and authored an influential book on algebra.

Where are Algorithms Used?

- Algorithms as traditionally studied in mathematics and computer science are designed to solve instances of certain problems such as solving algebraic equations or searching for a word in a text.
- All the expertise needed for their realization is encoded in their description by their designer.

What are Ecorithms?

- Ecorithms are types of learning algorithms that operate within a biological or ecological context.
- Ecorithms describe adaptive behaviors in systems that interact with their environment, such as biological organisms or ecosystems.
- While their realization is foreseeable, their course will vary according to the environment.

https://web.eecs.umich.edu/~movaghar/Valiant2014-Chaps1-2.pdf

Algorithms Versus Ecorithms

- Algorithms are usually defined after a finite number of steps using limited resources.
- Ecorithms, on the other hand, are defined in the learning model of PAC.
- The phenomena that they seek to explain are some of the most familiar to human experience: learning, resilience, and adaptation.

Overview

- Background
- What is Software Engineering (SE) for Artificial Intelligence (AI)?
- What are the applications of SE in AI?
  - White-Box Testing of DNNs
  - Black-Box Testing of DNNs
  - Formal Verification of DNNs
  - How can we evaluate or write tests for LLMs

What is Software Engineering (SE) for Artificial Intelligence (AI)?

SE for AI

- Using software engineering principles for AI development is essential for creating robust, scalable, and maintainable AI systems.
- By applying software engineering principles, AI systems can be developed more effectively, ensuring they are reliable, efficient, and aligned with user needs and ethical standards.

System Design and Architecture

- Software engineers design the overall architecture of AI systems, ensuring they are modular, scalable, and efficient.
- This involves selecting appropriate frameworks, libraries, and tools.

Model Development and Integration

- Engineers develop machine learning models and integrate them into larger software systems.
- This requires knowledge of both AI algorithms and software development practices.

Data Management

- Effective data management is crucial for AI.
- Software engineers design and implement data pipelines, ensuring data is collected, processed, and stored efficiently.

Testing and Validation

- Rigorous testing is essential to ensure AI models perform as expected.
- Engineers develop testing frameworks and methodologies to validate models and their integration into systems.

Deployment and Maintenance

- Deploying AI models in production environments and maintaining them over time requires robust software engineering practices.
- This includes using containerization, continuous integration/continuous deployment (CI/CD) pipelines, and monitoring tools.

Performance Optimization

- Engineers optimize the performance of AI systems, ensuring they run efficiently and meet performance requirements.
- This can involve tuning hyperparameters, optimizing code, and using specialized hardware like GPUs.

Ethical Considerations

- Software engineers also address ethical considerations in AI, such as ensuring fairness, transparency, and accountability.
- This involves implementing practices to mitigate bias and ensure compliance with regulations.

What is Deep Learning?

- The concept of deep learning typically involves the use of deep neural networks.
- Deep learning is a subset of machine learning that uses neural networks with multiple layers (hence "deep") to model complex patterns in data.
- These deep neural networks are designed to simulate the way the human brain processes information, allowing them to perform tasks such as image and speech recognition, natural language processing, and more.
  - https://www.ibm.com/topics/deep-learning
  - https://builtin.com/machine-learning/deep-learning

Deep Neural Networks (DNNs)

- Deep neural networks (DNNs) are a type of artificial neural network with multiple layers between the input and output layers.
- These layers allow the network to learn and model complex patterns and relationships within data.
- Each layer extracts increasingly abstract features from the input, enabling the network to perform tasks such as image and speech recognition, natural language processing, and more accurately.

Why DNNs are so dominant in today's AI applications?

- DNNs have become dominant in today's AI systems due to their superior performance and versatility.
- They can handle large datasets and complex models, making them suitable for various applications.
- Advances in computational power, such as GPUs, and algorithm improvements have enhanced their efficiency and effectiveness.
- This combination of factors has made DNNs a cornerstone of modern AI, driving significant advancements across various industries.

Why do Large Language Models (LLMs) like GPT-4 Use Deep Neural Network (DNN) Architecture?

- LLMs need to understand and generate human language, which involves recognizing patterns, context, and nuances. DNNs, with their multiple layers, are well-suited for capturing these complexities.
- Deep architectures allow LLMs to scale up, handling vast amounts of text data and learning from it efficiently. This scalability is essential for training models on large corpora.
- DNNs enable transfer learning, where a model trained on one task can be fine-tuned for another related task. This capability is handy for LLMs, which can be adapted to various language-related tasks.

The Nobel Prize in Physics 2024

The Nobel Prize in Physics 2024 was awarded jointly to John J. Hopfield and Geoffrey E. Hinton "for foundational discoveries and inventions that enable machine learning with artificial neural networks"

Ill. Niklas Elmehed © Nobel Prize Outreach

John J. Hopfield

Prize share: 1/2

Ill. Niklas Elmehed © Nobel Prize Outreach

Geoffrey Hinton

Prize share: 1/2

Some DNN Verification Tools

- α,β-CROWN (alpha-beta-CROWN)
- DNNV

- https://github.com/Verified-Intelligence/alpha-beta-CROWN
- https://github.com/dlshriver/dnnv

Major Companies using α,β-CROWN and DNNV

- Google: Utilizes α,β-CROWN, and DNNV for verifying the robustness of their neural networks against adversarial attacks.
- IBM: Employs α,β-CROWN, and DNNV in their research and development to ensure the security and reliability of AI models.
- Microsoft: Uses α,β-CROWN, and DNNV for formal verification of neural networks in various AI applications

SAT Solvers

- SAT (Boolean Satisfiability) solvers work with propositional logic to determine whether there exists an assignment of truth values to variables that make a given Boolean formula true.
- https://en.wikipedia.org/wiki/SAT_solver

SMT Solvers

- SMT (Satisfiability Modulo Theories) solvers extend SAT solvers by working with first-order logic and incorporating various theories such as arithmetic, bit-vectors, arrays, etc.
- https://en.wikipedia.org/wiki/Satisfiability_modulo_theories

Overview

- Background
- What is Software Engineering (SE) for Artificial Intelligence (AI)?
- What are the applications of SE in AI?
  - White-Box Testing of DNNs
  - Black-Box Testing of DNNs
  - Formal Verification of DNNs
  - How can we evaluate or write tests for LLMs

What are the applications of SE for AI?

White Box Testing of DNNs

A White-Box Testing for Deep Neural Networks Based on Neuron Coverage

- This paper introduces Test4Deep, an effective white-box testing DNN approach based on neuron coverage.
- It uses a differential testing framework to automatically verify inconsistent DNNs' behavior.
- Its contributions highlight Test4Deep's effectiveness in enhancing the reliability and robustness of deep neural networks through improved testing methodologies.

- https://web.eecs.umich.edu/~movaghar/WB TEsting TNNLS 2023 .pdf

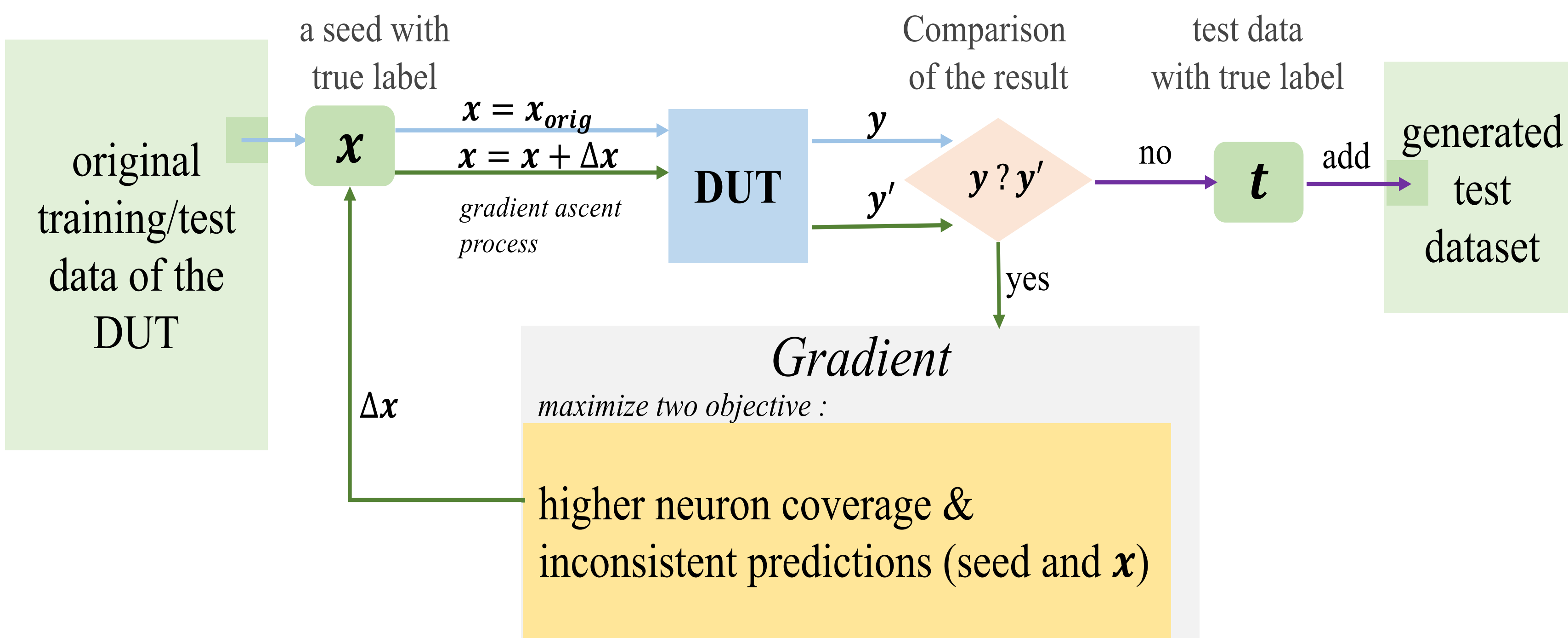


Fig. 1. Framework and workflow of Test4Deep.

TABLE I

SPECIFIC DESCRIPTIONS OF EXPERIMENTAL DATASETS AND DUTS [8]

Dataset	Dataset Description	DNN Description	Architecture	Neurons	Reported Acc.
MNIST	handwritten grayscale numerical images	LeNet family	LeNet-1[25]	52	98.30%
			LeNet-4[25]	148	98.90%
			LeNet-5[25]	268	99.05%
Imagenet	RGB images with classifications of various common things	Image classifiers	VGG-16[27]	14,888	92.60%
			VGG-19[27]	16,168	92.70%
			ResNet50[28]	94,059	96.43%
Contagio/ Virustotal	benign and malicious PDF documents described by 135 static features	Benign and malicious PDF detectors	PDF_D1: (200,200)	402	98.50%[a]
			PDF_D2: (200,200,200)	602	98.50%[a]
			PDF_D3: (200,200,200,200)	802	98.50%[a]

[a]: Same accuracy reported by [29].

TABLE II

RESULTS OF EFFECTIVENESS AND EFFICIENCY GENERATED BY TEST4DEEP (T4D.), DLFUZZ (DF.), AND DEEPXPLORE (DX.) IN NINE DUTS OF THREE DATASETS

DataSet	**DUT**	**Avg.NC (%)***			**# Cases***			**Time per Case† (s)**			**AD‡**			$Avg.\overline{l_2 dist.}$§		
		T4D.	*DF.*	*DX.*	*T4D.*	*DF.*	*DX.*	*T4D.*	*DF.*	*DX.*	*T4D.*	*DF.*	*DX.*	*T4D.*	*DF.*	*DX.*
MNIST	LeNet-1	69.73	50.09	46.29	186	116	109	1.01	2.19	2.38	491.3	11.99	402.04	0.018	0.018	1.65
	LeNet-4	72.65	70.96	71.66	190	140	125	0.92	2.33	2.41	502.12	11.89	291.68	0.018	0.018	1.06
	LeNet-5	78.43	76.93	72.19	195	168	123	0.90	2.27	2.38	490.98	11.75	382.52	0.017	0.018	0.98
Imagenet	VGG16	98.07	58.78	57.71	171	127	111	7.85	12.95	22.95	60.49	253.97	297.04	0.001	0.007	0.01
	VGG19	99.99	54.43	53.29	170	133	106	7.94	14.79	24.19	59.69	268.52	278.51	0.001	0.007	0.01
	ResNet50	99.99	76.81	77.11	177	152	118	18.28	42.94	23.08	45.59	189.09	290.47	0.001	0.007	0.01
Contagio/ Virustotal	PDF_D1	99.05	75.22	74.89	188	107	116	1.38	6.96	3.46	6.35	10.07	36.21	0.003	0.003	0.09
	PDF_D2	99.05	75.86	75.84	188	106	116	1.38	6.95	3.46	6.84	10.69	35.28	0.003	0.002	0.09
	PDF_D3	99.05	75.07	74.41	188	106	116	1.38	7.21	3.46	7.53	10.21	35.79	0.003	0.003	0.09

⋆: Average neuron coverage

*: The quantity of test cases generated by each test method.

†: Average time (measured in seconds) for each test case generation.

‡: Average diversity[8], i.e., average l_1 distance between generated test cases and seeds.

§: Average relative l_2 distance[30].

TABLE III

COMPARISONS OF QUANTITY, NEURON COVERAGE, AND GENERATION TIME OF EACH TEST CASE ACHIEVED BY TEST4DEEP (T4D.), DEEPXPLORE (DX.), AND DLFUZZ (DF.) WHICH USED TWO OPTIMAL NEURON SELECTION STRATEGIES IN LENET-4 AND RESNET50

DUT	Method	# Cases	Avg.NC (%)	Time per Case (s)
LeNet-4	*T4D.*	183	56.54	0.92
	DF.(23)⋆⋆	161	54.19	16.67
	DX.	141	48.39	6.59
ResNet50	*T4D.*	166	78.61	17.98
	*DF.(2)***	114	63.79	125.44
	DX.	178	52.89	19.31

⋆⋆: DLFuzz with combination strategy ($Strategy.2$ and $Strategy.3$)
**: DLFuzz with $Strategy.2$

What is the percentage improvement of average Neural Coverage (Ave.NC%) of Teast4Deep (T4D) to DLFUZZ (DF) in the two DDNs under Testing (DUTs) above?

Answer

Test4Deep (T4D) improved neuron coverage to DLFuzz (DF) in the two DUTs by 4.34% and 23.23%, respectively.

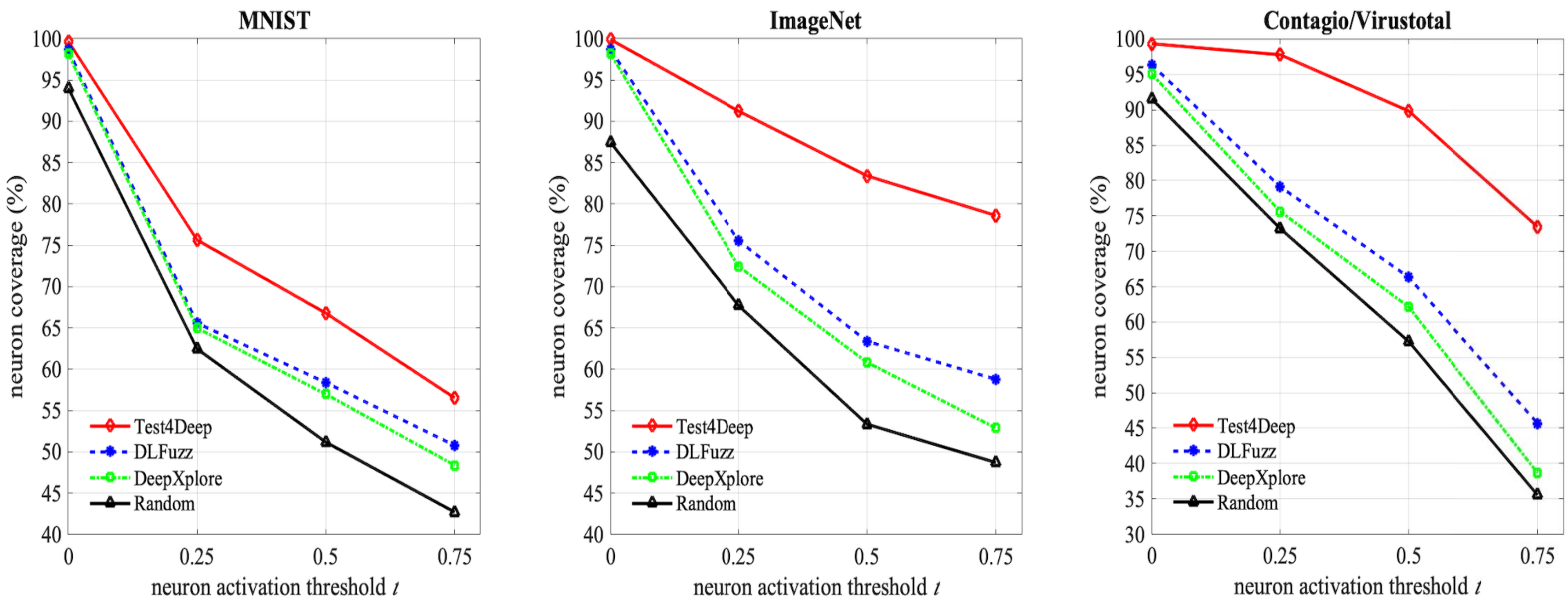


Fig. 2. Neuron coverage suppressed with growth of neuron activation threshold t. Neuron coverage by Test4Deep had the least decrease.

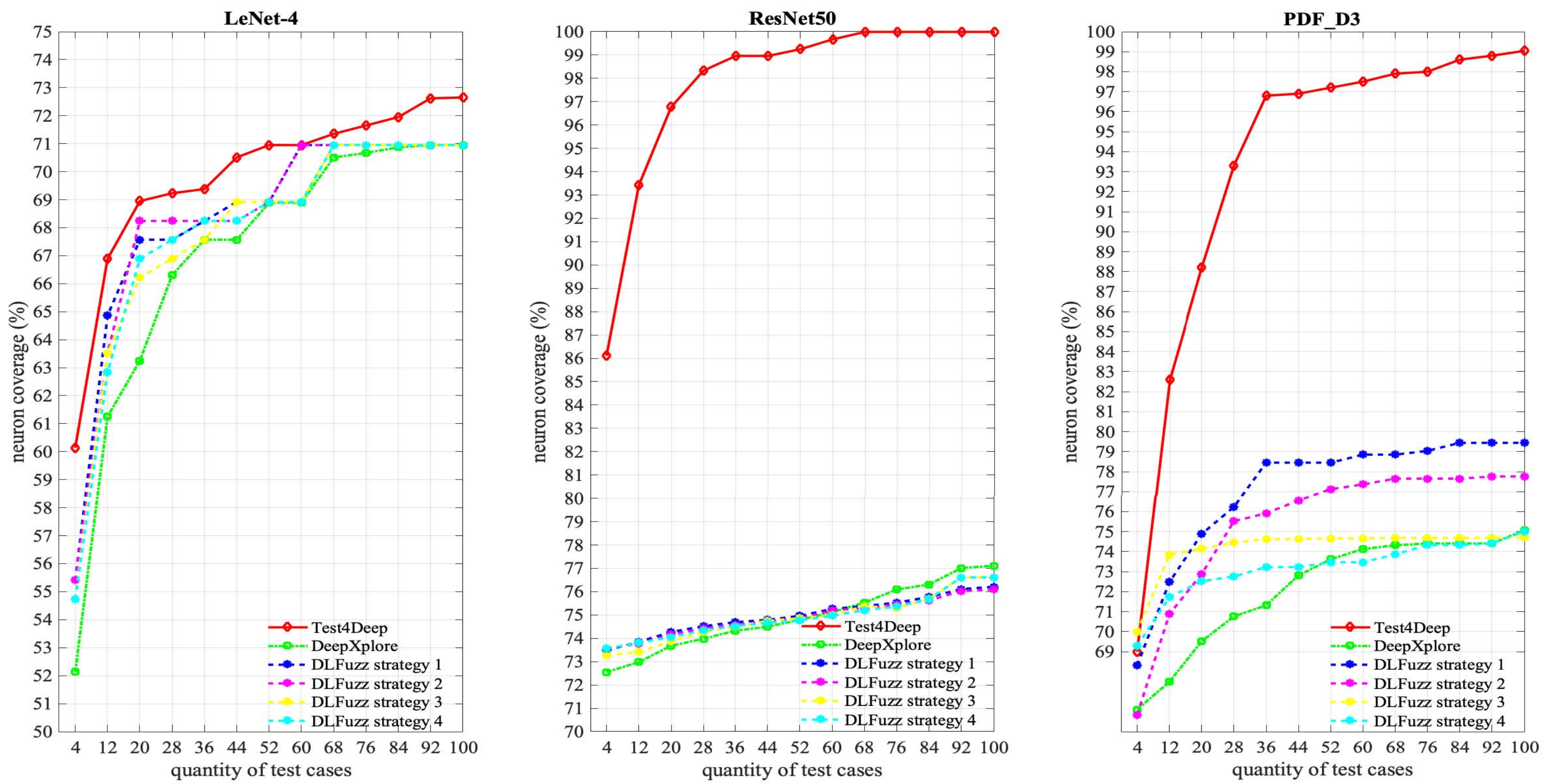


Fig. 3. Relationship between neuron coverage and the number of test cases which generated by Test4Deep, DeepXplore, and DLFuzz. DLFuzz with different neuron selection strategies were regarded as separate methods. Higher neuron coverage with the same number of seeds was better.

Neuron Coverage Improvement

- Test4Deep enhances neuron coverage by tracking and activating inactive neurons during testing.
- This approach ensures a more comprehensive examination of the neural network's internal logic.

Differential Testing Framework

- The paper introduces a differential testing framework that automatically verifies inconsistent behaviors in DNNs.
- This framework helps identify discrepancies between the original input and generated test data.

Efficiency and Effectiveness

- Compared to other state-of-the-art methods like DLFuzz and DeepXplore, Test4Deep achieves higher neuron coverage (by 32.87% and 35.69% respectively) while reducing testing time (by 58.37% and 53.24% respectively).
- It also generates more test cases with fewer perturbations.

Improving DNN Robustness

- By merging generated test cases and retraining the DNNs, Test4Deep can improve the accuracy and robustness of the networks.

Black Box Testing of DNNs

Black-Box Testing of Deep Neural Networks through Test Case Diversity

- This paper investigates diversity metrics as an alternative to white-box coverage criteria. Such metrics are required to be black-box and not rely on the execution and outputs of DNNs under test.
- It is shown that relying on the diversity of image features embedded in test input sets is a more reliable indicator than coverage criteria to effectively guide DNN testing.
- https://web.eecs.umich.edu/~movaghar/BBT DNN TSE 2023 - 1.pdf

Diversity Metrics

- The paper proposes the use of black-box diversity metrics to guide the testing of DNN models.
- These metrics help identify diverse test cases that can reveal different types of faults in the network.

What are faults?

- In this paper, faults refer to erroneous behaviors exhibited by deep neural networks (DNNs) that can lead to critical errors, especially in safety-critical systems.
- These faults can manifest as incorrect predictions or misclassifications when the DNN is presented with certain inputs.
- The paper investigates the use of black-box input diversity metrics to guide the testing of DNNs, aiming to detect these faults more effectively.

Geometric Diversity

- Geometric diversity is shown to be particularly effective in guiding the testing process.
- This metric helps identify similar mispredicted inputs caused by the same issues in the DNN model.

Coverage versus Faults

- The study finds that there is no direct correlation between traditional coverage metrics and the presence of faults in DNN models.
- Instead, diversity metrics provide a more reliable indicator for effective testing.

Main Result

- The paper demonstrates that relying on the diversity of image features embedded in test input sets is a more reliable method than coverage criteria for effectively guiding the testing of DNNs.

Feature-Guided Black-Box Safety Testing of Deep Neural Networks

- The paper focuses on image classifiers and proposes a feature-guided black-box approach to test the safety of deep neural networks that require no such knowledge.
- It uses object detection techniques such as SIFT (Scale Invariant Feature Transform) to extract features from an image.
- These features are converted into a mutable saliency distribution, where high probability is assigned to pixels that affect the composition of the image concerning the human visual system.

https://web.eecs.umich.edu/~movaghar/Black-Box Testing DNN 2018.pdf

Feature-Guided Approach

- The paper introduces a novel feature-guided approach for black-box safety testing of deep neural networks.
- This method leverages object detection techniques like SIFT (Scale Invariant Feature Transform) to extract features from images.

Adversarial Example Generation

- The authors formulate the crafting of adversarial examples as a two-player turn-based stochastic game.
- This game-theoretic approach helps in identifying minimal adversarial examples that can fool the neural network.

Safety Guarantees

- For Lipschitz networks, the paper identifies conditions that provide safety guarantees, ensuring that no adversarial examples exist under certain conditions.

Monte Carlo Tree Search

- The paper employs Monte Carlo tree search to explore the game state space gradually, searching for adversarial examples.
- This method is shown to be competitive with state-of-the-art white-box methods.

Application to Safety-Critical Systems

- The proposed method is demonstrated in safety-critical applications, such as traffic sign recognition in self-driving cars, highlighting its practical relevance and effectiveness.
- These contributions underscore the importance of robust and effective testing methodologies for ensuring the safety and reliability of deep neural networks.

Formal verification of DNNs

Deep Statistical Model Checking

- This paper presents several significant contributions to the field of neural network verification, particularly in the context of Markov Decision Processes (MDPs).
- Its contributions highlight the effectiveness of Deep Statistical Model Checking (DSMC) in providing a robust framework for the verification of neural networks within the context of MDPs.

https://web.eecs.umich.edu/~movaghar/Deep Statistical Model Checking 2020.pdf

Integration of Neural Networks and MDPs

- The paper introduces a method where a neural network (NN) is used to represent a policy that takes action decisions within an MDP.
- This integration results in a Markov chain, which can be analyzed using statistical model checking.

Scalable Verification Method

- The proposed method, termed Deep Statistical Model Checking (DSMC), is a scalable approach that extends traditional statistical model checking.
- It allows for the verification of systems incorporating neural networks by treating the NN as a determiner of the MDP.

Insight into NN Behavior

- DSMC provides deep insights into various aspects of neural network behavior, such as the safety risks induced by the NN, the performance of the NN compared to the optimal policy, and the impact of further training on the NN.

Light-Weight Verification

- The methodology described is a lightweight approach to checking the behavior of systems that incorporate neural networks, making it practical for real-world applications.

Book: Introduction to Neural Networks Verification

- This book provides a comprehensive overview of the methods and principles for verifying neural networks.
- It covers foundational concepts from formal verification and their adaptation to neural networks and deep learning.
- These contributions aim to provide formal guarantees on the safety, security, correctness, and robustness of neural networks, which are crucial for their deployment in real-world applications.

https://web.eecs.umich.edu/~movaghar/nnv_book-2021.pdf

Constraint-based techniques for DNN verification

- It discusses various techniques for constraint-based verification, including logic and satisfiability, encodings of neural networks, and neural theory solvers.
- These techniques involve formulating the verification problem as a set of constraints that the neural network must satisfy.
- These techniques are crucial for ensuring the reliability and safety of neural networks, especially in applications where correctness is critical.

Constraint Solving

- This involves encoding the neural network's behavior and the properties to be verified as mathematical constraints.
- These constraints are then solved using various techniques such as Satisfiability Modulo Theories (SMT) solvers.

Linear and Non-linear Constraints

- For linear regions of the network, constraints can be precisely encoded.
- However, non-linear behaviors, such as those introduced by activation functions like ReLU, require more complex handling.
- Techniques like piecewise linear approximation are often used to manage these non-linearities.

Abstraction Techniques

- Abstraction is used to simplify the verification problem by creating a more manageable representation of the neural network.
- This can involve over-approximating the network's behavior to ensure that all possible behaviors are considered, which can help in proving the properties of the network.

Combining Techniques

- Modern approaches often combine constraint solving with other techniques like abstraction and neuron splitting.
- Neuron splitting involves breaking down the verification problem into smaller subproblems, which can be solved more efficiently.

Scalability

- One of the main challenges addressed is the scalability of these techniques.
- As neural networks grow in size and complexity, the verification process becomes more computationally intensive.
- The book discusses various methods to improve the scalability of constraint-based verification.

Abstraction-based techniques for DNN verification

- The book explores abstraction techniques such as neural interval abstraction, zonotope abstraction, and polyhedron abstraction.
- Abstraction-based techniques for Deep Neural Network (DNN) verification are crucial for managing the complexity of verifying large and intricate networks.
- It also covers verification with abstract interpretation and abstract training of neural networks.

Over-approximation

- This technique involves creating a simplified version of the neural network that over-approximates its behavior.
- The idea is to ensure that if a property holds for the abstract (simplified) network, it will also hold for the original network.
- This makes the verification process more manageable.

Abstract Interpretation

- This method uses mathematical abstractions to represent sets of possible states of the neural network.
- By analyzing these abstract states, it is possible to infer properties about the network without having to examine every possible state explicitly.

Zonotope and Polyhedron Abstractions

- These are specific types of abstractions used to represent the possible values of neurons in the network.
- Zonotopes are geometric shapes that can efficiently represent linear transformations, while polyhedra can represent more complex relationships between neuron values.

Refinement

- When an over-approximation is too coarse and leads to spurious counterexamples (false positives), refinement techniques are used to make the abstraction more precise.
- This iterative process continues until the abstraction is accurate enough to verify the desired properties.

Scalability

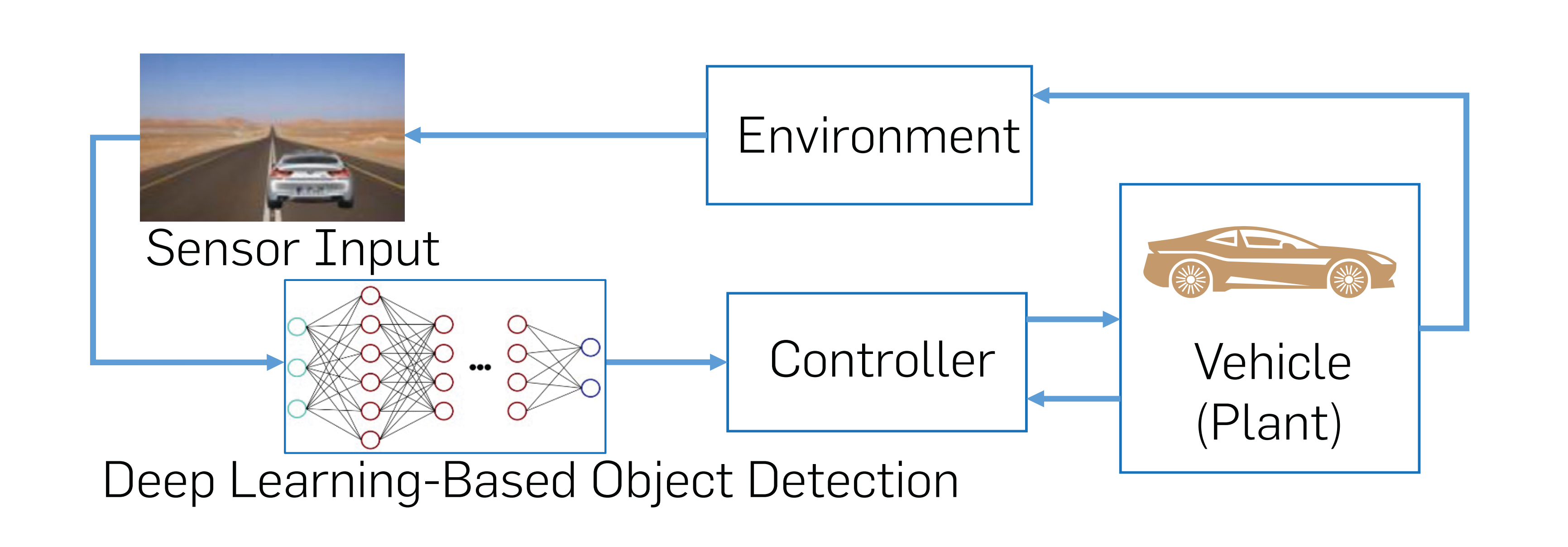
- One of the main advantages of abstraction-based techniques is their scalability.
- By reducing the complexity of the verification problem, these techniques make it feasible to verify larger and more complex neural networks.

Toward Verified Artificial Intelligence

- This paper discusses the goal of creating AI systems with strong, ideally provable, assurances of correctness based on mathematically specified requirements.
- The paper primarily focuses on the theoretical and methodological aspects of achieving Verified AI.
- It does not mention detailed case studies but discusses the challenges and principles needed to ensure that AI systems are verifiable and reliable.

https://web.eecs.umich.edu/~movaghar/Toward Verified Artificial Intelligence 2022.pdf

Figure 2. Example of a closed-loop cyber-physical system with machine-learning components (introduced in Dreossi et al.[5]).



Formal Modeling of Environment, Specification, and Learning Systems

- Formal modeling of the environment in which AI operates is usually a challenge.

- Defining precise and unambiguous specifications for AI behavior is not usually easy.

Modeling Learning Systems makes it difficult for the overall system to be formally verified.

Scalability and Design Correctness

- Developing scalable computational tools that can handle the complexity of AI systems is not usually practical.

- Designing correct AI systems from the outset is usually too difficult to be realized.

Applications: Safety-Critical Systems and Financial Systems

- Verified AI can be applied to systems where safety is paramount, such as autonomous vehicles, medical devices, and aerospace systems. Ensuring these systems operate correctly under all conditions is crucial.
- In financial technology, Verified AI can help create algorithms that are robust against errors and fraud, ensuring the integrity and reliability of financial transactions and trading systems.

Applications: Robotics, Cybersecurity, and Legal Systems

- Verified AI can be used in robotics to ensure that robots perform tasks accurately and safely, especially in environments where they interact closely with humans.
- AI systems designed to detect and respond to cyber threats can benefit from verification to ensure they correctly identify and mitigate threats without false positives or negatives.
- Verified AI can help ensure that AI systems comply with legal and regulatory requirements, reducing the risk of non-compliance and associated penalties.

How can we evaluate or write tests for LLMs

Large Language Models (LLMs)

- Large Language Models (LLMs) have transformed AI with their ability to process and generate human-like responses.
- These models can now tackle complex problems, but how do we know if they deliver reliable, actionable insights?
- The key lies in precise evaluation. Like any machine learning model, one should rigorously test LLMs to ensure accuracy, trustworthiness, and relevance.

Widespread Use of Benchmarks

- Standardized benchmarks like GLUE, SuperGLUE, and others are extensively used to evaluate LLMs.
- These benchmarks provide a common ground for comparing different models and are a staple in the research community.

- https://composio.dev/blog/llm-evaluation-guide/

Automated Testing Tools

- There are numerous automated tools and frameworks designed to streamline the evaluation process.
- These tools help in efficiently assessing various aspects of LLM performance, such as accuracy, fluency, and bias.
- https://www.lakera.ai/blog/large-language-model-evaluation

Human Evaluation

- Despite advances in automated testing, human evaluation remains crucial.
- Human judges assess the quality of the model's outputs based on criteria like coherence, relevance, and fluency, which are difficult to measure automatically.

Ethical and Bias Testing

- Evaluating LLMs for ethical considerations, such as bias and fairness, is a growing area of focus.
- Researchers are developing specific tests to identify and mitigate biases in LLM outputs.
- The field of LLM evaluation is continuously evolving, with ongoing research aimed at improving the robustness, fairness, and efficiency of evaluations.