大学共同利用機関法人 情報・システム研究機構
国立情報学研究所
National Institute of Informatics

eAI

Search-based Approaches to Enhancing Safety of Automated Driving Systems

Fuyuki ISHIKAWA
National Institute of Informatics, Japan
f-ishikawa@nii.ac.jp / https://research.nii.ac.jp/~f-ishikawa/en/

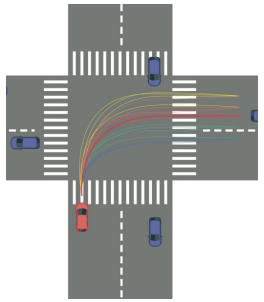
Fuyuki Ishikawa (NII, Japan)

- Test generation and failure analysis with CPS simulation
  - For automated driving controller (with Mazda) [ICST'20, ASE'21, TDSC'23, etc.]
  - For delivery robot services (with Panasonic) [ICECCS'23, CEC'23, ICST'24]
- Neural network repair for safe perception [SANER'22, SANER'23, ICST'23, GECCO'23]
- Responsibility-Sensitive Safety with black-box AI controller [FM'23]
  - With the Event-B formalism with refinement mechanisms
- Practice in Japanese industry
  - QA4AI: guidelines for AI quality management since 2019
    (just released a new version for LLM)
  - TopSE: 1-year course for engineers since 2005, including SE4AI/AI4SE

Overview

with MMSD Project (led by Ichiro Hasuo)

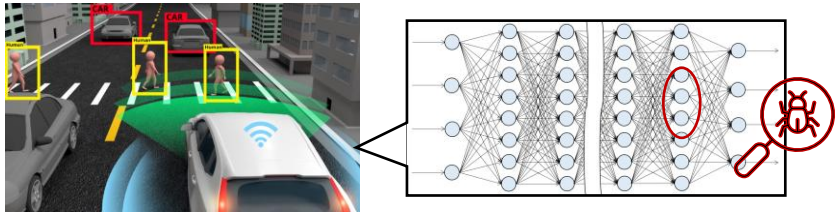
Testing/debugging of CPS [with Mazda]



(1) Search-based test generation and cause analysis for path-planning

"Engineerable AI" (eAI) Project

Risk-aware repair of neural networks [with Fujitsu, AISIN, etc.]



(2) Repair techniques with fault localization to identify "responsible neurons" for critical mis-recognitions

Research for Dependability of AI/CPS: Technical Approach

Traditional: detect, analyze, and repair program bugs

l.1	l.2	l.3	…	Result
✓	✓		…	PASS
✓		✓	…	FAIL
		✓	…	FAIL
…	…	…	…	…

Line	Fault impact
l.3	0.8
l.5	0.72
l.9	0.6
…	…

Complex programs

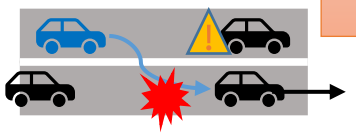
"Intelligent testing" (e.g., search-based)

Fault localization (e.g., spectrum-based)

Automated repair (e.g., search-based)

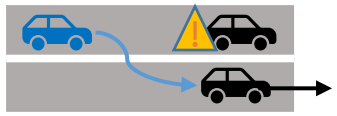
Transfer to AI/CPS, incl. Automotive Systems

Continuous
Fuzzy/open world



X	Y	Z	…	Danger
0.2	0.8	0.4	…	0.2
0.8	0.3	0.1	…	0.9
0.4	0.2	0.7	…	0.6
…	…	…	…	…

Factor	Safety impact
Large X	0.8
Small Y	0.72
Small Z	0.6
…	…



Driving systems

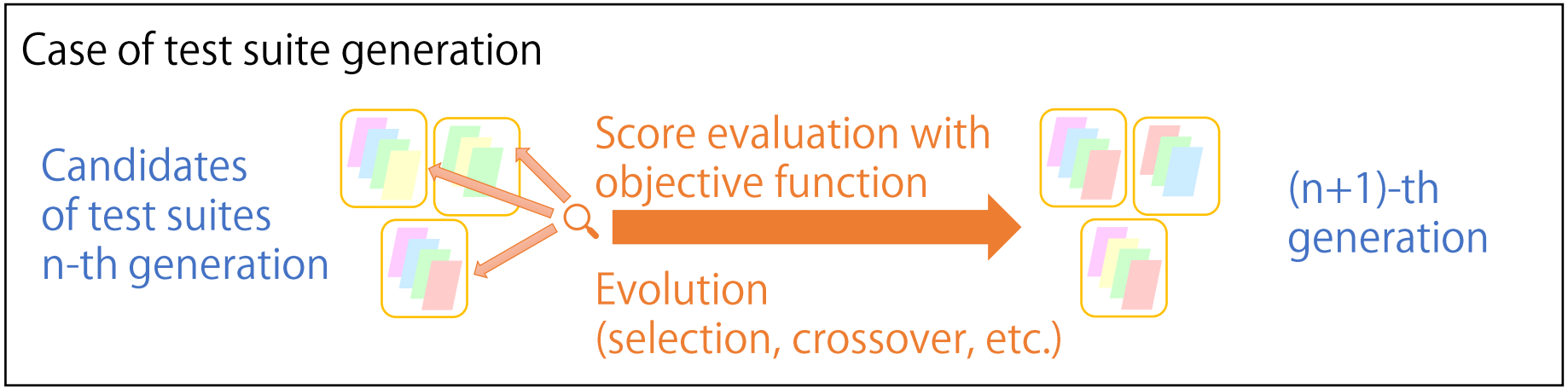
"Intelligent testing" for safety

Fault localization in continuous world

Automated repair of continuous behavior with multiple goals

Note: Search-based Software Engineering

- Tackle SE problems by (metaheuristics) optimization
  - Test input generation, program repair, configuration, …



[S. Ali et al., Systematic Review of the Application and Empirical Investigation of Search-Based Test Case Generation, 2010]

[https://code.fb.com/developer-tools/finding-and-fixing-software-bugs-automatically-with-sapfix-and-sapienz/]

Application in Facebook
(test input generation and program repair)

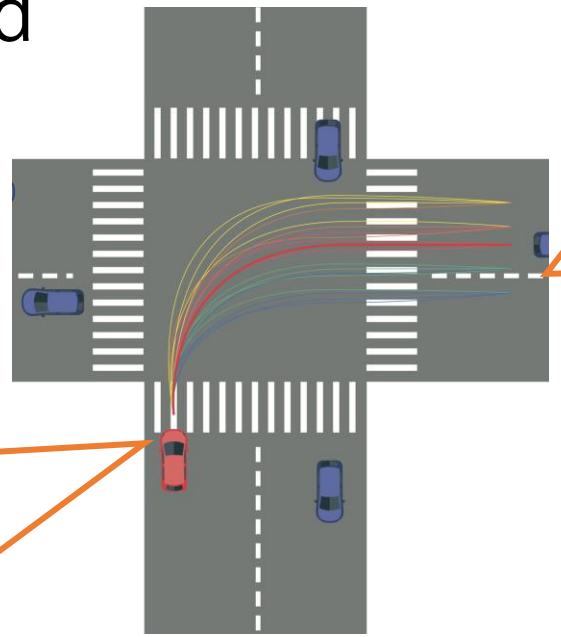
Target (1) Controller Function

- Path planning in autonomous driving
  - Short-term decision on steering and acceleration
  - Here, optimization-based

Dev. time

Weight design
in cost function

Distance to objects	10000
Abrupt steering	200
…	



Runtime

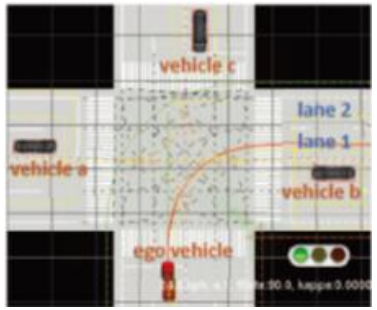
Cost of each path candidate

1. 150
2. 30000
3. …

Testing and debugging??

Research software provided by Mazda

Testing and Debugging for Path Planner



Ex. “right turn” case

Path planning component: periodically executed and decide the path to take, e.g., by optimization

Enormous space of test inputs (= simulation config.) such as position of other cars and signal timing

[with Mazda]

Search-based Test Generation

Generate only “avoidable crash”

Deal with priorities on multiple requirements

Define unique “coverage/adequacy” criteria

[ICST’20, ISSRE’20, ICECCS’20, ICST’21, ASE’21, ICST’22, RE’22, TOSEM’22]

Cause Analysis and Repair

Explain cause of crashes by fault localization

Repair multiple crash cases

[GECCO’20, ISSRE’20, TDSC’22]

Search-based Collision Detection?

We can search for and detect collision cases by using a "danger score" !

Search space

Simulator configuration
- Road shape
- Movement of pedestrians and other cars
- Initial location and velocity
- …

Objective function

$$danger(s^e_{t_i}, s^j_{t_i}) = \begin{cases} \vec{v}^{\,t_i}_{e|j} + K & \text{if } collision(s^e_{t_i}, s^j_{t_i}), \\ \dfrac{\vec{v}^{\,t_i}_{e|j}}{\|s^e_{t_i}.p, s^j_{t_i}.p\|^2} & \text{otherwise.} \end{cases}$$

Collision case: bad if the relative speed is high
Non-collision case: bad if the relative speed is high and the distance is small

Detected collisions are not due to the ego-car
- *Even "attacks" by other cars*
- *But "collision of our fault" is non-specifiable*

Example Work: Detection of "Avoidable" Collision

A collision is likely to our fault if it can be avoided by very small change of the weight design of our car

Search space

Simulator configuration
- Road shape
- Movement of pedestrians and other cars
- Initial location and velocity
- …

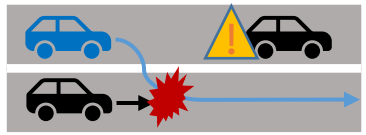
+ Weight repair

Objective function

1. The weight repair largely changes the "danger score" (especially, changing a collision case into a non-collision case)
2. The weight repair is small

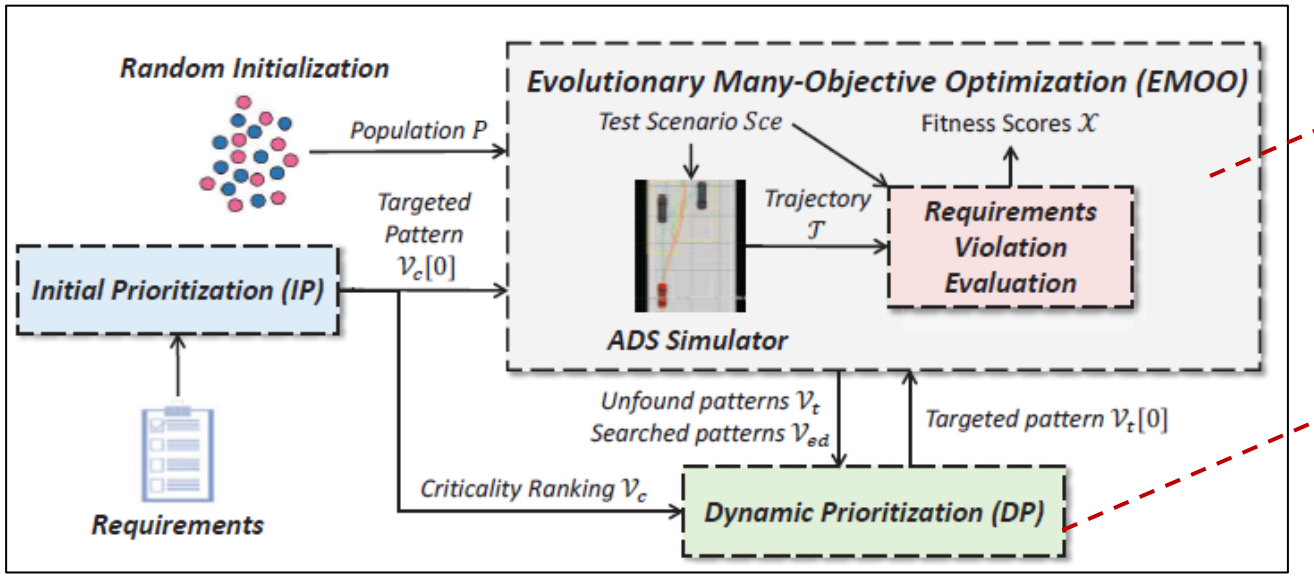
Note: scenarios (e.g., overtaking) can be specifiable by an objective or the initial setting

We could generate collision cases that need analysis/fix



[Calo+, Generating Avoidable Collision Scenarios for Testing Autonomous Driving Systems, ICST'20]

Example Work: Detecting Multiple Violation Patterns

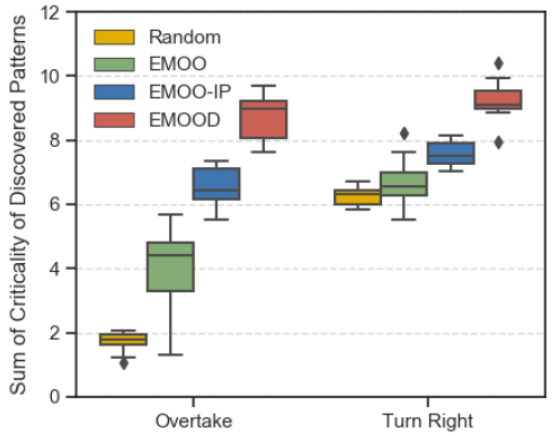


Optimization to generate scenarios given a goal to find a specific violation pattern in the multiple requirements

Dynamical goal update of search:

- search for worse violation patterns by evolving the found ones
- decrease priorities for violation patterns that turned out difficult to occur e.g., lateral acceleration in take over

Requirements with multiple aspects and metrics such as safety, comfort, rule conformance, etc. with different levels of significance

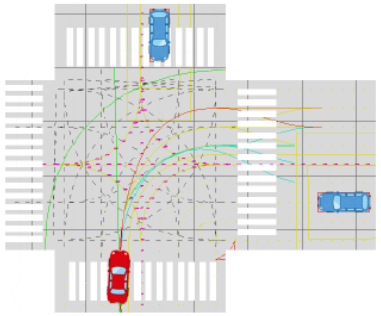


Detected scenarios of more diverse and critical violation patterns

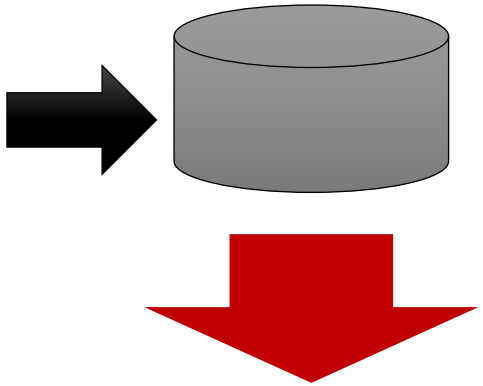
[Luo+, Targeting Requirements Violations of Automated Driving Systems by Dynamic Evolutionary Search, ASE'21]

Example Work: Analyzing Causes of Crash

Crash found in a right-turn scenario



(a) $\mathcal{S}_{RightTurn}$



Running variations of scenarios for a detected crash scenario by slightly changing behavior configurations

Extracting how each factor affects or not the crash, or "danger score" to be more granular

Example of insights

Danger score increases when our car is configured to have more penalty on too large lateral acceleration

Steering and braking amounts increases when the danger score is high

→ This time, the crash could be avoided by flexibly making a large steering

→ Discuss the design parameter on the penalty of large steering!

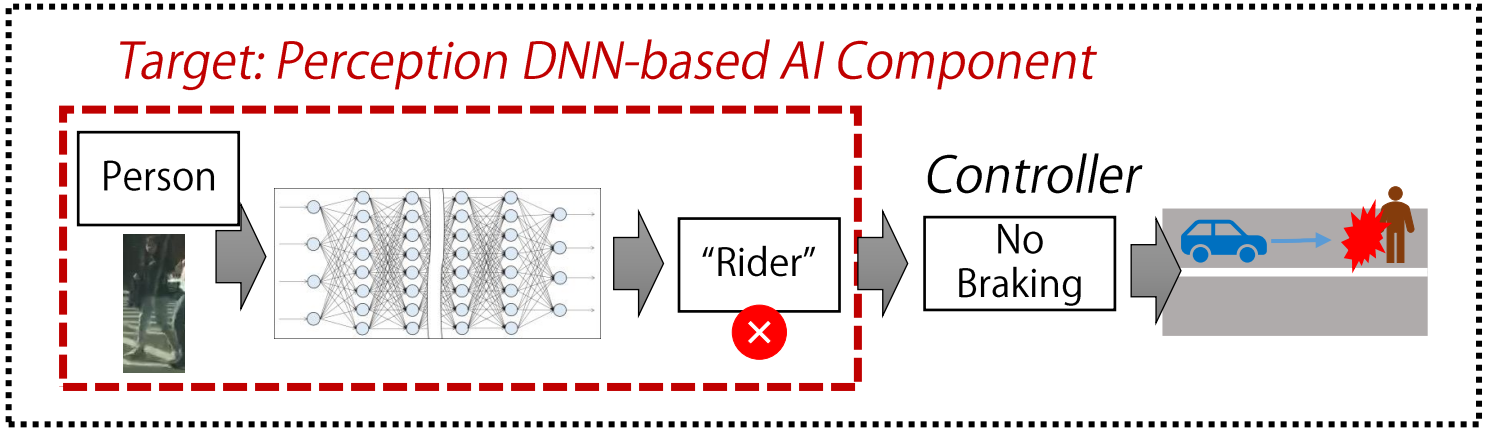
[Zhang+, An Incremental Approach for Understanding Collision Avoidance of an Industrial Path Planner, TDSC'22]

Target (2) Perception Function

- DNN-based perception in autonomous driving

  (DNN: deep neural networks)

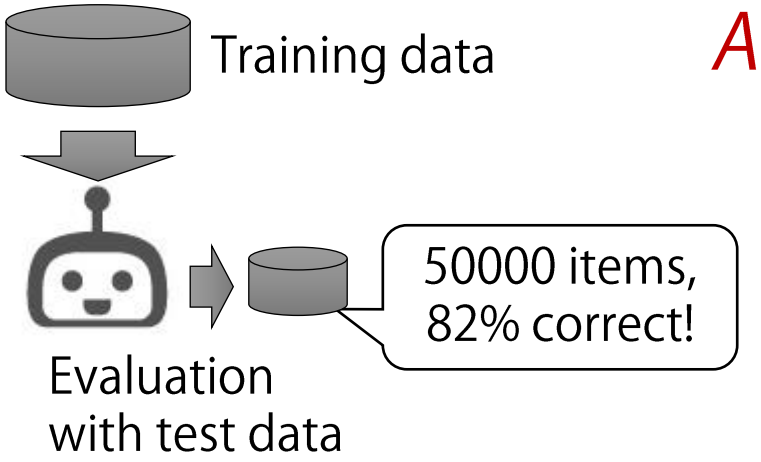
  - Heavily affects the control decision
  - Sometimes difficult to complement with other mechanisms



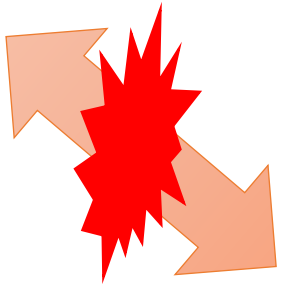
Testing and debugging??

Note: Traditional Safety vs. Traditional AI Performance

Traditional AI Quality



Construction as a whole
Average over the dataset total



Individual evaluation
Iterative improvement

Traditional Safety

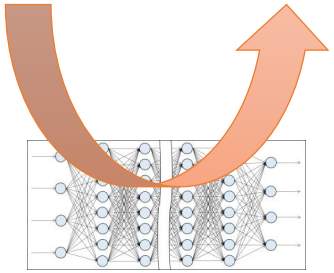
Faults	Hazards	Evaluation/Countermeasure
The wire to the brake lamp broken	Hit by the following car when sudden/large braking	Risk unacceptable → Duplicate the wire
…	…	…

Benchmark with Risk Analysis on Perception Component

ID	AI Error Modes	Scenes	Hazards	Risk Levels	AI Eval.	Acceptable
001	Misclassify: pedestrian to rider	Pedestrian in front of ego-car	No braking, hit the pedestrian	5	Error 4%	Y
002	Misclassify: rider to pedestrian	Closely following car	Unnecessary braking, hit at the rear side	3	Error 35%	N
…	…	…	…	…	…	…

1. Analysis by safety experts

2. Risk-aware fine-grained AI evaluation



3. Risk-aware AI improvement

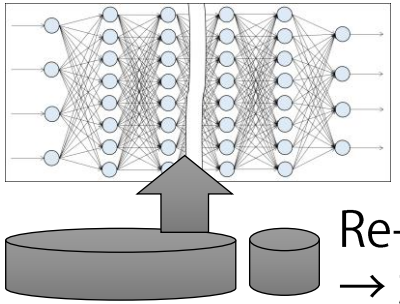
Example of Setting:
- Improve the majority over 16 metrics
- Improve to preserve and satisfy more of 10 constraints

Defined by an industrial working group

Technical Approach for Performance Alignment

Before

Update to adjust performance after training, tests, operation?



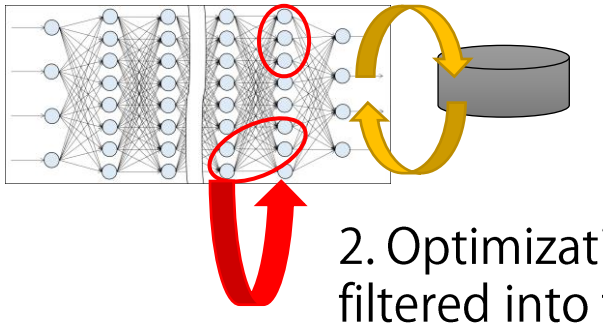
"Changing anything changes everything"

Re-training with additional data
→ **Shuffling parameters of over millions**

Uncertain/uncontrollable outcome

Unexpected regression (degradation)

Our Approach



1. Analysis causes of critical errors

2. Optimization of parameters filtered into tens

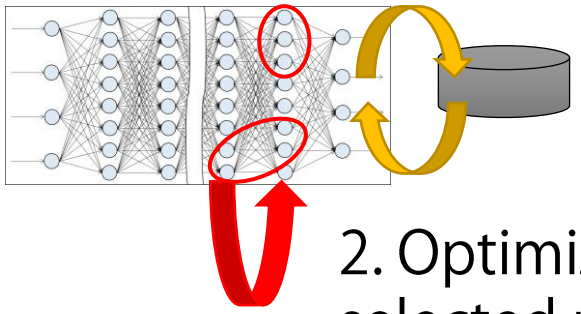
By limiting the fix targets,

- Effective for critical errors
- Control regression
- Multiple solutions obtained for trade-off analysis

Analogy with fault localization and automated repair on program code

Baseline of DNN Repair

Arachne (v1) [Sohn+, 2019]



1, analyze impact of parameters for failed inputs, i.e., target of repair

By looking at both of
- Gradient of loss
- Output value

2. Optimize only the selected parameters

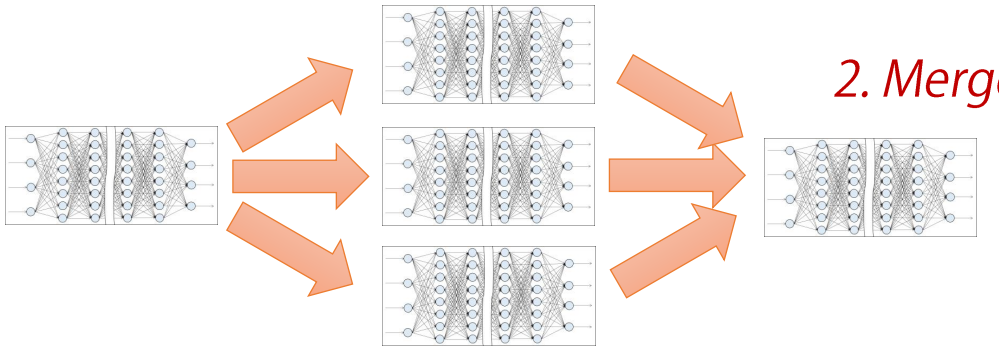
Objective function is integration of
- How much we could fix the model for the target inputs (originally failed)
- How much we had degradation for other succeeded inputs (originally passed)

(there can be other approaches, e.g., applying spectrum-based fault localization as for program code)

Example Work: Repair for Prioritized Multiple Targets (1)

- Specific focus: priorities over multiple targets
- Technical approach
  - Make a fix for each error type, then merge
    - e.g., “pedestrian -> rider” error

1. Create a fix candidate for each error type



2. Merge them by considering priorities/risks

Key point: fault localization works better for each error type, not mixture of all the errors

[Li Calsei+, Distributed Repair of Deep Neural Networks, ICST’23]

Example Work: Repair for Prioritized Multiple Targets (2)

- Integrated safety score as evaluation criteria

$$\begin{aligned} \mathrm{REM} = & -\mathrm{mw} \cdot \left(\begin{array}{l} \mathrm{rw}_1 \cdot (MR^{ped} + MR^{car,rider}) + \\ \mathrm{rw}_2 \cdot (MR^{rider} + MR^{car,truck} + MR^{bicy}) + \\ \mathrm{rw}_3 \cdot (MR^{ped,rider} + MR^{rider,ped} + \\ MR^{motor,ped}) \end{array} \right) \\ & +\mathrm{aw} \cdot \left(\begin{array}{l} \mathrm{rw}_4 \cdot AC^{ped} + \mathrm{rw}_5 \cdot (AC^{car} + AC^{bicy}) + \\ \mathrm{rw}_6 \cdot AC^{rider} \end{array} \right) \end{aligned}$$

- Improvement by performance controllability of our method

ENETB7	Arachne	Arachne_REM	DistrRep	SplitTrain_NW	SplitTrain_W	FullTrain_NW	FullTrain_W
AVG DELTA	-1.13	1.40	7.53	0.29	0.13	0.83	-0.22
MAX DELTA	0.99	2.90	8.38	1.16	1.32	1.33	0.59
MIN DELTA	-2.61	0.34	6.75	-1.12	-1.18	0.24	-1,55

Baseline repair methods (with/without awareness of weights for risk score)

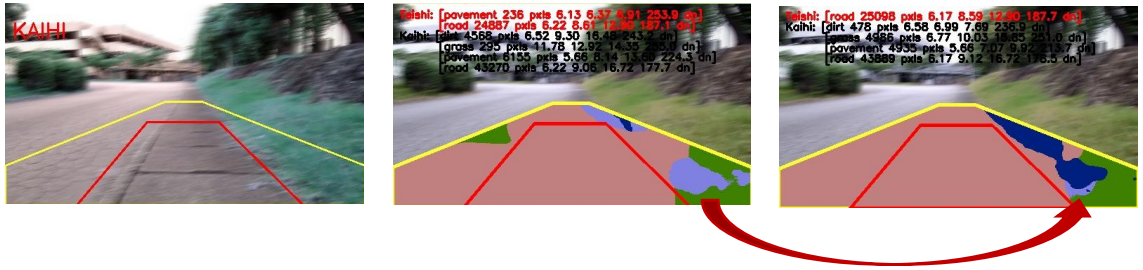
Re-training method (with different configurations)

[Li Calsei+, Distributed Repair of Deep Neural Networks, ICST'23]

Application Studies

- Fixing perception errors for personal mobility

Research prototype in AISIN



- Updating AI while avoiding “regressions”
  - Or, catastrophic forgetting

In Fujitsu

Further Opportunities with Power of LLM??

- Power of LLM (large language models) for
  - Analysis based on commonsense
    - e.g., We can ask LLM how likely the detected crash case occures
  - Interaction for obtaining preferences and validation from human
    - e.g., We can use LLM to explain the results and ask for a decision regarding the trade-off over different recognition targets
  - Exploration with the open terminology
    - e.g., We can ask LLM to think of difficult situations for our AI
  - … (rather than asking to do anything!)

Example Work: Explorative Testing of Image Recognition

- Search of "systematic faults" of perception AI
  - The previous work on DNN repair assumed human annotations on the dataset, e.g., "person close to our car", "in snowy day"
  - The real world contains much more diversity and edge cases

➡ "Likely difficult" object/background attributes proposed by GPT + AI evaluation with Stable Diffusion

e.g., fire truck around breakwater (surrounded by water, in more general context), is likely to be missed,
(obtained by exploring graph of concepts, about the background in this case)

[Torikoshi+, AdaSniper: An Adaptive Automated Approach for Systematic Error Detection in Image Recognition Models, FOSE'24]
[Yokoyama+, Investigating the Applicability of Image Generative Models to Weakness Detection Tasks, SES'24]

Summary

- Increasing focus on AI/CPS in the real world
  - Fuzzy requirements and open environments
  - Optimization/learning-based complex implementation
- Focus today: search/optimization-based approach
  - For detecting, analyzing, and addressing “faults” as causes of failures
- ➧ Beyond: Towards hybrid-approaches
  - We also use formal verification such as theorem proving
  - Combining LLM is adding different types of capabilities