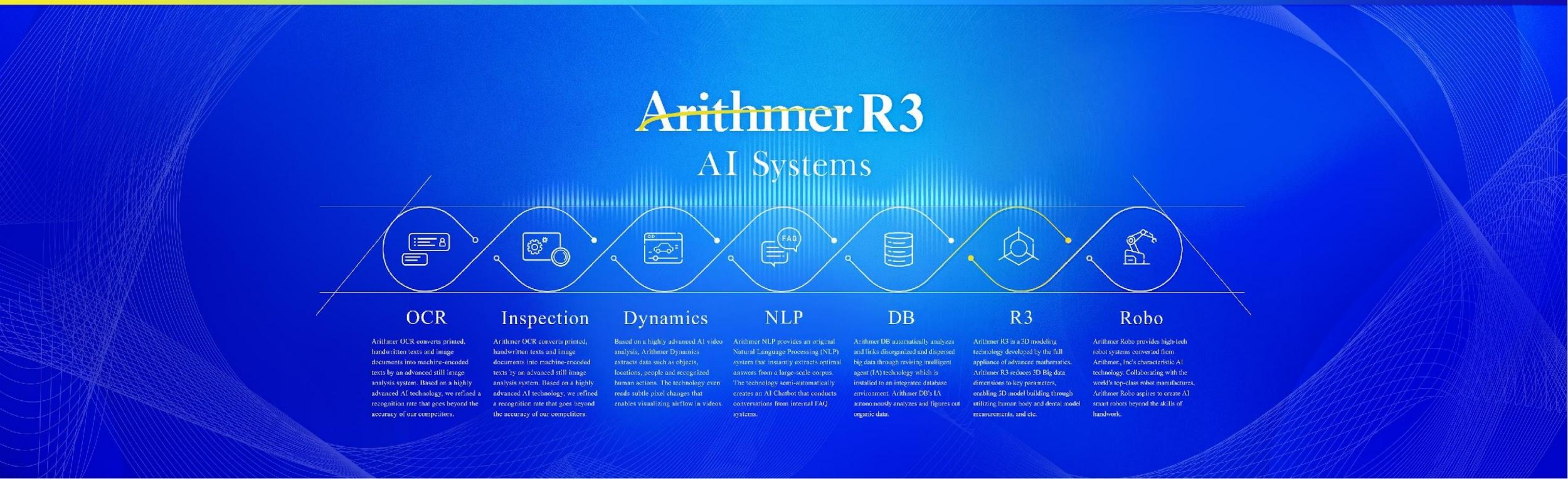


OCR

Arithmer OCR converts printed, handwritten texts and image documents into machine-encoded texts by an advanced still image analysis system. Based on a highly advanced AI technology, we refined a recognition rate that goes beyond the accuracy of our competitors.

Inspection

Arithmer OCR converts printed, handwritten texts and image documents into machine-encoded texts by an advanced still image analysis system. Based on a highly advanced AI technology, we refined a recognition rate that goes beyond the accuracy of our competitors.

Dynamics

Based on a highly advanced AI video analysis, Arithmer Dynamics extracts data such as objects, locations, people and recognized human actions. The technology even reads subtle pixel changes that enables visualizing airflow in videos.

NLP

Arithmer NLP provides an original Natural Language Processing (NLP) system that instantly extracts optimal answers from a large-scale corpus. The technology semi-automatically creates an AI Chatbot that conducts conversations from internal FAQ systems.

DB

Arithmer DB automatically analyzes and links disorganized and dispersed big data through revising intelligent agent (IA) technology which is installed to an integrated database environment. Arithmer DB's IA autonomously analyzes and figures out organic data.

R3

Arithmer R3 is a 3D modeling technology developed by the full appliance of advanced mathematics. Arithmer R3 reduces 3D Big data dimensions to key parameters, enabling 3D model building through utilizing human body and dental model measurements, and etc.

Robo

Arithmer Robo provides high-tech robot systems converted from Arithmer., Inc's characteristic AI technology. Collaborating with the world's top-class robot manufactures, Arithmer Robo aspires to create AI smart robots beyond the skills of handwork.

Tests for AI model

2021/09/02
Masaaki Uesaka

Arithmer

Today’s topic

I would like to introduce the testing of machine learning models based on the contents of the book [Sato et al, 2021].

But as a result of surveying papers related to the contents of the book, **the contents have expanded beyond the book.**

I will skip the detail in my talk and please read this slide and search the references if you are interested in this topic.

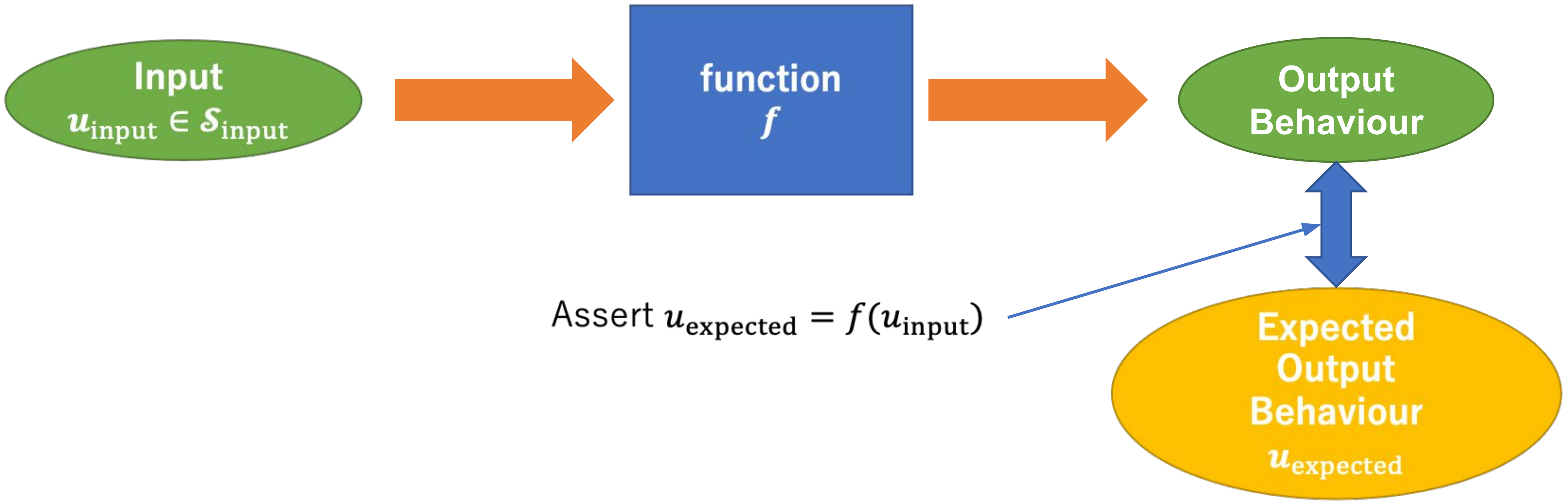
We focus on the following four topics:

- Metamorphic test
- Neuron coverage
- Adversarial minimum distortion (最大安全半径)
- Formal verification (網羅検証)

We can access the code in [Sato et al, 2021] by purchasing it.
(based on keras & tensorflow. not pytorch)

[Sato et al, 2021] 佐藤直人･小川秀人･來間啓伸･明神智之 (2021).『AIソフトウェアのテスト―答えのない答え合わせ [4つの手法]』リックテレコム

Problems in testing of machine learning


Input
$u_{\text{input}} \in S_{\text{input}}$
function
f
Output Behaviour
Assert $u_{\text{expected}} = f(u_{\text{input}})$
Expected Output Behaviour
u_{expected}

In ordinary software, (unit) test is done by **checking whether the output of function in the specified input is equal to expected one.**

Arithmer

Some techniques for software tests

Equivalence class test

- Divide the input spaces to equivalence classes in which the expected outputs are same.
- Take a representative from each equivalence class
- Create the test case for the representatives.

Assert: $f(2) = 1,\ \ f(25) = 2,\ \ f(324) = 3,\ \ f(5678) = 4$

e.g. The function which returns the number of digits in decimal notation

Input value	Expected output
0～9	1
10～99	2
100～999	3
1000～9999	4

Boundary value test

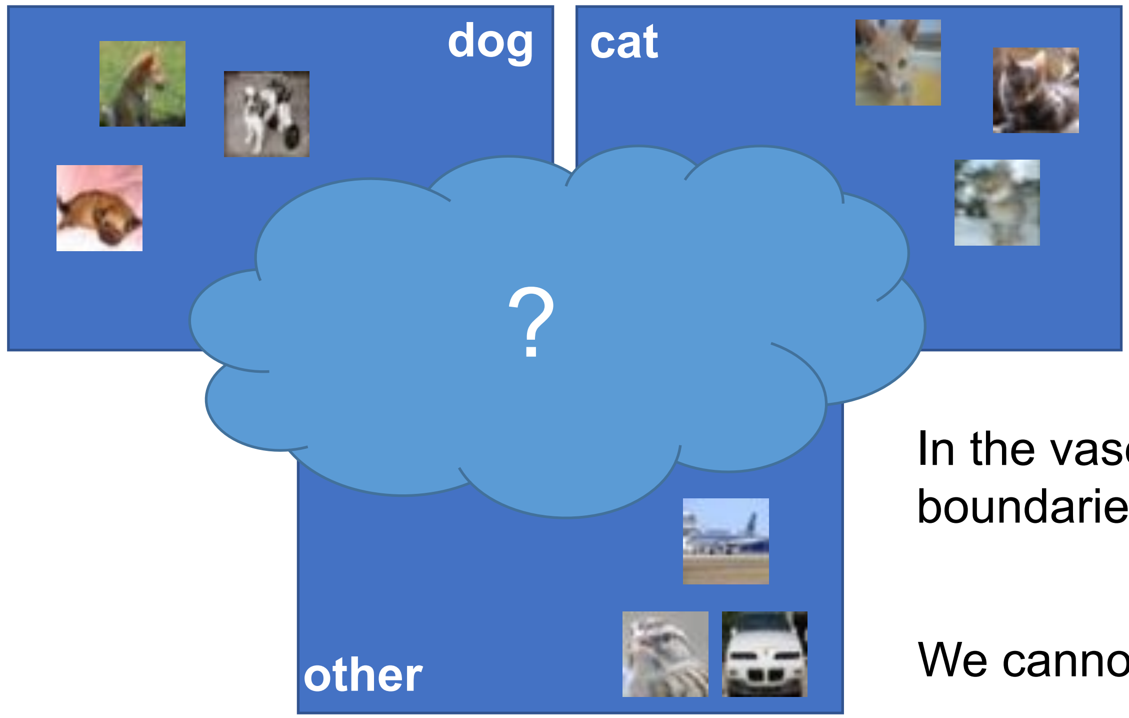
- Divide the input spaces to equivalence classes
- Take the nearest values to neighboring classes for each class
- Create the test cases for these boundary values
- This is applicable if the input space is ordered.

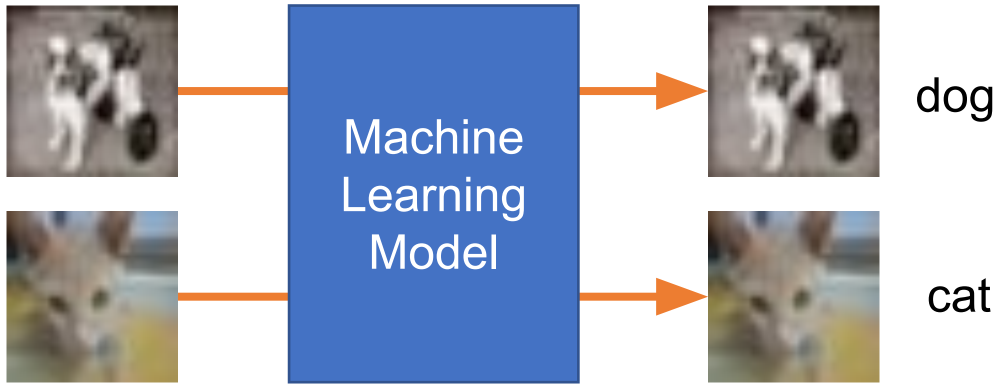
Assert: $f(9) = 1,\ \ f(10) = 2,\ \ f(99) = 2,\ \ f(100) = 3,\ \ \cdots$

Problems in test for machine learning model

Can we apply the software test techniques for machine learning model?

We do not know the entire boundary of the dog images and cat images.





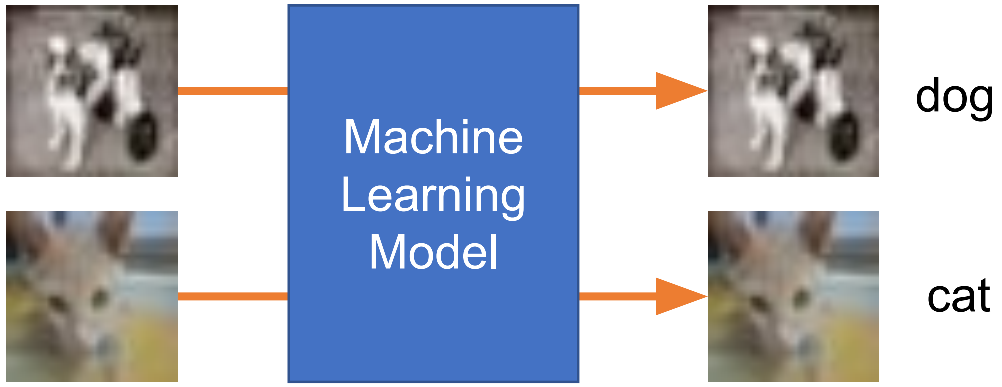
In the vase space of images, we cannot determine the boundaries of each category in advance.

We cannot apply the boundary test.

Problems in test for machine learning model

Can we apply the software test techniques for machine learning model?

- The expected behavior of the classification depends on the human judgements.
- Sometimes **output of the model differs from the human judgements.**



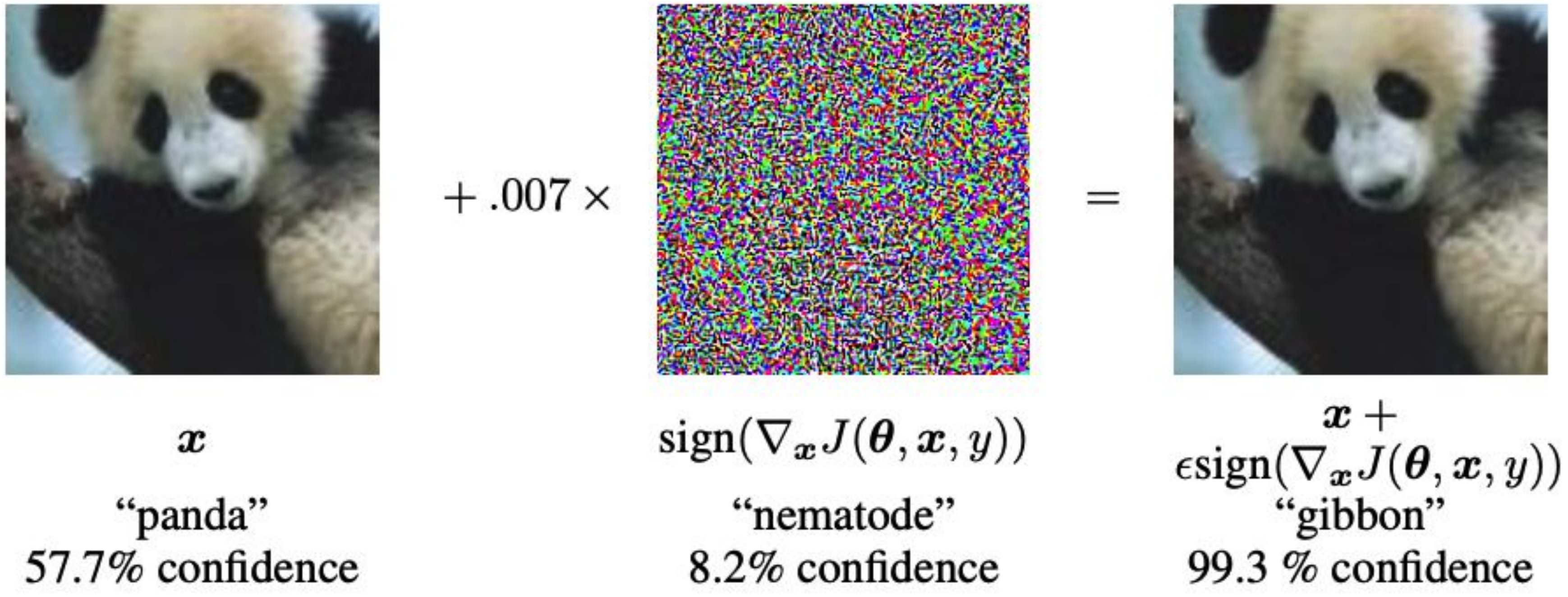
Deterministic assertion test is not applicable
At most, we can make the statistical one:

Assert that the probability that the labeling of human and ML model coincides is high enough to make it practical.

→ Precision, Recall, F1 score,…

Problems in test for machine learning model

If we admit the probabilistic assertion, we encounter the problem of **adversarial attack**.


$+ .007 \times$
$=$
$\boldsymbol{x}$
"panda"
57.7% confidence
$\mathrm{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"nematode"
8.2% confidence
$\boldsymbol{x} + \epsilon \mathrm{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$
"gibbon"
99.3 % confidence

[Goodfellow et al., 2015]

ML models are easily fooled by small perturbations unrecognizable to humans.

Methods of Adversarial Attacks

Fast Gradient Sign Method [Goodfellow et al, 2015]

Corresponding label of *x*

$$x_{\text{adv}} = x + \varepsilon \, \text{sign} \left(\nabla_x J(x, y; \theta) \right)$$

Cost function of training

Carlini-Wagner Method [Carlini & Wagner, 2017]

Trained model

$$x_{\text{adv}} = x + \delta$$

$$\delta = \underset{\delta}{\text{argmin}} \, [\|\delta\|_{L^p} + cL(f(x+\delta), y)$$

Measure of correspondence

Projected Gradient Descent Method [Madry et al, 2018]

$$x_{\text{adv}} = x + \delta$$

$$\delta = \underset{\delta}{\text{argmin}} \, L(f(x+\delta), y) \ \text{s.t.} \quad \|\delta\|_{L^p} \le \varepsilon$$

[Goodfellow et al., 2015] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. ICLR.
[Carlini & Wagner, 2017] Carlini, N., & Wagner, D. (2017). Towards Evaluating the Robustness of Neural Networks. 2017 IEEE Symposium on Security and Privacy (SP), 39–57.
[Madry et al, 2018] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards Deep Learning Models Resistant to Adversarial Attacks. ICLR

Intermediate Summary

In ML model, the ordinary software test is difficult to apply because

(1) Tests cannot be deterministic; **they can only be evaluated statistically**.

(2) In the first place, **it is not possible to identify the equivalence classes and their boundaries** in advance in a very high dimensional data space.

(3) Even if we accept statistical evaluation, **the existence of adversarial attacks does not guarantee the robustness of the inference**.

These are due to **the inductive nature** of the ML model construction.

- There are no clear functional requirements. (**The expected output for the input cannot be perfectly assumed.**)

- Because it learns only from a limited amount of data (though large), **it is not perfectly clear whether the trained model will make valid inferences in other cases** (even if generalization performance is improved).

Arithmer

What is the metamorphic test?

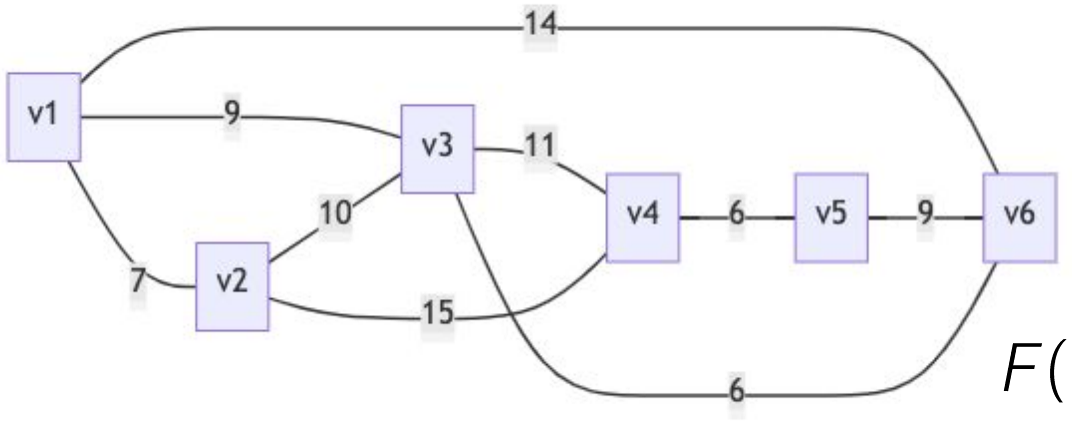
Metamorphic test itself is proposed in [Chen et al, 1998].
This technique is to uncover software error **in the production phase and in the absence of test oracle.**

Example: Shortest path problem on the weighted graph

$G = (V, E, d)$ undirected weighted graph with vertices *V* and edges *E* with distance function *d*

Task: For given $v_1, v_2 \in V$, Find the shortest path $v_1 \to w_1 \to w_2 \to \cdots \to w_n \to v_2$ connecting from v_1 to v_2 and its total distance $d_{\min}$:

$$F(v_1, v_2) = (v_1 \to w_1 \to w_2 \to \cdots \to w_n \to v_2, d_{\min})$$



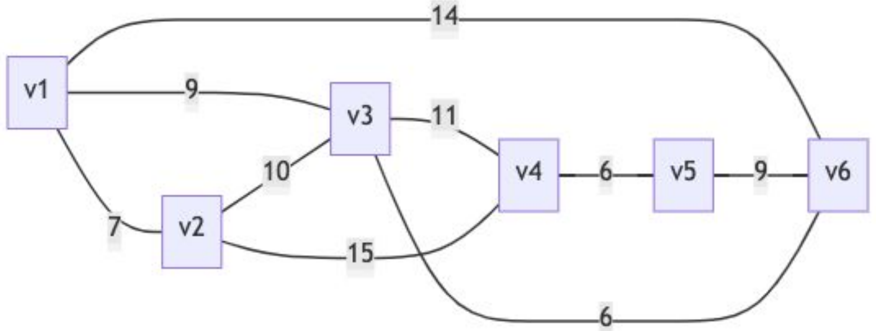
$$F(v_1, v_5) = (v_1 \to v_6 \to v_5, 23$$

[Chen et al, 1998] Chen, T. Y., Cheung, S. C., & Yiu, S. M. (2020). Metamorphic Testing: A New Approach for Generating Next Test Cases. ArXiv:2002.12543

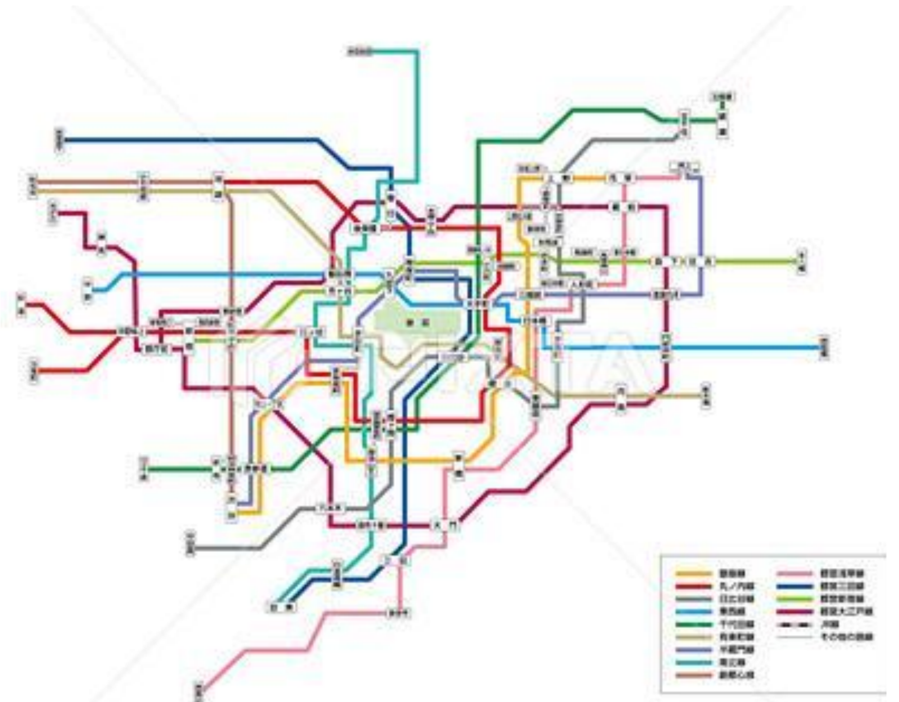
What is the metamorphic test?

- Of course, if the graph G is small, we can find manually the shortest path. So we can make the test case manually.
- But in production phase, G is **generally large**. **We cannot construct the test case for such large graph!!**
- How can we test for real-world input in the production phase?

→**Metamorphic testing!!!**



$$F(v_1, v_5) = (v_1 \to v_6 \to v_5, 23)$$



[Chen et al, 1998] Chen, T. Y., Cheung, S. C., & Yiu, S. M. (2020). Metamorphic Testing: A New Approach for Generating Next Test Cases. ArXiv:2002.12543

Examples of metamorphic test

Suppose that

$$F(v_1, v_2) = (v_1 \to w_1 \to w_2 \to \cdots \to w_n \to v_2, d_{\min})$$

Then the following statements should also be valid:

Assertion 1: If we exchange v_1 and v_2, the resulting path become reversed and minimal distance is unchanged:

$$F(v_2, v_1) = (v_2 \to w_n \to w_{n-1} \to \cdots \to w_1 \to v_1, d_{\min})$$

Assertion 2: For any vertex w_k in the shortest path, the shortest path connecting v_1 and w_k is $v_1 \to w_1 \to \cdots \to w_k$ and that connecting w_k and v_2 is $w_k \to w_{k+1} \to \cdots \to v_2$ (subproblem optimality):

$$F(v_1, w_k) = (v_1 \to w_1 \to w_2 \to \cdots \to w_k, d'_{\min})$$
$$F(w_k, v_2) = (w_k \to w_{k+1} \to \cdots \to w_n \to v_2, d''_{\min})$$
$$d_{\min} = d'_{\min} + d''_{\min}$$

Assertion 3: The shortest path connecting w_1 and w_n, which are next to v_1 and v_2 respectively, is $w_1 \to w_2 \to \cdots \to w_n$, the partial path of $v_1 \to w_1 \to \cdots \to w_n \to v_2$:

$$F(w_1, w_2) = (w_1 \to w_1 \to w_2 \to \cdots \to w_n, d'_{\min})$$
$$d_{\min} = d(v_1, w_1) + d'_{\min} + d(w_n, v_2)$$

[Chen et al, 1998] Chen, T. Y., Cheung, S. C., & Yiu, S. M. (2020). Metamorphic Testing: A New Approach for Generating Next Test Cases. ArXiv:2002.12543

Arithmer

Examples of metamorphic test

$$F(v_1, v_2) = (v_1 \to w_1 \to w_2 \to \cdots \to w_n \to v_2, d_{\min})$$

Assertion 1 $F(v_2, v_1) = (v_2 \to w_n \to w_{n-1} \to \cdots \to w_1 \to v_1, d_{\min})$

Assertion 2 $F(v_1, w_k) = (v_1 \to w_1 \to w_2 \to \cdots \to w_k, d'_{\min})$

$F(w_k, v_2) = (w_k \to w_{k+1} \to \cdots \to w_n \to v_2, d''_{\min})$

$d_{\min} = d'_{\min} + d''_{\min}$

Assertion 3 $F(w_1, w_2) = (w_1 \to w_1 \to w_2 \to \cdots \to w_n, d'_{\min})$

$d_{\min} = d(v_1, w_1) + d'_{\min} + d(w_n, v_2)$

These assertions are independent of the expected output of $F(v_1, v_2)$, which is nearly unknown.
So, we can test these assertion for some output of the function F, which is even not tested.

[Chen et al, 1998] Chen, T. Y., Cheung, S. C., & Yiu, S. M. (2020). Metamorphic Testing: A New Approach for Generating Next Test Cases. ArXiv:2002.12543

Formulation

$f: S_{\mathrm{in}} \to S_{\mathrm{out}}$ the function to be tested

Ordinary test

- $x \in S_{\mathrm{in}}$: input
- $\hat{y} \in S_{\mathrm{out}}$: expected output of f with input x

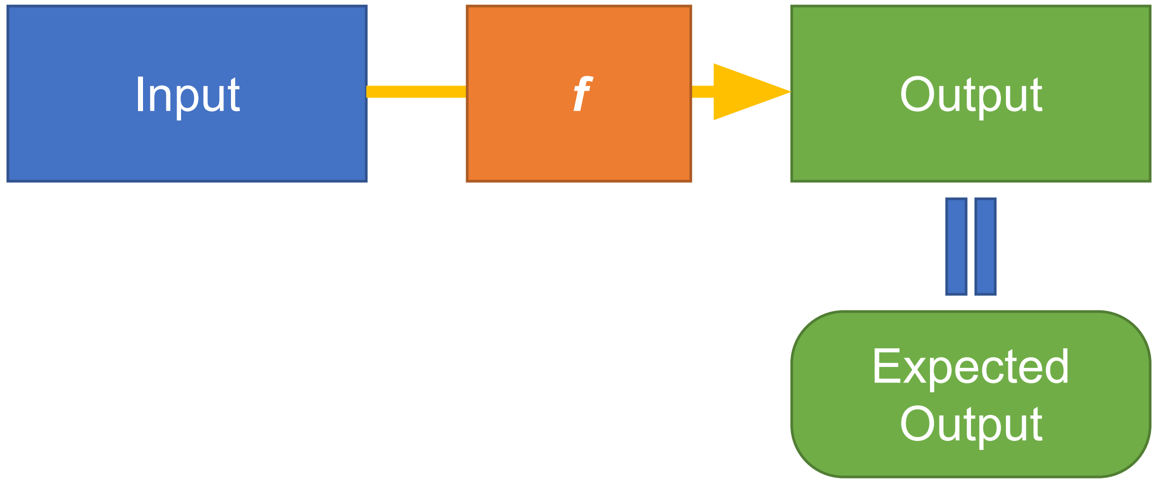
Assert $\hat{y} = f(x)$

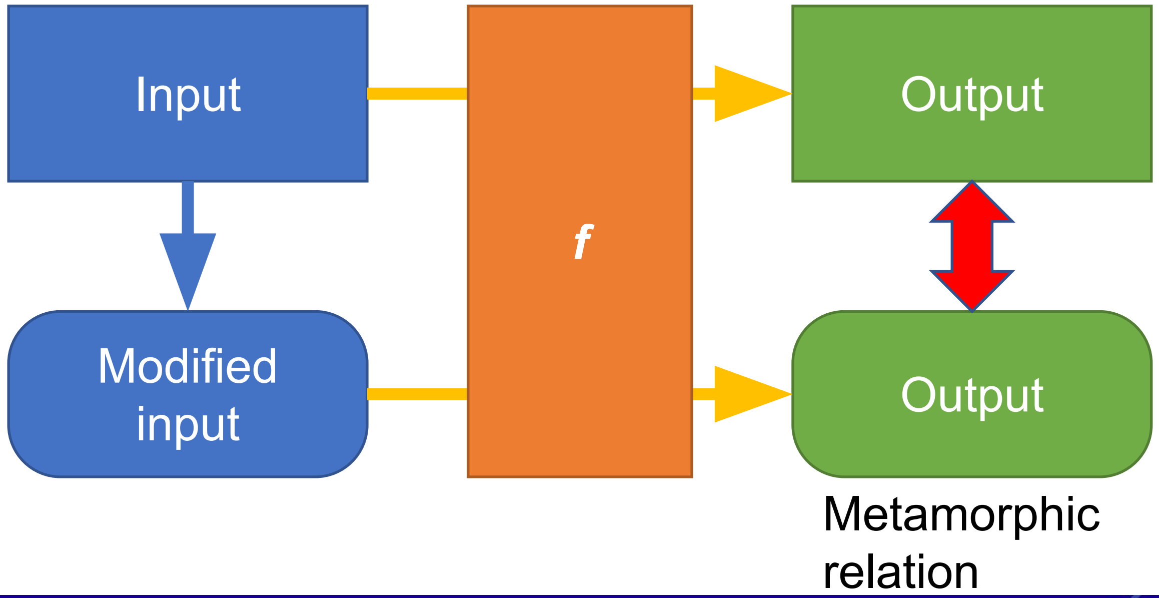
Metamorphic test

We don't need the expected output.

- $x \in S_{\mathrm{in}}$: input
- $g: S_{\mathrm{in}} \to S_{\mathrm{in}}$: input modification function
- $R \subset S_{\mathrm{out}} \times S_{\mathrm{out}}$: binary relation on S_{out}

Assert $(f(x), f(g(x))) \in R$

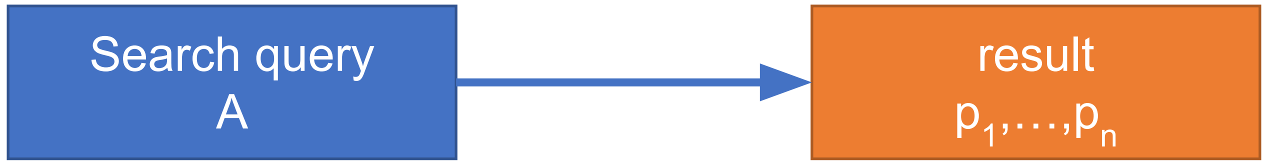




Application of metamorphic test

Application in search engine [Zhou et al, 2016]

Metamorphic test is designed in order to estimate the quality of search engine.



Examples of metamorphic relations

Relation name	Input modification	Expected relation
MPSite	If the domain of p_i is d_i, "A" → "A & site:d_i"	The result still contains p_i.
MPTitle	If the title of p_i is t_i, "A" → "A & t_i"	The result still contains p_i.
MPReverseJD	If A have the form "A_1 & A_2 & … & A_n", "A_1 & A_2 & … & A_n" → "A_n & … & A_2 & A_1"	The results of A and reversed one have large similarity. (measured by Jaccard coefficient)

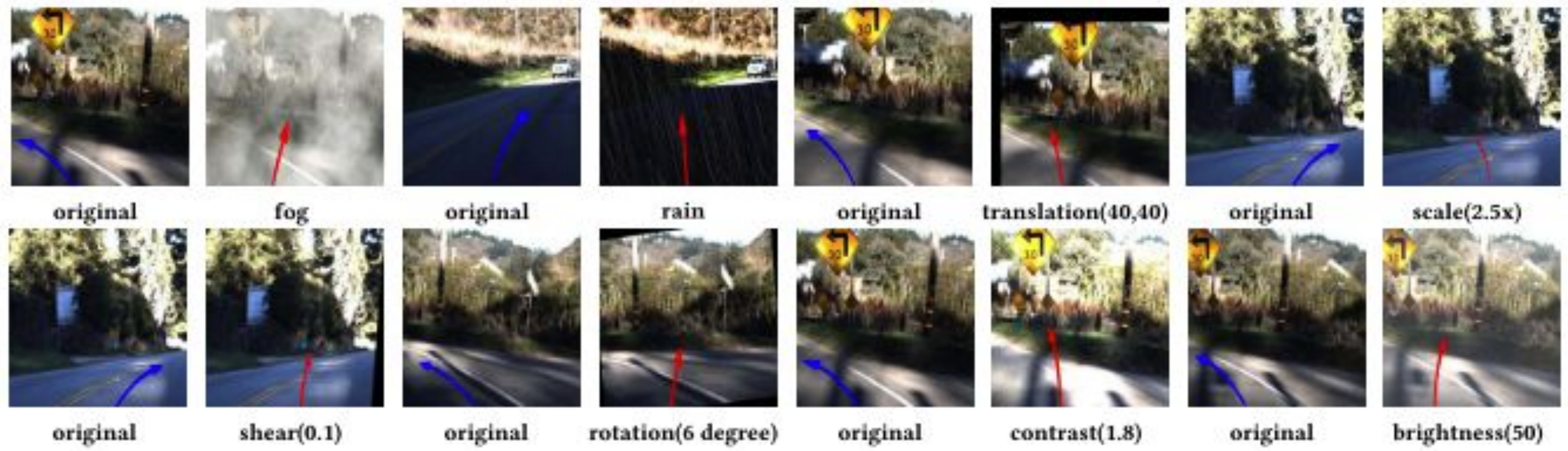
[Zhou et al, 2016] Zhou, Z. Q., Xiang, S., & Chen, T. Y. (2016). Metamorphic Testing for Software Quality Assessment: A Study of Search Engines. *IEEE Transactions on Software Engineering*, *42*(3), 264–284.

Arithmer

Metamorphic test in ML model

Also in machine learning system, the concept of metamorphic test is applicable.

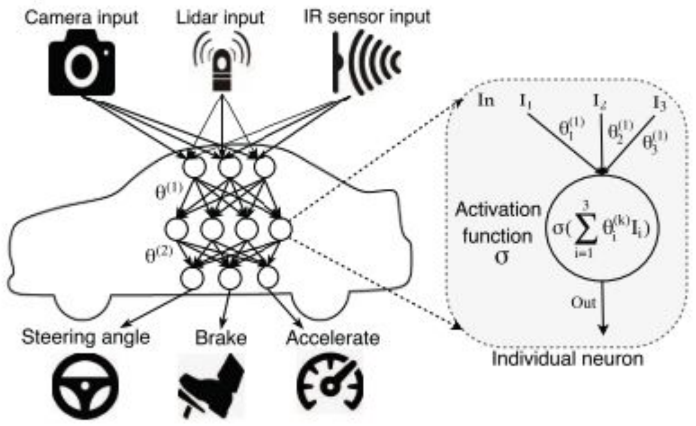
Example in image recognition for autonomous cars (DeepTest) [Tian et al, 2018]



Erroneous behaviors found by metamorphic test

Applying the several transformation which is expected not to be affected to the steering angle, the metamorphic test oracles are created.
They find the erroneous behaviors by these synthetic images.

Remark: They also use the **neuron coverage** for finding erroneous behavior (explain later)



[Tian et al, 2018] Tian, Y., Pei, K., Jana, S., & Ray, B. (2018). DeepTest. Proceedings of the 40th International Conference on Software Engineering, 2018-May, 303–314.

Other examples of metamorphic test

[Zhang et al, 2018] constructs the metamorphic test oracles by synthetic images created by GAN.

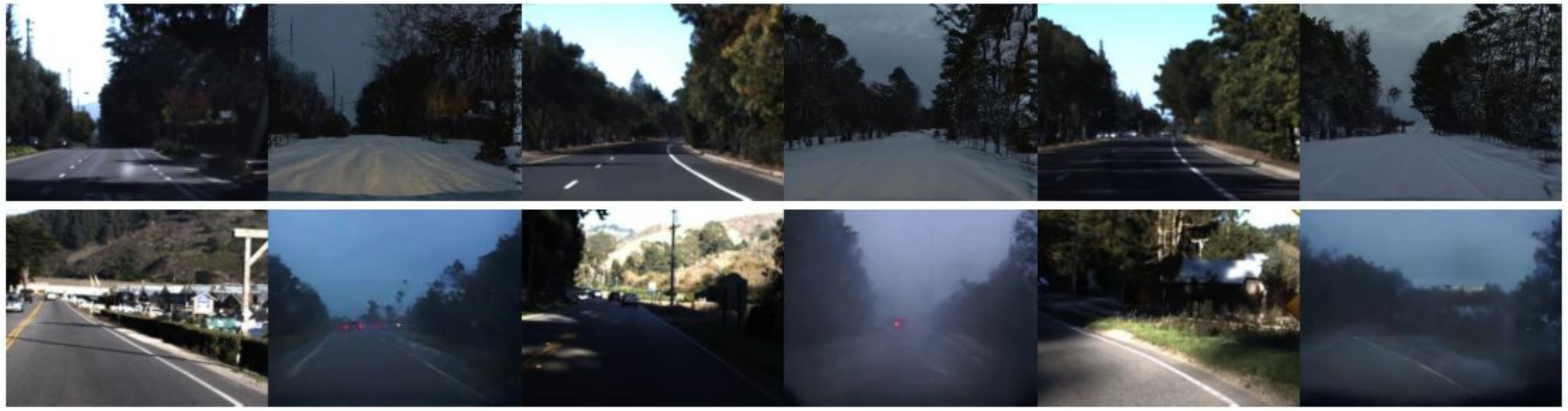
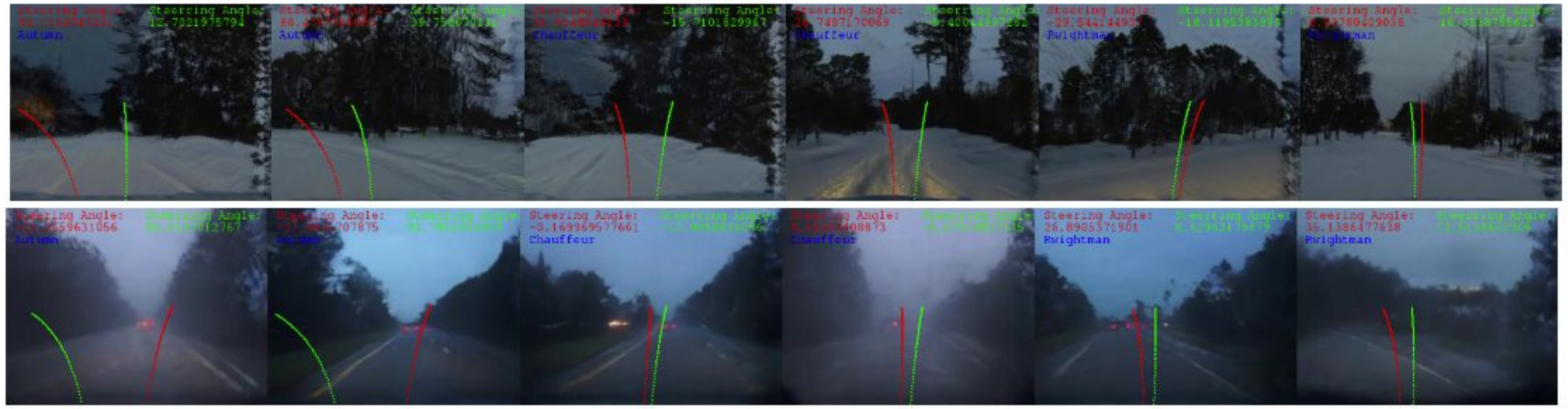


Figure 8: Real and GAN-generated images



(a) Autumn (b) Chauffeur (c) Rwightman

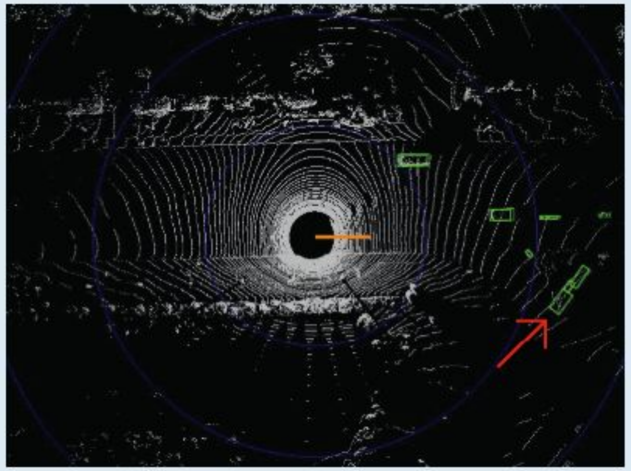
Figure 9: Inconsistency of steering angle prediction on real and synthetic images

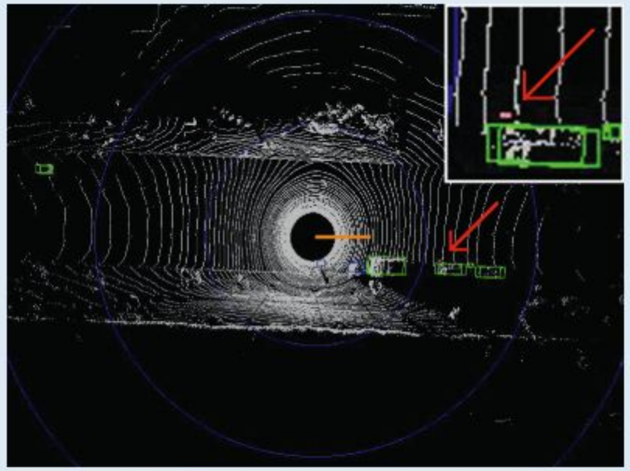
[Zhang et al, 2018] Zhang, M., Zhang, Y., Zhang, L., Liu, C., & Khurshid, S. (2018). DeepRoad: GAN-Based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems. *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 132–142.

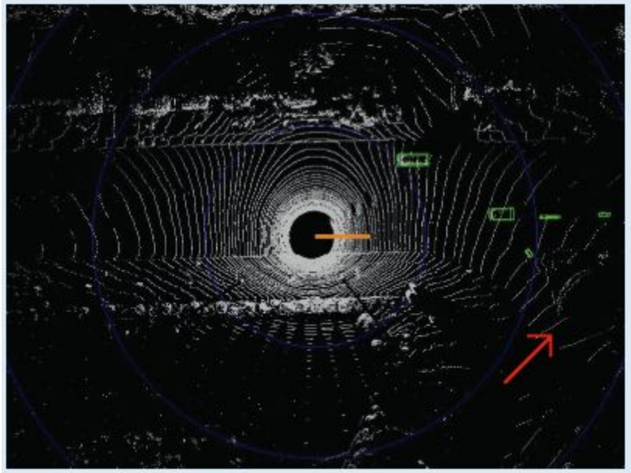
Other examples of metamorphic test

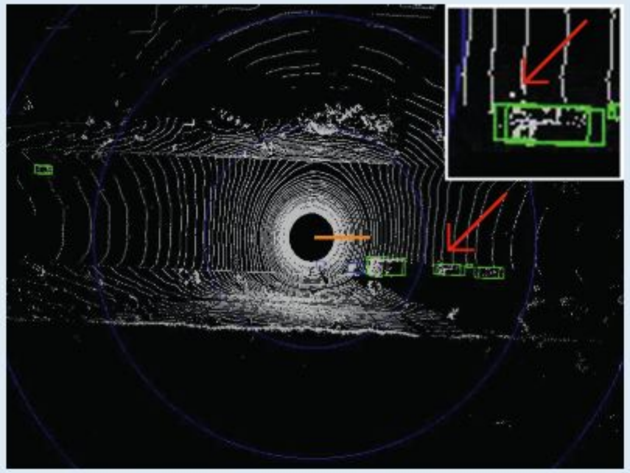
[Zhou et al, 2019] uses metamorphic test on LIDAR data in the Baidu Apollo autonomous car system.

By adding random data outside the ROI, they find the fatal behavior that the pedestrian or other car in the ROI cannot be detected by small random noise.


(a) Original: 101,676 LiDAR data points; the green boxes were generated by the Apollo system to represent the detected cars.


(c) Original: 104,251 LiDAR data points; the small pink mark was generated by the Apollo system to represent a detected pedestrian.

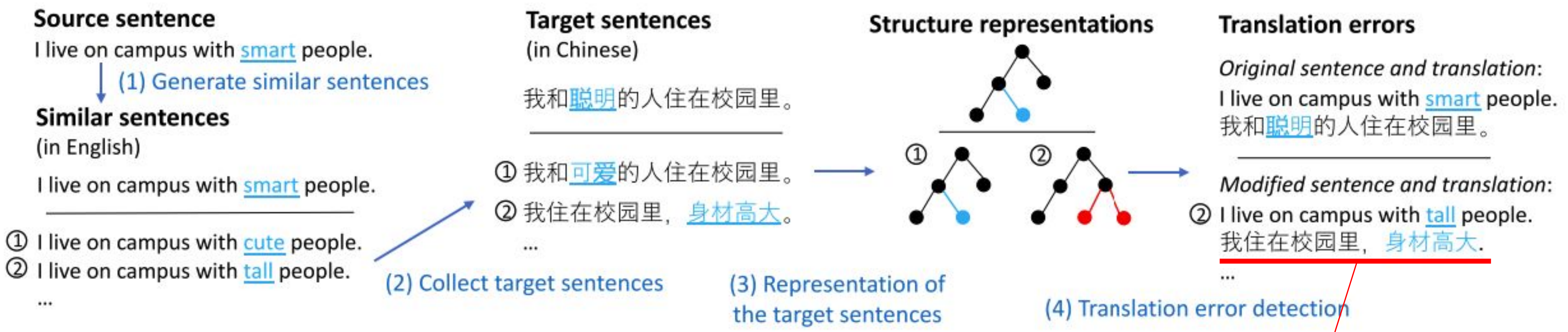

(b) After adding 1,000 random data points outside the ROI, the three cars inside the ROI could no longer be detected.


(d) After adding only 10 random data points outside the ROI, the pedestrian inside the ROI could no longer be detected.

[Zhou et al, 2019] Zhou, Z. Q., & Sun, L. (2019). Metamorphic testing of driverless cars. *Communications of the ACM*, *62*(3), 61–67.

Other examples of metamorphic test

[He et al, 2020] uses metamorphic test on machine translation.


Source sentence
I live on campus with smart people.
(1) Generate similar sentences
Similar sentences
(in English)
I live on campus with smart people.
① I live on campus with cute people.
② I live on campus with tall people.
…
(2) Collect target sentences
Target sentences
(in Chinese)
我和聪明的人住在校园里。
① 我和可爱的人住在校园里。
② 我住在校园里，身材高大。
…
Structure representations
①
②
(3) Representation of the target sentences
Translation errors
Original sentence and translation:
I live on campus with smart people.
我和聪明的人住在校园里。
Modified sentence and translation:
② I live on campus with tall people.
我住在校园里，身材高大.
…
(4) Translation error detection

I live on campus and am tall.
(erroneous behavior)

[He et al, 2020] He, P., Meister, C., & Su, Z. (2020). Structure-invariant testing for machine translation. *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 961–973.

What is neuron coverage?

$y = f(x; W_1, \ldots, W_L; b_1, \ldots, b_L)$ Deep neural network

$$\begin{aligned} z_0(x) &= x, \\ z_j(x) &= \sigma(W_j z_{j-1}(x) + b_j), \\ z_j(x) &= \begin{pmatrix} z_{j,1}(x) & \ldots & z_{j,n_j}(x) \end{pmatrix}^T \in \mathbb{R}^{n_j}, \\ y &= W_L z_L(x) \end{aligned}$$

$x \in \mathbb{R}^{n_0}$

$W_j \in \mathbb{R}^{n_j \times n_{j-1}}, b_j \in \mathbb{R}^{n_j}$

Activation function like ReLU

$n_{\text{neuron}} := n_1 + \cdots + n_l$ The number of neurons

The number of "activated" neurons

Definition.
For some threshold $t > 0$, **Neuron Coverage** for test input set T is defined as follows:

$$\mathrm{NC}(T, t) := \frac{\boxed{\#\{(j,k) \mid z_{j,k}(x) \geq t \text{ for any } x \in T\}}}{n_{\text{neuron}}}$$

What is neuron coverage?

Definition.
For some threshold $t > 0$, **Neuron Coverage** for test input set T is defined as follows:

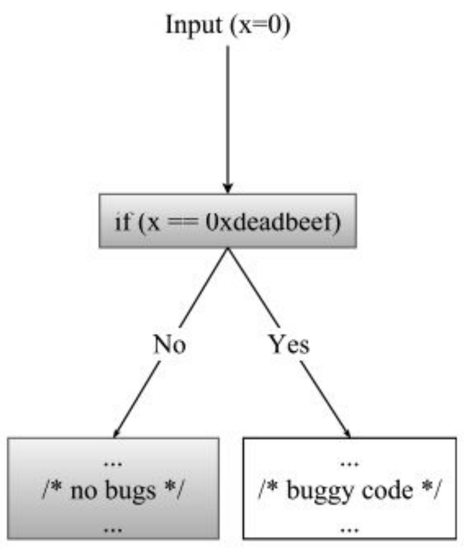
$$\mathrm{NC}(T, t) := \frac{\#\{(j, k) \mid z_{j,k}(x) \geq t \text{ for any } x \in T\}}{n_{\mathrm{neuron}}}$$

NC is proposed in [Pei et al, 2017].
Idea: analogy of coverage rate

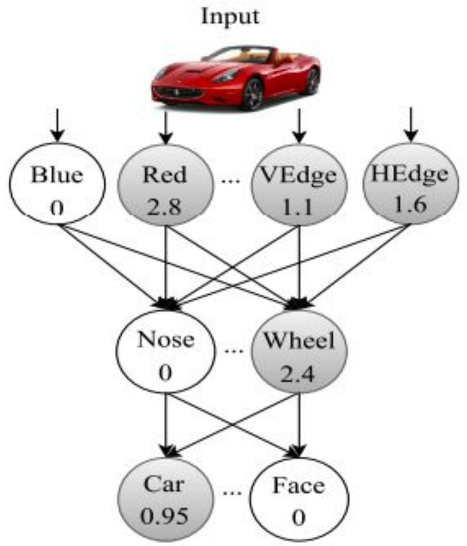
If the neuron value is large, this neuron strongly influences the output through the downstream layers.

Low NC implies that a significant number of neurons have no effect on the output for the test data set.

In order to test the effect on the output for all neurons, **we need to create test data that will have a high NC.**



(a) A program with a rare branch



(b) A DNN for detecting cars and faces

[Pei et al, 2017] Pei, K., Cao, Y., Yang, J., & Jana, S. (2017). DeepXplore. *Proceedings of the 26th Symposium on Operating Systems Principles*, 1–18.

Arithmer

How to create the high NC test data?

[Pei et al, 2017] proposes to use **the gradient ascending** and it is widely used.

$$z_0(x) = x, \qquad x \in \mathbb{R}^{n_0}$$

$$z_j(x) = \sigma(W_j z_{j-1}(x) + b_j), \qquad W_j \in \mathbb{R}^{n_j \times n_{j-1}},\ b_j \in \mathbb{R}^{n_j}$$

$$z_j(x) = \begin{pmatrix} z_{j,1}(x) & \dots & z_{j,n_j}(x) \end{pmatrix}^T \in \mathbb{R}^{n_j},$$

$$y = W_L z_L(x)$$

Suppose that $\boldsymbol{z_{j,k}(x)}$ is not activated. We modify the input $\boldsymbol{x}$ as follows:

$$x' = x + \varepsilon \nabla_x z_{j,k}(x)$$

Update the data in the direction of increasing the value of inactive neurons

If we restrict the transform of the input by the transformation function $\boldsymbol{g(x,\eta)}$ with parameters $\boldsymbol{\eta}$, we calculate the parameters $\boldsymbol{\eta}$ as

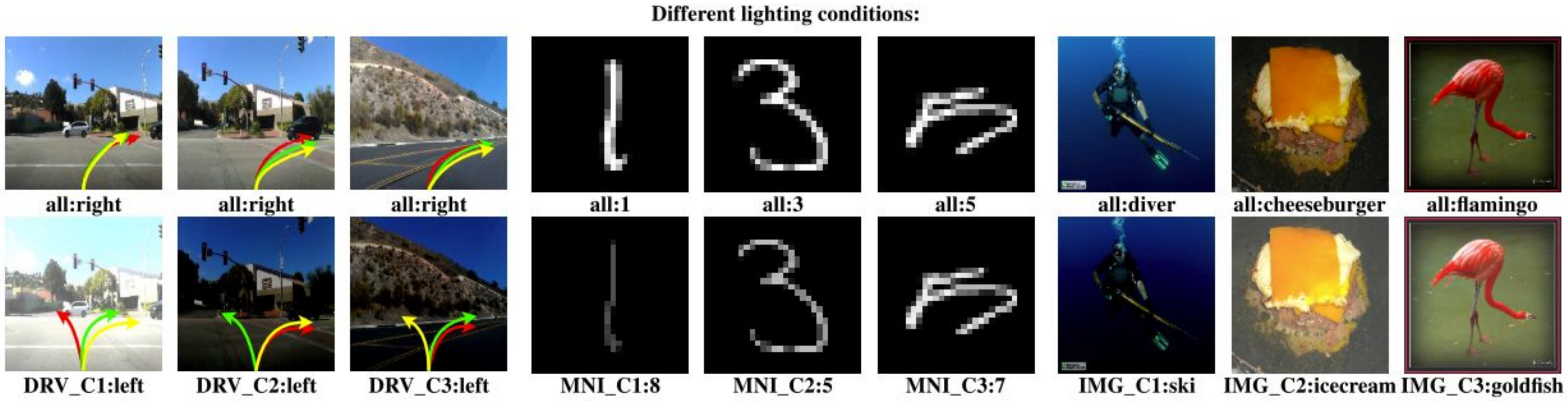
$$x' = g(x, \eta'),$$

$$\eta' = \eta + \varepsilon \nabla_\eta z_{j,k}(g(x,\eta)).$$

Application of neuron coverage

DeepXplore in [Pei et al, 2017]

Erroneous behaviors by changing light conditions



For the set of neural networks $f^{(1)}, \dots, f^{(n)}$, set

Make the outputs of NNs differently

$$L_{j,\ell,m}(x) = \boxed{\sum_{j \neq k} f^{(k)}(x)[c] - \lambda_1 f^{(j)}(x)[c]} + \boxed{\lambda_2 z^{(j)}_{\ell,m}(x)}$$

Raise NC

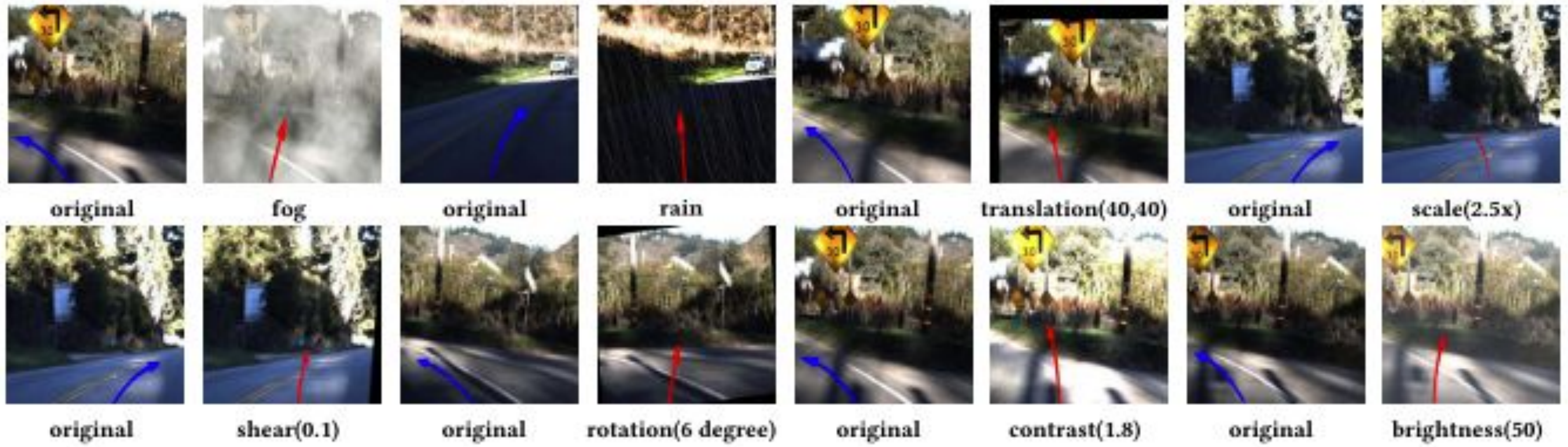
$$x' = x + \varepsilon \nabla_x L_{j,\ell,m}(x)$$

[Pei et al, 2017] Pei, K., Cao, Y., Yang, J., & Jana, S. (2017). DeepXplore. *Proceedings of the 26th Symposium on Operating Systems Principles*, 1–18.

Arithmer

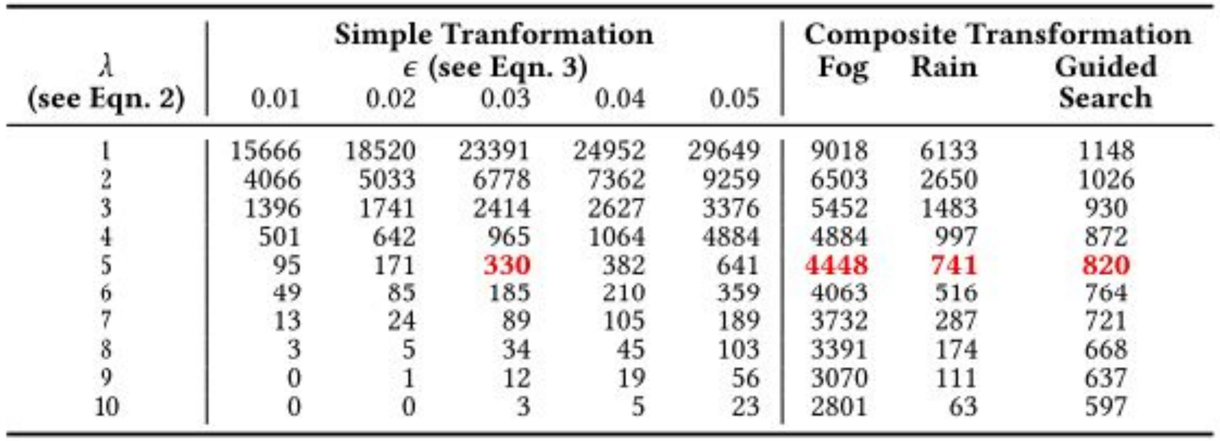
Application of neuron coverage

[Tian et al, 2018] DeepTest (bis.)



Erroneous behaviors found by metamorphic test

They use the NC for finding the erroneous behavior like DeepXplore.



λ (see Eqn. 2)	Simple Tranformation ε (see Eqn. 3) 0.01	0.02	0.03	0.04	0.05	Composite Transformation Fog	Rain	Guided Search
1	15666	18520	23391	24952	29649	9018	6133	1148
2	4066	5033	6778	7362	9259	6503	2650	1026
3	1396	1741	2414	2627	3376	5452	1483	930
4	501	642	965	1064	4884	4884	997	872
5	95	171	330	382	641	4448	741	820
6	49	85	185	210	359	4063	516	764
7	13	24	89	105	189	3732	287	721
8	3	5	34	45	103	3391	174	668
9	0	1	12	19	56	3070	111	637
10	0	0	3	5	23	2801	63	597

Metamorphic test and neuron coverage can be combined to find errors efficiently.

Arithmer

Is neuron coverage meaningful?

Some studies indicate that increasing NC is not necessarily meaningful for testing adversarial attack.

[Harel-Canada et al, 2020]

Generate adversarial images that also improve NC.

$$x_{\mathrm{adv}} = x + \delta$$

$$\delta = \underset{\delta}{\operatorname{argmin}} \left[\boxed{\|\delta\|_{L^p} + cL(f(x+\delta), y)} + \lambda \sum_{j=1}^{L} \boxed{\mathrm{KL}(z_j(x+\delta), \mathrm{unif})} \right]$$

Kullback-Leibler divergence

Carlini-Wagner type adversarial attack
(We can also choose other attacks)

Uniform distribution in [0,1]

This term measures how different the output of the j-th layer is from uniform distribution

$\Downarrow$

This also measures whether the neurons in the j-th layer is unbiasedly activated
= high NC

[Harel-Canada et al, 2020] Harel-Canada, F., Wang, L., Gulzar, M. A., Gu, Q., & Kim, M. (2020). Is neuron coverage a meaningful measure for testing deep neural networks? *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 851–862.

Is neuron coverage meaningful?

For these high-NC adversarial images, they measures the following indices:

Attack Success Rate is the classification error rate for adversarial test data set.

Naturalness measures how natural the generated image is.

- Frèchet Inception Distance: a measure of similarity between two dataset of images.
- Inception Score: a measure of naturalness based on salience and diversity

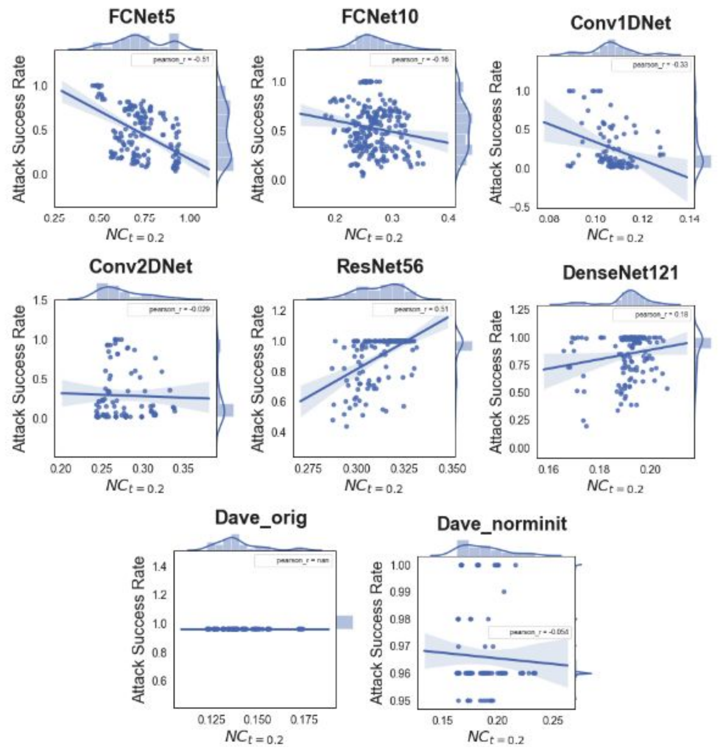
Output impartiality measures how the predictions of the model for adversarial images are biased.

$$\mathrm{OI}_k(T) = \frac{\sum_{x \in T, f(x)=c_k} P_{T,k} \log P_{T,k}}{\log \# C}, \quad P_{T,k} := \frac{\#\{x \in T \mid f(x) = c_k\}}{\# T}$$

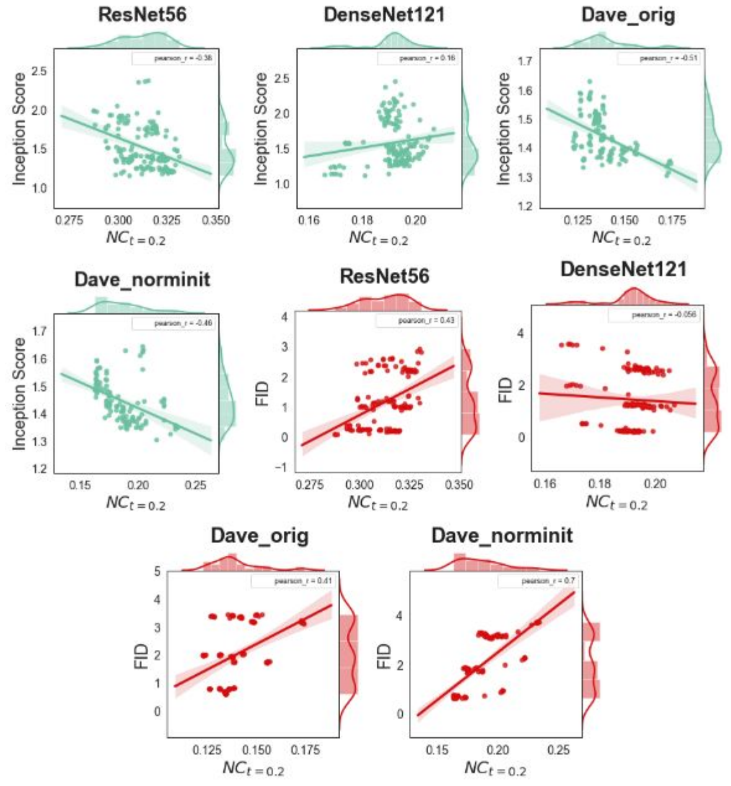
[Harel-Canada et al, 2020] Harel-Canada, F., Wang, L., Gulzar, M. A., Gu, Q., & Kim, M. (2020). Is neuron coverage a meaningful measure for testing deep neural networks? *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 851–862.

Is neuron coverage meaningful?

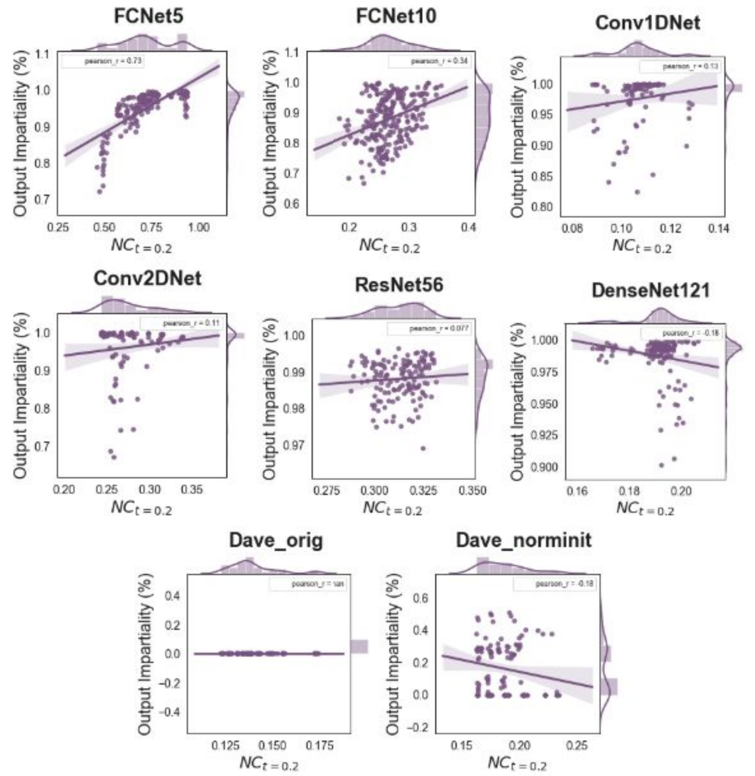
Attack success rate



Naturalness



Output Impartiality



They cannot support the hypothesis that these indices is positively correlated with NC.

[Harel-Canada et al, 2020] Harel-Canada, F., Wang, L., Gulzar, M. A., Gu, Q., & Kim, M. (2020). Is neuron coverage a meaningful measure for testing deep neural networks? *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 851–862.

Arithmer

Is neuron coverage meaningful?

Speaker's Remark

- Contexts seems to be different between DeepTest & DeepXplore and [Harel-Canada et al, 2020].
    - Deeptest & DeepXplore
        - The deformation method of image is restricted and recognizable by human
        - Finding erroneous behavior of the model is the main concern.
    - [Harel-Canada et al, 2020]
        - The deformation is unrecognizable by human.
        - Finding adversarial attack is the main concern, which is more severe.

- NC+metamorphic test effectively detects **erroneous behavior in response to changes in data that are not in the test dataset but are likely to be real and recognizable by humans.**

- For adversarial attack, naturalness and output impartiality are more useful indices.

Minimal adversarial distortion

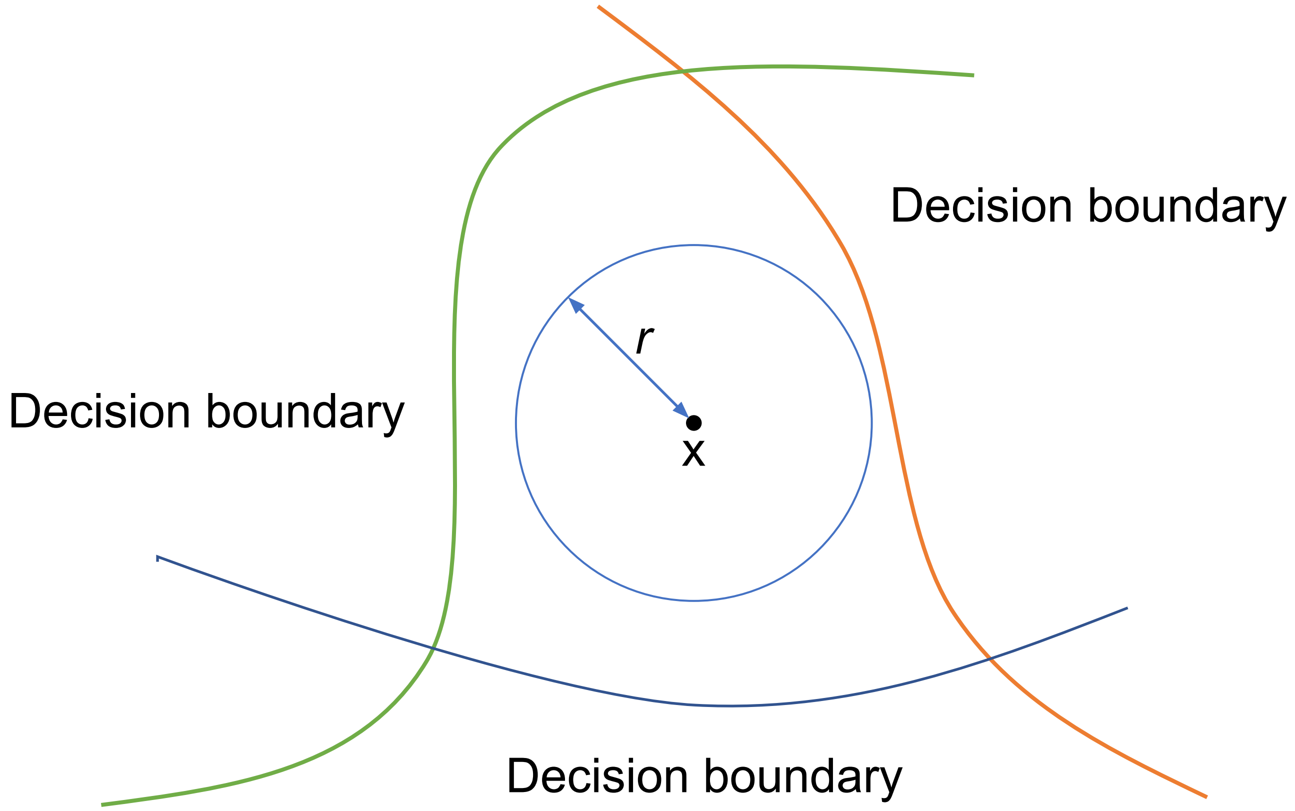
How can we assure the robustness of ML model including adversarial attack?

In order to do it, it is sufficient to estimate **how much noise is allowed to be added without change the inference results.**

Definition.
Assume that we treat the classification.
For the input x, the number $r > 0$ is **the minimal adversarial distortion** if $f(x+\delta) = f(x)$ holds for **any** perturbation δ of the input with $\|\delta\| \leq r$.

Can we calculate minimal adversarial distortion in advance?

We can estimate the lower bound.



Arithmer

Lower bound of adversarial minimum distortion

CROWN [Zhang et al, NIPS 2018]

$$z_0(x) = x, \qquad x \in \mathbb{R}^{n_0}$$

$$z_j(x) = \sigma(W_j z_{j-1}(x) + b_j), \qquad W_j \in \mathbb{R}^{n_j \times n_{j-1}}, b_j \in \mathbb{R}^{n_j}$$

$$z_j(x) = \begin{pmatrix} z_{j,1}(x) & \dots & z_{j,n_j}(x) \end{pmatrix}^T \in \mathbb{R}^{n_j},$$

$$y = W_L z_L(x)$$

The proof can be done by backward induction on the layer number *j* and some elementary calculations (if you have the energy).
The proof for the simple case is given in [Sato et al, 2021].

Theorem [Zhang et al, NIPS 2018]
For the fixed input $\boldsymbol{x}$ and $\boldsymbol{\varepsilon > 0}$, there exists two vectors $\boldsymbol{\gamma^{(L)}, \gamma^{(U)} \in \mathbb{R}^{n_L}}$ such that

$$\gamma^{(L)} \le f(x+\delta) \le \gamma^{(U)} \text{ for } \forall \delta \in \mathbb{R}^{n_0} \text{ with } \|\delta\|_p \le \varepsilon,$$

Moreover, these two functions are calculated explicitly as

$$\gamma_k^{(L)} = \varepsilon \|\Lambda_{k,:}^{(0)}\|_q + \Lambda_{k,:}^{(0)} x + \sum_{j=1}^{L} \Lambda_{k,:}^{(j)} (b_j + \Delta_{:,k}^{(j)})$$

$$\gamma_k^{(L)} = -\varepsilon \|\Omega_{k,:}^{(0)}\|_q + \Omega_{k,:}^{(0)} x + \sum_{j=1}^{L} \Omega_{k,:}^{(j)} (b_j + \Theta_{:,k}^{(j)})$$

[Zhang et al, NIPS 2018] Zhang, H., Weng, T. W., Chen, P. Y., Hsieh, C. J., & Daniel, L. (2018). Efficient neural network robustness certification with general activation functions. NeurIPS, 4939–4948.

Arithmer

Lower bound of adversarial minimum distortion

$$\Lambda^{(j-1)} = \begin{cases} E_{n_L} & j = L \\ (\Lambda^{(j)} W_j) \odot \lambda^{(j-1)} & j = 1, \ldots, L-1 \end{cases}, \quad \Omega^{(j-1)} = \begin{cases} E_{n_L} & j = L \\ (\Omega^{(j)} W_j) \odot \omega^{(j-1)} & j = 1, \ldots, L-1 \end{cases}$$

$$\lambda^{(j)}_{k,\ell} = \begin{cases} \alpha^{(j)}_{\mathrm{U},\ell} & j \neq 0 \text{ and } (\Lambda^{(j+1)} W_{j+1})_{k,\ell} \geq 0, \\ \alpha^{(j)}_{\mathrm{L},\ell} & j \neq 0 \text{ and } (\Lambda^{(j+1)} W_{j+1})_{k,\ell} < 0, \\ 1 & j = 0. \end{cases}, \quad \omega^{(j)}_{k,\ell} = \begin{cases} \alpha^{(j)}_{\mathrm{U},\ell} & j \neq 0 \text{ and } (\Omega^{(j+1)} W_{j+1})_{k,\ell} \geq 0, \\ \alpha^{(j)}_{\mathrm{L},\ell} & j \neq 0 \text{ and } (\Omega^{(j+1)} W_{j+1})_{k,\ell} < 0, \\ 1 & j = 0. \end{cases}$$

$$\Delta^{(j)}_{k,\ell} = \begin{cases} \beta^{(j)}_{\mathrm{U},\ell} & j \neq 0 \text{ and } (\Lambda^{(j+1)} W_{j+1})_{k,\ell} \geq 0, \\ \beta^{(j)}_{\mathrm{L},\ell} & j \neq 0 \text{ and } (\Lambda^{(j+1)} W_{j+1})_{k,\ell} < 0, \\ 0 & j = 0. \end{cases}, \quad \Theta^{(j)}_{k,\ell} = \begin{cases} \beta^{(j)}_{\mathrm{U},\ell} & j \neq 0 \text{ and } (\Omega^{(j+1)} W_{j+1})_{k,\ell} \geq 0, \\ \beta^{(j)}_{\mathrm{L},\ell} & j \neq 0 \text{ and } (\Omega^{(j+1)} W_{j+1})_{k,\ell} < 0, \\ 0 & j = 0. \end{cases}$$

$\boldsymbol{\alpha}_{\mathrm{L}}^{(j)}, \boldsymbol{\alpha}_{\mathrm{U}}^{(j)}, \boldsymbol{\beta}_{\mathrm{L}}^{(j)}, \boldsymbol{\beta}_{\mathrm{U}}^{(j)} \in \mathbb{R}^{n_j}$ are defined to satisfy the following inequality:

Prescribed range of the neuron values

$$\alpha^{(j)}_{\mathrm{L},\ell}\left(\xi + \beta^{(j)}_{\mathrm{L},\ell}\right) \leq \sigma(\xi) \leq \alpha^{(j)}_{\mathrm{U},\ell}\left(\xi + \beta^{(j)}_{\mathrm{U},\ell}\right), \quad \xi \in [l^{(j)}_\ell, r^{(j)}_\ell].$$

[Zhang et al, NIPS 2018] Zhang, H., Weng, T. W., Chen, P. Y., Hsieh, C. J., & Daniel, L. (2018). Efficient neural network robustness certification with general activation functions. NeurIPS, 4939–4948.

Arithmer

Lower bound of adversarial minimum distortion

[Zhang et al, 2018]

Theorem

For the fixed input $\boldsymbol{x}$ and $\boldsymbol{\varepsilon > 0}$, there exists two vectors $\boldsymbol{\gamma^{(L)}, \gamma^{(U)}} \in \mathbb{R}^{n_L}$ such that

$$\gamma^{(\mathrm{L})} \leq f(x+\delta) \leq \gamma^{(\mathrm{U})} \text{ for } \forall \delta \in \mathbb{R}^{n_0} \text{ with } \|\delta\|_p \leq \varepsilon,$$

Moreover, these two functions are calculated explicitly as

$$\gamma_k^{(\mathrm{L})} = \varepsilon \|\Lambda_{k,:}^{(0)}\|_q + \Lambda_{k,:}^{(0)} x + \sum_{j=1}^{L} \Lambda_{k,:}^{(j)} (b_j + \Delta_{:,k}^{(j)})$$

$$\gamma_k^{(\mathrm{L})} = -\varepsilon \|\Omega_{k,:}^{(0)}\|_q + \Omega_{k,:}^{(0)} x + \sum_{i=1}^{L} \Omega_{k,:}^{(j)} (b_j + \Theta_{:,k}^{(j)})$$

Then if *c* is the inferenced label of the input *x*,
we have the lower bound of adversarial minimum distortion by

$$\widetilde{\varepsilon} = \min_{t \neq c} \max\{\varepsilon > 0 \mid \gamma_c^{(\mathrm{L})} - \gamma_t^{(\mathrm{U})} > 0\}$$

[Zhang et al, NIPS 2018] Zhang, H., Weng, T. W., Chen, P. Y., Hsieh, C. J., & Daniel, L. (2018). Efficient neural network robustness certification with general activation functions. NeurIPS, 4939–4948.

Arithmer

Other techniques to find the lower bound

[Boopathy et al, 2019] extends [Zhang et al, 2018] to CNN case.
The code is available as **CNN-Cert** on https://github.com/IBM/CNN-Cert

[Weng et al, 2019] **PROVEN**: The probalistic version of the robustness estimate
Assume that the inference label of x is c and the random input X follows some given distribution $\mathcal{D}$ with mean x.
Moreover, we assume that the following inequality holds.

$$A_t^{(\mathrm{L})}x + d^{(\mathrm{L})} \le f(x)_c - f(x)_t \le A_t^{(\mathrm{U})}x + d^{(\mathrm{U})}, \quad t \ne c$$

Then the following inequality holds for any a:

$$\gamma^{(\mathrm{L})} := \mathbb{P}[A_t^{(\mathrm{L})}x + d^{(\mathrm{L})} > a] \le \mathbb{P}[f(x)_c - f(x)_t > a] \le \mathbb{P}[A_t^{(\mathrm{U})}x + d^{(\mathrm{U})} > a] =: \gamma^{(\mathrm{U})}, \quad t \ne c$$

$\boldsymbol{\gamma^{(L)}}, \boldsymbol{\gamma^{(\mathrm{U})}}$ can be estimated by using Hoeffding's inequality.

[Boopathy et al, 2019] Boopathy, A., Weng, T.-W., Chen, P.-Y., Liu, S., & Daniel, L. (2019). CNN-Cert: An Efficient Framework for Certifying Robustness of Convolutional Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*, 3240–3247.
[Weng et al, 2019] Weng, L., Chen, P.-Y., Nguyen, L., Squillante, M., Boopathy, A., Oseledets, I., & Daniel, L. (2019). PROVEN: Verifying Robustness of Neural Networks with a Probabilistic Approach. PMLR.

Can we check the assertion directly?

Example: We construct the machine learning model predicting housing prices based on housing attributes.
Can we check the following assertion?

If the living area is more than 6,000 square feet, the inferred result is more than $400,000.

At first glance, it looks like it can't be done...

- It is impossible to test on all possible data where the living area is more than 6000 square feet;
- No matter how many data are tested, there is no guarantee that the above claim will hold 100% of the time.

However, given that we are trying to find **counterexample**s to the above claim, it would be nice if we could find at least one example as follows:
Find the data where the living area is more than 6000 square feet but the inferred result is less than or equal to $400,000.

"and"

$y_{\text{price}} = f(x_{\text{area}}, \ldots)$ ML model

Find the data *x* such that $x_{\text{area}} \geq 6000 \wedge f(x) \leq 40000$

Formal verification

$$x_{\text{area}} \geq 6000 \wedge f(x) \leq 400000$$

From this logical formula, we consider eliminating *f*.

It may seem impossible, but for example, if *f* is a deep neural network, then only affine transformations and activation functions (ReLU) appear in the definition of *f*.

$$\begin{aligned} z_0(x) &= x, \\ z_j(x) &= \sigma(W_j z_{j-1}(x) + b_j), \\ z_j(x) &= \begin{pmatrix} z_{j,1}(x) & \ldots & z_{j,n_j}(x) \end{pmatrix}^T \in \mathbb{R}^{n_j}, \\ y &= W_L z_L(x) \end{aligned}$$

So, for example, when the activation function is ReLU, if you try hard, you should be able to eliminate *f* from the above equation and write it down using only linear inequalities and "and/or".

$$\begin{aligned} \sigma(x) \geq a &\iff ((a > 0) \wedge (x \geq a)) \vee (a \leq 0 \\ \sigma(x) < a &\iff ((a > 0) \wedge (x < a)) \vee (a \leq 0 \end{aligned}$$

In this way, we can reduce the problem to find the counterexample to the **satisfiability problem**:

Satisfiability problem:
For given propositional logical formula, determine if there exists the true/false assignment to atomic formula so that the whole proposition is made true.

This problem is NP-complete but efficient algorithms are being actively researched.

Application of formal verification

[Huang et al, 2017] applies formal verification to find the adversarial images.

$$f(x) = \text{"boat"} \wedge f(x+\delta) = \text{"truck"}$$

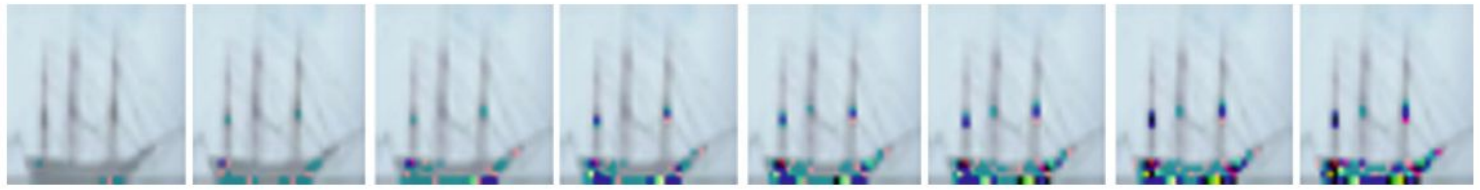


Fig. 9. An illustrative example of mapping back to input layer from the Cifar-10 mataset: the last image classifies as a truck.

[Huang et al, 2017] Huang, X., Kwiatkowska, M., Wang, S., & Wu, M. (2017). Safety Verification of Deep Neural Networks. In R. Majumdar & V. Kunčak (Eds.), *Computer Aided Verification* (pp. 3–29). Springer International Publishing.

Application of formal verification

[Sato et al, 2020] applies formal verification to decision tree ensemble model and find the violation range (the range of the values which does not satisfy the assertion)

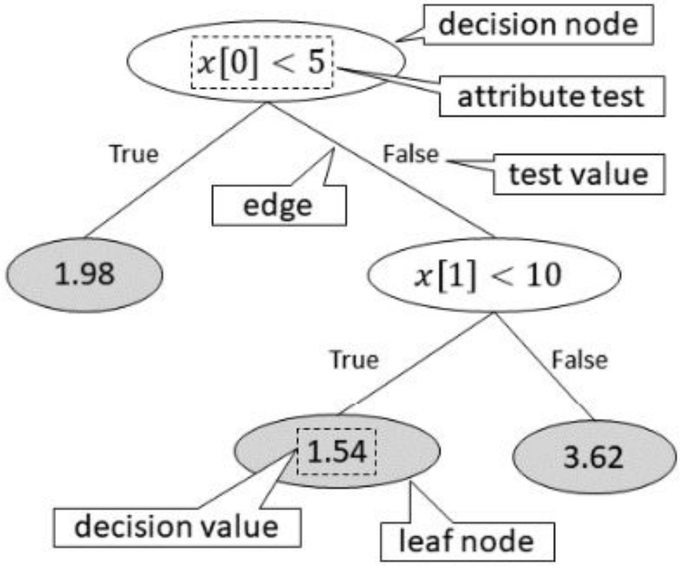
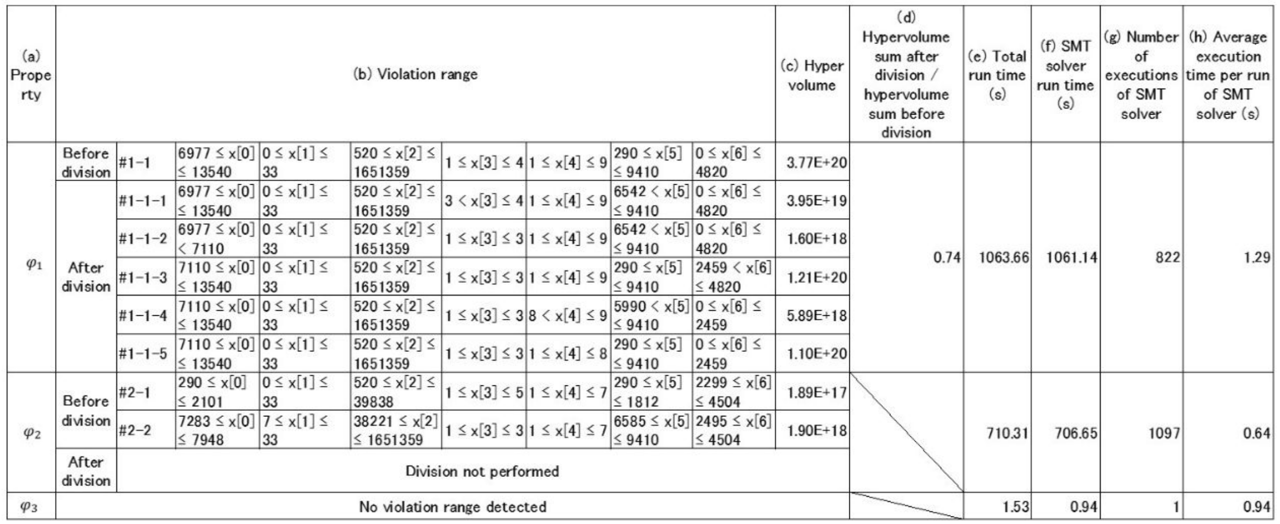


Fig. 2 Structure of decision tree

$$f_i^p := \left(\bigwedge_{j=1}^{d} \operatorname{attr}\left(n_j^p\right)(x) = \operatorname{tv}\left(\left\langle n_j^p, n_{j+1}^p \right\rangle\right) \right)$$
$$\rightarrow \left(y_i = \operatorname{dv}\left(n_{d+1}^p\right)\right)$$



(a) Property	(b) Violation range									(c) Hyper volume	(d) Hypervolume sum after division / hypervolume sum before division	(e) Total run time (s)	(f) SMT solver run time (s)	(g) Number of executions of SMT solver	(h) Average execution time per run of SMT solver (s)
φ_1	Before division	#1-1	6977 ≤ x[0] ≤ 13540	0 ≤ x[1] ≤ 33	520 ≤ x[2] ≤ 1651359	1 ≤ x[3] ≤ 4	1 ≤ x[4] ≤ 9	290 ≤ x[5] ≤ 9410	0 ≤ x[6] ≤ 4820	3.77E+20	0.74	1063.66	1061.14	822	1.29
	After division	#1-1-1	6977 ≤ x[0] ≤ 13540	0 ≤ x[1] ≤ 33	520 ≤ x[2] ≤ 1651359	3 < x[3] ≤ 4	1 ≤ x[4] ≤ 9	6542 < x[5] ≤ 9410	0 ≤ x[6] ≤ 4820	3.95E+19					
		#1-1-2	6977 ≤ x[0] < 7110	0 ≤ x[1] ≤ 33	520 ≤ x[2] ≤ 1651359	1 ≤ x[3] ≤ 3	1 ≤ x[4] ≤ 9	6542 < x[5] ≤ 9410	0 ≤ x[6] ≤ 4820	1.60E+18					
		#1-1-3	7110 ≤ x[0] ≤ 13540	0 ≤ x[1] ≤ 33	520 ≤ x[2] ≤ 1651359	1 ≤ x[3] ≤ 3	1 ≤ x[4] ≤ 9	290 ≤ x[5] ≤ 9410	2459 < x[6] ≤ 4820	1.21E+20					
		#1-1-4	7110 ≤ x[0] ≤ 13540	0 ≤ x[1] ≤ 33	520 ≤ x[2] ≤ 1651359	1 ≤ x[3] ≤ 3	8 < x[4] ≤ 9	5990 < x[5] ≤ 9410	0 ≤ x[6] ≤ 2459	5.89E+18					
		#1-1-5	7110 ≤ x[0] ≤ 13540	0 ≤ x[1] ≤ 33	520 ≤ x[2] ≤ 1651359	1 ≤ x[3] ≤ 3	1 ≤ x[4] ≤ 8	290 ≤ x[5] ≤ 9410	0 ≤ x[6] ≤ 2459	1.10E+20					
φ_2	Before division	#2-1	290 ≤ x[0] ≤ 2101	0 ≤ x[1] ≤ 33	520 ≤ x[2] ≤ 39838	1 ≤ x[3] ≤ 5	1 ≤ x[4] ≤ 7	290 ≤ x[5] ≤ 1812	2299 ≤ x[6] ≤ 4504	1.89E+17		710.31	706.65	1097	0.64
		#2-2	7283 ≤ x[0] ≤ 7948	7 ≤ x[1] ≤ 33	38221 ≤ x[2] ≤ 1651359	1 ≤ x[3] ≤ 3	1 ≤ x[4] ≤ 7	6585 ≤ x[5] ≤ 9410	2495 ≤ x[6] ≤ 4504	1.90E+18					
	After division	Division not performed													
φ_3	No violation range detected											1.53	0.94	1	0.94

Application result to housesales prediction problem

[Sato et al, 2020] SATO, N., KURUMA, H., NAKAGAWA, Y., & OGAWA, H. (2020). Formal Verification of a Decision-Tree Ensemble Model and Detection of Its Violation Ranges. *IEICE Transactions on Information and Systems*, *E103.D*(2), 363–378.

Summary of four testing method

Metamorphic test uses the relationship between the output of the base input and that of the modified input. The erroneous behavior can be captured.

Increasing the **neuron coverage** is helpful to find the erroneous examples efficiently by combining metamorphic test.

If you are worried about adversarial attack, you may check **the adversarial minimum distortion**. The lower bound of this quantity can be estimated like backpropagation.

For more comprehensive search of counterexamples for some assertion, the **formal verification** is helpful.

Behaviorist's perspective
Focus on how the ML model behaves.

- Metamorphic test
- Neuron Coverage

Anatomist's perspective
Follow the detailed structure of the ML model.

- Adversarial minimum distortion
- Formal verification

This kind of detailed analysis is possible due to the fact that **the model structure of DNNs and the like is a composition of simple functions.**

Practical viewpoint

Metamorphic test is easy to introduce in application.

- We do not need the special libraries or special codes.
- The key is to define the metamorphic relation appropriately according to the target domain.
    - Driving support system: make the images foggy, rainy, rotated, ...etc
    - Chatbot: change a question into a different expression without changing the meaning.
    - ...
- It is possible to use parameter search to find defects in the model without relying on neuron coverage.

CNN-Cert is available to estimate **the adversarial minimum distortion.**
- This is based on tensorflow and the applicable models are limited.

More important thing is **how to raise the robustness and quality of ML model based on these testing method.**

- Erroneous behavior
    - data augmentation (we can utilize the metamorphic test to augment the data)
    - preprocessing
- Adversarial attack
    - adversarial training

Arithmer

References (I)

[Sato et al, 2021] 佐藤直人・小川秀人・來間啓伸・明神智之 (2021). 『AIソフトウェアのテスト―答えのない答え合わせ [4つの手法]』リックテレコム

[Goodfellow et al., 2015] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. In Y. Bengio & Y. LeCun (Eds.), 3rd International Conference on Learning Representations.

[Carlini & Wagner, 2017] Carlini, N., & Wagner, D. (2017). Towards Evaluating the Robustness of Neural Networks. 2017 IEEE Symposium on Security and Privacy (SP), 39–57.

[Madry et al, 2018] Madry, A., Makelov, A., Schmidt, L., Tsipras, D., & Vladu, A. (2018). Towards Deep Learning Models Resistant to Adversarial Attacks. 6th International Conference on Learning Representations.

[Chen et al, 1998] Chen, T. Y., Cheung, S. C., & Yiu, S. M. (2020). Metamorphic Testing: A New Approach for Generating Next Test Cases. ArXiv:2002.12543

[Zhou et al, 2016] Zhou, Z. Q., Xiang, S., & Chen, T. Y. (2016). Metamorphic Testing for Software Quality Assessment: A Study of Search Engines. *IEEE Transactions on Software Engineering*, *42*(3), 264–284.

Arithmer

References (II)

[Tian et al, 2018] Tian, Y., Pei, K., Jana, S., & Ray, B. (2018). DeepTest. Proceedings of the 40th International Conference on Software Engineering, 2018-May, 303–314.

[Zhang et al, 2018] Zhang, M., Zhang, Y., Zhang, L., Liu, C., & Khurshid, S. (2018). DeepRoad: GAN-Based Metamorphic Testing and Input Validation Framework for Autonomous Driving Systems. *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, 132–142.

[Zhou et al, 2019] Zhou, Z. Q., & Sun, L. (2019). Metamorphic testing of driverless cars. *Communications of the ACM*, *62*(3), 61–67.

[He et al, 2020] He, P., Meister, C., & Su, Z. (2020). Structure-invariant testing for machine translation. *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*, 961–973.

[Pei et al, 2017] Pei, K., Cao, Y., Yang, J., & Jana, S. (2017). DeepXplore. *Proceedings of the 26th Symposium on Operating Systems Principles*, 1–18.

[Harel-Canada et al, 2020] Harel-Canada, F., Wang, L., Gulzar, M. A., Gu, Q., & Kim, M. (2020). Is neuron coverage a meaningful measure for testing deep neural networks? *Proceedings of the 28th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 851–862.

Arithmer

References (III)

[Zhang et al, NIPS2018] Zhang, H., Weng, T. W., Chen, P. Y., Hsieh, C. J., & Daniel, L. (2018). Efficient neural network robustness certification with general activation functions. *Advances in Neural Information Processing Systems*, *2018-Decem*(NeurIPS), 4939–4948.

[Boopathy et al, 2019] Boopathy, A., Weng, T.-W., Chen, P.-Y., Liu, S., & Daniel, L. (2019). CNN-Cert: An Efficient Framework for Certifying Robustness of Convolutional Neural Networks. *Proceedings of the AAAI Conference on Artificial Intelligence*, *33*, 3240–3247.

[Weng et al, 2019] Weng, L., Chen, P.-Y., Nguyen, L., Squillante, M., Boopathy, A., Oseledets, I., & Daniel, L. (2019). PROVEN: Verifying Robustness of Neural Networks with a Probabilistic Approach. In K. Chaudhuri & R. Salakhutdinov (Eds.), *Proceedings of the 36th International Conference on Machine Learning* (Vol. 97, pp. 6727–6736). PMLR.

[Huang et al, 2017] Huang, X., Kwiatkowska, M., Wang, S., & Wu, M. (2017). Safety Verification of Deep Neural Networks. In R. Majumdar & V. Kunčak (Eds.), *Computer Aided Verification* (pp. 3–29). Springer International Publishing.

[Sato et al, 2020] SATO, N., KURUMA, H., NAKAGAWA, Y., & OGAWA, H. (2020). Formal Verification of a Decision-Tree Ensemble Model and Detection of Its Violation Ranges. *IEICE Transactions on Information and Systems*, *E103.D*(2), 363–378.

Arithmer 株式会社
〒106-6040
東京都港区六本木一丁目６番１号 泉ガーデンタワー 38/40F(受付)
03-5579-6683
https://arithmer.co.jp/

Arithmer