Testing Deep Neural Networks

CS453 Automated Software Testing

Shin Yoo | COINSE@KAIST

Caveat Emptor

- Today’s topic is the bleeding edge: I myself probably have more questions than answers.
- The field is moving extremely fast.
- I will also assume that everyone has the very basic knowledge of Deep Neural Networks (who doesn’t these days?) but will cover elementary facts :)

Course Re-cap

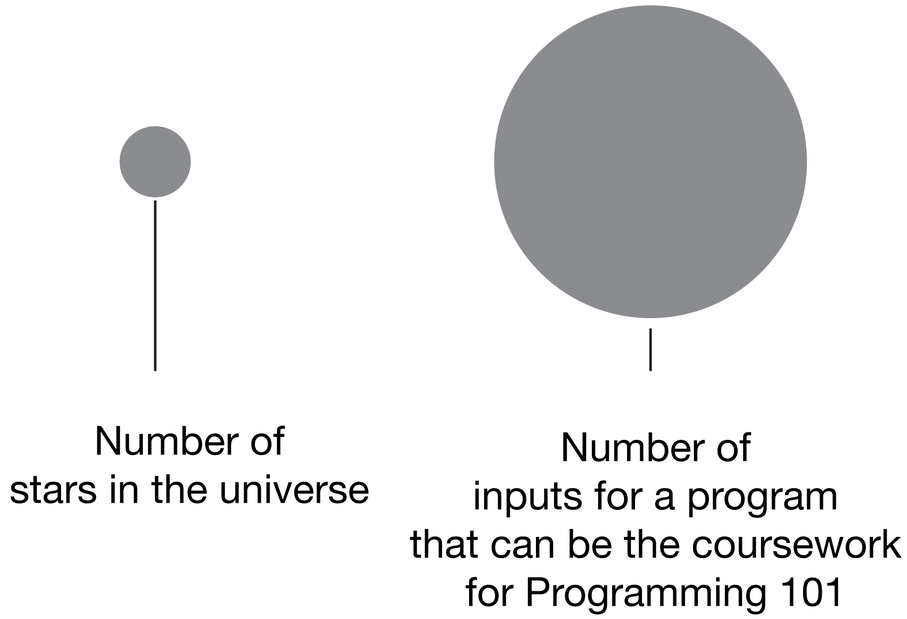
```java
public class Triangle {
    public enum TriangleType {
        INVALID, SCALENE, EQUALATERAL, ISOCELES
    }

    public static TriangleType classifyTriangle(int a, int b, int c) {
        // Sort the sides so that a <= b <= c
        if (a > b) {
            int tmp = a;
            a = b;
            b = tmp;
        }

        if (a > c) {
            int tmp = a;
            a = c;
            c = tmp;
        }

        if (b > c) {
            int tmp = b;
            b = c;
            c = tmp;
        }

        if (a + b <= c) {
            return TriangleType.INVALID;
        } else if (a == b && b == c) {
            return TriangleType.EQUALATERAL;
        } else if (a == b || b == c) {
            return TriangleType.ISOCELES;
        } else {
            return TriangleType.SCALENE;
        }
    }
}
```

Testing is really about sampling inputs.

Testing is really about sampling inputs.

(from a huuu-u-u-g-eee space)

- 32bit integers: between -2^{31} and $2^{31}-1$, there are 4,294,967,295 numbers
- The program takes three: all possible combination is close to 8^{28}
- Approximated number of stars in the known universe is 10^{24}
- Not. Enough. Time. In. The. Whole. World.



We can break down complex inputs down
to clearly defined primitives.

```
@given(st.lists(st.integers()))

@given(st.tuples(st.booleans(), st.text()))
```

Examples are from
input generator annotations of Hypothesis,
a kind-of Python implementation of Quickcheck.

https://hypothesis.readthedocs.io/en/latest/quickstart.html

We now have very sophisticated sampling methods.

Pex for fun

T3

SUSHI

EVOSUITE

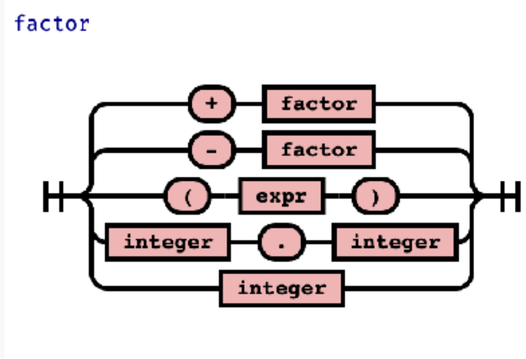
KLEE

Randoop

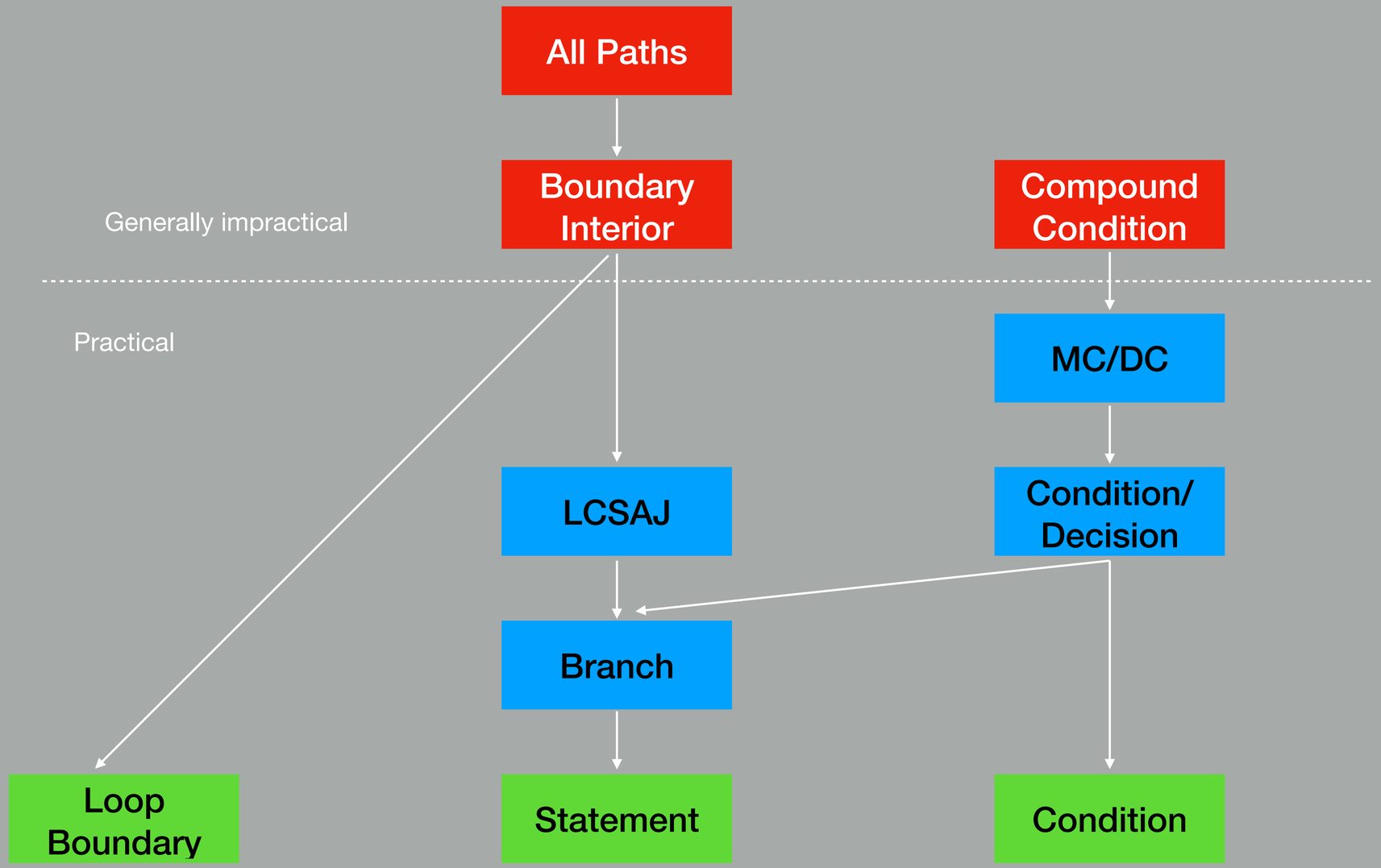
Automatic unit test generation for Java

```
american fuzzy lop 0.47b
```

sapienz



Testers like coverage (… but why?🧐)

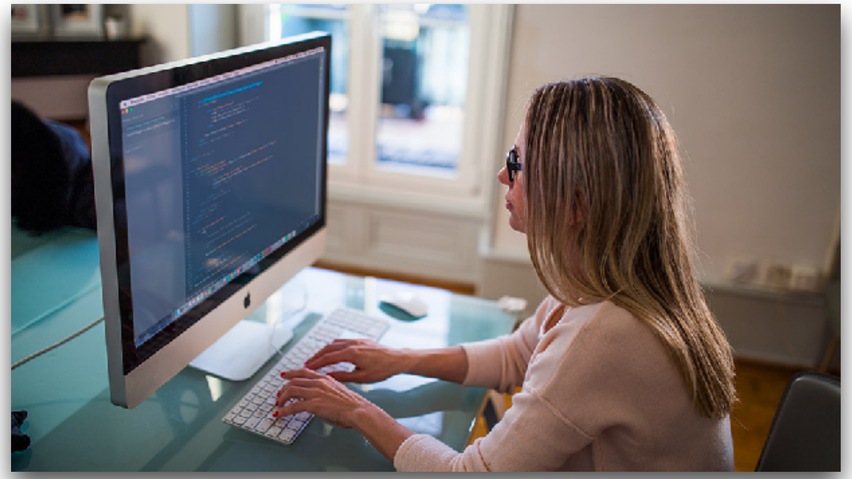


(Borrowed from Dr. Gregory Gay)

““I expect a high level of coverage. Sometimes managers require one. There is a subtle difference.”

“What’s a good code coverage to have?” by Harm Pauw
https://www.scrum.org/resources/blog/whats-good-code-coverage-have

Programmers are pretty competent.



Q: what do the programmers and the monkeys have in common when it comes to programming?

A: they write buggy code.

Test oracles are important.

Definition 2.6 (Ground Truth). *The* ground truth oracle*, $\mathcal{G}$, is a total test oracle that always gives the "right answer".*

We can now define soundness and completeness of a test oracle with respect to $\mathcal{G}$.

Definition 2.7 (Soundness). *The test oracle D is* sound *iff*

$$D(a) \Rightarrow \mathcal{G}(a).$$

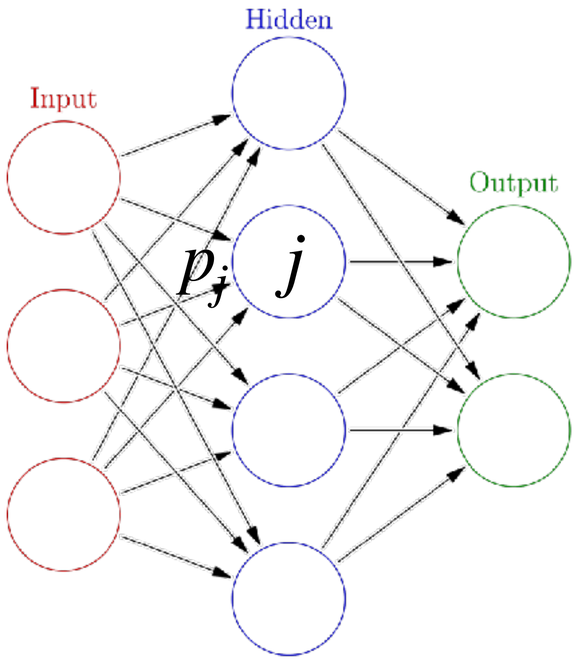
Definition 2.8 (Completeness). *The test oracle D is* complete *iff*

$$\mathcal{G}(a) \Rightarrow D(a).$$

E. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo. The oracle problem in software testing: A survey. IEEE Transactions on Software Engineering, 41(5):507–525, May 2015.

Moving onto the DNN world…

Artificial Neural Network



https://en.wikipedia.org/wiki/Artificial_neural_network#/media/File:Colored_neural_network.svg

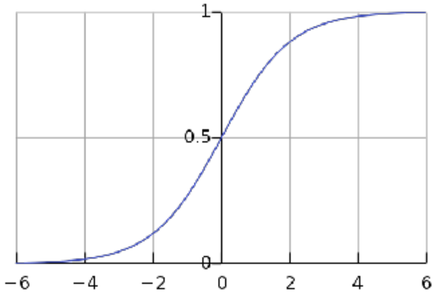
Activation: $a_j(t)$

Threshold: θ

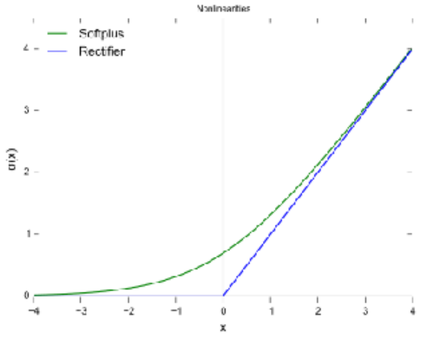
Propagation: $p_j(t) = \sum_i o_j(t) w_{ij}$

Activation function: $a_j(t+1) = f(a_j(t), p_j(t), \theta)$

Output function: $o_j(t) = f_{out}(a_j(t))$

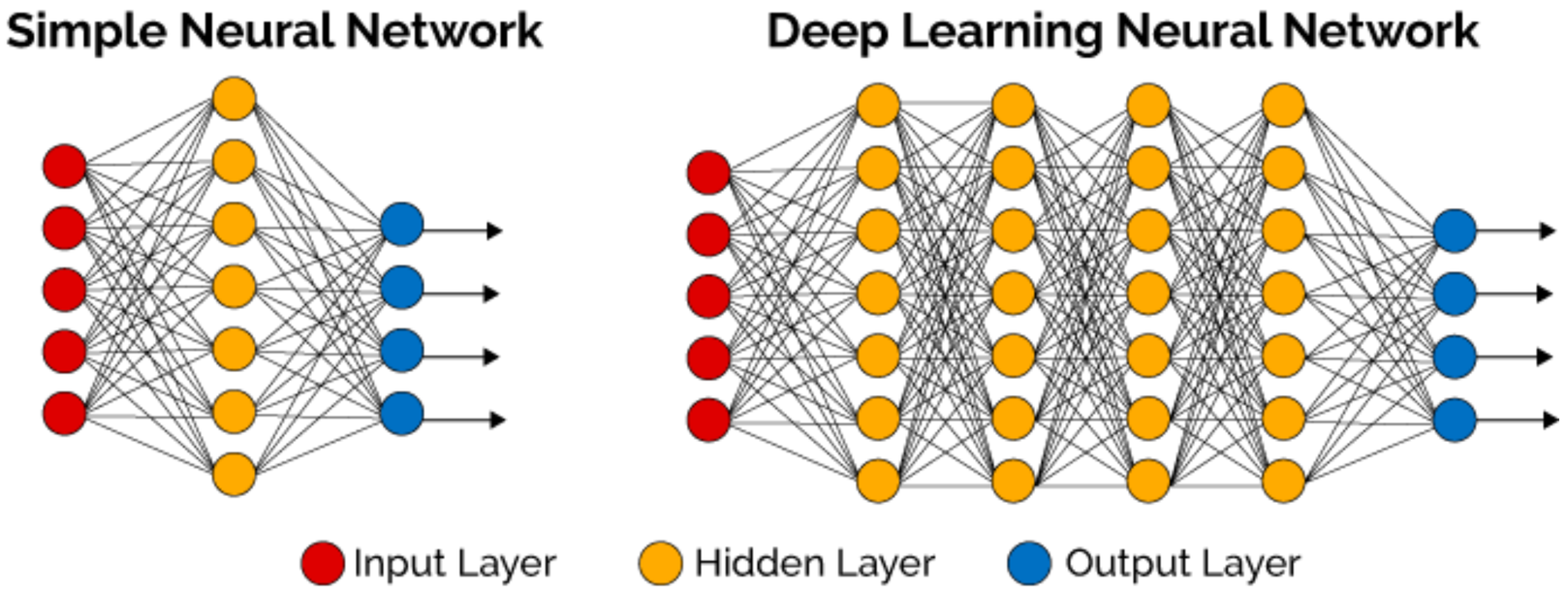


https://en.wikipedia.org/wiki/Sigmoid_function#/media/File:Logistic-curve.svg



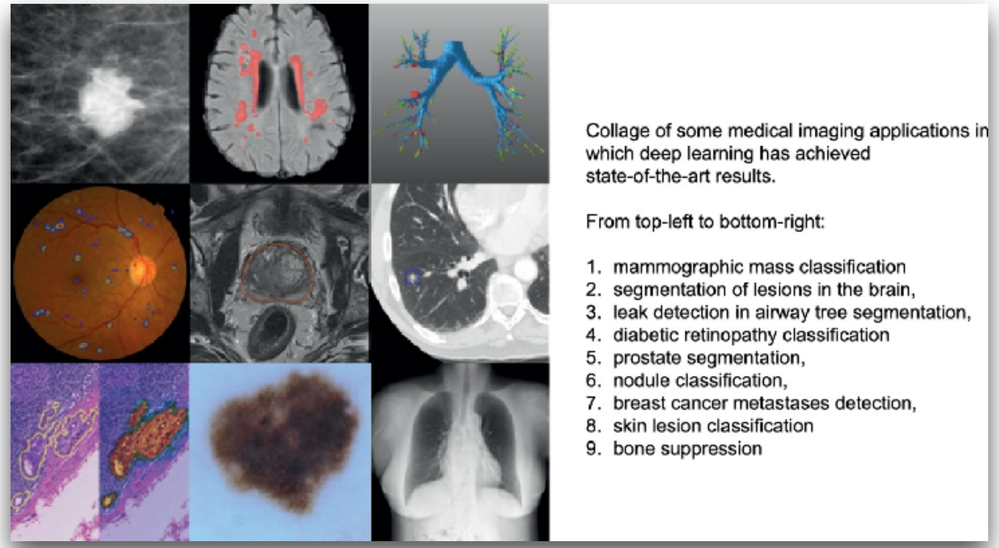
https://en.wikipedia.org/wiki/Rectifier_(neural_networks)#/media/File:Rectifier_and_softplus_functions.svg

Deep Neural Networks



Hardware parallelism (GPUs), advances in back-propagation methods,
and other innovations made DNNs surprisingly effective.

DL systems are being adopted in safety critical domains.



Umm, shouldn’t we test these?

“Tesla said shortly after the accident that the car’s sensors failed to recognize the white truck against the bright sky.”

https://www.siliconvalley.com/2016/07/26/feds-driver-in-fatal-tesla-autopilot-crash-was-speeding/

Traditional Code	DL System
Specification	Training Data
Logic as Control Flow	Logic as Data Flow
Written	Trained
Tested	Tested
For Faults	Faults?
Patched	Retrained?

Outputs are not exactly discrete 😨

```
if (a + b <= c) {
  return TriangleType.INVALID;
} else if (a == b && b == c) {
  return TriangleType.EQUALATERAL;
} else if (a == b || b == c) {
  return TriangleType.ISOCELES;
} else {
  return TriangleType.SCALENE;
}
```

vs.

$$\sigma(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \text{for } i = 1,\ldots,K \text{ and } \mathbf{z} = \left(z_1, \ldots, z_K\right) \in \mathbb{R}^K$$

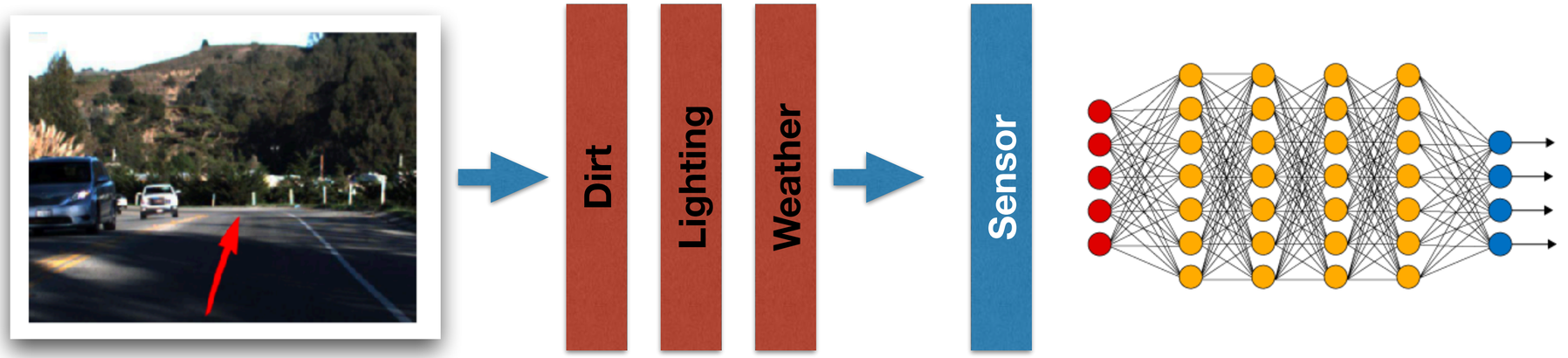
Inputs are more complicated, perhaps even stochastic 😨

```
int x = 42;
```

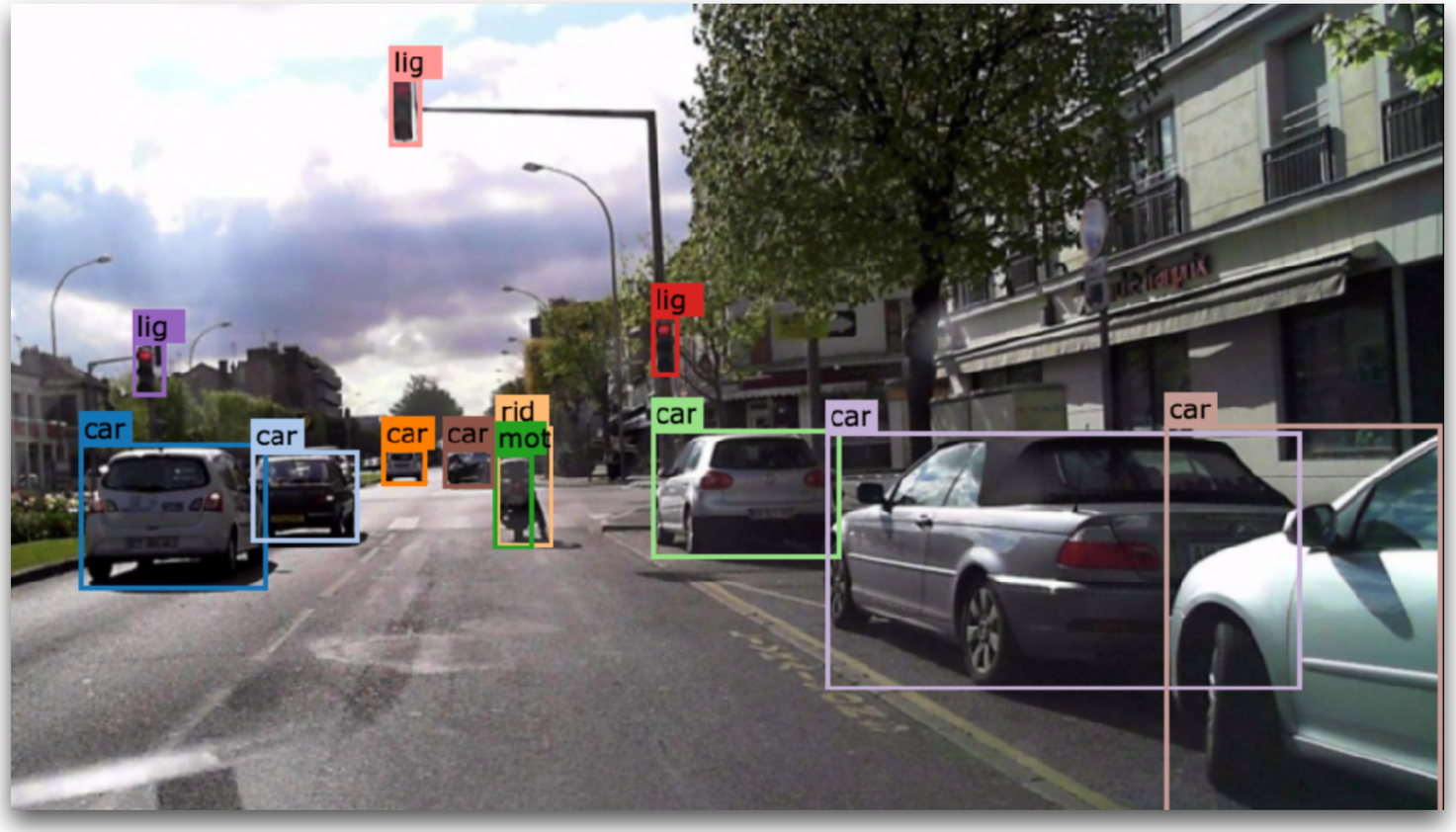
a du path

```
if(x == 42){…
```

VS.



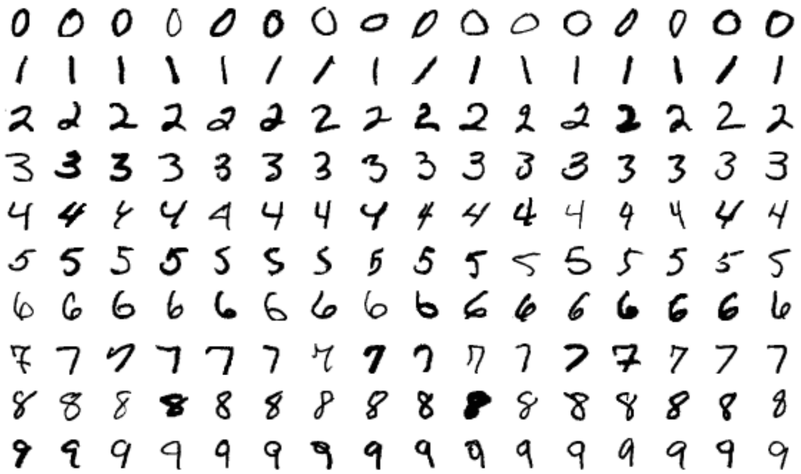
Can we (randomly) sample these inputs? 😨



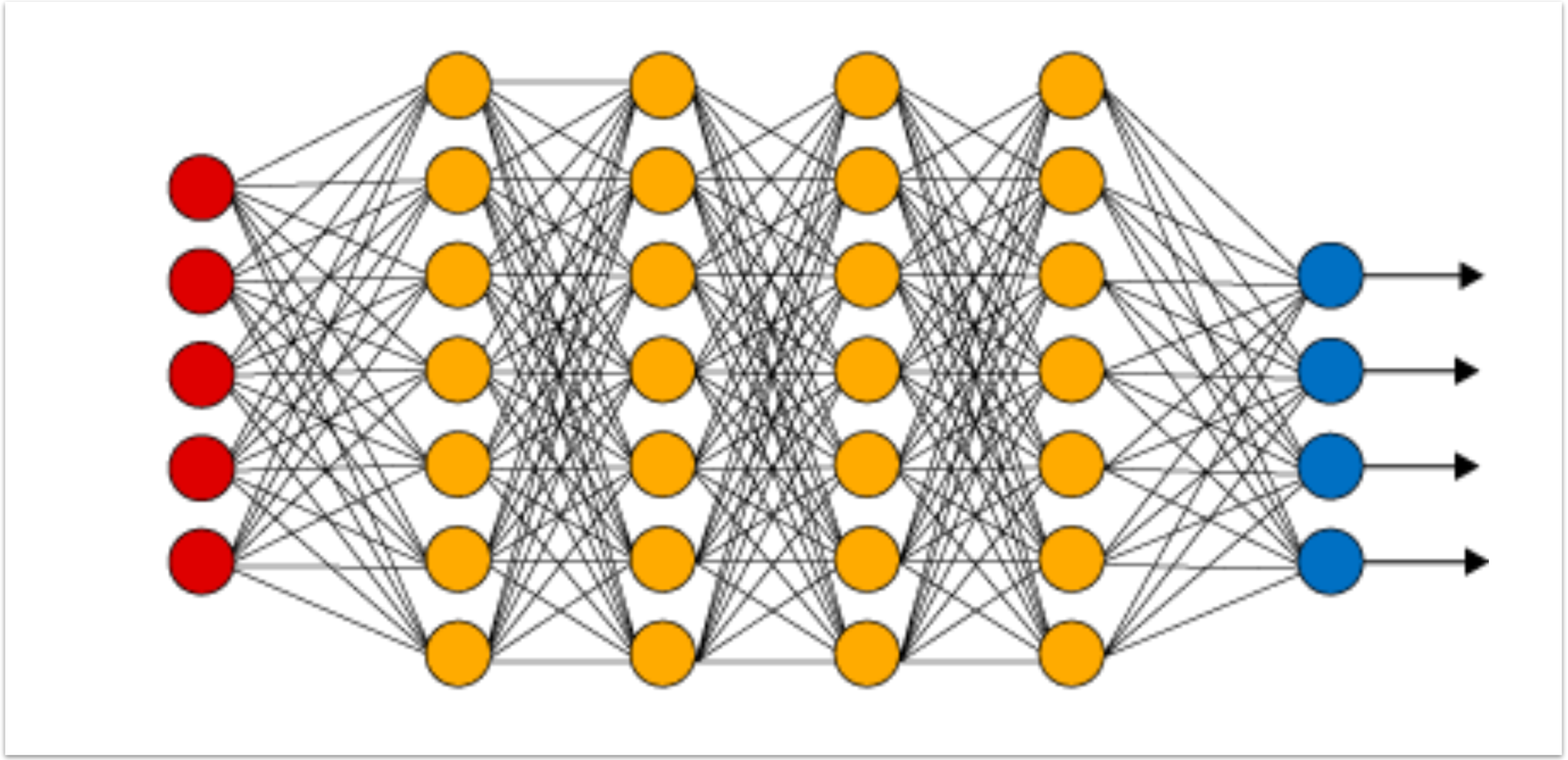
https://www.technologyreview.com/the-download/611380/researchers-have-released-the-largest-self-driving-car-data-set-yet/

Semantic Manifold Conundrum 😨

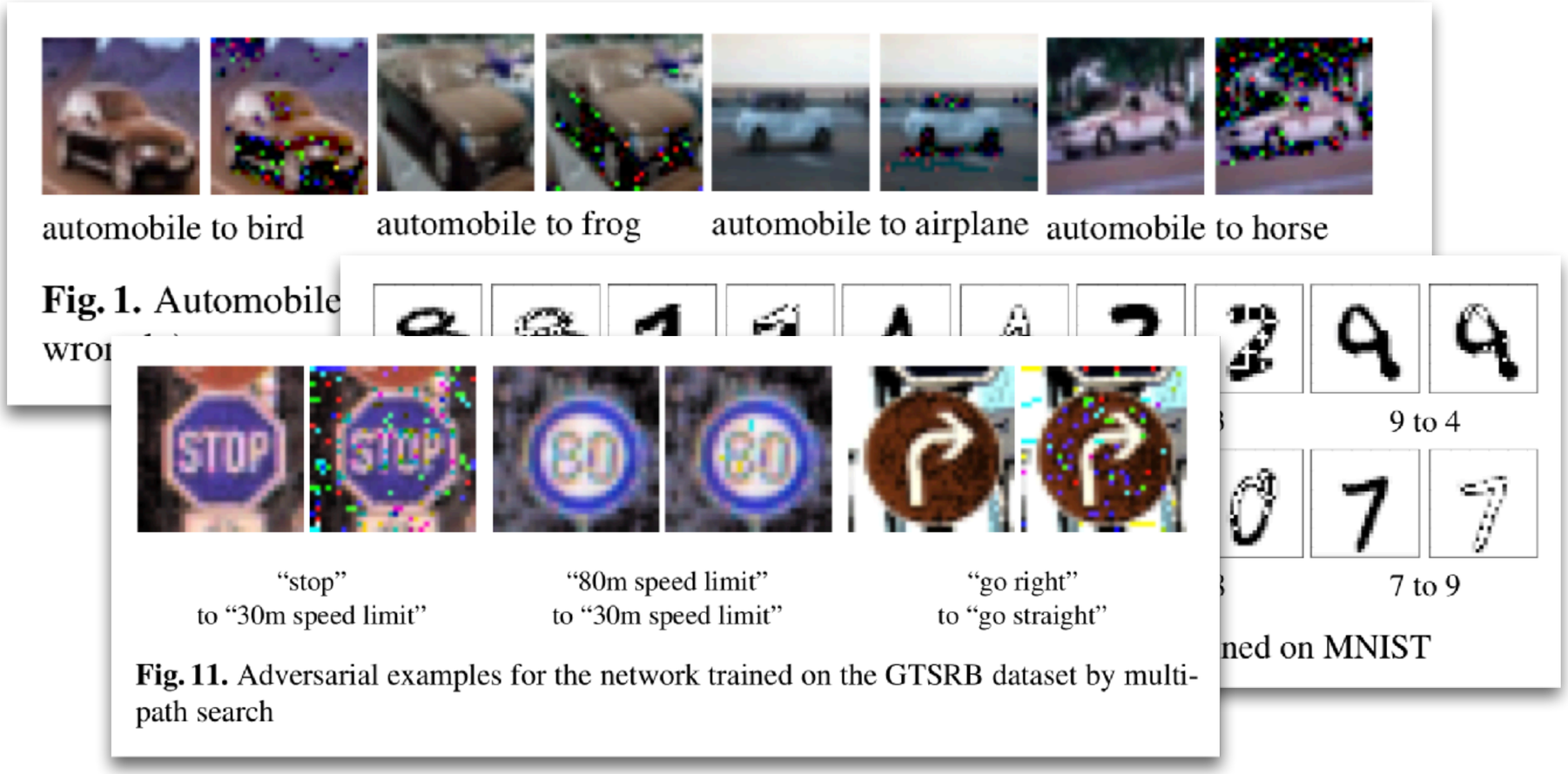
- Space of possible MNIST images (28 by 28): 2^{784}
- Space of meaningful digit images: size unknown, but much smaller than 2^{784}
- We do not know how to sample only from the manifold of meaningful digits
- In fact, DNNs perform well exactly because they approximate this manifold well



Very little to cover, at least structurally 😨



Training seems less competent than programmers 😨😨



Safety verification of deep neural networks,
iaowei Huang, Marta Kwiatkowska, Sen Wang, Min Wu (https://arxiv.org/abs/1610.06940)

Training seems less competent than programmers 😨😨😨

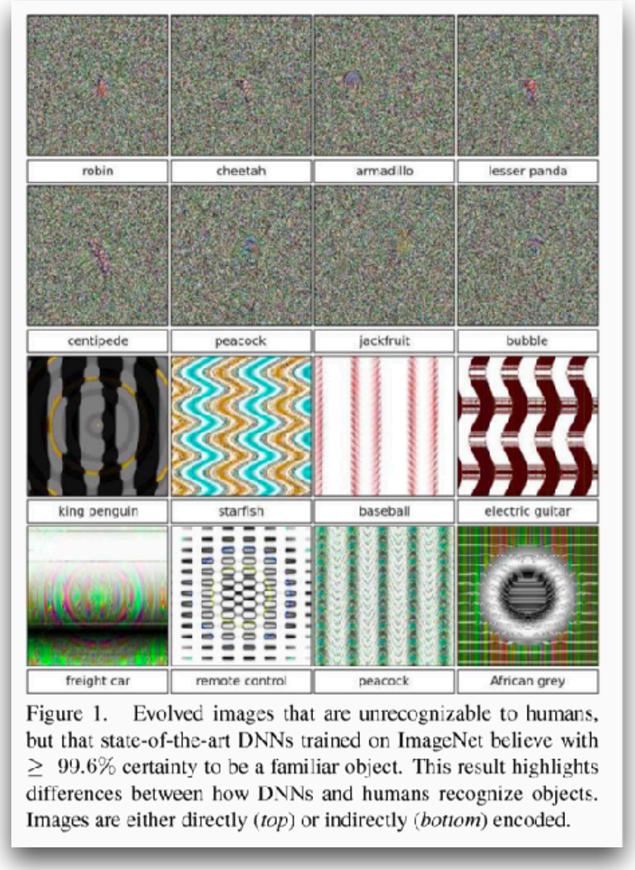


Figure 1. Evolved images that are unrecognizable to humans, but that state-of-the-art DNNs trained on ImageNet believe with $\geq$ 99.6% certainty to be a familiar object. This result highlights differences between how DNNs and humans recognize objects. Images are either directly (*top*) or indirectly (*bottom*) encoded.

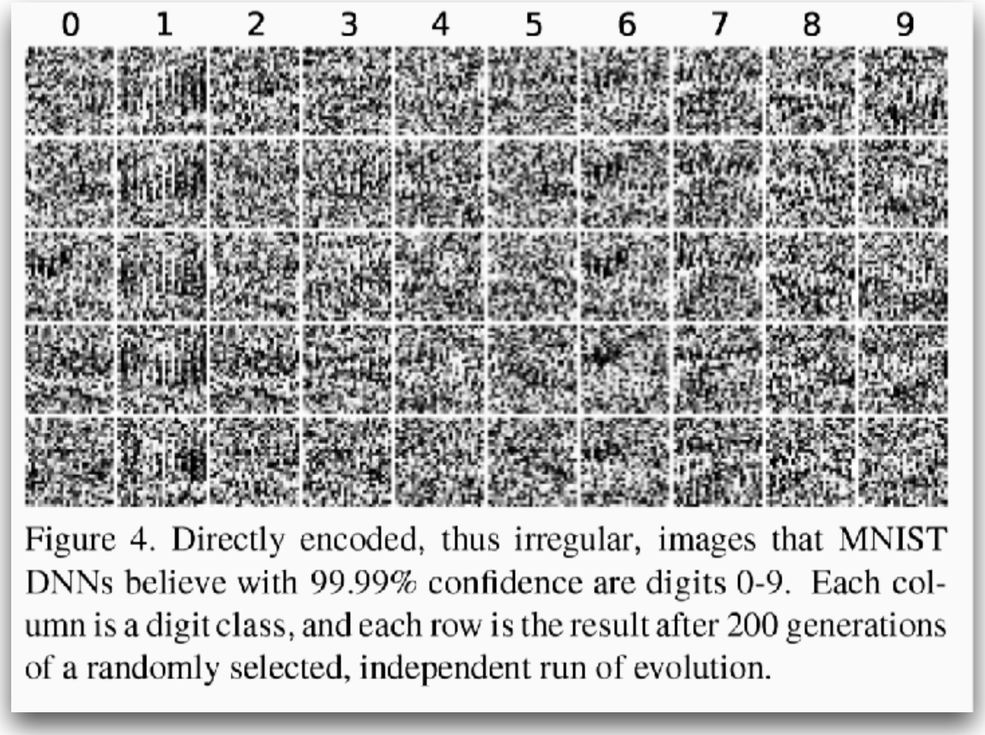


Figure 4. Directly encoded, thus irregular, images that MNIST DNNs believe with 99.99% confidence are digits 0-9. Each column is a digit class, and each row is the result after 200 generations of a randomly selected, independent run of evolution.

A. M. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence pre- dictions for unrecognizable images. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 427–436, 2015.

We will have to learn to reason about probabilistic oracles 😨

Deterministic

Definition 2.6 (Ground Truth). *The* ground truth oracle*, $\mathcal{G}$, is a total test oracle that always gives the "right answer".*

We can now define soundness and completeness of a test oracle with respect to $\mathcal{G}$.

Definition 2.7 (Soundness). *The test oracle D is* sound *iff*

$$D(a) \Rightarrow \mathcal{G}(a).$$

Definition 2.8 (Completeness). *The test oracle D is* complete *iff*

$$\mathcal{G}(a) \Rightarrow D(a).$$

Probabilistic

Definition 2.9 (Probablistic Soundness and Completeness). *A probabilistic test oracle $\tilde{D}$ is* probabilistically sound *iff*

$$P(\tilde{D}(w) = 1) > \frac{1}{2} + \epsilon \Rightarrow \mathcal{G}(w)$$

and $\tilde{D}$ is probabilistically complete *iff*

$$\mathcal{G}(w) \Rightarrow P(\tilde{D}(w) = 1) > \frac{1}{2} + \epsilon$$

where ϵ is non-negligible.

E. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo. The oracle problem in software testing: A survey. IEEE Transactions on Software Engineering, 41(5):507–525, May 2015.

Yes.

https://www.siliconvalley.com/2016/07/26/feds-driver-in-fatal-tesla-autopilot-crash-was-speeding/

“Tesla said shortly after the accident that the car’s sensors failed to recognize the white truck against the bright sky.”

Adversarial Examples

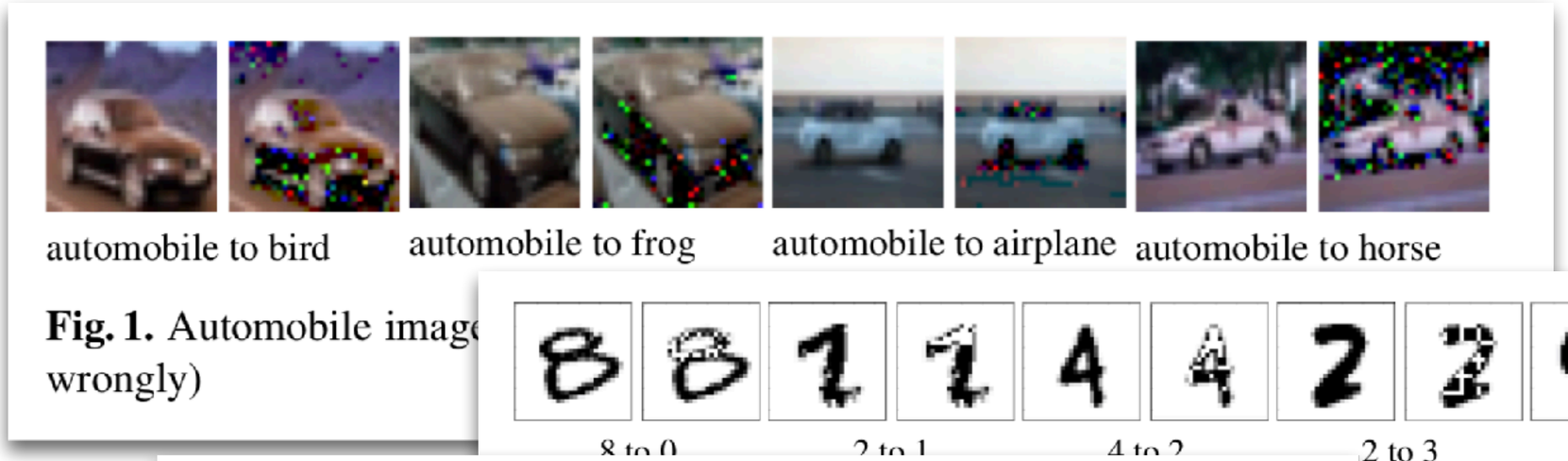


Fig. 1. Automobile image
wrongly)

"stop"
to "30m speed limit"

"80m speed limit"
to "30m speed limit"

"go right"
to "go straight"

Fig. 11. Adversarial examples for the network trained on the GTSRB dataset by multi-path search

Safety verification of deep neural networks,
Xiaowei Huang, Marta Kwiatkowska, Sen Wang, Min Wu (https://arxiv.org/abs/1610.06940)

Mantra of this course

- AKA things I went on and on and on about:
  - Oracle is a mechanism that decides whether the observed behaviour is as expected….
  - Exhaustive testing is not possible, so we rely on adequacy criteria…
  - Testing is really all about input sampling….
- Are these still all true with DNNs?

Oracle may not be deterministic 😰

- Logical behaviours are no longer discrete. Unlike branches and equivalence partitions, there is no clear boundary between decision outcomes of DNNs.
- Inputs are, to some extent, stochastic. Imagine a crossing, for which an autonomous car is supposed to stop after recognising it. Depending on sensor status, the same scene can be processed as different bits.

Going Metamorphic

Metamorphic testing is a surprisingly effective conceptual tool
for testing DNNs (at least so far).

Given that DNN() produces the output “car”,

MT suggests that DNN() should also produce the output “car”.

Input MR: images are perceptively identical to human eyes.
Output MR: class labels should be identical.

Probabilistic Oracles are harder to reason about

Deterministic

Definition 2.6 (Ground Truth). *The* ground truth oracle*, $\mathcal{G}$, is a total test oracle that always gives the "right answer".*

We can now define soundness and completeness of a test oracle with respect to $\mathcal{G}$.

Definition 2.7 (Soundness). *The test oracle D is* sound *iff*

$$D(a) \Rightarrow \mathcal{G}(a).$$

Definition 2.8 (Completeness). *The test oracle D is* complete *iff*

$$\mathcal{G}(a) \Rightarrow D(a).$$

Probabilistic

Definition 2.9 (Probablistic Soundness and Completeness). *A probabilistic test oracle $\tilde{D}$ is* probabilistically sound *iff*

$$P(\tilde{D}(w) = 1) > \frac{1}{2} + \epsilon \Rightarrow \mathcal{G}(w)$$

and $\tilde{D}$ is probabilistically complete *iff*

$$\mathcal{G}(w) \Rightarrow P(\tilde{D}(w) = 1) > \frac{1}{2} + \epsilon$$

where ϵ is non-negligible.

E. Barr, M. Harman, P. McMinn, M. Shahbaz, and S. Yoo. The oracle problem in software testing: A survey. IEEE Transactions on Software Engineering, 41(5):507–525, May 2015.

So how do we estimate the quality of testing?

Adequacy Criteria

- With traditional software, the code embodies the logic of the program. Which is why structural coverage can work as an adequacy for testing: more code being executed is correlated with more diverse logical behaviours being explored.
- DNN code DOES NOT embody the logic. In fact, hardly anything to explore.

```
import tensorflow as tf
mnist = tf.keras.datasets.mnist

(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0

model = tf.keras.models.Sequential([
  tf.keras.layers.Flatten(),
  tf.keras.layers.Dense(512, activation=tf.nn.relu),
  tf.keras.layers.Dropout(0.2),
  tf.keras.layers.Dense(10, activation=tf.nn.softmax)
])
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
```

https://medium.com/tensorflow/standardizing-on-keras-guidance-on-high-level-apis-in-tensorflow-2-0-bad2b04c819a

Test Adequacy for DNNs

- Given two sets of inputs, how do we know which one is better for testing DNNs?
- The more diverse one.

DeepXplore (SOSP 2017)

Neuron Coverage

$$\mathrm{NCov}(T, \boldsymbol{x}) = \frac{|\{n \mid \forall x \in T, \mathbf{out}(n, \boldsymbol{x}) > t\}|}{|N|}$$

Intuition: inputs that activate more nodes above a given threshold are using wider and different parts of the network, and therefore making use of a wider range of learnt features.

K. Pei, Y. Cao, J. Yang, and S. Jana. Deepxplore: Automated whitebox testing of deep learning systems. In Proceedings of the 26th Symposium on Operating Systems Principles, SOSP '17, pages 1–18, New York, NY, USA, 2017. ACM.

DeepGauge (ASE 2018)

k Multi-section Neuron Coverage (kMNC)

$$\text{KMNCov}(T,k) = \frac{\sum_{n \in N} \left| \left\{ S_i^n \mid \exists \mathbf{x} \in T : \phi(\mathbf{x}, n) \in S_i^n \right\} \right|}{k \times |N|}$$

Neuron Boundary Coverage (NBC)

$$\textbf{UpperCornerNeuron} = \left\{ n \in N \mid \exists \mathbf{x} \in T : \phi(\mathbf{x}, n) \in (\text{high}_n, +\infty) \right\}$$

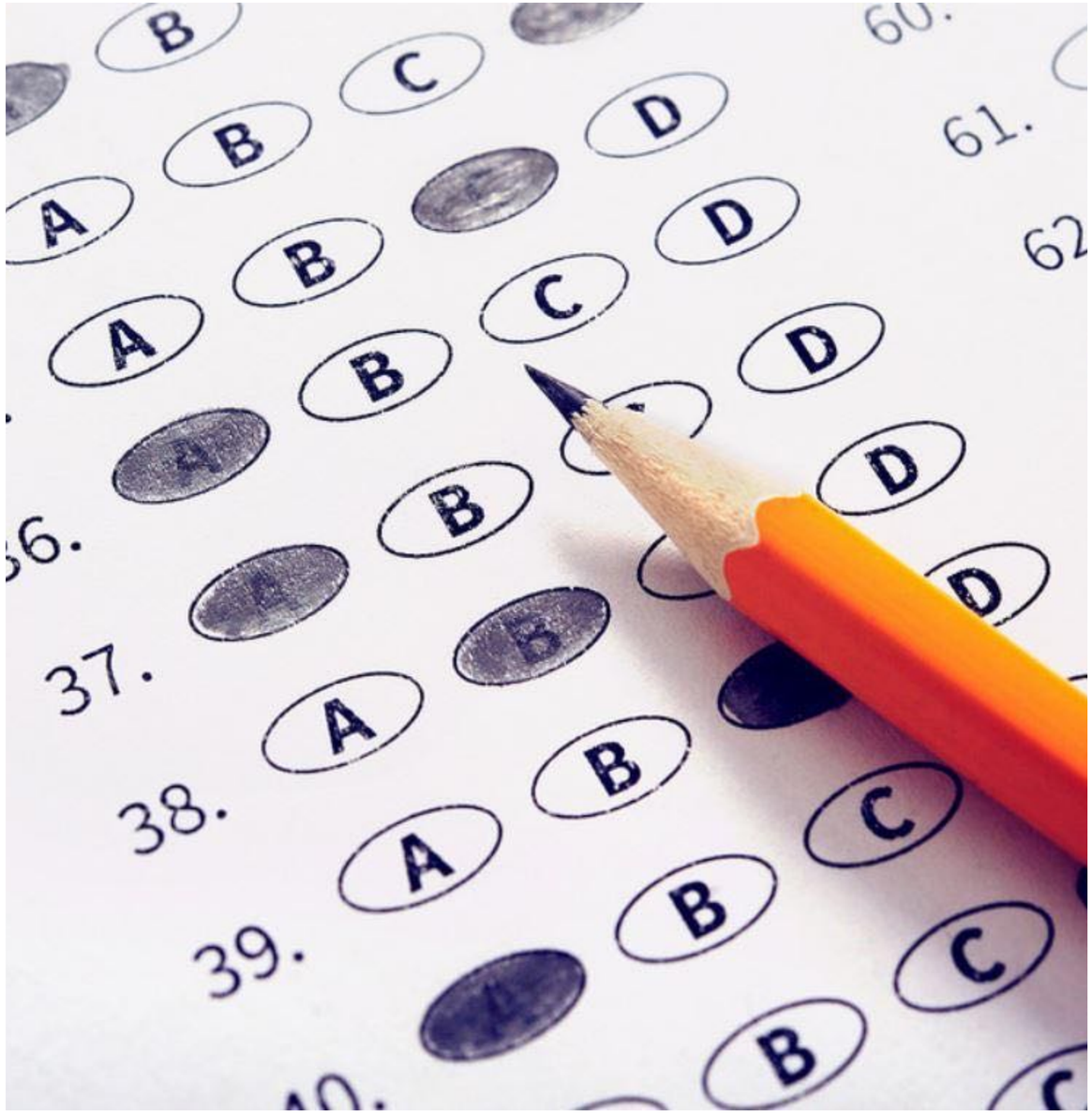
$$\textbf{LowerCornerNeuron} = \left\{ n \in N \mid \exists \mathbf{x} \in T : \phi(\mathbf{x}, n) \in (-\infty, \text{low}_n) \right\}$$

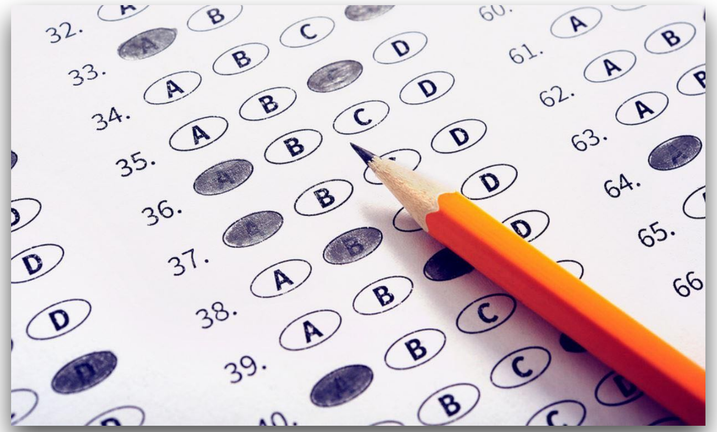
$$\text{NBCov}(T) = \frac{|\ \textbf{UpperCornerNeuron}\ | + |\ \textbf{LowerCornerNeuron}\ |}{2 \times |N|}$$

Strong Neuron Activation Coverage (SNAC)

$$\text{SNACov}(T) = \frac{|\ \textbf{UpperCornerNeuron}\ |}{|N|}$$

For (machine) learners, what is a good test of their knowledge?

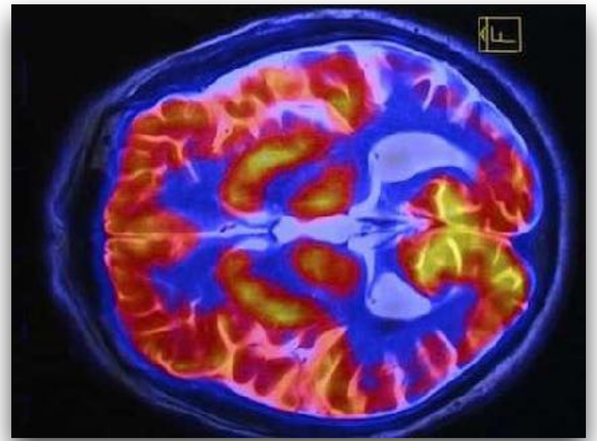




"I want to write good
test questions"

"Good questions should
challenge learner's knowledge
in diverse ways"

"So I am going to take
a brain MRI while the
learner is taking the exam!"



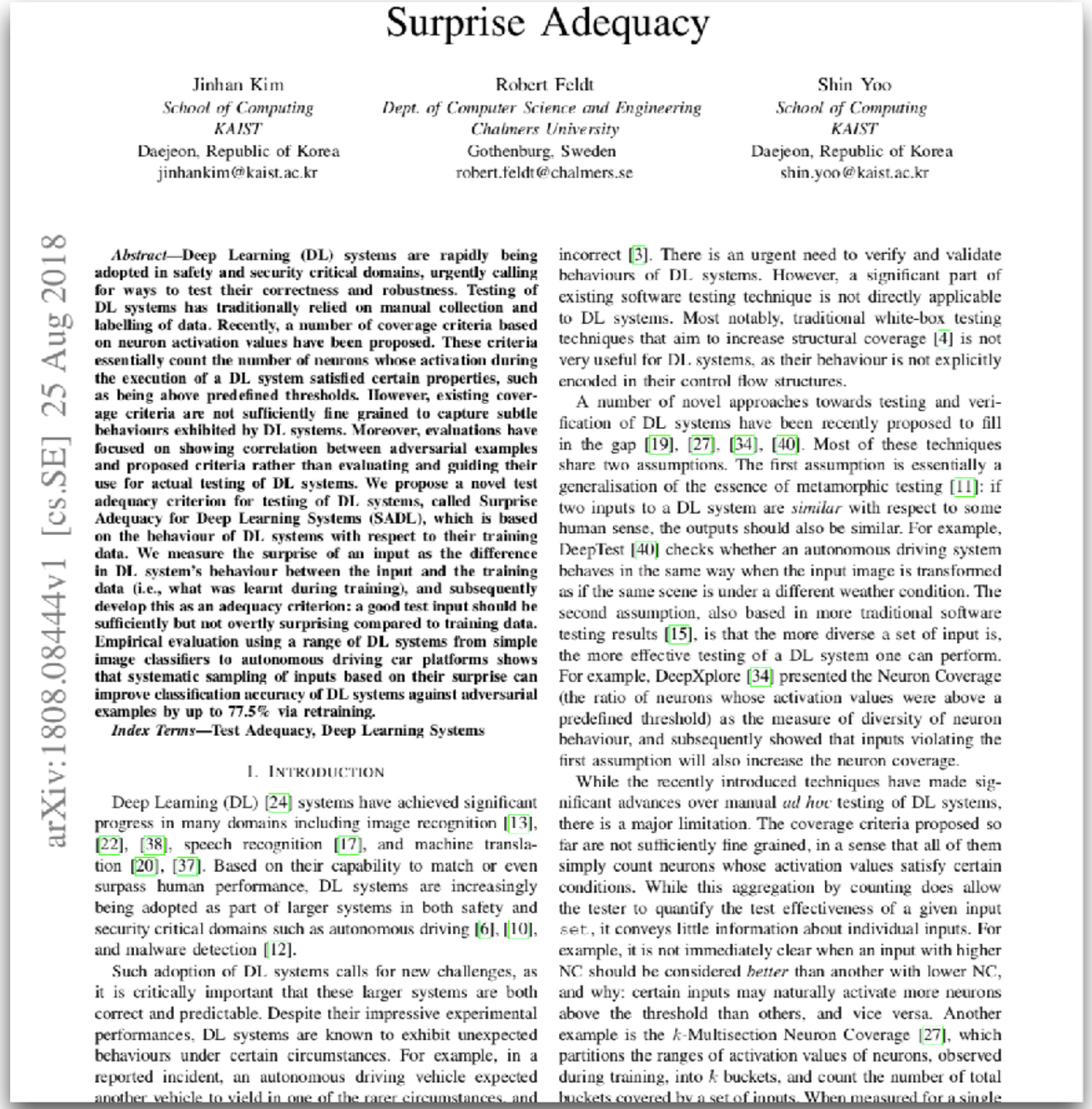
"The more areas light up,
the more diverse my questions are,
and the better they are too!"

Obvious Limits

- There is some truths in the argument, but:
- The correlation between diversity and neuron activation is very ad-hoc, and **uninterpretable**.
- You can only ever compare sets of inputs: given two individual inputs, which one is better?

Surprise Adequacy (ICSE 2019)

- Our argument: “A good exam question is one that is **reasonably surprising** to the (machine) learner: it should be **sufficiently different from exercises in the textbook**, **but not so much as to be irrelevant** to the course."



Surprise Adequacy

Jinhan Kim
School of Computing
KAIST
Daejeon, Republic of Korea
jinhankim@kaist.ac.kr

Robert Feldt
Dept. of Computer Science and Engineering
Chalmers University
Gothenburg, Sweden
robert.feldt@chalmers.se

Shin Yoo
School of Computing
KAIST
Daejeon, Republic of Korea
shin.yoo@kaist.ac.kr

***Abstract*—Deep Learning (DL) systems are rapidly being adopted in safety and security critical domains, urgently calling for ways to test their correctness and robustness. Testing of DL systems has traditionally relied on manual collection and labelling of data. Recently, a number of coverage criteria based on neuron activation values have been proposed. These criteria essentially count the number of neurons whose activation during the execution of a DL system satisfied certain properties, such as being above predefined thresholds. However, existing coverage criteria are not sufficiently fine grained to capture subtle behaviours exhibited by DL systems. Moreover, evaluations have focused on showing correlation between adversarial examples and proposed criteria rather than evaluating and guiding their use for actual testing of DL systems. We propose a novel test adequacy criterion for testing of DL systems, called Surprise Adequacy for Deep Learning Systems (SADL), which is based on the behaviour of DL systems with respect to their training data. We measure the surprise of an input as the difference in DL system's behaviour between the input and the training data (i.e., what was learnt during training), and subsequently develop this as an adequacy criterion: a good test input should be sufficiently but not overtly surprising compared to training data. Empirical evaluation using a range of DL systems from simple image classifiers to autonomous driving car platforms shows that systematic sampling of inputs based on their surprise can improve classification accuracy of DL systems against adversarial examples by up to 77.5% via retraining.**

***Index Terms*—Test Adequacy, Deep Learning Systems**

I. Introduction

Deep Learning (DL) [24] systems have achieved significant progress in many domains including image recognition [13], [22], [38], speech recognition [17], and machine translation [20], [37]. Based on their capability to match or even surpass human performance, DL systems are increasingly being adopted as part of larger systems in both safety and security critical domains such as autonomous driving [6], [10], and malware detection [12].

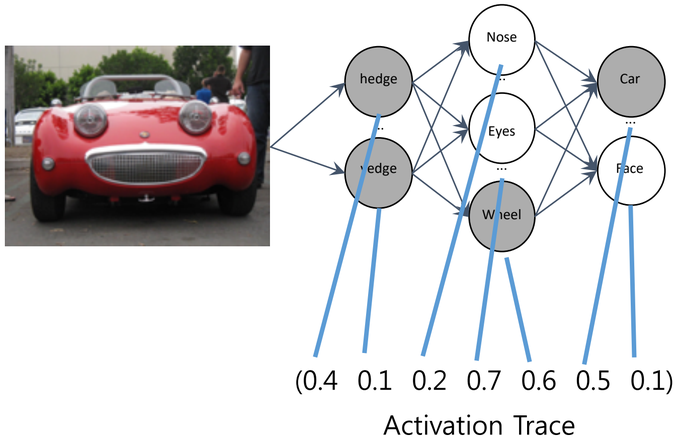
Such adoption of DL systems calls for new challenges, as it is critically important that these larger systems are both correct and predictable. Despite their impressive experimental performances, DL systems are known to exhibit unexpected behaviours under certain circumstances. For example, in a reported incident, an autonomous driving vehicle expected another vehicle to yield in one of the rarer circumstances, and incorrect [3]. There is an urgent need to verify and validate behaviours of DL systems. However, a significant part of existing software testing technique is not directly applicable to DL systems. Most notably, traditional white-box testing techniques that aim to increase structural coverage [4] is not very useful for DL systems, as their behaviour is not explicitly encoded in their control flow structures.

A number of novel approaches towards testing and verification of DL systems have been recently proposed to fill in the gap [19], [27], [34], [40]. Most of these techniques share two assumptions. The first assumption is essentially a generalisation of the essence of metamorphic testing [11]: if two inputs to a DL system are *similar* with respect to some human sense, the outputs should also be similar. For example, DeepTest [40] checks whether an autonomous driving system behaves in the same way when the input image is transformed as if the same scene is under a different weather condition. The second assumption, also based in more traditional software testing results [15], is that the more diverse a set of input is, the more effective testing of a DL system one can perform. For example, DeepXplore [34] presented the Neuron Coverage (the ratio of neurons whose activation values were above a predefined threshold) as the measure of diversity of neuron behaviour, and subsequently showed that inputs violating the first assumption will also increase the neuron coverage.

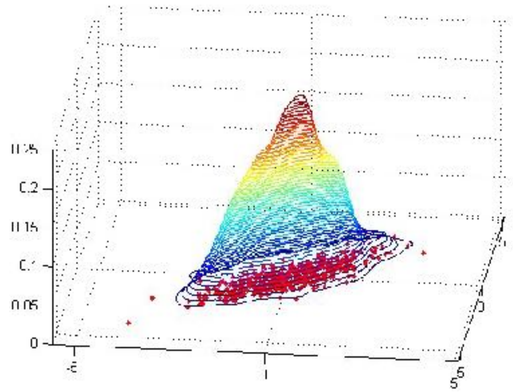
While the recently introduced techniques have made significant advances over manual *ad hoc* testing of DL systems, there is a major limitation. The coverage criteria proposed so far are not sufficiently fine grained, in a sense that all of them simply count neurons whose activation values satisfy certain conditions. While this aggregation by counting does allow the tester to quantify the test effectiveness of a given input `set`, it conveys little information about individual inputs. For example, it is not immediately clear when an input with higher NC should be considered *better* than another with lower NC, and why: certain inputs may naturally activate more neurons above the threshold than others, and vice versa. Another example is the k-Multisection Neuron Coverage [27], which partitions the ranges of activation values of neurons, observed during training, into k buckets, and count the number of total buckets covered by a set of inputs. When measured for a single

Caveats

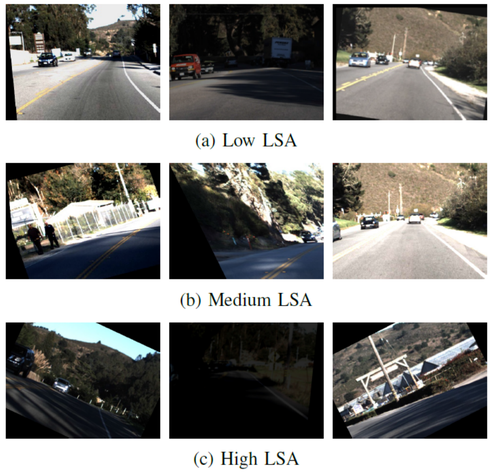
- Unlike academic exams, the concept of “reasonably different” is hard to quantify for ML inputs. More importantly, automating that judgement is even harder.
- So it turns out we cannot completely escape the MRI-like nature of our testing approach. That is, we do access the internals of DNNs.



Learnt Knowledge
(from training data)



Summarisation
(KDE or point-cloud)

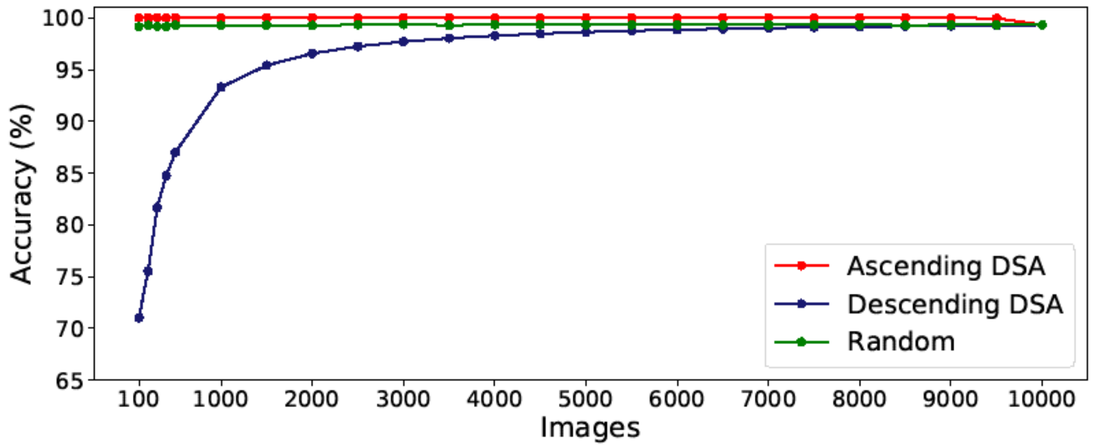


(a) Low LSA

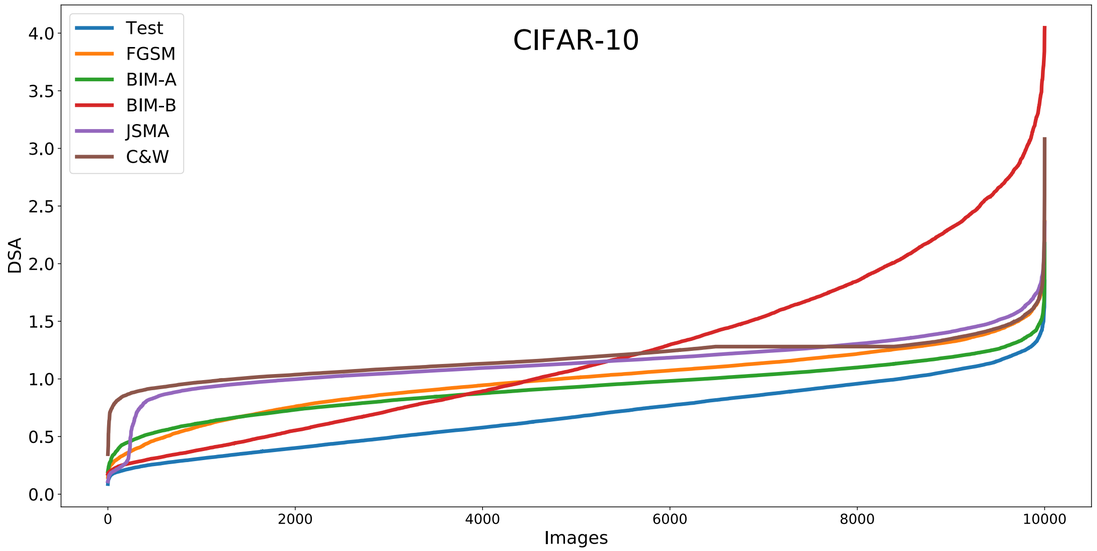
(b) Medium LSA

(c) High LSA

Quantitative Surprise Measure of New Input
Against the Summarisation



More surprising questions
are harder to answer correctly.



Trick questions (=adversarial
examples) are very surprising.

What are the actual benefits?

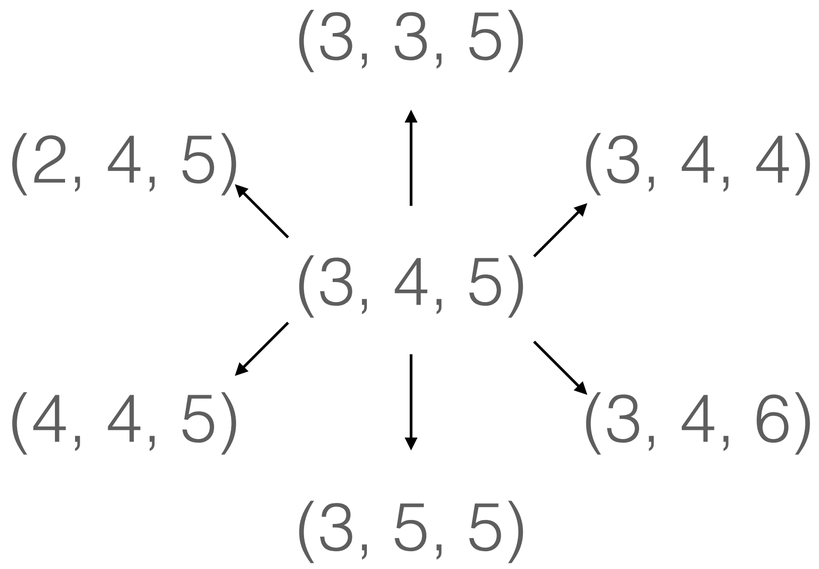
- In the ML context, the actual model correctness for an input is a given concept - you compare it to the label.
- In the SE context, the actual model correctness for an input is **EXPENSIVE**, as you have to manually label any new input!
    - SA has been successfully applied to semantic object segmentation (for autonomous driving) or NLP tasks such as question answering: both are tasks that require high labelling cost.

What about input sampling?

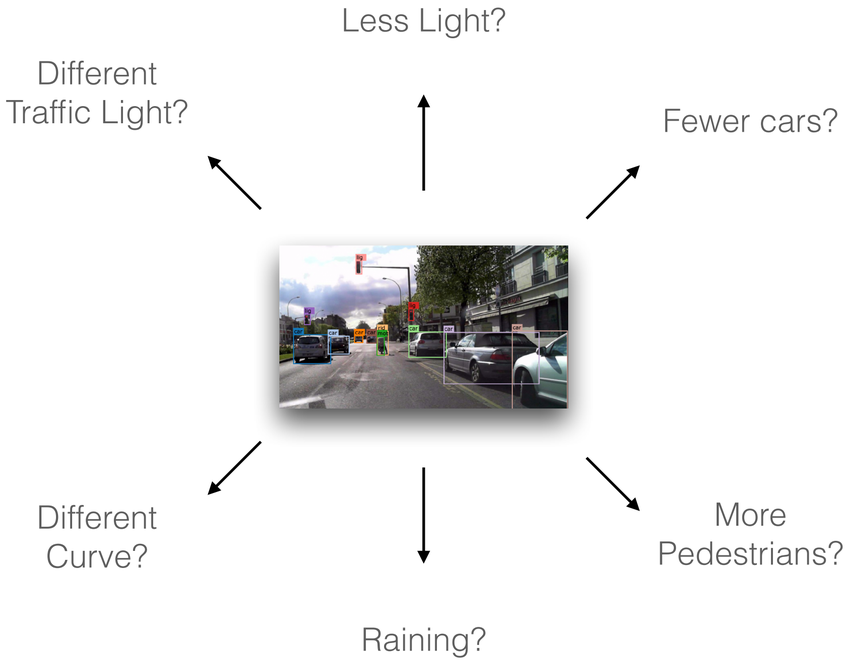
Input Sampling/Search

- In many cases, interesting applications of DNNs handle sensory input (image and audio) or highly unstructured input (natural language).
  - What is a random scene on a road? How do we sample it?
  - What is a random sentence? What is a neighbour of that sentence?
- Random sampling is not so easy now. Many benchmarks are manually generated (not only labels!).

Taming Dimensionality



Primitive Type Neighbourhood



Perceptive Input Neighbourhood

Extreme Sparsity in Input Space

- MNIST dataset of handwritten digits contains 28 by 28 black and white pixels (784 pixels).
  - The number of all possible inputs: 2^{784}
  - The number of all recognisable digit images: ? but probably much smaller than 2^{784}
- What will happen if we ignore the manifold of meaningful images?

Deep Neural Networks are Easily Fooled: High Confidence Predictions for Unrecognizable Images (CVPR 2015)

Use Genetic Algorithm to generate an imagethat is classified with high confidence

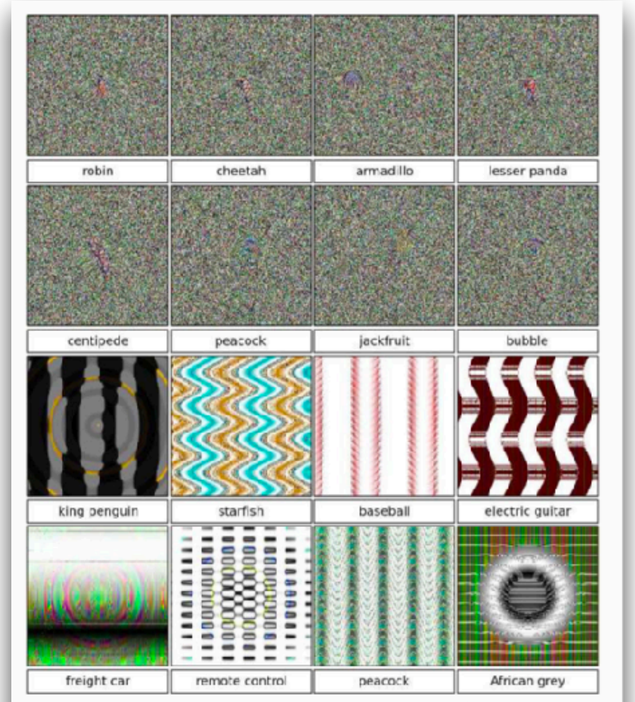


Figure 1. Evolved images that are unrecognizable to humans, but that state-of-the-art DNNs trained on ImageNet believe with $\geq$ 99.6% certainty to be a familiar object. This result highlights differences between how DNNs and humans recognize objects. Images are either directly (*top*) or indirectly (*bottom*) encoded.

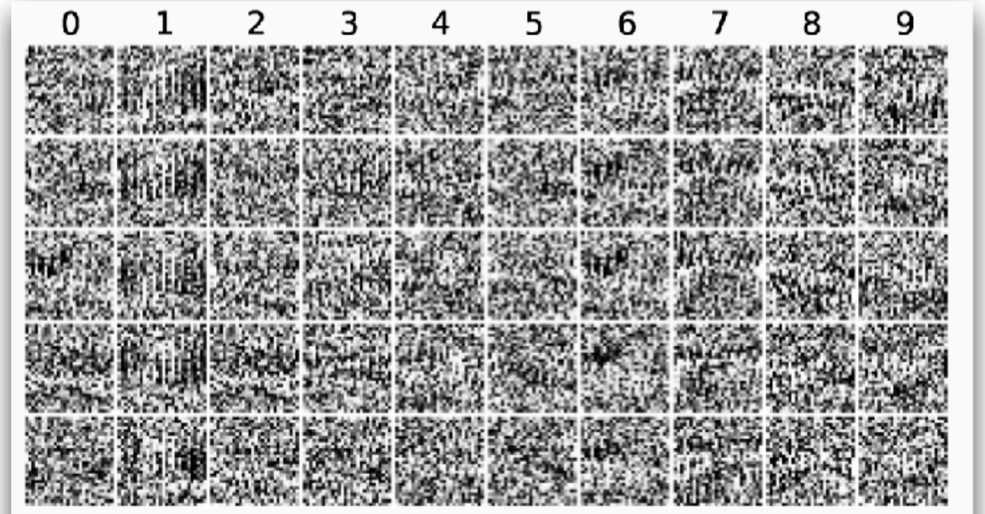
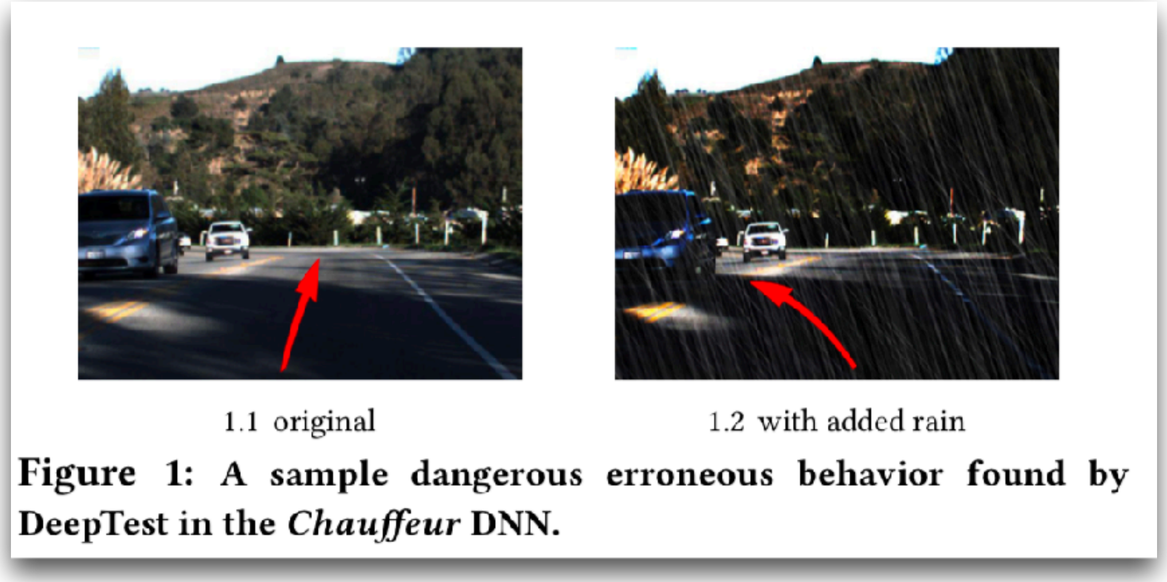


Figure 4. Directly encoded, thus irregular, images that MNIST DNNs believe with 99.99% confidence are digits 0-9. Each column is a digit class, and each row is the result after 200 generations of a randomly selected, independent run of evolution.

A. M. Nguyen, J. Yosinski, and J. Clune. Deep neural networks are easily fooled: High confidence pre- dictions for unrecognizable images. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 427–436, 2015.

DeepTest (ICSE 2018)



1.1 original 1.2 with added rain

Figure 1: A sample dangerous erroneous behavior found by DeepTest in the *Chauffeur* DNN.

A neighbour of a clear weather road scene
is the same road in a rainy day.

In other words, DeepTest uses weather condition effects
(photoshopped) as a metamorphic relationship on inputs.

Y. Tian, K. Pei, S. Jana, and B. Ray. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In Proceedings of the 40th International Conference on Software Engineering, ICSE '18, pages 303–314, New York, NY, USA, 2018. ACM.

DeepRoad (ASE 2018)

(a) Snow

(b) Rain

Figure 2: Snowy and rainy scenes synthesized by DeepRoad

We really had this coming: they apply GAN to mimic weather condition.

M. Zhang, Y. Zhang, L. Zhang, C. Liu, and S. Khurshid. Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems. In Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering, ASE 2018, pages 132–142, New York, NY, USA, 2018. ACM.

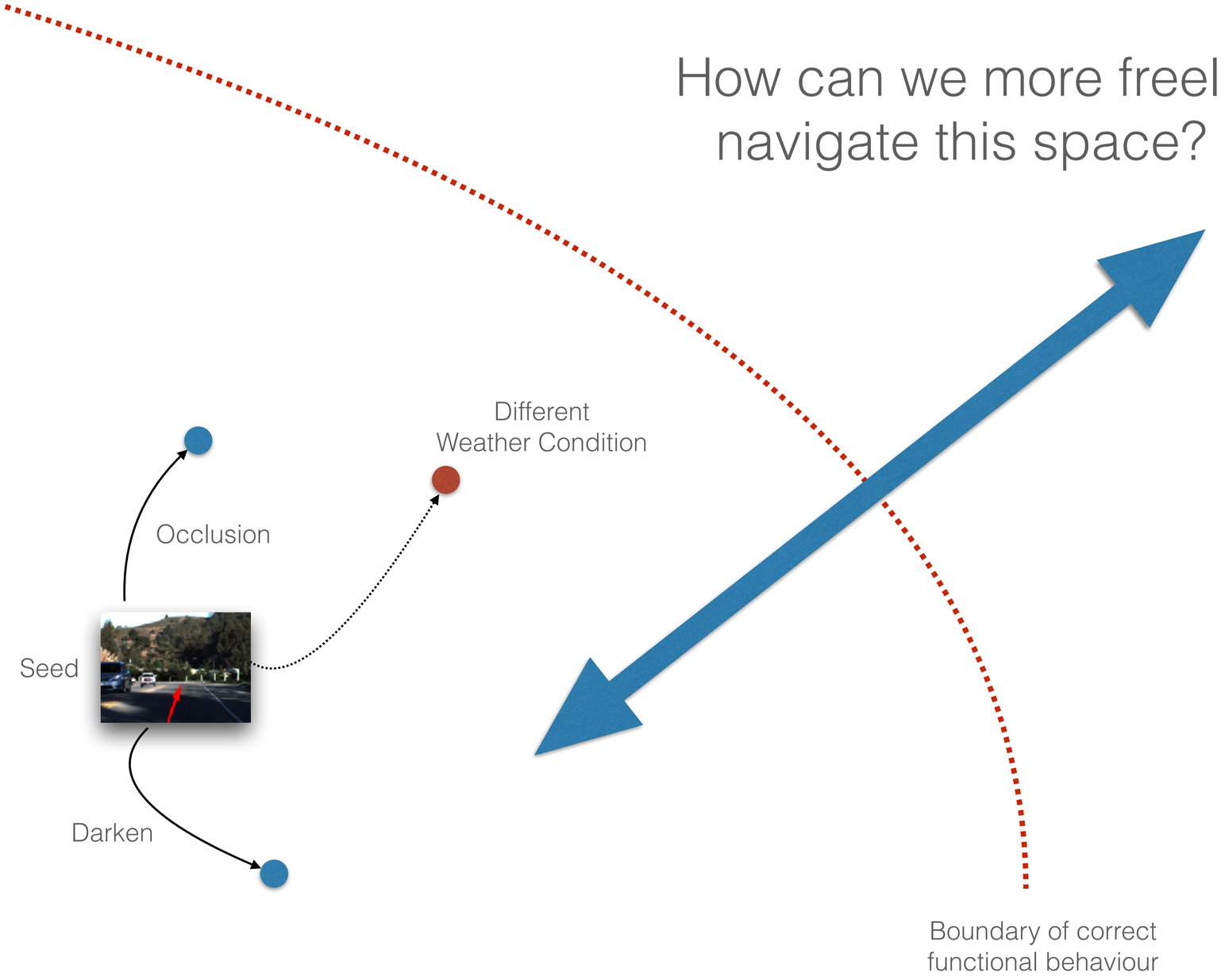
So far…

- Right or wrong (oracle) is not clear, and
- We do not know which adequacy criteria to target, and
- Searching for input is not as easy as with simpler data.
- Any good news?

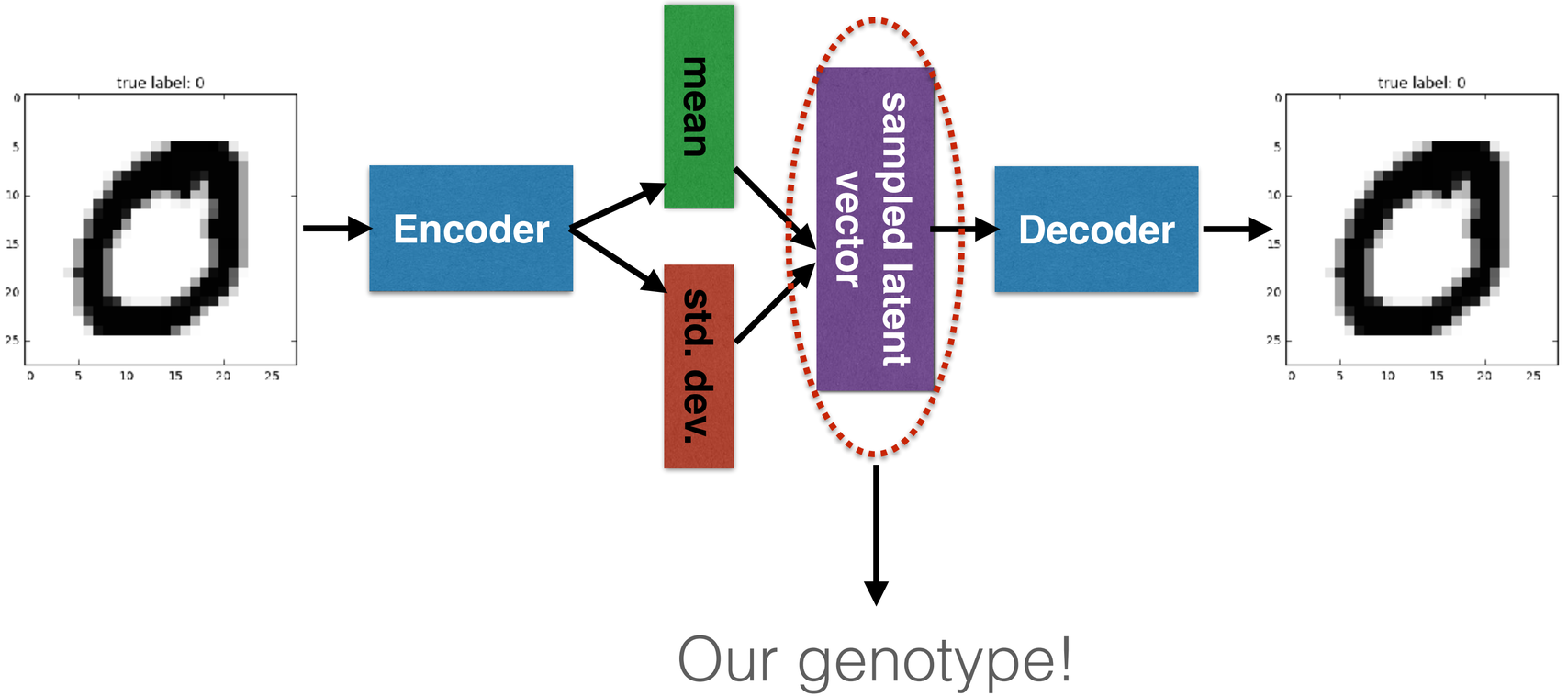
Everything becomes a number

- All internal states are numbers.
- Execution path can be captured as a numeric vector.
- Concepts like distance and similarity become much easier to handle against all types of inputs.

How can we more freely navigate this space?

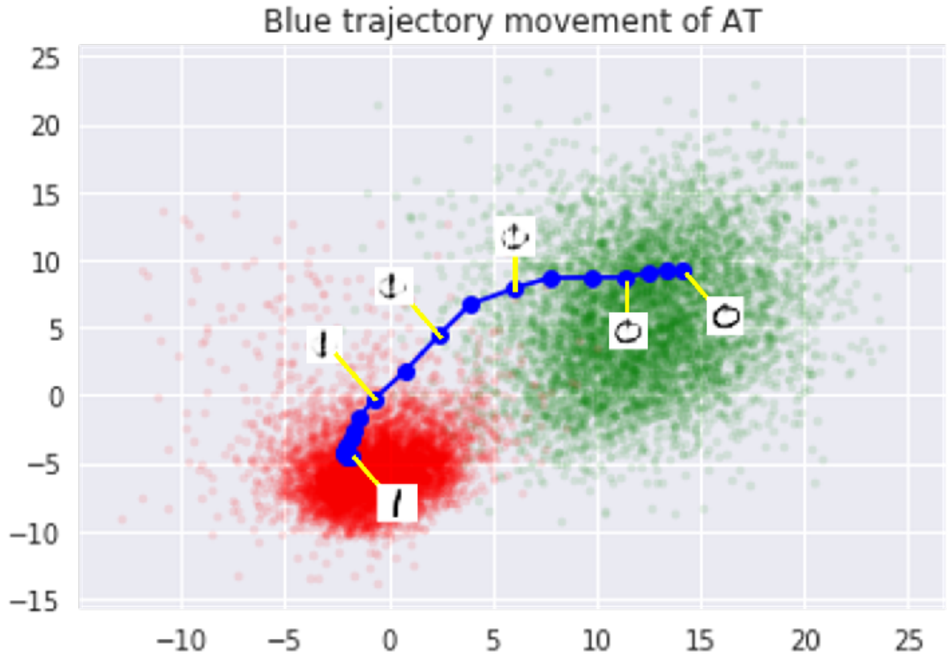


Variational Auto-Encoders (VAEs)



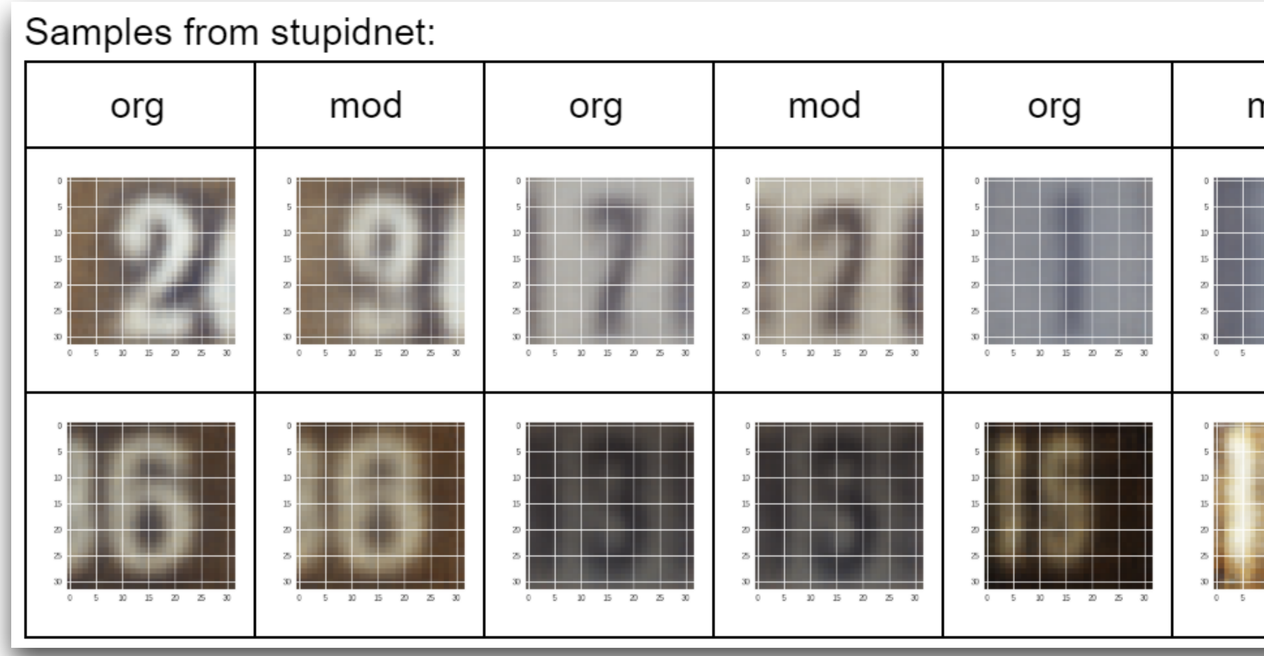
VAE + GA = Search Based Input Data Generation

- Representation: a latent vector that fits our VAE
- Fitness: Surprise Adequacy of the image decoded from a candidate solution (i.e., a latent vector)
- We visualise the search trajectory using Activation Trace (i.e., the output of a specific layer of the DNN) and PCA



Search-based Differential Testing

- Search in the latent space using VAE
- Fitness is the difference in results between a stupidnet and a properly trained ResNet
- Start from the original image, search for the modified image



Summary

- DNNs require total rethinking of how we test software.
- Despite a very fast moving research landscape, we are still in a very early stage, with many open questions.
- If you are interested in applied AI/ML, from a very unique angle, perhaps…? 🧐