

# Safety and Reliability for Learning Systems

Rüdiger Ehlers, Clausthal University of Technology

Marktoberdorf Summer School, August 2019

# Course content

## Motivation

- 1 Why do we use learning systems?
- 2 Why do we need to verify them (or do other types of quality assurance)?

## Core content

- 1 A formal methods view on machine learning
- 2 Verifying learned models (with MILP solvers)
- 3 Enforcing the correctness of systems that adapt at runtime
- 4 Overview of some latest approaches on these problems

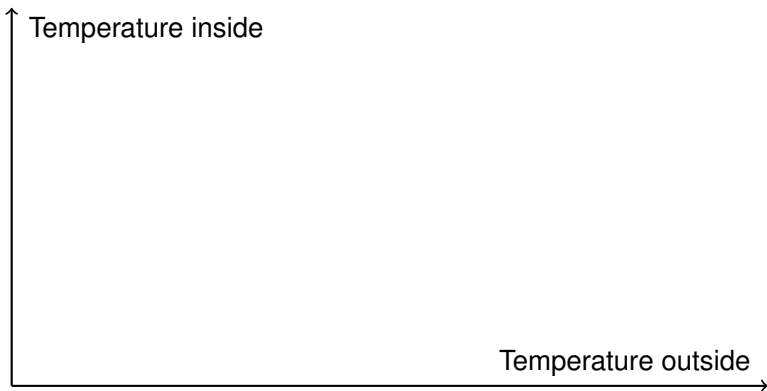
## Closing

- 1 Outlook

A stylized illustration of a glowing yellow lightbulb with grey rays emanating from its top. The lightbulb is bisected by a horizontal brown band. The word "Motivation" is written in blue, bold, sans-serif font across the center of this band.

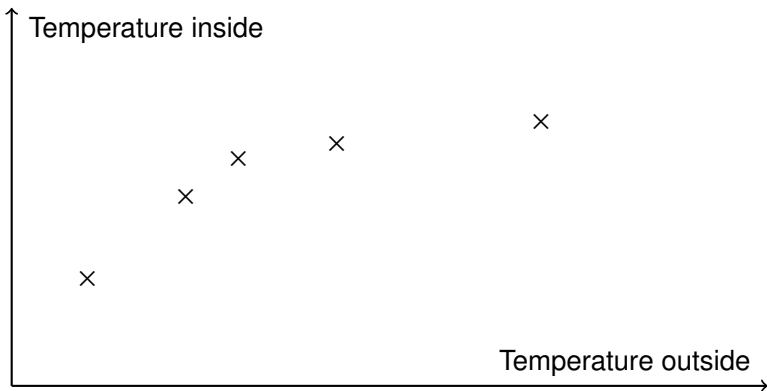
**Motivation**

## Why do we use learning systems?



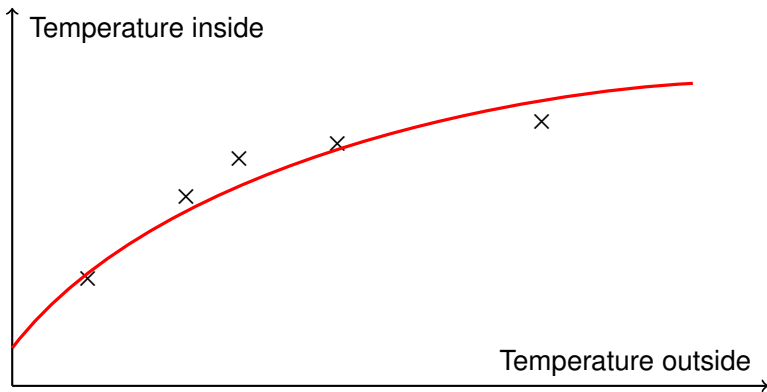
Example for using machine learning/regression to derive the relationship between outside and inside temperature of a house.

## Why do we use learning systems?



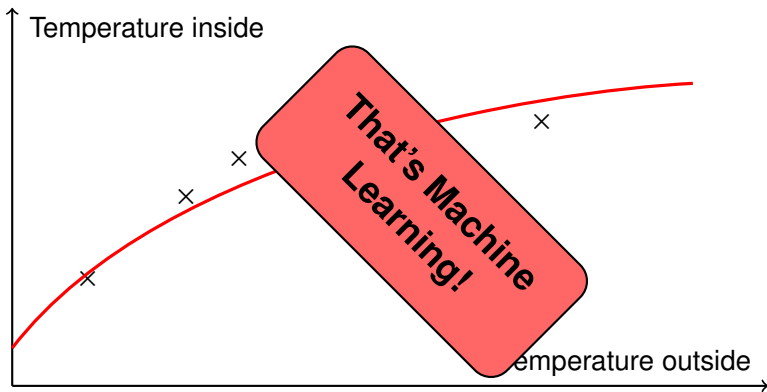
Example for using machine learning/regression to derive the relationship between outside and inside temperature of a house.

# Why do we use learning systems?



Example for using machine learning/regression to derive the relationship between outside and inside temperature of a house.

## Why do we use learning systems?



Example for using machine learning/regression to derive the relationship between outside and inside temperature of a house.

## Why do we use learning systems?

We may want to use a learning system if...

...we want to capture real-world phenomena

- 1 that are too difficult to describe,
- 2 that do not have a closed-form formal model,
- 3 that are at least partially unknown, or
- 4 for which we do not have the computational power for an engineered solution.

## Example: Too difficult to describe

### Example

We search for a function  $f$  that detects *digits* from images.

$$f\left(\begin{array}{c} \blacksquare \\ 5 \\ \blacksquare \end{array}\right) = 5$$

### Training data



## Example: No closed-form formal model

Orr and  
Westenskow,  
1994

# Environment partially unknown

Gu et al. (2016)

## Example: Lack of computational resources

### ACAS Xu

Source: Julian et al., 2018

### Learning problem

#### Collision avoidance advisory:

←, ?, √, ?, →

#### Input:

- The 5 values on the left
- Last advisory
- Time until loss of vertical separation of the aircraft

### Motivation

The advisory table with discretized input values takes 2 GB of memory → too much (Katz et al., 2017)

## Why the need for verification?

### Quality assurance

In safety-critical contexts, *correctness* properties need to be satisfied (think of self-driving cars!)

In contexts that are not safety critical, formal verification can be used to check if the learning process yielded something useful.

## Why the need for verification?

### Quality assurance

In safety-critical contexts, *correctness* properties need to be satisfied (think of self-driving cars!)

In contexts that are not safety critical, formal verification can be used to check if the learning process yielded something useful.

### But does this make sense?

If we had good specifications, we wouldn't need machine learning in the first place, right?

## Why does it make sense to verify?

<b>Reason for using ML</b>	<b>What can be verified?</b>
RWP too difficult to describe	Some boundary conditions that are known
RWP does not have a closed-form formal model	
RWP at least partially unknown	Properties of the RWP that <i>are</i> known
We do not have the computational power for an engineered solution	We often have a part of the specification

RWP = Real-world phenomena

## Example for verification

ACAS Xu, Katz et al., 2017, Property 2

*If the intruder is distant and is significantly slower than ownship, the advisory should be ✓.*

## Example for verification

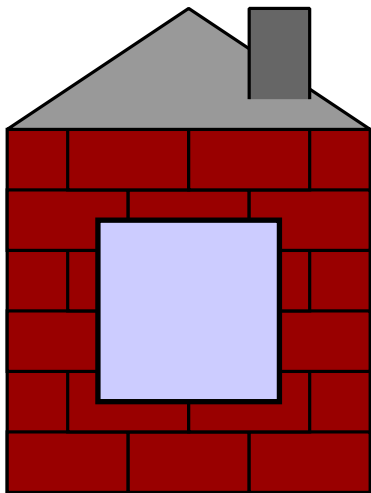
ACAS Xu, Katz et al., 2017, Property 2

*If the intruder is distant and is significantly slower than ownship, the advisory should be ✓.*

Formalization

$(\rho \geq 55947.691 \wedge v_{own} \geq 1145 \wedge v_{int} \leq 60) \rightarrow output = \checkmark$

## Take-home messages



### Machine learning...

... is useful in some application domains due to a lack of a precise model or it being too complex to handle.

### Verifying learned models...

... makes sense whenever there are known *boundary conditions* for the learned model.

# References I

- Shixiang Gu, Ethan Holly, Timothy P. Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation. *CoRR*, abs/1610.00633, 2016. URL <http://arxiv.org/abs/1610.00633>.
- Kyle D. Julian, Mykel J. Kochenderfer, and Michael P. Owen. Deep neural network compression for aircraft collision avoidance systems. *CoRR*, abs/1810.04240, 2018. URL <http://arxiv.org/abs/1810.04240>.
- Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, and Mykel J. Kochenderfer. Reluplex: An efficient SMT solver for verifying deep neural networks. In *Computer Aided Verification - 29th International Conference, CAV 2017, Heidelberg, Germany, July 24-28, 2017, Proceedings, Part I*, pages 97–117, 2017. doi: 10.1007/978-3-319-63387-9\_5. URL [https://doi.org/10.1007/978-3-319-63387-9\\_5](https://doi.org/10.1007/978-3-319-63387-9_5).
- Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010. URL <http://yann.lecun.com/exdb/mnist/>.
- Joseph A. Orr and Dwayne R. Westenskow. A breathing circuit alarm system based on neural networks. *Journal of Clinical Monitoring*, 10(2):101–109, Mar 1994. ISSN 1573-2614. doi: 10.1007/BF02886822. URL <https://doi.org/10.1007/BF02886822>.