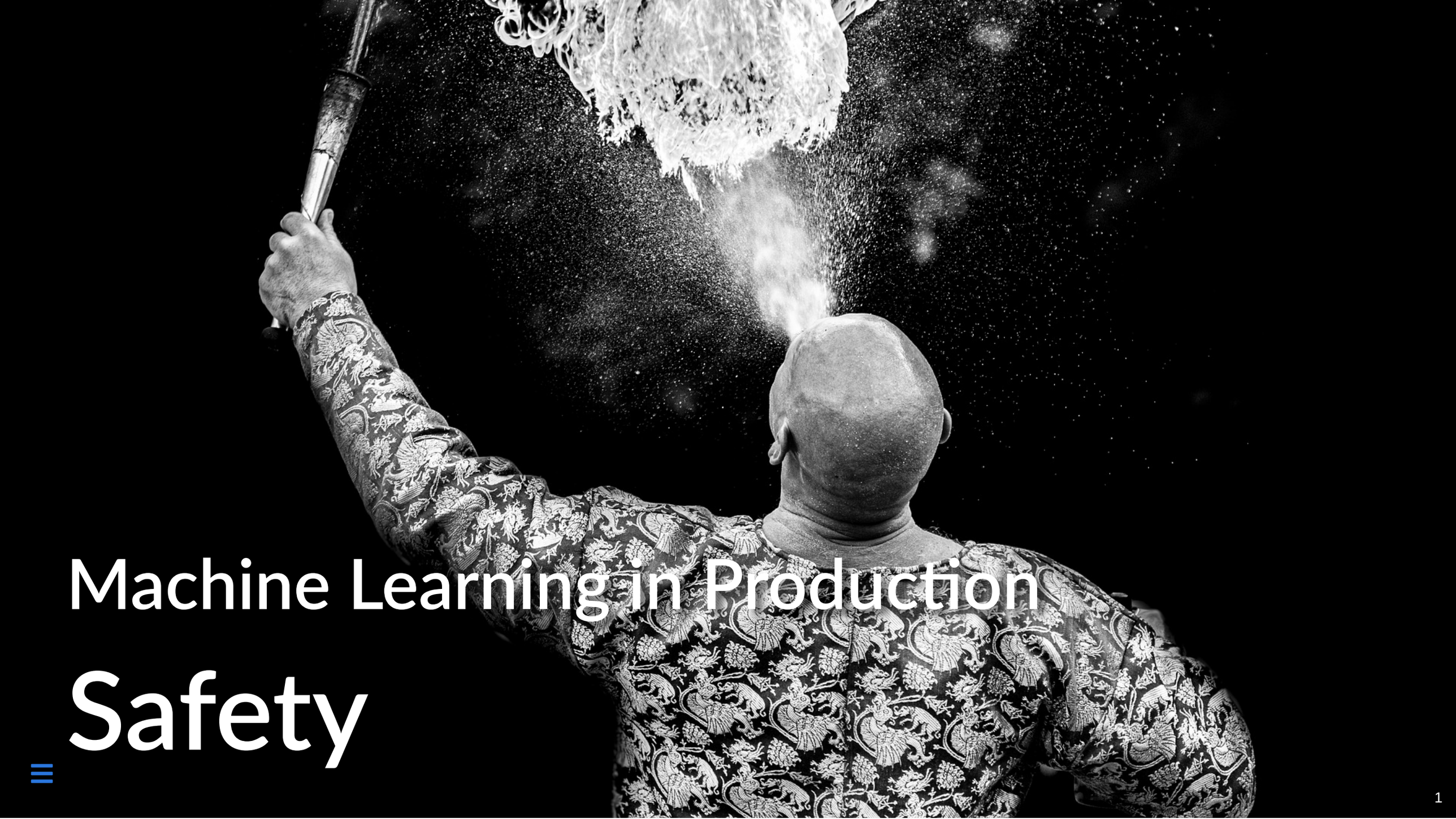


Machine Learning in Production

Safety

Mitigating more mistakes...

Fundamentals of Engineering AI-Enabled Systems

Holistic system view: AI and non-AI components, pipelines, stakeholders, environment interactions, feedback loops

Requirements:
System and model goals
User requirements
Environment assumptions
Quality beyond accuracy
Measurement
Risk analysis
Planning for mistakes

Architecture + design:
Modeling tradeoffs
Deployment architecture
Data science pipelines
Telemetry, monitoring
Anticipating evolution
Big data processing
Human-AI design

Quality assurance:
Model testing
Data quality
QA automation
Testing in production
Infrastructure quality
Debugging

Operations:
Continuous deployment
Contin. experimentation
Configuration mgmt.
Monitoring
Versioning
Big data
DevOps, MLOps

Teams and process: Data science vs software eng. workflows, interdisciplinary teams, collaboration points, technical debt

Responsible AI Engineering

Provenance, versioning, reproducibility

Safety

Security and privacy

Fairness

Interpretability and explainability

Transparency and trust

Ethics, governance, regulation, compliance, organizational culture

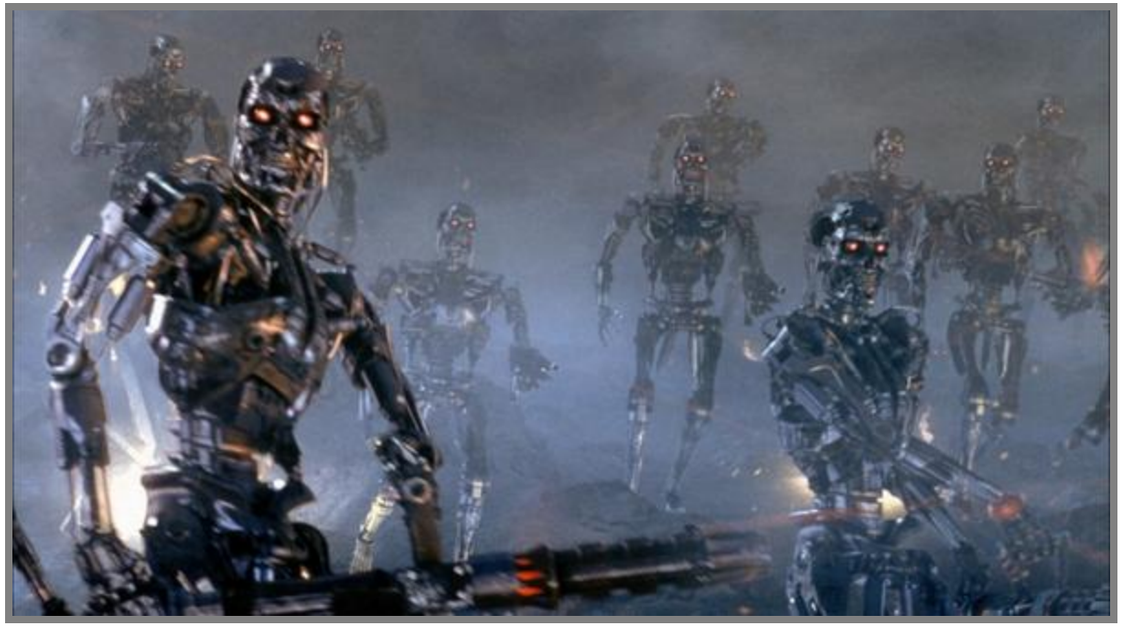
Reading

S. Mohseni et al., Practical Solutions for Machine Learning Safety in Autonomous Vehicles. SafeAI Workshop@AAAI (2020).

Learning Goals

- Understand safety concerns in traditional and AI-enabled systems
- Understand the importance of ML robustness for safety
- Apply hazard analysis to identify risks and requirements and understand their limitations
- Discuss ways to design systems to be safe against potential failures
- Suggest safety assurance strategies for a specific project
- Describe the typical processes for safety evaluations and their limitations

AI Safety



Amodei, Dario, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. "Concrete problems in AI safety." arXiv preprint arXiv:1606.06565 (2016).

Your Favorite AI Dystopia?

Existential AI Risk

Existential risk and AI alignment common in research

Funding through *longtermism* branch of effective altruism
(Longtermism is the view that positively influencing the longterm future is a key moral priority of our time.)

Ord estimates 10% existential risk from unaligned AI in 100 years

Our view: AI alignment not a real concern for the kind of ML-enabled products we consider here

Ord, Toby. The precipice: Existential risk and the future of humanity. Hachette Books, 2020.

Speaker notes

Relevant for reinforcement learning and AGI

More pressing AI isks?

AI

Ethicists fire back at 'AI Pause' letter they say 'ignores the actual harms'

Devin Coldewey @techcrunch / 8:18 PM EDT • March 31, 2023

"Those hypothetical risks are the focus of a dangerous ideology called longtermism that ignores the actual harms resulting from the deployment of AI systems today," they wrote, citing worker exploitation, data theft, synthetic media that props up existing power structures and the further concentration of those power structures in fewer hands.

The AI Alignment Problem

AI is optimized for a specific objective/cost function

- Inadvertently cause undesirable effects on the environment
- e.g., Transport robot: Move a box to a specific destination
- Side effects: Scratch furniture, bump into humans, etc.,

Side effects may cause ethical/safety issues (e.g., social media optimizing for clicks, causing teen depression)

Difficult to define sensible fitness functions:

- Perform X *subject to common-sense constr. on the environment*
- Perform X *but avoid side effects to the extent possible*

Reward Hacking

PlayFun algorithm pauses the game of Tetris indefinitely to avoid losing

When about to lose a hockey game, the PlayFun algorithm exploits a bug to make one of the players on the opposing team disappear from the map, thus forcing a draw.

Self-driving car rewarded for speed learns to spin in circles

Example: Coast Runner

Reward Hacking

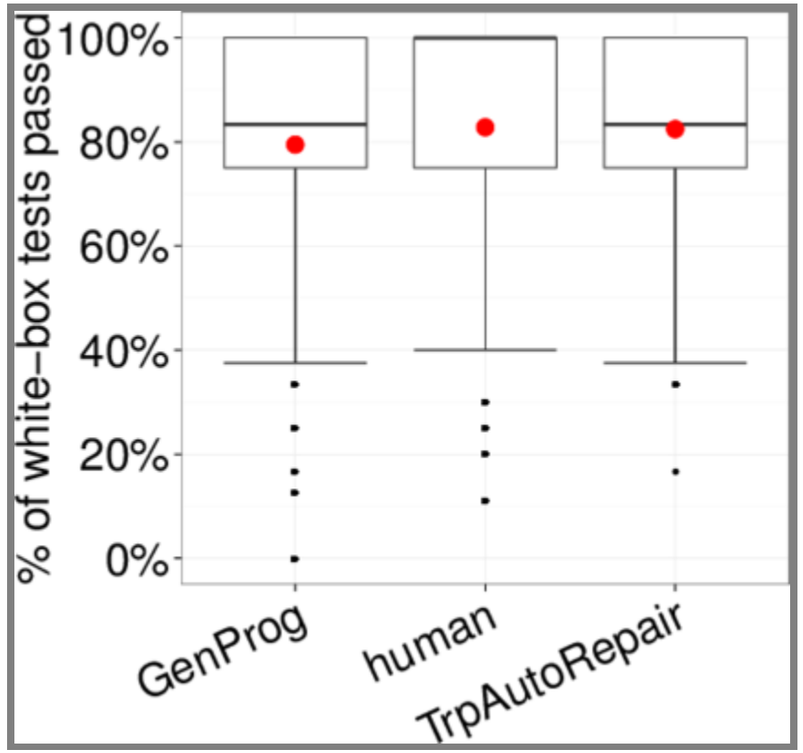
- AI can be good at finding loopholes to achieve a goal in unintended ways
- Technically correct, but does not follow *designer's informal intent*
- Many possible causes, incl. partially observed goals, abstract rewards, feedback loops
- In general, a very challenging problem!
  - Difficult to specify goal & reward function to avoid all possible hacks
  - Requires careful engineering and iterative reward design

Speaker notes

program repair

Gaming the model

This is a problem in automatic bug fixing, and manual bug fixing, too!



AI Alignment Problem = Requirements Problem

Recall: "World vs. machine"

- Identify stakeholders in the environment & possible effects on them
- Anticipate side effects, feedback loops
- Constrain the scope of the system
- Perfect contracts usually infeasible, undesirable

But more requirements engineering unlikely to be only solution

Practical Alignment Problems

Does the model goal align with the system goal? Does the system goal align with the user's goals?

- Profits (max. accuracy) vs fairness
- Engagement (ad sales) vs enjoyment, mental health
- Accuracy vs operating costs

Test model *and* system quality *in production*

(see requirements engineering and architecture lectures)

System Safety

Defining Safety

Prevention of a system failure or malfunction that results in:

- Death or serious injury to people
- Loss or severe damage to equipment/property
- Harm to the environment or society

Safety is a system concept

- Can't talk about software/ML being "safe"/"unsafe" on its own
- Safety is defined in terms of its effect on the **environment**

Safety of AI-Enabled Systems

Dr. Emily Slackerman Ackerman
@EmilyEAckerman · **Follow**

i (in a wheelchair) was just trapped *on* forbes ave by one of these robots, only days after their independent roll out. i can tell that as long as they continue to operate, they are going to be a major accessibility and safety issue. [thread]

pittnews.com
Everything we know about the Starship food delivery robots
The white, 2-foot tall battery-powered delivery robots will be sharing the sidewalk with Oakland pedestrians starting sometime in late …

7:27 PM · Oct 21, 2019

3.8K **Reply** **Copy link**

Speaker notes

Systems can be unsafe in unexpected ways

Safety of AI-Enabled Systems

Speaker notes

Systems can be unsafe in unexpected ways

Safety != Reliability

Reliability = absence of defects, mean time between failure

Safety = prevents accidents, harms

Can build safe systems from unreliable components (e.g. redundancy, safeguards)

System may be unsafe despite reliable components (e.g. stronger gas tank causes more severe damage in incident)

Accuracy is usually about reliability!

Safety is a broad concept

Recall: Legal vs ethical

Safety analysis not only for regulated domains (nuclear power plants, medical devices, planes, cars, ...), and includes harms beyond physical harms/injury

Many end-user applications have a safety component

Start with requirements and hazard analyses

Speaker notes

Do the right thing, even without regulation

Safety Engineering

Safety Engineering

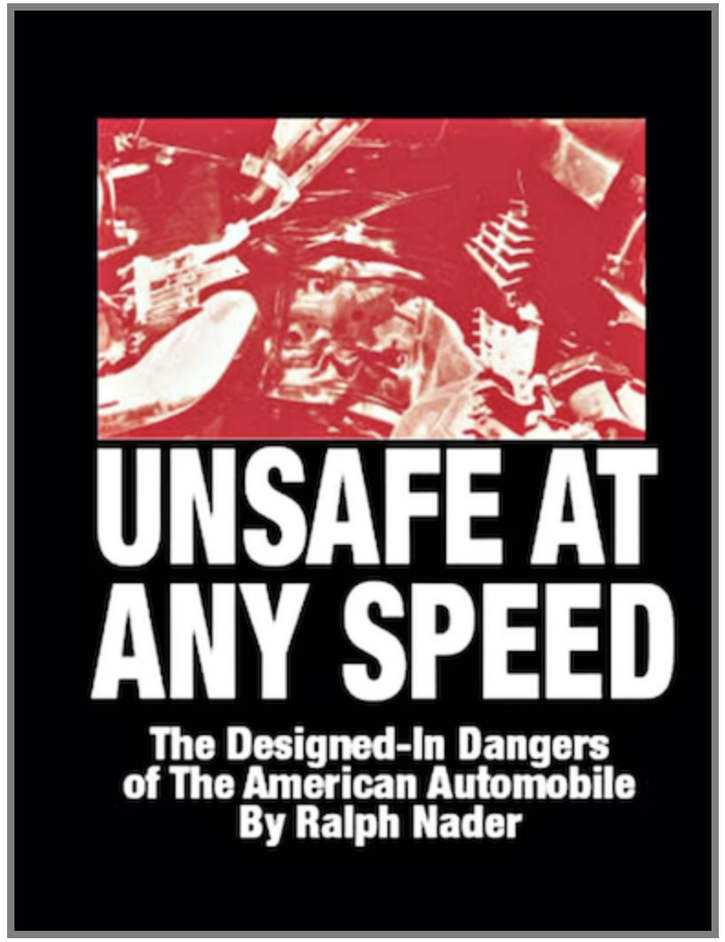
Safety Engineering: An engineering discipline which assures that engineered systems provide acceptable levels of safety.

Typical safety engineering process:

- Identify relevant hazards & safety requirements
- Identify potential root causes for hazards
- For each hazard, develop a mitigation strategy
- Provide evidence that mitigations are properly implemented

Case Study: Self-Driving Car

How did traditional vehicles become safer?



National Traffic & Motor Safety Act (1966):

- Mandatory design changes (head rests, shatter-resistant windshields, safety belts)
- Road improvements (center lines, reflectors, guardrails)
- Significant reduction (13-46%) in traffic fatalities

Autonomous Vehicles: What's different?

Ford Taps the Brakes on the Arrival of Self-Driving Cars

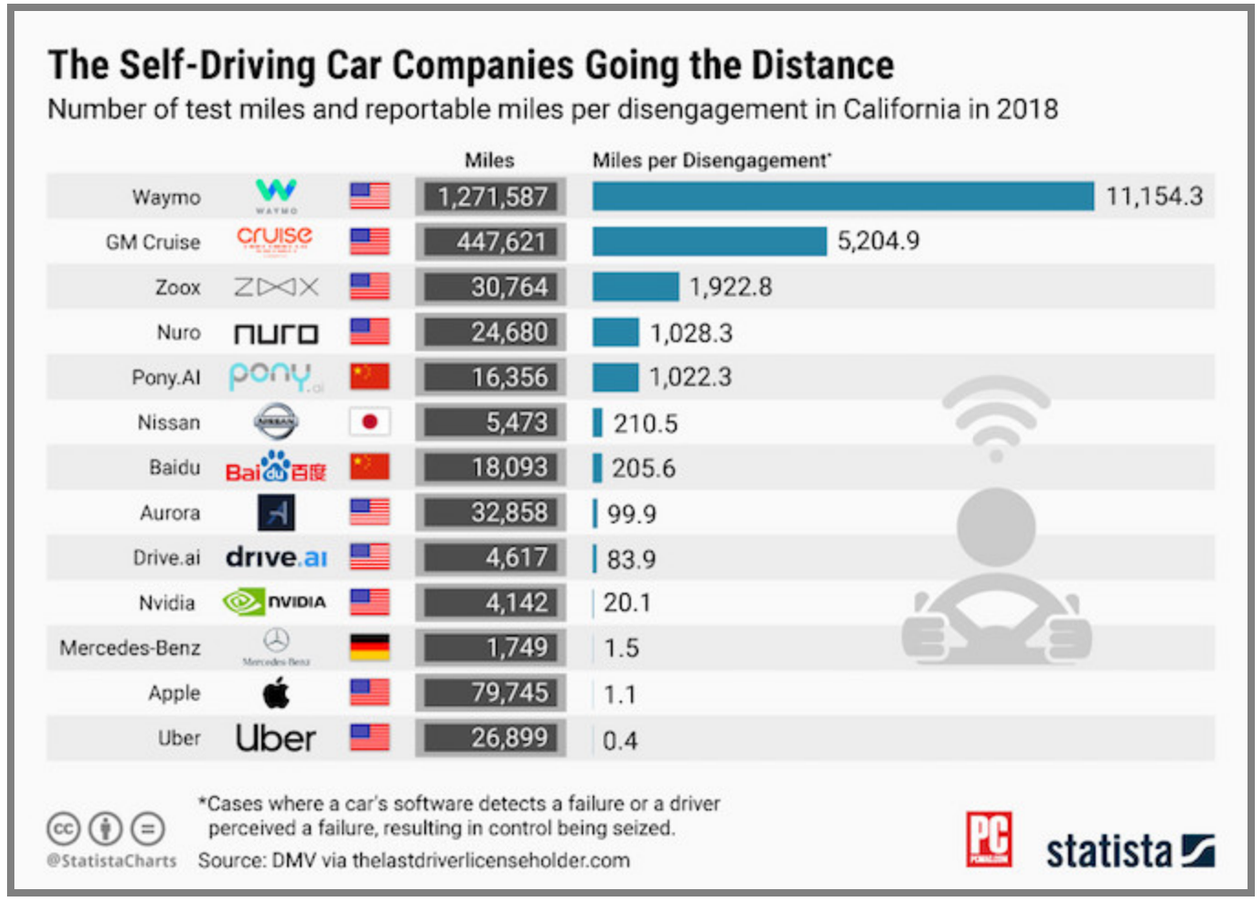
HYPE CYCLE —

The hype around driverless cars came crashing down in 2018

Top Toyota expert throws cold water on the driverless car hype

- In traditional vehicles, humans ultimately responsible for safety
  - Built-in safety features (lane keeping, emergency braking)
  - i.e., safety = human control + safety mechanisms
- Use of AI in autonomous vehicles: Perception, control, routing, etc.,
  - Inductive training: No explicit requirements or design insights
  - **Can ML achieve safe design solely through lots of data?**

Demonstrating Safety



Q. More miles tested => safer?

Challenge: Edge/Unknown Cases

- Gaps in training data; ML unlikely to cover all unknown cases
- Is this a unique problem for AI? What about humans?

Improving Safety of ML-Enabled Systems

Anticipate problems (hazard analysis, FTA, FMEA, HAZOP, ...)

Anticipate the existence of unanticipated problems

Plan for mistakes, design mitigations (recall earlier lecture!)

- Human in the loop
- Undoable actions, failsoft
- Guardrails
- Mistaked detection
- Redundancy, ...

Demonstrating and Documenting Safety

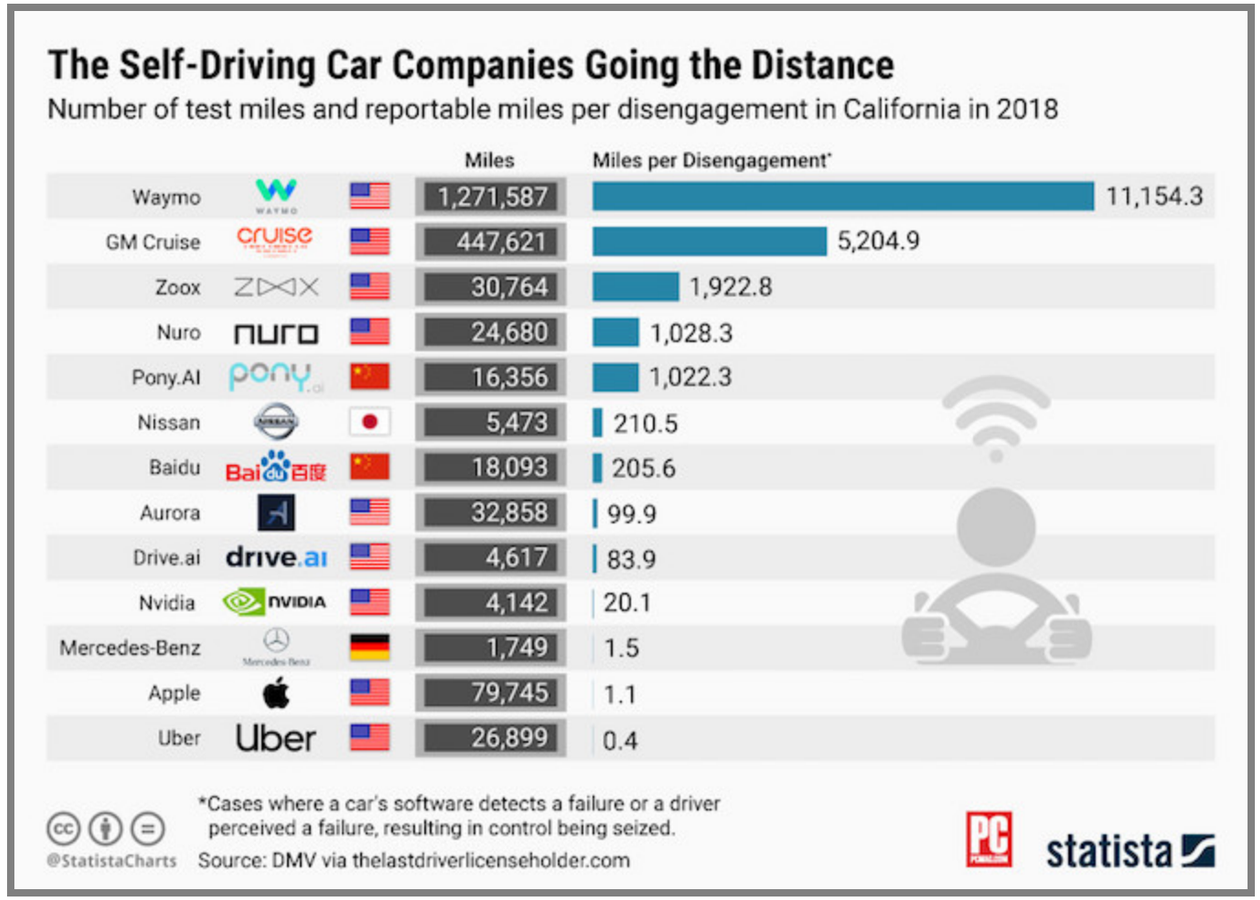
Demonstrating Safety

Two main strategies:

1. **Evidence of safe behavior in the field**
    - Extensive field trials
    - Usually expensive
2. **Evidence of responsible (safety) engineering process**
    - Process with hazard analysis, testing mitigations, etc
    - Not sufficient to assure safety

Most standards require both

Demonstrating Safety

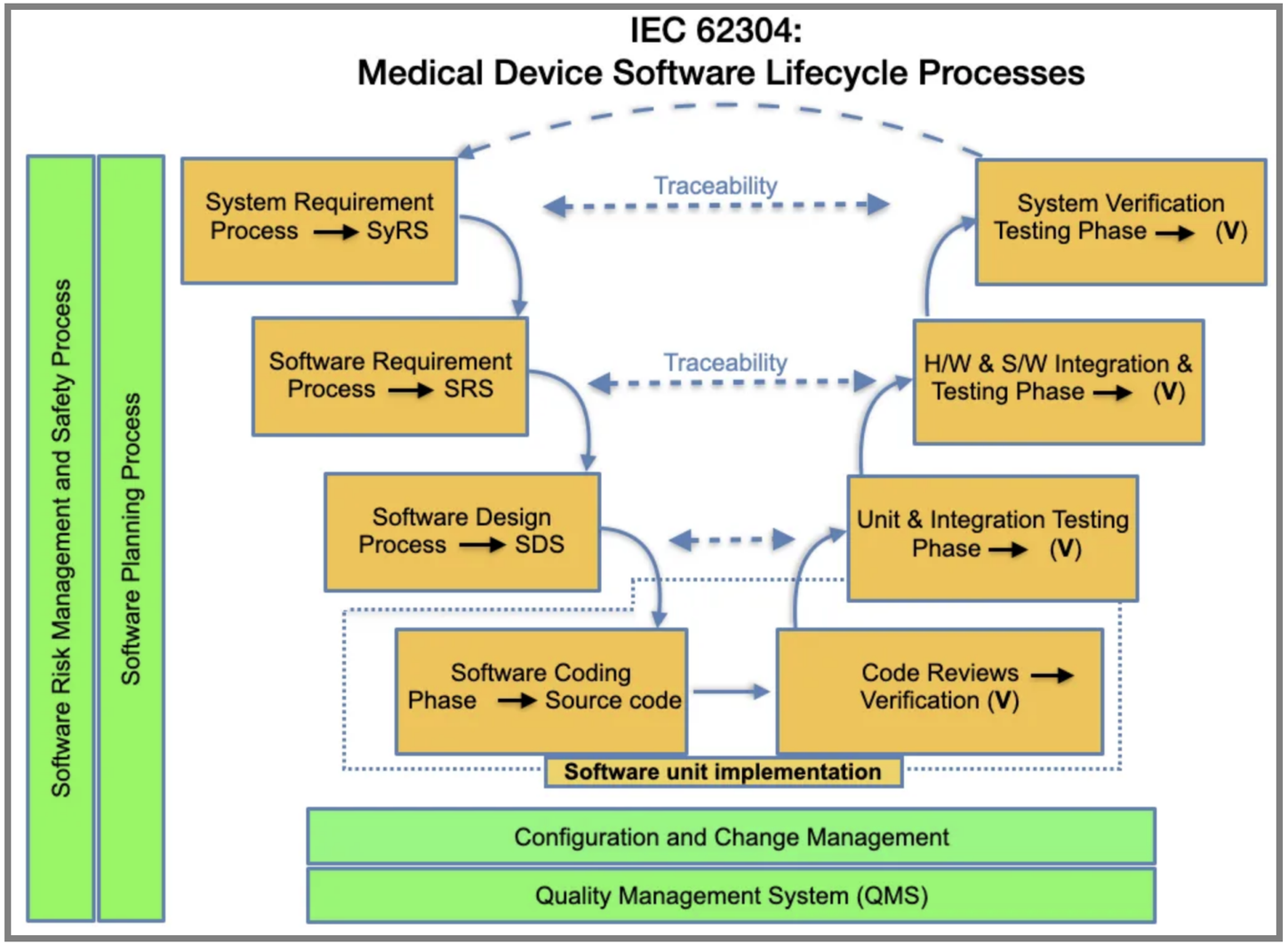


How do we demonstrate to a third-party that our system is safe?

Safety & Certification Standards

- Guidelines & recommendations for achieving an acceptable level of safety
- Examples: DO-178C (airborne systems), ISO 26262 (automotive), IEC 62304 (medical software), Common Criteria (security)
- Typically, **prescriptive & process-oriented**
  - Recommends use of certain development processes
  - Requirements specification, design, hazard analysis, testing, verification, configuration management, etc.,
- Limitations
  - Most not designed to handle ML systems (exception: UL 4600)
  - Costly to satisfy & certify, but effectiveness unclear (e.g., many FDA-certified products recalled due to safety incidents)
- Good processes are important, but not sufficient; provides only indirect evidence for system safety

Certification Standards: Example



Demonstrating Safety

Two main strategies:

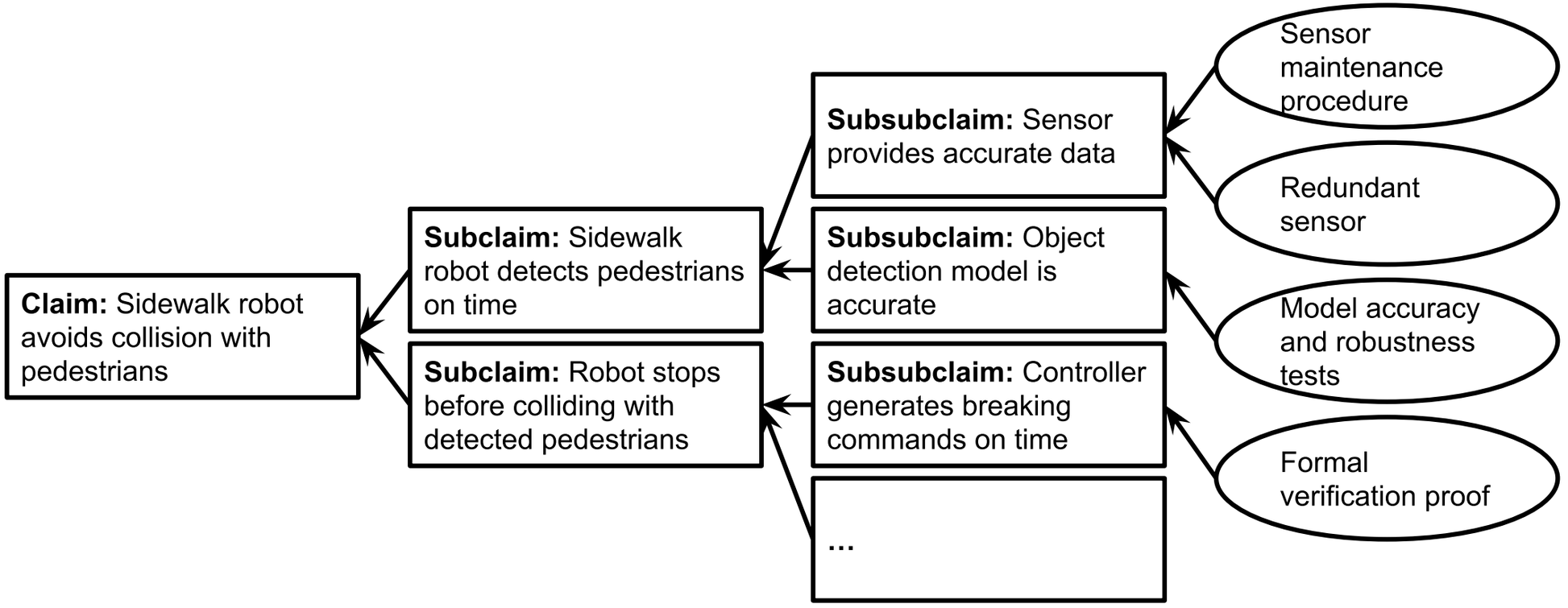
1. **Evidence of safe behavior in the field**
    - Extensive field trials
    - Usually expensive
2. **Evidence of responsible (safety) engineering process**
    - Process with hazard analysis, testing mitigations, etc
    - Not sufficient to assure safety

Most standards require both, but often not sufficient!

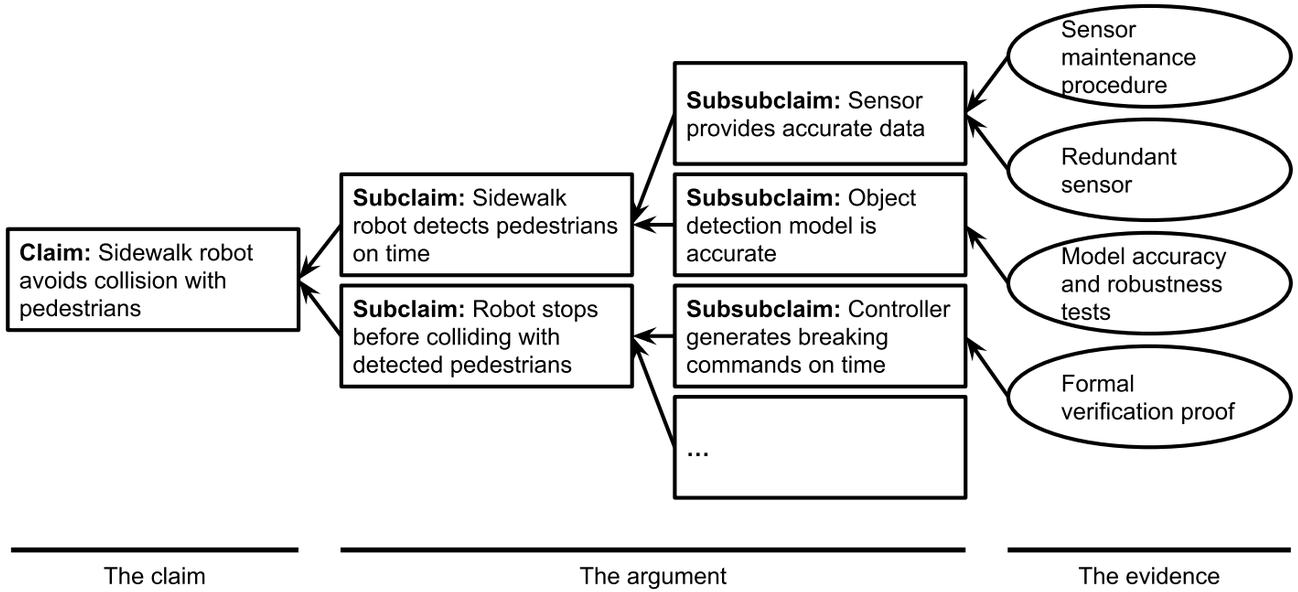
Assurance (Safety) Cases

- An emerging approach to demonstrating safety
- An explicit argument that a system achieves a desired safety requirement, along with supporting evidence
- Structure:
  - Argument: A top-level claim decomposed into multiple sub-claims
  - Evidence: Testing, software analysis, formal verification, inspection, expert opinions, design mechanisms…

Documenting Safety with Assurance (Safety) Cases



Assurance Cases: Example



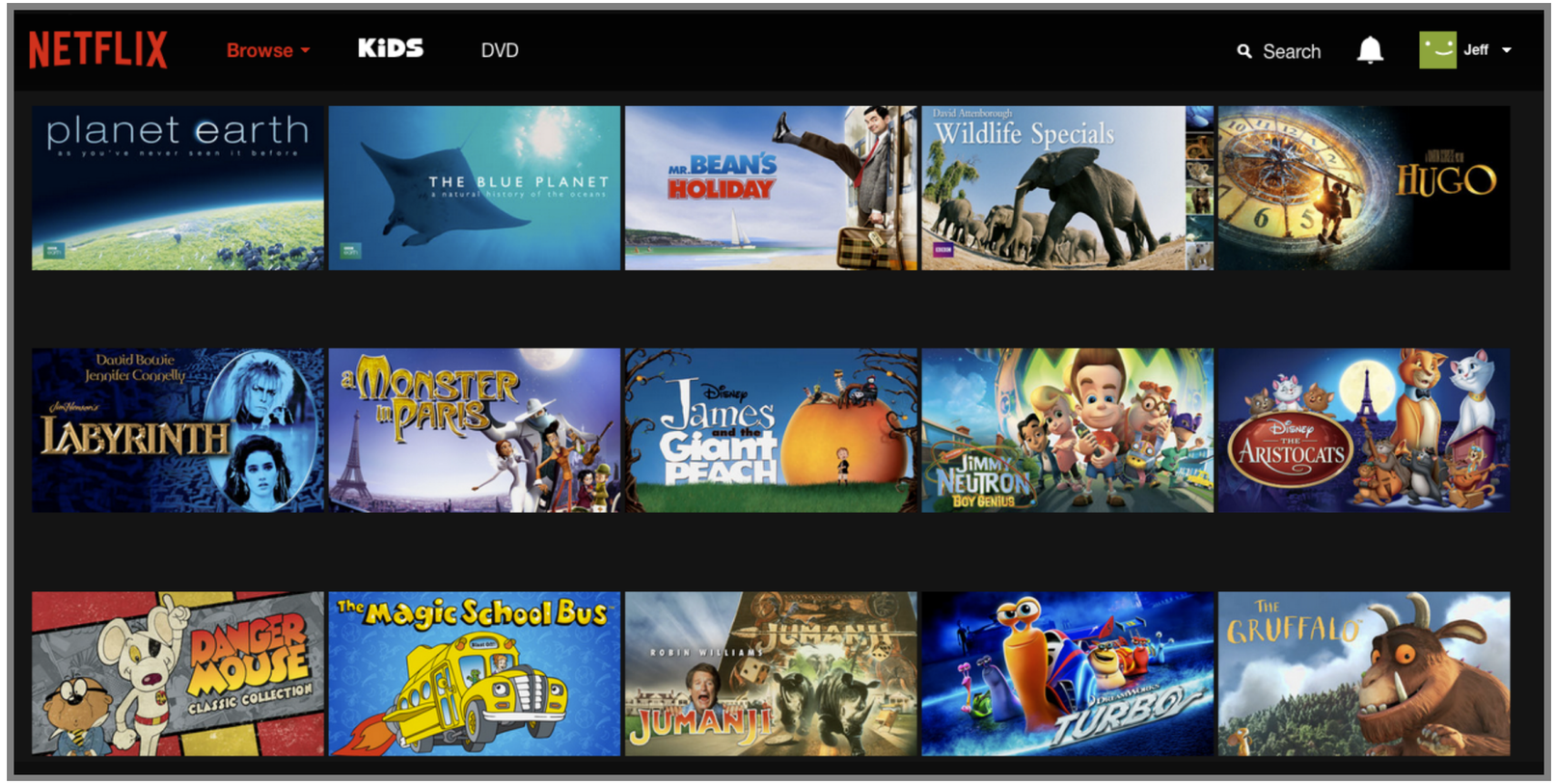
Questions to think about:

- Do sub-claims imply the parent claim?
- Am I missing any sub-claims?
- Is the evidence strong enough to discharge a leaf claim?

Assurance Cases: Example

Aurora Safety Case

Discussion: Assurance Case for Recommender



Assurance Cases: Benefits & Limitations

- Provides an explicit structure to the safety argument
  - Easier to navigate, inspect, and refute for third-party auditors
  - Provides traceability between system-level claims & low-level evidence
  - Can also be used for other types of system quality (security, reliability, etc.,)
- Challenges and pitfalls
  - Informal links between claims & evidence, e.g., Does the sub-claims actually imply the top-level claim?
  - Effort in constructing the case & evidence: How much evidence is enough?
  - System evolution: If system changes, must reproduce the case & evidence
- Tools for building & analyzing safety cases available
  - e.g., ASCE/GSN from Adelard
  - But ultimately, can't replace domain knowledge & critical thinking

Testing is still important!

Other Challenges

- Safe Exploration
  - Exploratory actions "in production" may have consequences
  - e.g., trap robots, crash drones
- Robustness to Drift
  - Drift may lead to poor performance that may not even be recognized
- Scalable Oversight
  - Cannot provide human oversight over every action (or label all possible training data)
  - Use indirect proxies in telemetry to assess success/satisfaction

Robustness for ML-based Systems

Robustness

Environment sometimes **deviates** from expected, normal conditions

- Extreme weathers, unexpected obstacles, etc.,
- Erratic user behaviors; unusually high service demand
- Adversarial actors; users trying to game your system, etc.,

Does your system work reasonably well under these deviations? i.e., is it *robust*?

Most safety-critical systems require some level of robustness

- Not enough to show that system is safe in normal conditions

Defining Robustness for ML:

- A prediction for input x is robust if the outcome is stable under minor perturbations to the input:
  - $\forall x' \,.\, d(x, x') < \epsilon \Rightarrow f(x) = f(x')$
  - distance function d and permissible distance ϵ depends on the problem domain!
- A model is said to be robust if most predictions are robust
- An important concept in safety and security settings
  - In safety, perturbations tend to be random or predictable (e.g., sensor noise due to weather conditions)
  - In security, perturbations are intentionally crafted (e.g., adversarial attacks)

Robustness and Distance for Images

- Slight rotation, stretching, or other transformations
- Change many pixels minimally (below human perception)
- Change only few pixels
- Change most pixels mostly uniformly, e.g., brightness

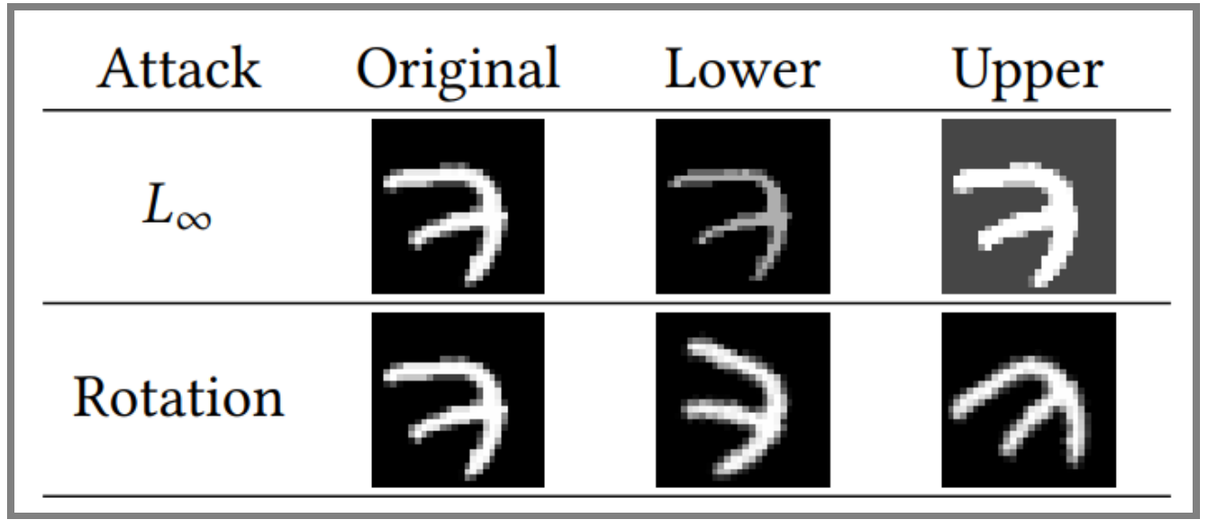
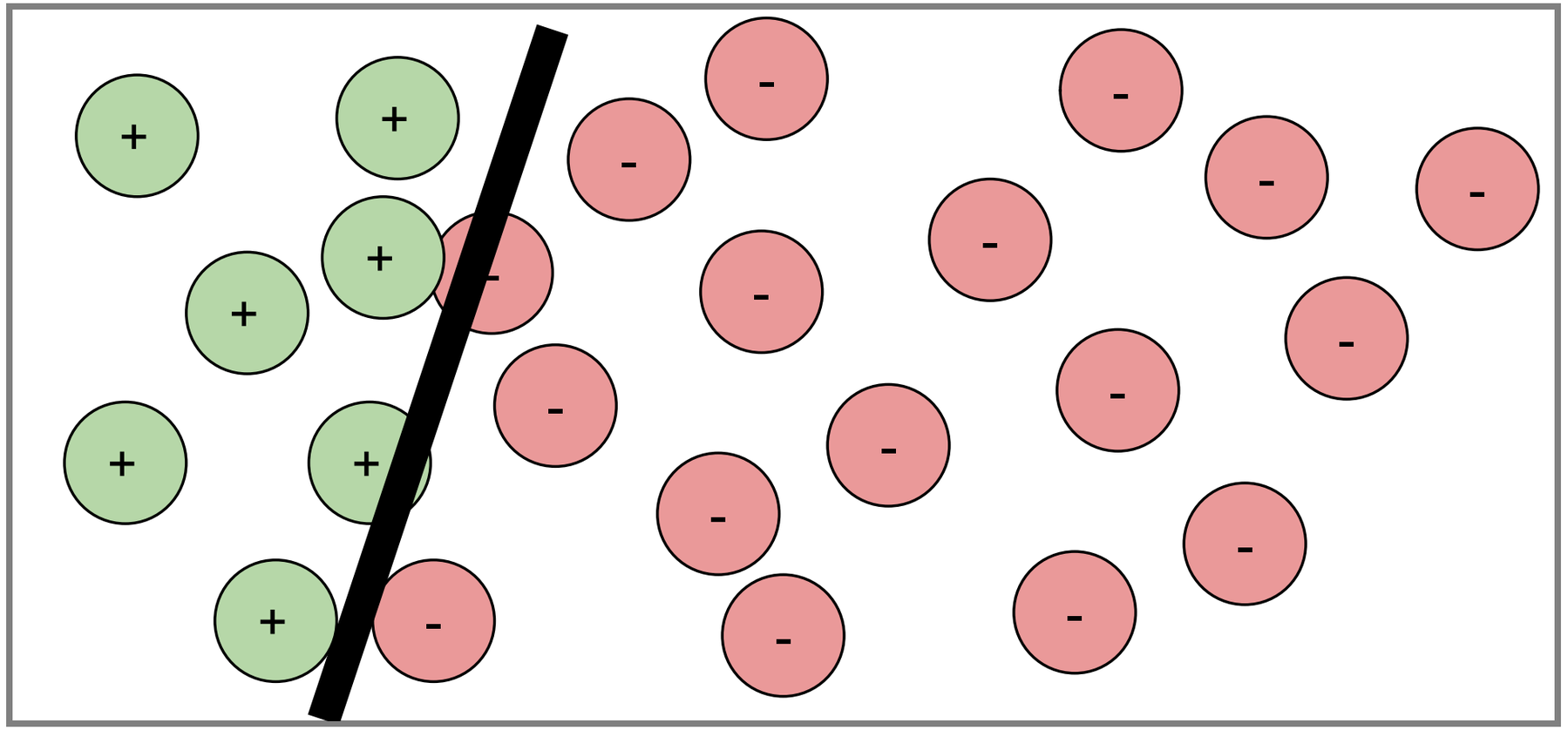


Image: *An abstract domain for certifying neural networks*. Gagandeep et al., POPL (2019).

No Model is Fully Robust

- Every useful model has at least one decision boundary
- Predictions near that boundary are not (and should not) be robust



Evaluating ML Robustness

- Lots of on-going research (especially for DNNs)
- Formal verification
  - Constraint solving or abstract interpretation over computations in neuron activations
  - Conservative abstraction, may label robust inputs as not robust
  - Currently not very scalable
  - Example: *An abstract domain for certifying neural networks*. Gagandeep et al., POPL (2019).
- Sampling
  - Sample within distance, compare prediction to majority prediction
  - Probabilistic guarantees possible (with many queries, e.g., 100k)
  - Example: *Certified adversarial robustness via randomized smoothing*. Cohen, Rosenfeld, and Kolter, ICML (2019).

ML Robustness: Limitations

- Lots of on-going research (especially for DNNs)
- Mostly input-centric, focusing on small (ϵ) perburtations
  - Common use case: Robustness against adversarial attacks
  - Q. But do these pertubations matter for safety?
- In practice: Perturbations result from environmental changes!
  - Which parts of the world does the software sense?
  - Can those parts change over time? Can the sensors be noisy, faulty, etc.,? (these are **domain-specific**)
  - What input pertburbations could be caused by from these changes/noise…?

Robustness testing

Evaluates the degree to which a system or component functions correctly in the presence of invalid inputs or stressful or off-nominal environmental conditions.

Speaker notes

this has a rich history outside of ML!

Robustness in a Safety Setting

- Does the model detect stop signs under normal conditions?
- Does the model detect stop signs under deviations?
  - **Q. What deviations do we care about?**

Robustness in a Safety Setting

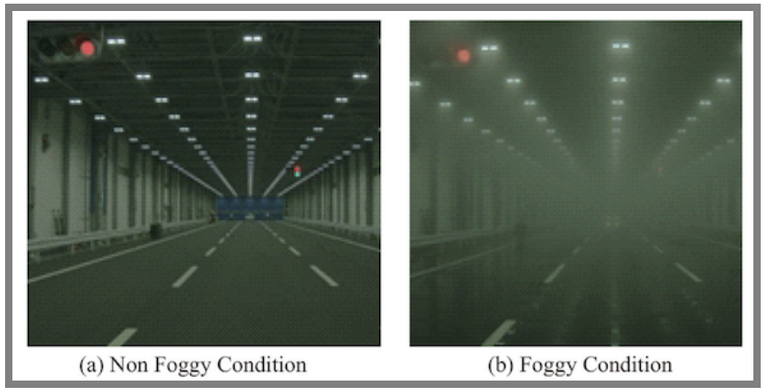
- Does the model detect stop signs under normal settings?
- Does the model detect stop signs under deviations?
  - Poor lighting? In fog? With a tilted camera? Sensor noise?
  - With stickers taped to the sign?

Image: David Silver. Adversarial Traffic Signs. Blog post, 2017

Improving Robustness for Safety

Q. How do we make ML-based systems more robust?

Improving Robustness for Safety

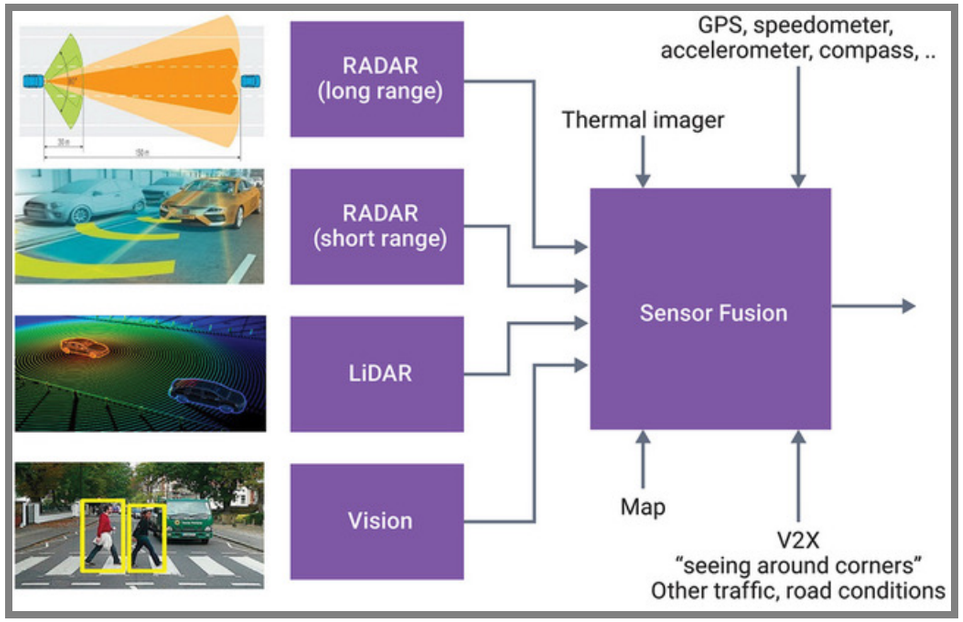


Learn more robust models

- Test/think about domain-specific deviations that might result in perturbations to model input (e.g., fogs, snow, sensor noise)
- Curate data for those abnormal scenarios or augment training data with transformed inputs

Automated driving recognition technologies for adverse weather conditions. Yoneda et al., (2019).

Improving Robustness for Safety



Design mechanisms

- Deploy redundant components for critical tasks (e.g., vision + map)
- Ensemble learning: Combine models with different biases
- Multiple, independent sensors (e.g., LiDAR + radar + cameras)

Improving Robustness for Safety

Design mechanisms

- Deploy redundant components for critical tasks (e.g., vision + map)
- Ensemble learning: Combine models with different biases
- Multiple, independent sensors (e.g., LiDAR + radar + cameras)

Robustness checking at inference time

- Handle inputs with non-robust predictions differently (e.g. discard or output low confidence score for outliers)
- Downside: Raises cost of prediction; may not be suitable for time-sensitive applications (e.g., self-driving cars)

Breakout: Robustness

Scenario: Medical use of transcription service, dictate diagnoses and prescriptions

As a group, tagging members, post to `#lecture`:

> *1. What safety concerns can you anticipate?*
> *2. What deviations are you concerned about?*
> *3. How would you improve the robustness of the overall system?*

FMEA

Reminder: Fault-Tree Analysis

- Town-down, *backward* search for the root cause of issues
  - from final outcomes to initiating events
- Issues (TOP events) need to be known upfront
- Quantitative analysis possible
- Useful for understanding faults post-hoc
- Where do outcomes come from?

Failure Mode and Effects Analysis (FMEA)

	Function	Potential Failure Mode	Potential Effect(s) of Failure	SEV i	Potential Cause(s) of Failure	OCC i	Current Design Controls (Prevention)	Current Design Controls (Detection)	DET i	RPN i	Recommended Action(s)
1	Provide required levels of radiation	Radiation level too high for the required intervention	Over radiation of the patients.		Technician did not set the radiation at the right level.			Current algorithm resets to normal levels after imaging each patient.			Modify software to alert technician to unusually high radiation levels before activating.
2		Radiation at lower level than required	Patient fails to receive enough radiation.		Software does not respond to hardware mechanical setting.			Failure detection included in software			Include visual / audio alarm in the code when lack of response.
3											Improve recovery protocol.
4	Protect patients from unexpected high radiation	Higher radiation than required	Radiation burns		sneak paths in software			Shut the system if radiation level does not match the inputs.			Perform traceability matrix.

- A **forward search** technique to identify potential hazards
- Widely used in aeronautics, automotive, healthcare, food services, semiconductor processing, and (to some extent) software

FMEA Process

(a) Identify system components

(b) Enumerate potential failure modes

- *for ML component: Always suspect prediction may be wrong*

(c) For each failure mode, identify:

- Potential hazardous effect on the system
- Method for detecting the failure
- Potential mitigation strategy

FMEA Excerpt: Autonomous Car

Component	Failure Mode	Failure Effects	Sev	Potential Causes	Occ	Det	Recommended Action	RPN
Sensors								
Vision-based camera	Poor visibility	Outcome depends on whether other sensors remain operational and how the controller compensates for the loss of data. Collision is possible.	5	Driving at night, poor weather (heavy rain, snow, or fog), dirt or obstruction over lens	10	2	If confidence in sensor data is low, pull over or alert human driver to take control	100
	Hardware failure		5	Manufacturing fault, or at end of life cycle	4	4	Annual inspection	80
LIDAR	Poor visibility		5	Poor weather (heavy rain, snow, or fog), dirt or obstruction over sensor	8	2	If confidence in sensor data is low, pull over or alert human driver to take control	80
	LIDAR interference		5	Other AVs in the area using LIDAR	10	2	Laser signal should be coded with ID to prevent interference	100
	Positional error (bias error or noise)		4	Intrinsic to sensor	10	2	Measurement uncertainty should be conveyed to decision-making algorithm	80
	Hardware failure		5	Manufacturing fault, or at end of life cycle	3	4	Annual inspection	60

Excerpt of an FMEA table for analyzing components in an autonomous vehicle, from David Robert Beachum. Methods for assessing the safety of autonomous vehicles. University of Texas Theses and Dissertations (2019).

"Wrong Prediction" as Failure Mode?

"Wrong prediction" is a very cause grained failure mode of every model

May not be possible to decompose further

However, may evaluate causes of wrong prediction for better understanding, as far as possible --> FTA?

FMEA Summary

Forward analysis: From components to possible failures

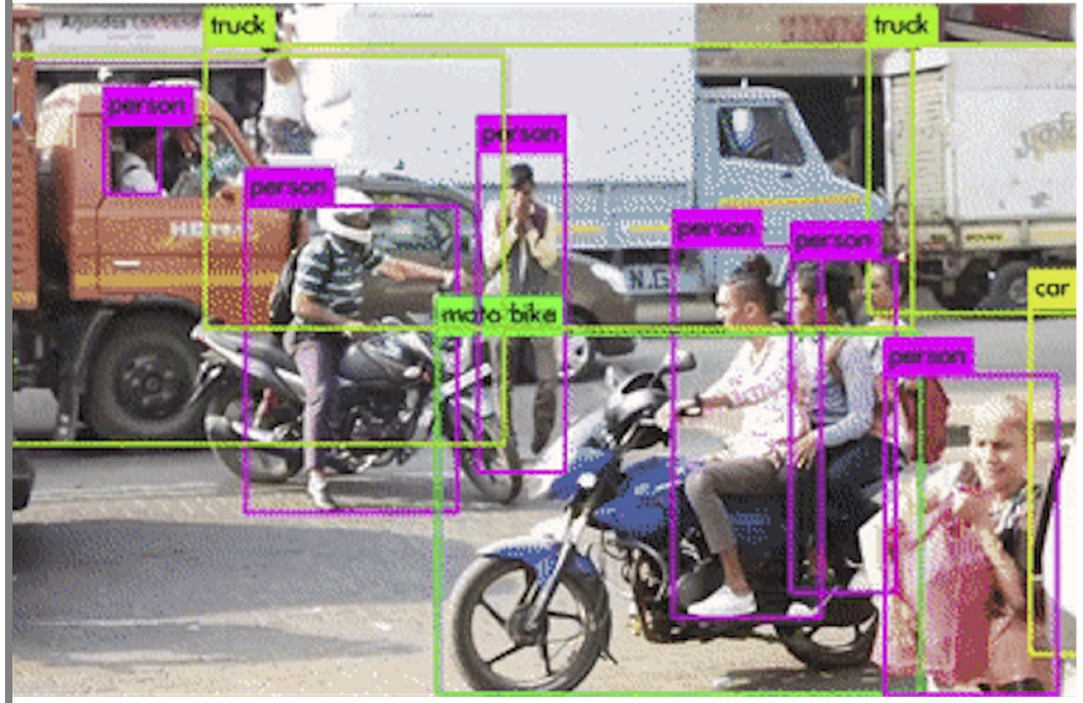
Focus on single component failures, no interactions

Identifying failure modes may require domain understanding

HAZOP

Hazard and Interoperability Study (HAZOP)

Identify hazards and component fault scenarios through guided inspection of requirements



Guide Word	Meaning
NO OR NOT	Complete negation of the design intent
MORE	Quantitative increase
LESS	Quantitative decrease
AS WELL AS	Qualitative modification/increase
PART OF	Qualitative modification/decrease
REVERSE	Logical opposite of the design intent
OTHER THAN / INSTEAD	Complete substitution
EARLY	Relative to the clock time
LATE	Relative to the clock time
BEFORE	Relating to order or sequence
AFTER	Relating to order or sequence

Hazard and Operability Study (HAZOP)

A **forward search** method to identify potential hazards

For each component, use a set of **guide words** to generate possible deviations from expected behavior

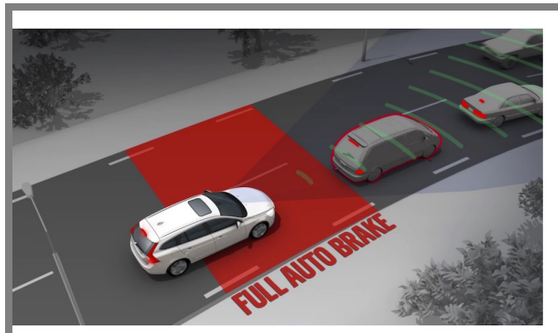
Consider the impact of each generated deviation: Can it result in a system-level hazard?

Guide Word	Meaning
NO OR NOT	Complete negation of the design intent
MORE	Quantitative increase
LESS	Quantitative decrease
AS WELL AS	Qualitative modification/increase
PART OF	Qualitative modification/decrease
REVERSE	Logical opposite of the design intent
OTHER THAN / INSTEAD	Complete substitution
EARLY	Relative to the clock time
LATE	Relative to the clock time
BEFORE	Relating to order or sequence
AFTER	Relating to order or sequence

HAZOP Example: Emergency Braking (EB)

Specification: EB must apply a maximum braking command to the engine.

- **NO OR NOT**: EB does not generate any braking command.
- **LESS**: EB applies less than max. braking.
- **LATE**: EB applies max. braking but after a delay of 2 seconds.
- **REVERSE**: EB generates an acceleration command instead of braking.
- **BEFORE**: EB applies max. braking before a possible crash is detected.



Guide Word	Meaning
NO OR NOT	Complete negation of the design intent
MORE	Quantitative increase
LESS	Quantitative decrease
AS WELL AS	Qualitative modification/increase
PART OF	Qualitative modification/decrease
REVERSE	Logical opposite of the design intent
OTHER THAN / INSTEAD	Complete substitution
EARLY	Relative to the clock time
LATE	Relative to the clock time
BEFORE	Relating to order or sequence
AFTER	Relating to order or sequence

HAZOP & ML

In addition to traditional analysis: Analyze possible mistakes of all ML components

Original guidewords: NO OR NOT, MORE, LESS, AS WELL AS, PART OF, REVERSE, OTHER THAN / INSTEAD, EARLY, LATE, BEFORE, AFTER

Additional ML-specific guidewords: WRONG, INVALID, INCOMPLETE, PERTURBED, and INCAPABLE.

Breakout: Automated Train Doors

Analyze the vision component to detect obstacles in train doors

NO OR NOT, MORE, LESS, AS WELL AS, PART OF, REVERSE, OTHER THAN / INSTEAD, EARLY, LATE, BEFORE, AFTER, WRONG, INVALID, INCOMPLETE, PERTURBED, and INCAPABLE.

Using HAZOP: As a group answer in `#lecture`, tagging group members:

- *What is the specification of the perception component?*
- *What are possible deviations from the specification?*
- *What are potential hazards resulting from these deviations?*
- *What possible mitigations would you consider? (e.g., human in the loop, undoable actions, guardrails, mistake detection and recovery, containment)*

HAZOP: Benefits & Limitations

- Easy to use; encourages systematic reasoning about component faults
- Can be combined with FTA/FMEA to generate faults (i.e., basic events in FTA)
- Potentially labor-intensive; relies on engineer's judgement
- Does not guarantee to find all hazards (but also true for other techniques)

Remarks: Hazard Analysis

None of these methods guarantee completeness

- You may still be missing important hazards, failure modes

Intended as structured approaches to thinking about failures

- But cannot replace human expertise and experience

Designing for Safety

See Lecture **Planning for Mistakes**

Safety Assurance with ML Components

- Consider ML components as unreliable, at most probabilistic guarantees
- Testing, testing, testing (+ simulation)
  - Focus on data quality & robustness
- *Adopt a system-level perspective!*
- Consider safe system design with unreliable components
  - Traditional systems and safety engineering
  - Assurance cases
- Understand the problem and the hazards
  - System level, goals, hazard analysis, world vs machine
  - Specify *end-to-end system behavior* if feasible

Summary

- Defining safety: absence of harm to people, property, and environment -- consider broadly; safety != reliability
- *Adopt a safety mindset!*
- Assume all components will eventually fail in one way or another, especially ML components
- Hazard analysis to identify safety risks and requirements; classic safety design at the system level
- Robustness: Identify & address relevant deviations
- AI alignment: AI goals are difficult to specify precisely; susceptible to negative side effect & reward hacking

Further Readings

- Borg, Markus, Cristofer Englund, Krzysztof Wnuk, Boris Duran, Christoffer Levandowski, Shenjian Gao, Yanwen Tan, Henrik Kaijser, Henrik Lönn, and Jonas Törnqvist. “Safely entering the deep: A review of verification and validation for machine learning and a challenge elicitation in the automotive industry.” Journal of Automotive Software Engineering. 2019
- Leveson, Nancy G. Engineering a safer world: Systems thinking applied to safety. The MIT Press, 2016.
- Salay, Rick, and Krzysztof Czarnecki. “Using machine learning safely in automotive software: An assessment and adaption of software process requirements in ISO 26262.” arXiv preprint arXiv:1808.01614 (2018).
- Mohseni, Sina, Mandar Pitale, Vasu Singh, and Zhangyang Wang. “Practical Solutions for Machine Learning Safety in Autonomous Vehicles.” SafeAI workshop at AAAI’20, (2020).
- Huang, Xiaowei, Daniel Kroening, Wenjie Ruan, James Sharp, Youcheng Sun, Emese Thamo, Min Wu, and Xinping Yi. “A survey of safety and trustworthiness of deep neural networks: Verification, testing, adversarial attack and defence, and interpretability.” Computer Science Review 37 (2020).
- Amodei, Dario, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. "Concrete problems in AI safety." arXiv preprint arXiv:1606.06565 (2016).