Continuous Argument Engineering: Tackling Uncertainty in Machine Learning based Systems

Fuyuki Ishikawa, National Institute of Informatics, Japan

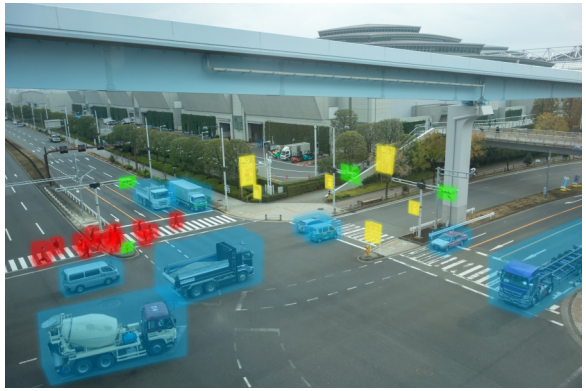
Yutaka Matsuno, Nihon University, Japan

Overview

- Increasing uncertainty in autonomous systems

  (Not to mention increasing demand for assurance)

  - Specifically, use of machine learning is introducing a different level of uncertainty



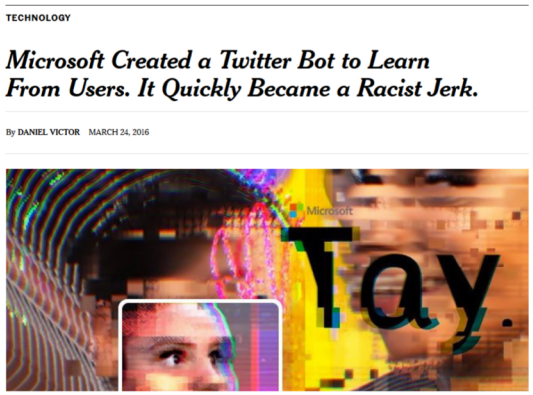
Uncertainty-awareness in assurance cases towards their effective use

AI and ML

- Everyone is talking about AI and ML, recently more about risks and concerns

Accidents of autonomous cars

[http://www.dailymail.co.uk/news/article-3677101/Tesla-told-regulators-fatal-Autopilot-crash-nine-days-happened.html]



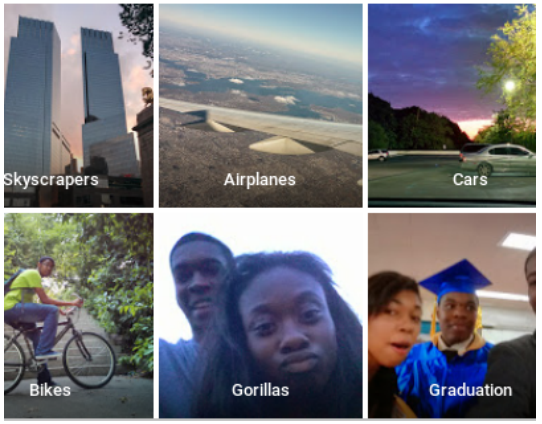
TECHNOLOGY

Microsoft Created a Twitter Bot to Learn From Users. It Quickly Became a Racist Jerk.

By DANIEL VICTOR MARCH 24, 2016

Improper online learning

Technically unsolved problems at Google Photo

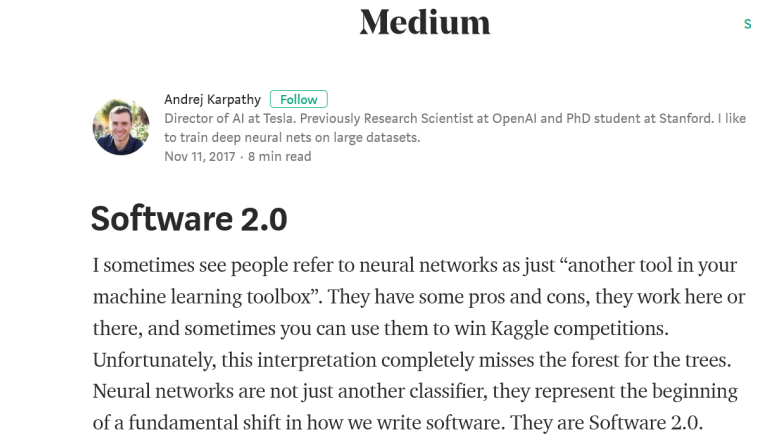


[https://www.nytimes.com/2016/03/25/technology/microsoft-created-a-twitter-bot-to-learn-from-users-it-quickly-became-a-racist-jerk.html]

[https://www.theguardian.com/technology/2015/jul/01/google-sorry-racist-auto-tag-photo-app]
[https://www.theguardian.com/technology/2018/jan/12/google-racism-ban-gorilla-black-people]

Software 2.0 !?

- Let us focus on ML
  - Present movement on AI was driven by ML, specifically advance in deep learning techniques
  - With ML, we construct a software component by deriving from training data: the rule that governs the behavior is not directly given by us
  - In the Japanese industry, the term “inductive software construction” got some popularity to capture the essential difference


Medium

Andrej Karpathy Follow
Director of AI at Tesla. Previously Research Scientist at OpenAI and PhD student at Stanford. I like to train deep neural nets on large datasets.
Nov 11, 2017 · 8 min read

Software 2.0

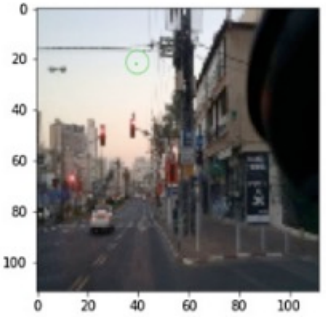
I sometimes see people refer to neural networks as just “another tool in your machine learning toolbox”. They have some pros and cons, they work here or there, and sometimes you can use them to win Kaggle competitions. Unfortunately, this interpretation completely misses the forest for the trees. Neural networks are not just another classifier, they represent the beginning of a fundamental shift in how we write software. They are Software 2.0.

[https://medium.com/@karpathy/software-2-0-a64152b37c35]

Resulting Characteristics

Uncertainty on the implementation (not only the requirements and environments)

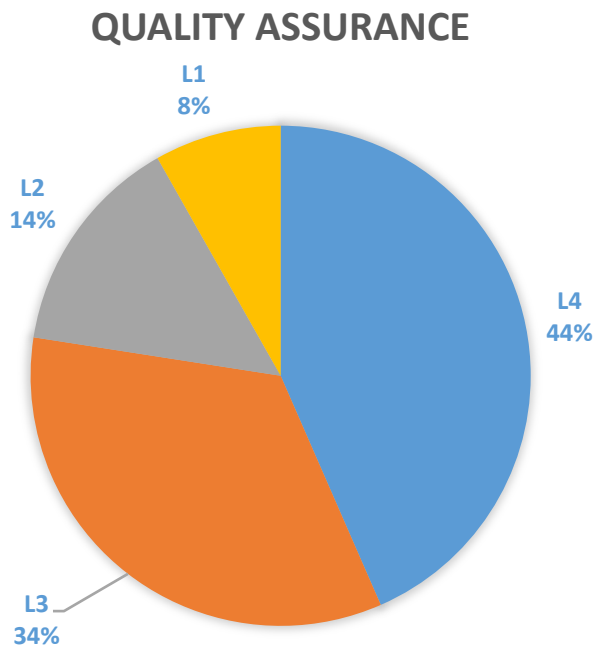
- Often difficult/costly or impossible to define the right output for each arbitrary input
- Imperfect and has limitation on performance
  - Impossible to estimate the performance beforehand
  - Impossible to describe the boundary of what can be done and what cannot be done
- Cannot deductively explain why each output is obtained
- Fragile against change of the training data
- Have adversarial examples (slight input changes cause large output change)



1 pixel attack to wrong recognition of signal color
[Wicker et al., Feature-Guided Black-Box Safety Testing of Deep Neural Networks, 2018]

Impacts on Engineers

- Expected “paradigm change”:
  - Most highly in decision making (requirements & acceptance) and testing & quality assurance
  - From a survey over 270 people in Japan



1. We can use existing methods, frameworks, or tools
2. We already have dedicated methods, etc.
3. We can apply the same approaches but methods, etc., are still immature
4. We need to use new approaches as the existing ones do not work anymore

Our Approach

- Investigate uncertainty-aware usages of assurance cases
  - Snapshot record and analysis of the current situation for ever-changing aspects (rather than one-short acceptance/release check)
  - Awareness of intrinsic uncertainty that should be taken into consideration for risk analysis

    which parts of GSN become intrinsically uncertain?

⇨ *GSN Extensions? Patterns? Tool supports?*

(1) Uncertainty of Goal Decomposition

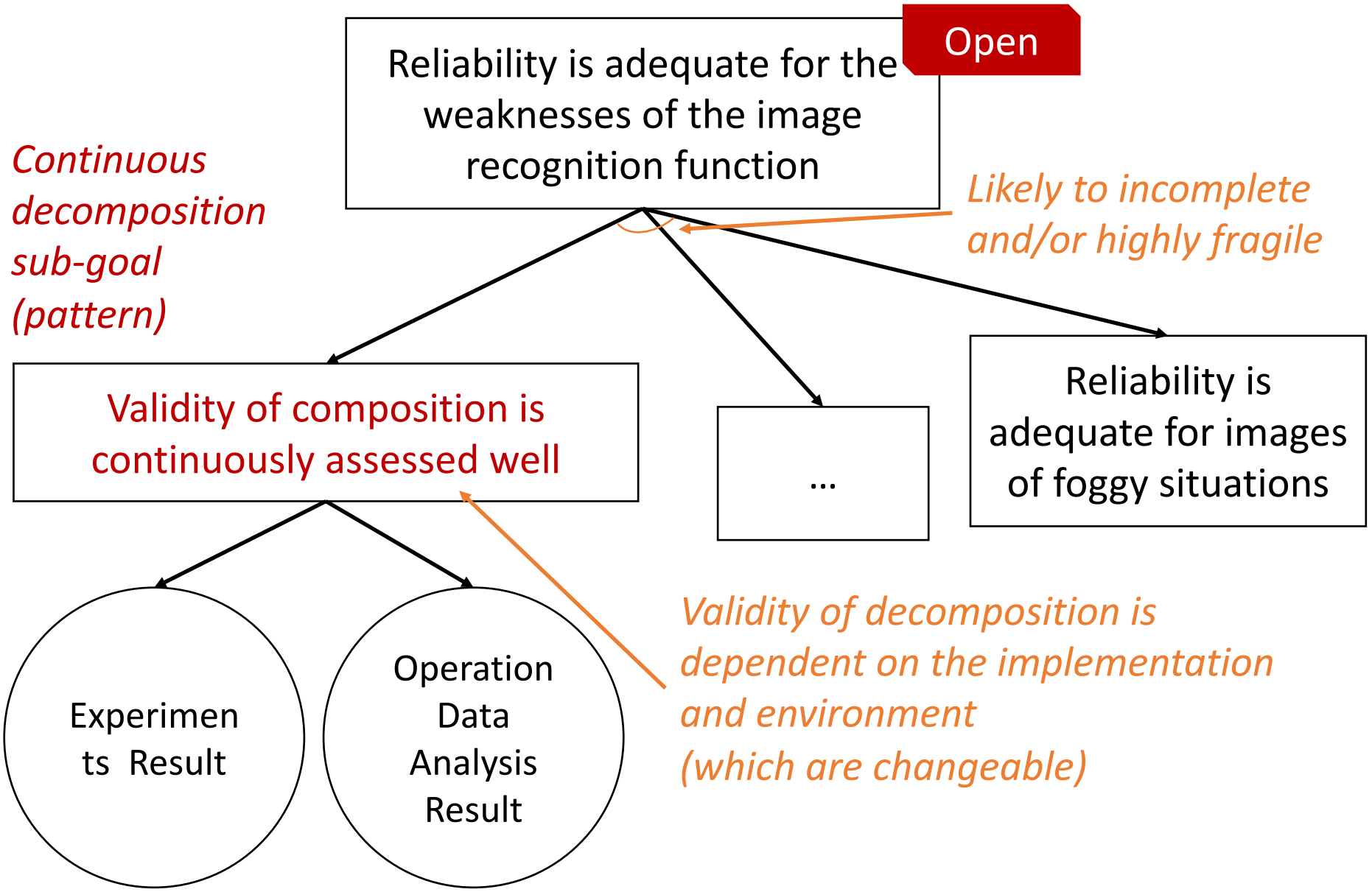
- Completeness of goal decomposition
  - One of the key factors in GSN
  - Preferably MECE (Mutually Exclusive and Collectively Exhaustive)
- With ML
  - Deals with the real world or human perceptions

  ➡ Intrinsically impossible to have exhaustive goal decomposition
  - Unexplainable

  ➡ Significance and necessity of a sub-goal can be only known after experiments and operations

Distinguish explicitly (e.g., aware of risks)!

(1) Uncertainty of Goal Decomposition



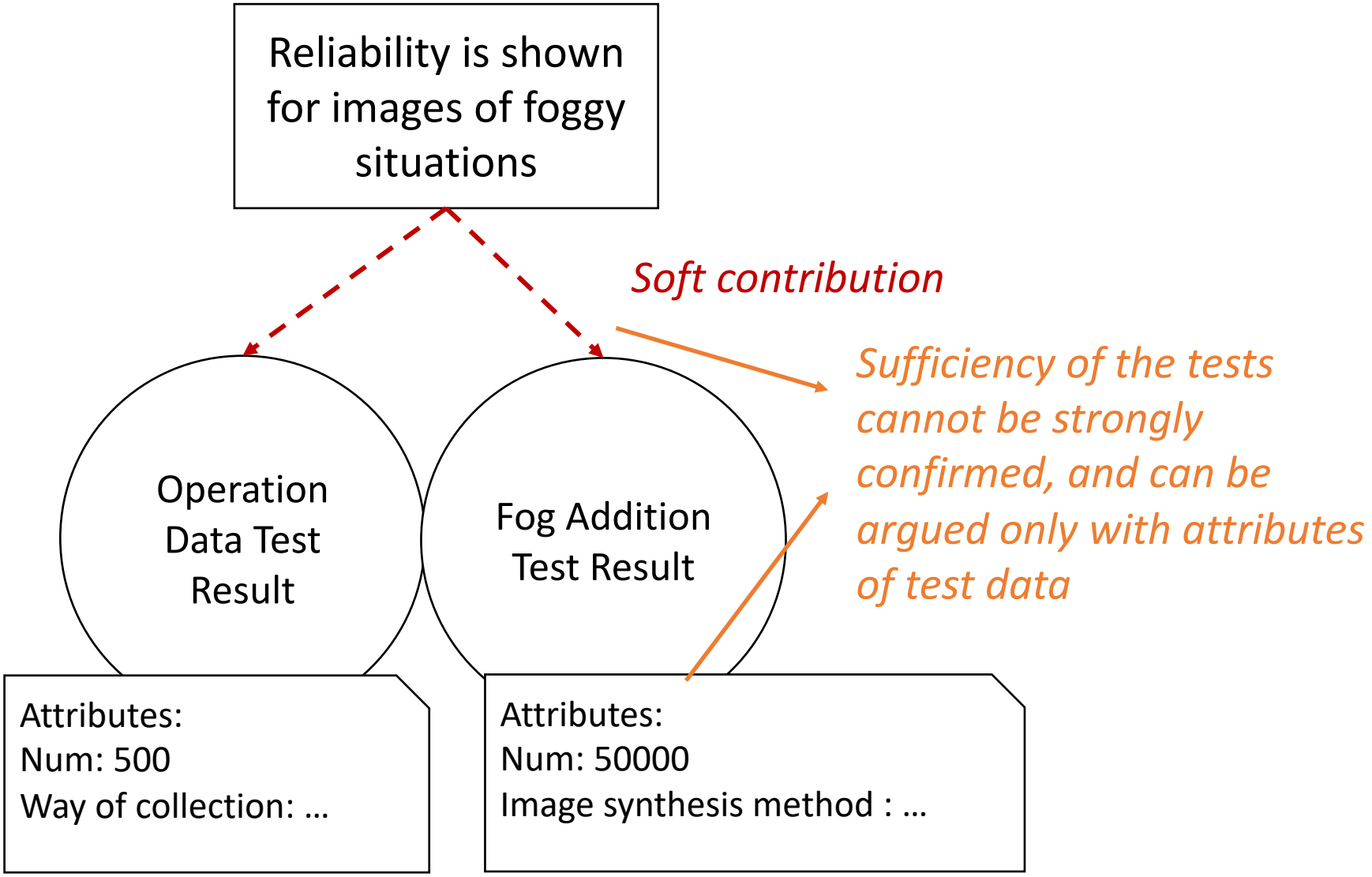
(2) Uncertainty of Evidence Contributions

- Attachment of an evidence to a goal
  - Basically the evidence considered sufficient
  - Sufficiency established deductively or empirically (e.g., test coverage, FTA, etc.)
- With ML
  - No deductive reasoning: almost no information of untested data (i.e., “equivalent class” unapplicable)

➡ A little confidence of the sufficiency of an evidence (typically a test data set)

Distinguish explicitly (e.g., aware of risks)!

(2) Uncertainty of Evidence Contributions

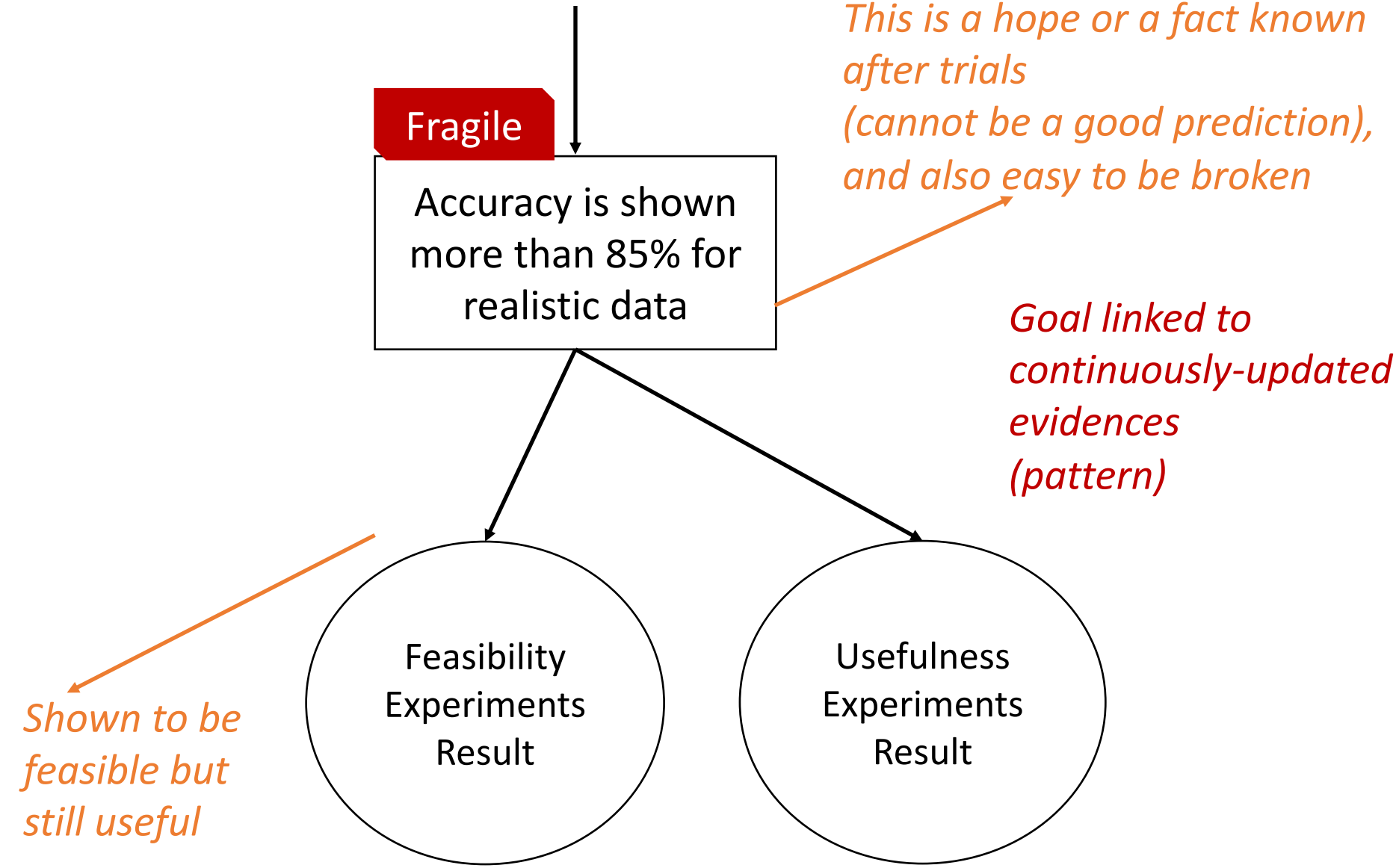


(3) Uncertainty of Feasible Goals

- Concreteness of a leaf goal
  - Significant to show its satisfaction
    e.g., representation in a quantified metric
- With ML
  - Impossible to predict the implementable performance (e.g., accuracy)
  - ➡ Concrete goals are decided in a bottom-up way by experiments and operations, and also fragile

Distinguish explicitly (e.g., aware of risks)!

(3) Uncertainty of Feasible Goals



Summary

- Investigate uncertainty-aware usages of assurance cases
  - Snapshot record and analysis of the current situation for ever-changing aspects (rather than one-short acceptance/release check)
  - Awareness of intrinsic uncertainty that should be taken into consideration for risk analysis

    which parts of GSN become intrinsically uncertain?
- *GSN Extensions, patterns, and tool supports*

Future work: case studies and tool implementations