

# Principles of AI Engineering

## Chapter 2: System with Machine Learning

Prof. Dr. Steffen Herbold

Credit:

Based on contents from Christian Kästner (<https://github.com/ckaestne/seai>)

# Contents

- The role of ML in a system
- Model goals vs. system goals
- Embedding ML systems in the real world
- Human-AI interaction
- Operating production ML systems
- Deductive and inductive reasoning
- Managing software complexity

# The role of ML in a System

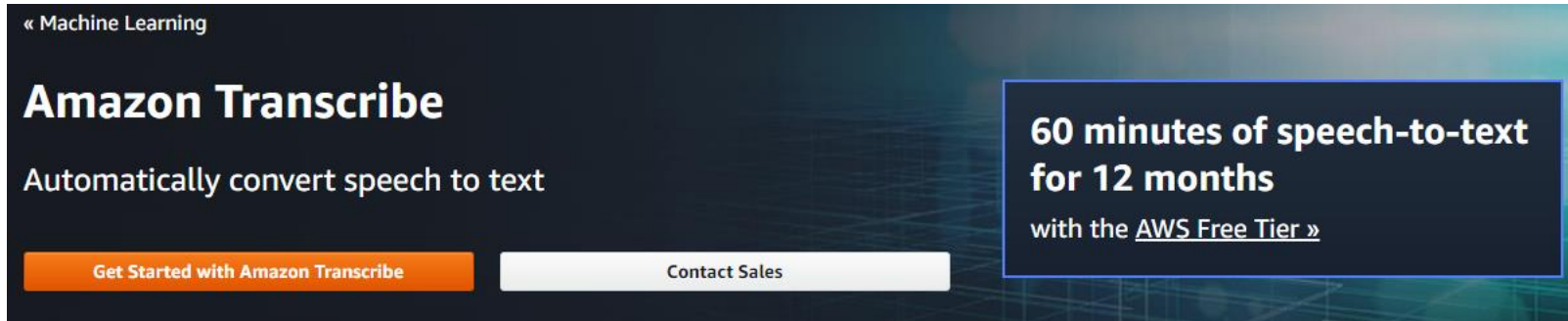
# Machine learning as a small feature in a system



<https://ttlc.intuit.com/community/choosing-a-product/help/about-the-audit-risk-meter/00/25924>

**Augments a working system with an additional optional feature**

# Machine learning as a core component

A promotional banner for Amazon Transcribe. It features a dark blue background with a subtle grid pattern. The text is white and orange. A blue-bordered box on the right highlights a free tier offer.

« Machine Learning

## Amazon Transcribe

Automatically convert speech to text

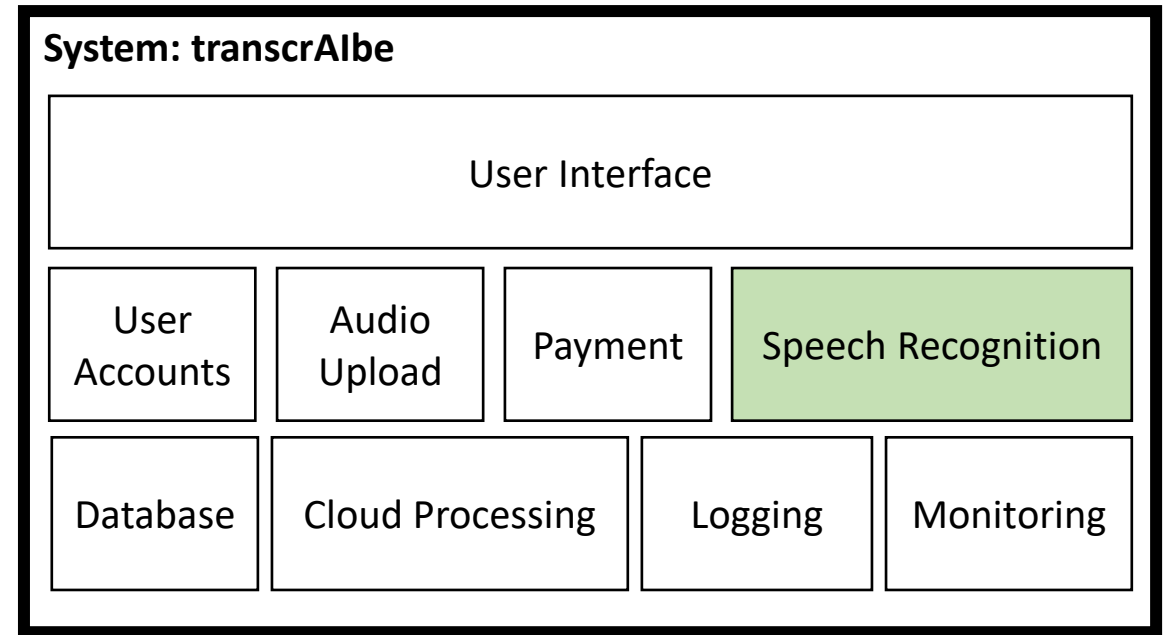
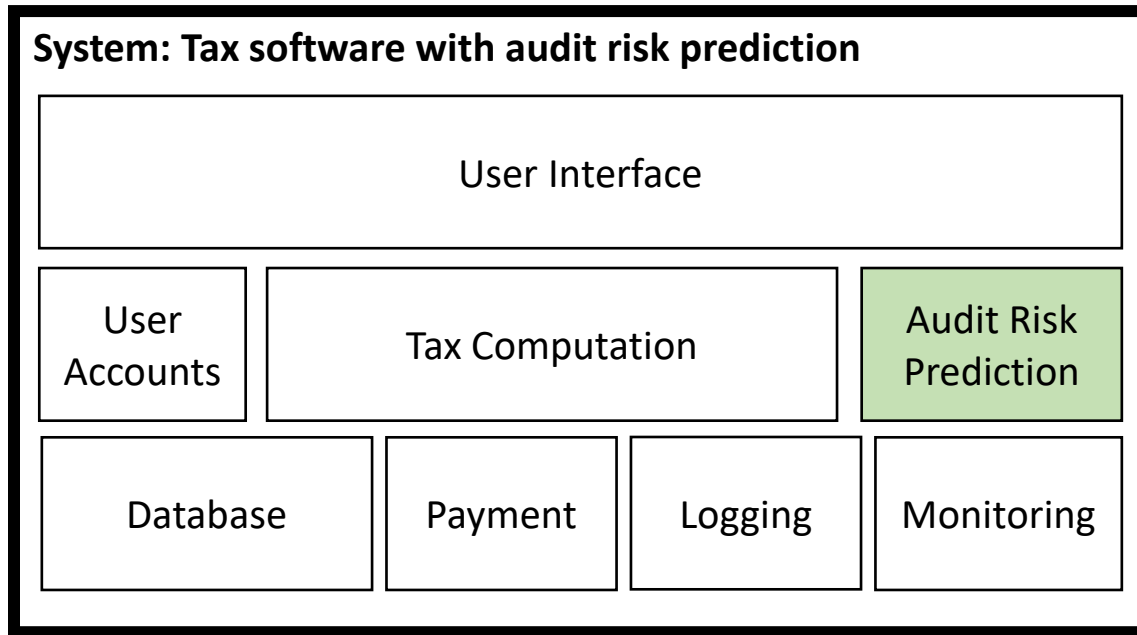
[Get Started with Amazon Transcribe](#) [Contact Sales](#)

**60 minutes of speech-to-text  
for 12 months**  
with the [AWS Free Tier »](#)

<https://aws.amazon.com/transcribe/>

**Enables a new product and is a major selling point**

... but in both cases a fairly small part of the system

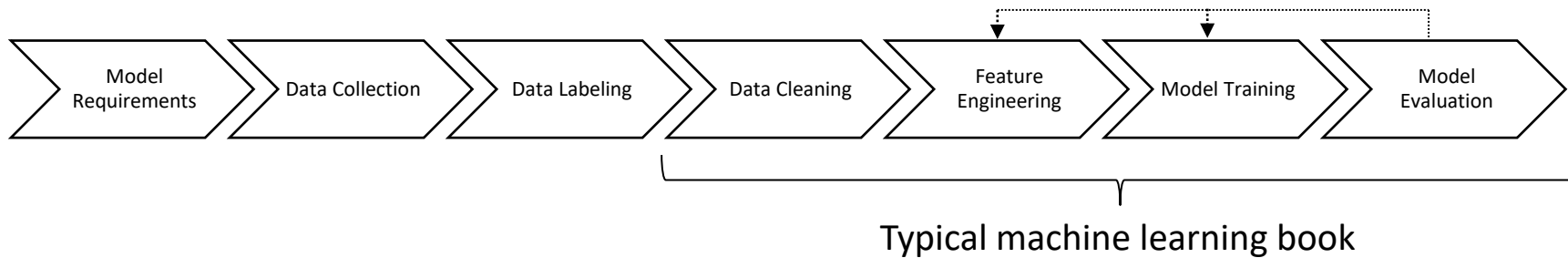


# Many examples of ML in production

- Product recommendation on Amazon
- Surge price calculation for Uber
- Inventory planning at Walmart
- Search for new oil fields by Shell
- Smart app suggestion on Android
- Social profiling for placing advertisements by Facebook
- ...

# Model-centric vs. system-wide focus

Traditional model focus (data science):

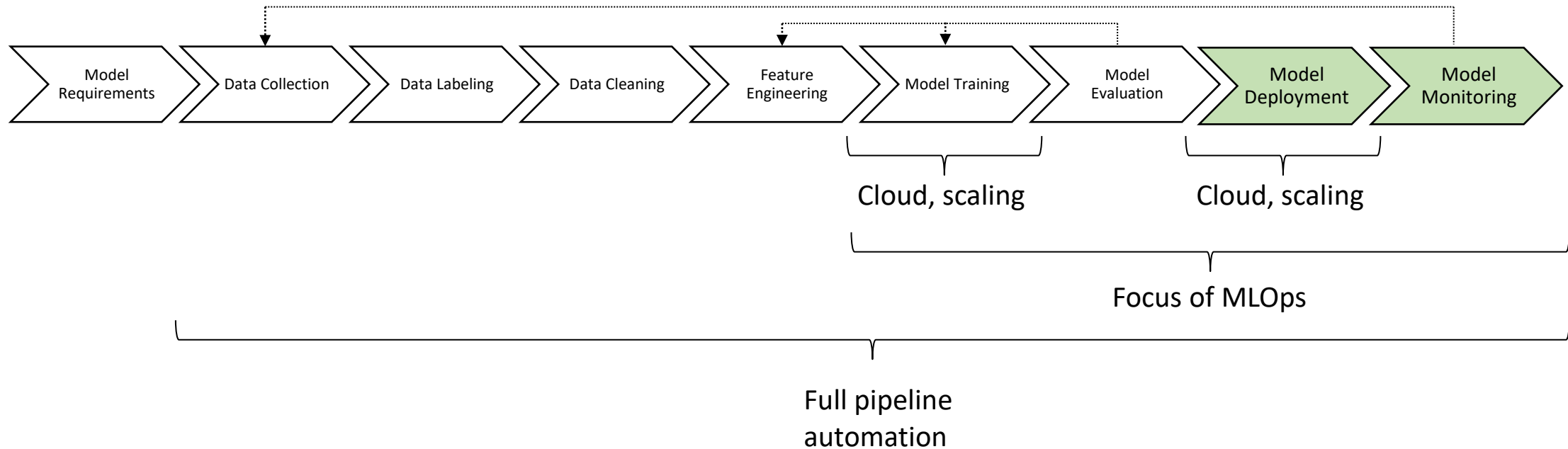


- Focus on:
  - Building models for given a given data set
  - Evaluation of the accuracy of the model



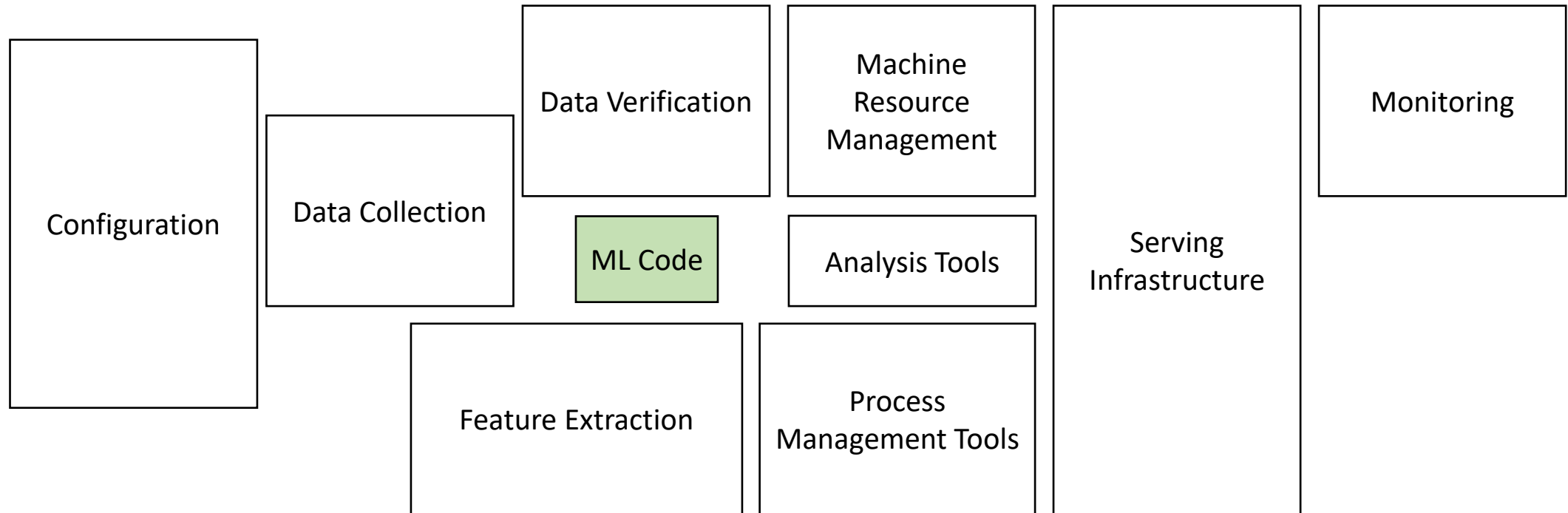
# Model-centric vs. system-wide focus

Automating pipelines and MLOps (ML engineering)



- Focus on:
  - Experimenting
  - Deploying
  - Scaling training and serving
  - Model monitoring and updating

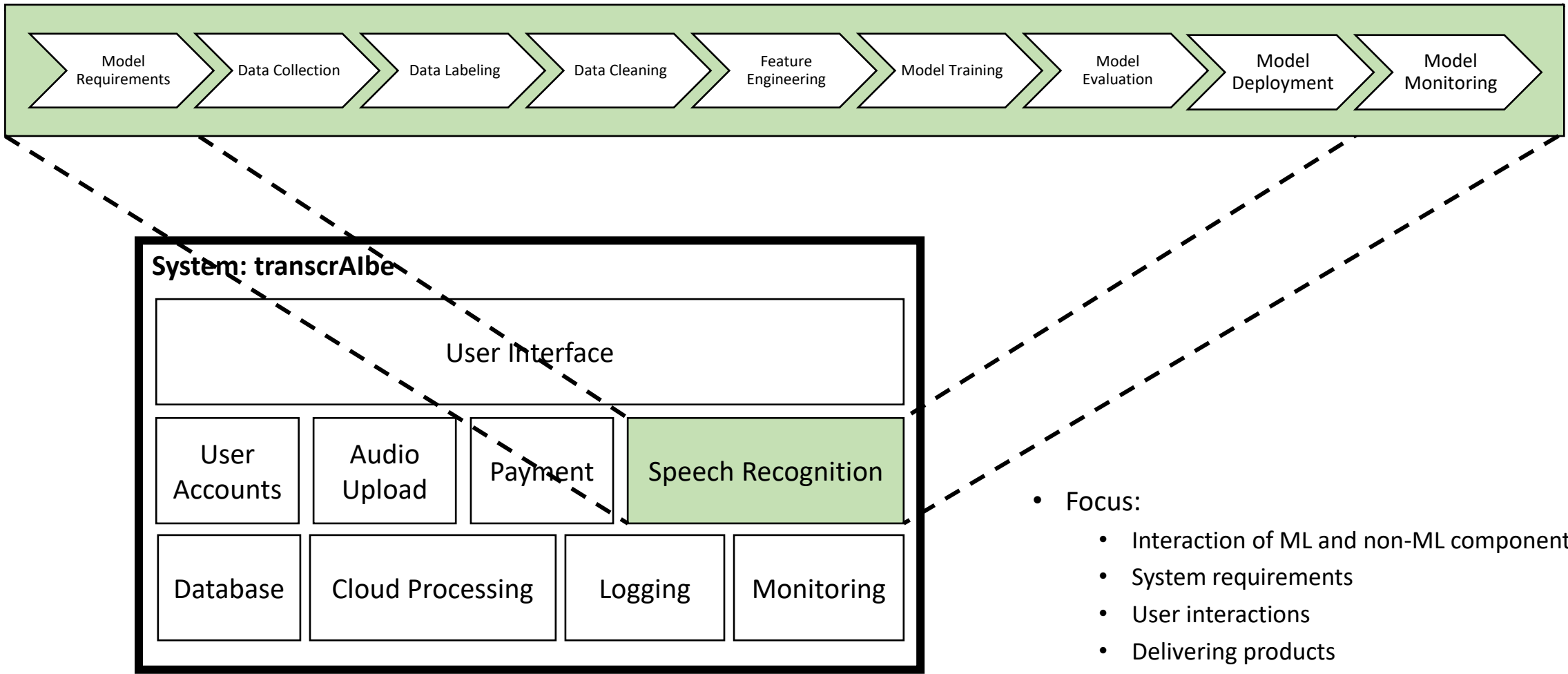
# MLOps infrastructure



**ML Code only a small part of the MLOps code!**

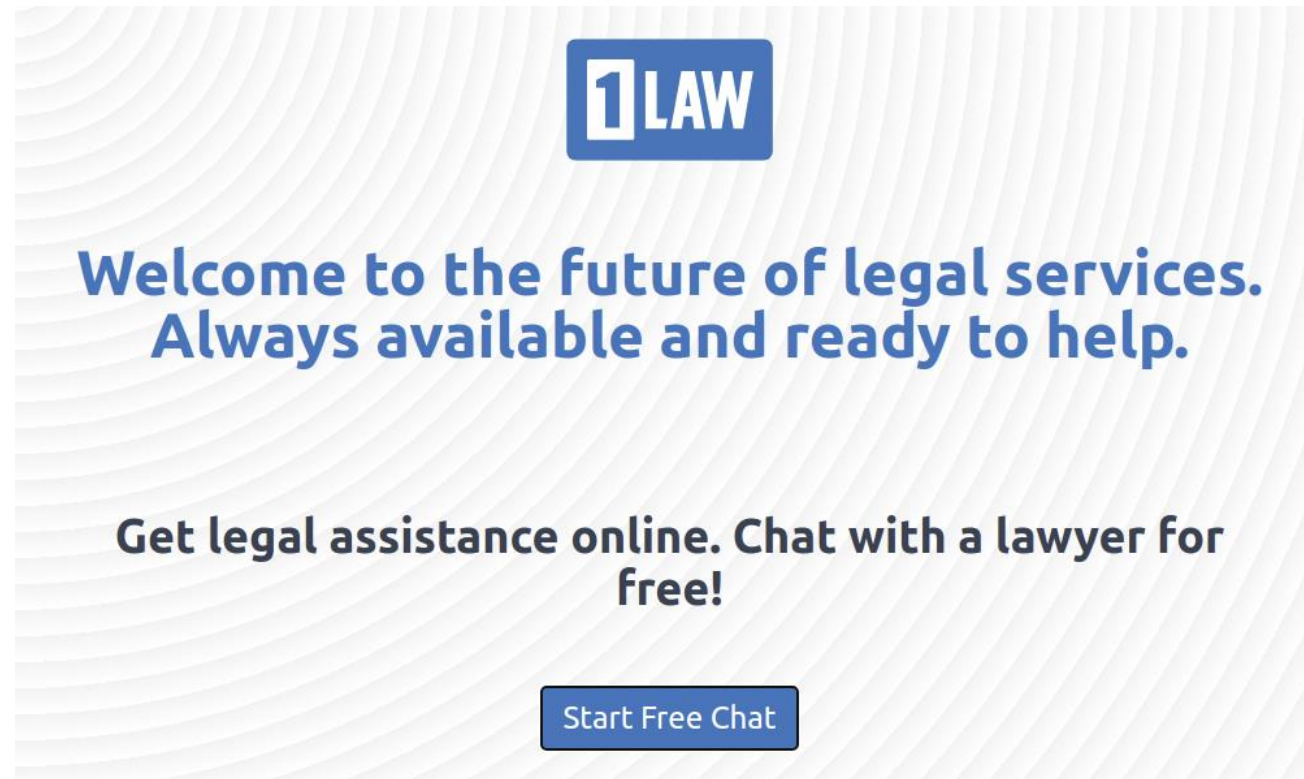
Note: Amount of other code depends a lot on tooling. Modern cloud platforms provide many aspects “out of the box”. 10

# View on ML-enabled system in production



Model goals vs. system goals

# Case study: Self-help legal chatbot



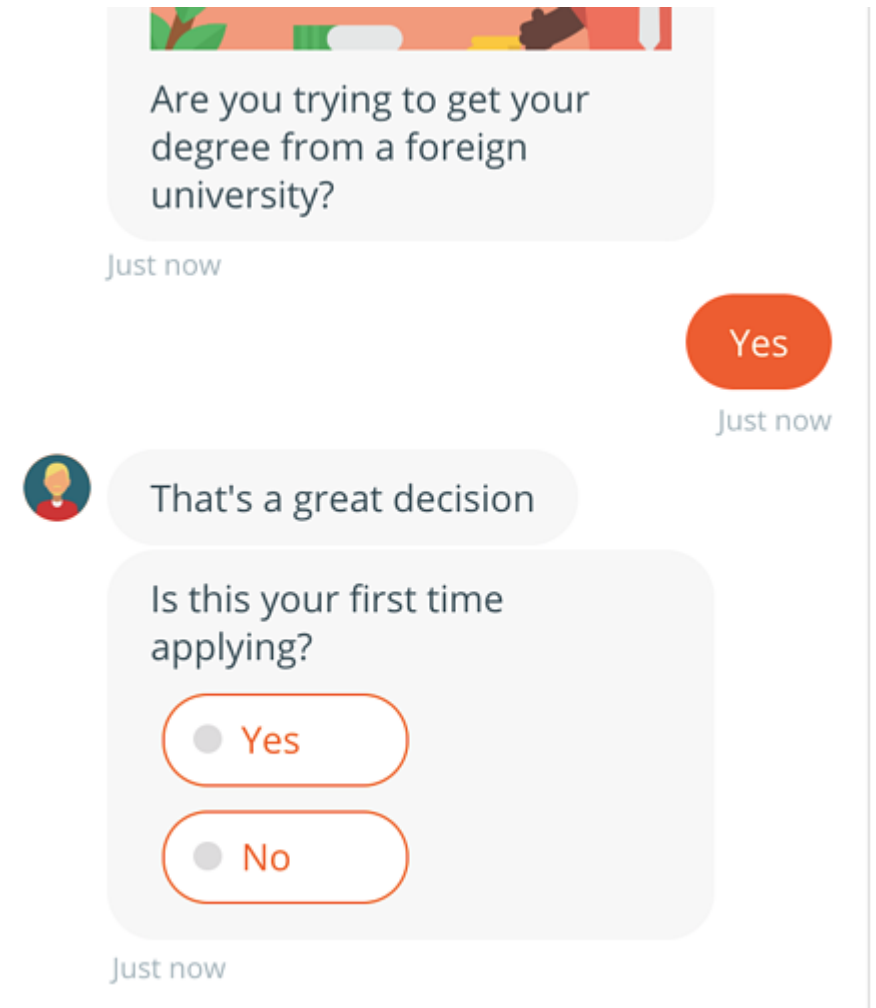
<https://www.1law.com/>

Based on: Passi, S., & Sengers, P. (2020). Making data science systems work. *Big Data & Society*, 7(2).


Note: all screenshots on the next slides are for illustration and not from the actual system

# Previous system: Guided chat

- Pre-defined set of questions and answer
- Often multiple choice
  - Just think of the guidance at telefon hotlines until you get to speak to a human
- Hard to match pre-defined content to open questions
  - „Please state your topic“ allows many ways to phrase an answer!
- Old-fashioned
- Common solution:
  - If automation fails, involve a human early
  - → Expensive!



# Initial vision



**We need a  
better chatbot!**

**Paul (director of technology of Law&You):**

People in legal chat do not always follow a script. We want to find ways to anticipate conversational pathways. We are agnostic to technology as long as it provides value—user information. (Fieldwork Notes, 31 July 2017)

- Help users with simple tasks
- Connect them with lawyers when needed
- Modernize appearance

# Buy or build?!

ChatGPT • Enterprise

Enterprise-grade security & privacy and the most powerful version of ChatGPT yet.

[Contact sales](#)

[Read launch announcement](#)

xt



Clearbit

descript

Khan Academy



Quizlet

ChatBot

Product ▾

Pricing

Integrations ▾

Resources ▾

## With ChatBot, automating customer service is a breeze

An all-in-one platform to build and launch conversational chatbots without coding.

Enter your business email

[Sign up free](#)

✓ Free 14-day trial ✓ No credit card required

<https://www.chatbot.com/>

# AWS Chatbot

ChatOps für AWS

[Probieren Sie AWS Chatbot aus](#)

<https://aws.amazon.com/chatbot/>

[Log in](#)

[Sign up free](#)

[Home](#) / [Products](#) / [Azure Bot Service](#)

## Azure Bot Service

A comprehensive development environment for designing and building enterprise-grade conversational AI

[Start free](#)

[Already using Azure? Get started with Bot Service >](#)

<https://azure.microsoft.com/services/bot-services>

Botsify

Products ▾

Partner ▾

Resources ▾

Pricing

Demo

Sign in

[Signup](#)

[Become whitelabel partner and resell chatbots to your clients!](#)

## Build Conversational Flows With No-Code Chatbot Platform

Botsify is a managed chatbot platform that provide unified chat automation for your business. Get omnichannel live-chat service connected with multiple platforms to set autoresponses

[Get Free Trial](#)

[Explore Platform Now](#)



[Book Demo Now](#)

[Free 1-1 Product Tour](#)



<https://botsify.com/>



# Data science challenges

- Infrastructure
  - Understand chat bot infrastructure and capabilities
- Knowing topics
  - Identify what users talk about
  - Train/test concepts with past chat logs
- Guiding conversations
  - Support for open-ended conversations
    - Requires detecting topic and finding a good response
    - Intent modeling!
  - Is talk about parents and children on topic when discussing divorce?
    - Many corner cases!

# Live exercise



What are the system goals?

How are the data science challenges related to them?

# Status meeting



**Paul**  
Director of  
Technology

Maybe we need to think about it like an 80/20 rule. In some cases, it works well, but for some, it is harder. 80% of everything is fine, and in the remaining 20%, we try to do our best.

The trouble is how to automatically recognize what is 80 and what is 20.

I agree. Let us focus on that. We just want value. Tech is secondary.

It is harder than it sounds. One of the models is a matching model trained on pairs of legal questions and answers. 60,000 of them. It seems large but is small for ML.

That's a lot. Can it answer a question about say visa renewal?

If there exists a question like that in training data, then yes. But with just 60,000, the model can easily overfit, and then for anything outside, it would just fail.

I see what you are saying. Edge cases are interesting from an academic perspective, but for a business the first and foremost thing is value. You are trying to solve an interesting problem. I get it. But I feel that you may have already solved it enough to gain business value.



**Remy**  
Director of  
Data Science

# System goals for the chatbot

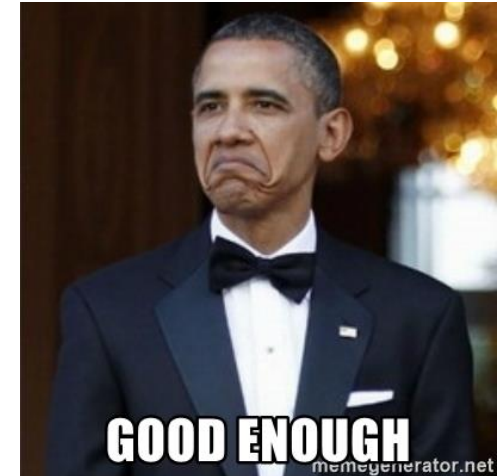
- Collect user data and sell to lawyers
- Signal technical competency to lawyers
- Acceptable to fail!
  - If the conversation is too complicated for the bot, connect with lawyer
- Solving edge cases is not important



**Paul:** Edge cases are important, but the end goal is user information, monetizing user data. We are building a legal self-help chatbot, but a major business use case is to tell people: 'here, talk to this lawyer.' We *do* want to connect them with a lawyer. Even for 20%, when our bot fails, we tell users that the problem cannot be done through self-help. Let us get you a lawyer, right? That is what we wanted in the first place.

# Accuracy is not everything in practice!

- “Good enough” may be good enough
- Prediction critical for the system or just a gimmick?
- Costs for better predictions are important
  - May need way more data, much longer training times
  - May need other types of data and raise privacy concerns
- User interfaces can help mitigate many problems
  - “User Experience” (UX) matters often more than predictions
- Using high confidence predictions can help offset issues

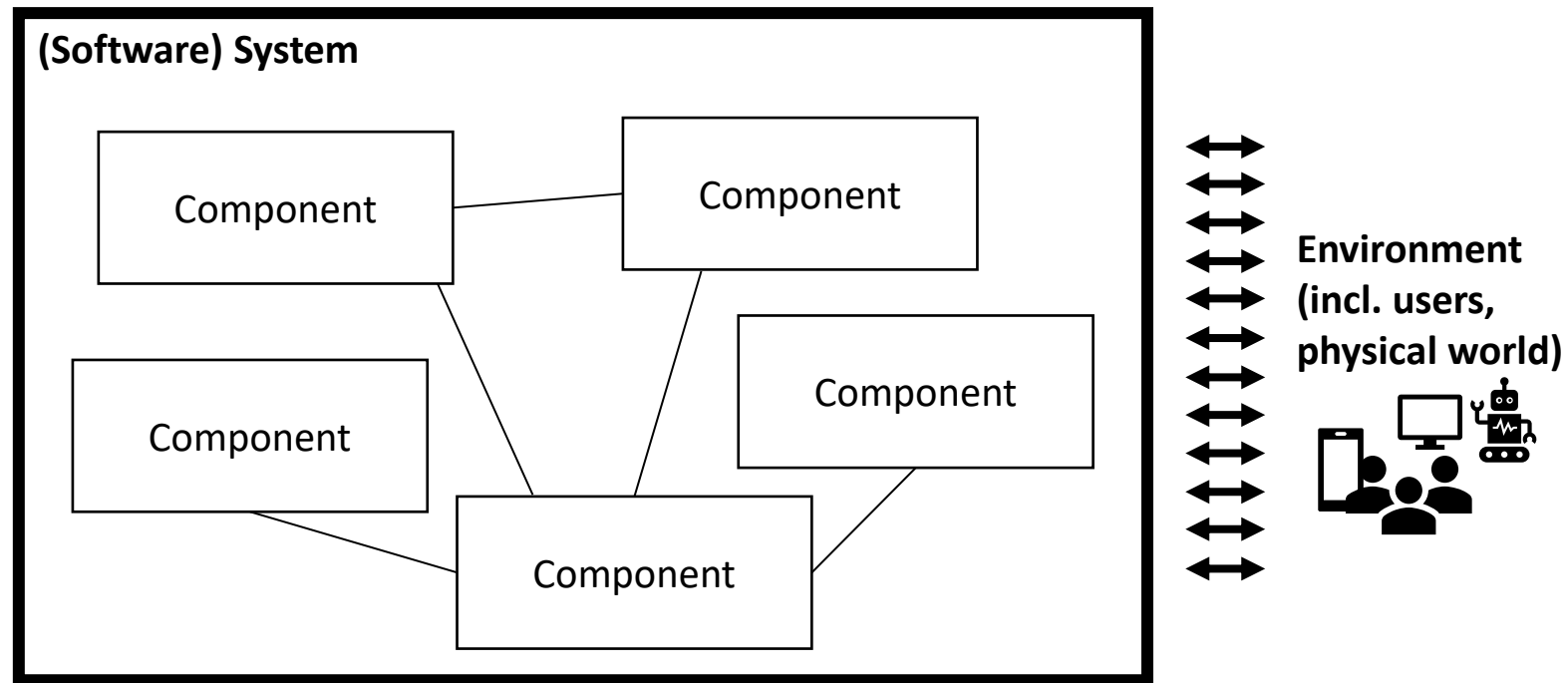


# Benchmark vs. production

- In 2012, researchers already complained about the focus on benchmarks in ML
- Standard benchmarks not directly related to real-world problems (Iris, MNIST, ImageNet, ...)
  - Focus on abstract metrics, not real-world impact
  - Accuracy instead of Return on Investment (ROI)
- Lack of follow-through with deployments in real-world use cases
- Ignores important design choices
  - How to collect data
  - What problem to solve
  - Design of the human-AI interface
- **This got worse since then!**
  - E.g., good Large Language Model (LLM) evaluations are extremely difficult, because the benchmarks are getting less and less representative

# Embedding ML systems in the real world

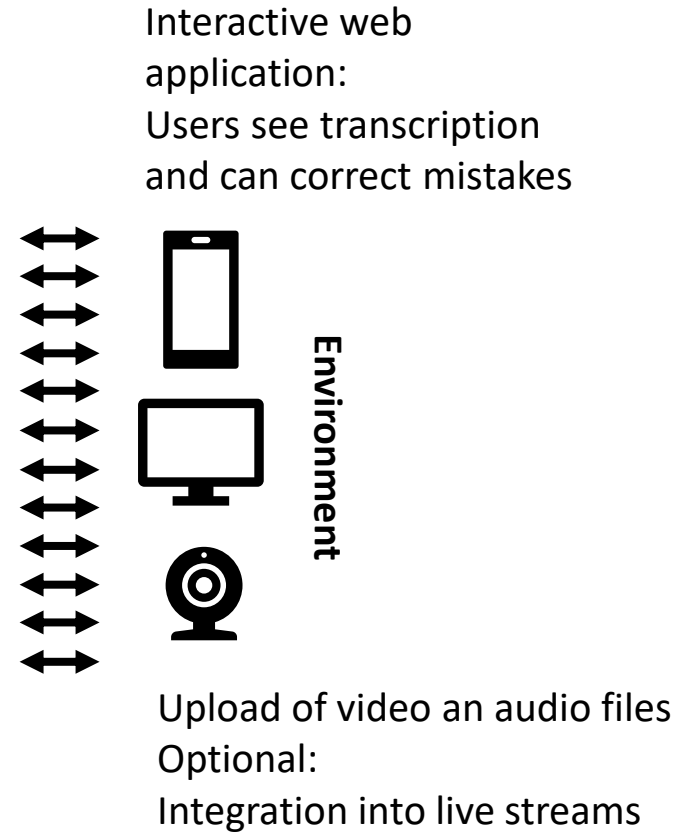
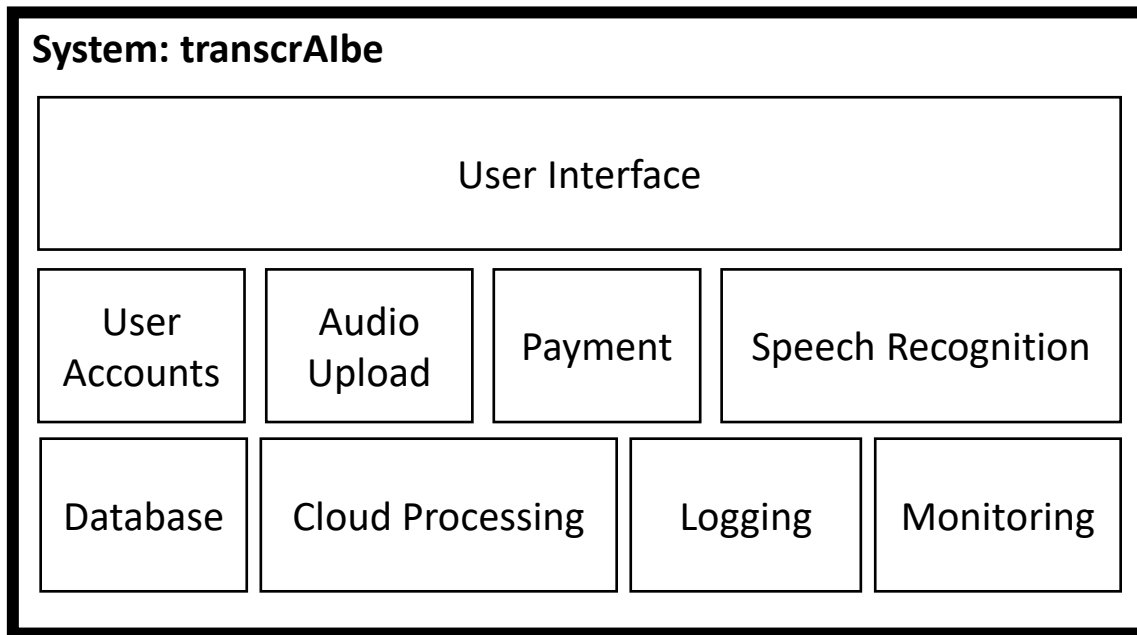
# Systems thinking



**Consider how your system as a whole interacts with the world!**



# Example 1: Speech recognition



# Example 2: Audit risk



Static reporting of results. No interaction.

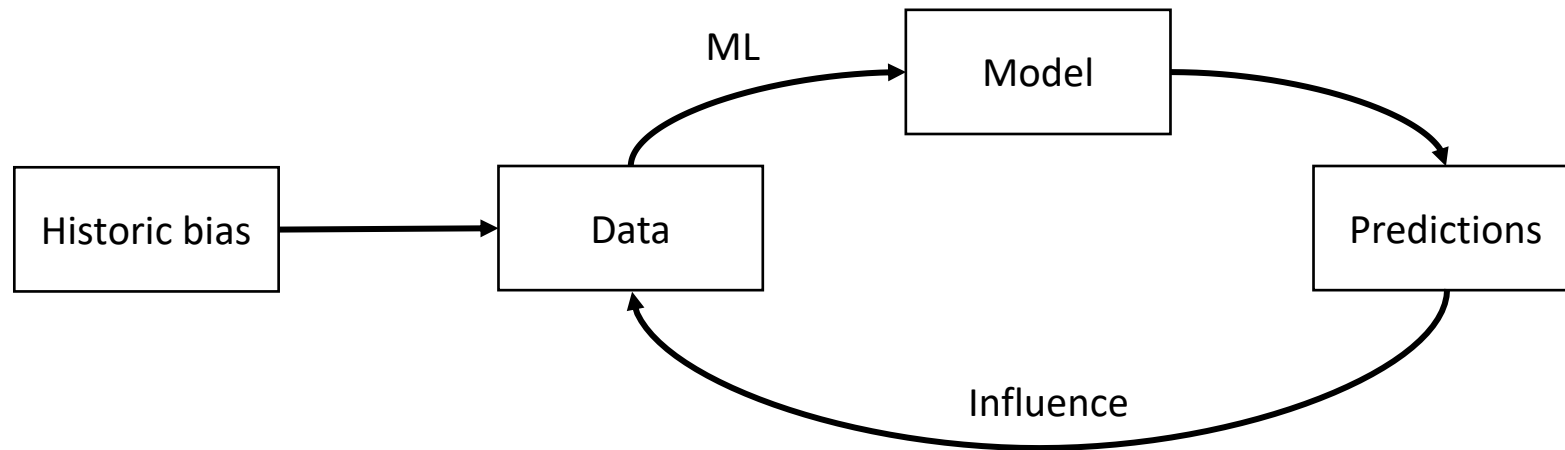
# Example 3: Robotics



Spot by Boston Dynamics

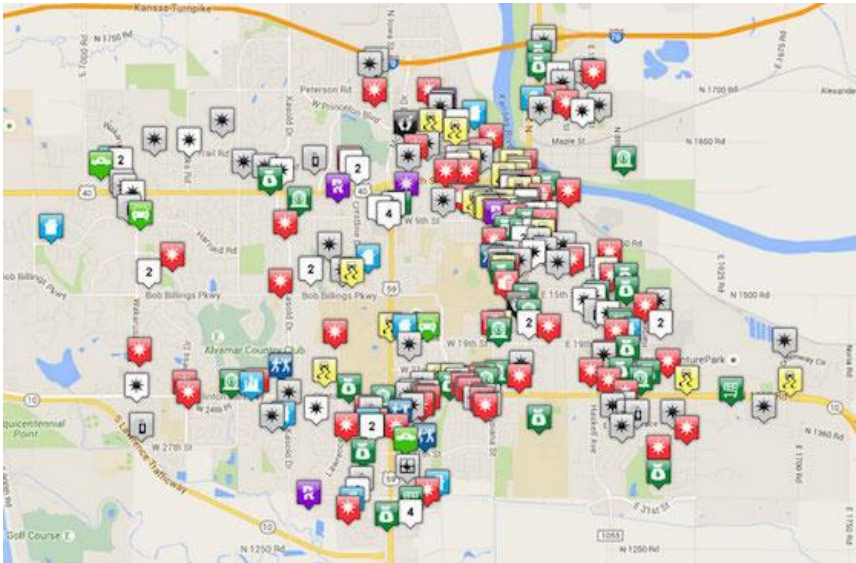
Observes and actively interacts with physical environment

# System $\leftrightarrow$ World feedback loops



**Can encode, enforce, and even enhance existing bias!**

# Example 4: Predictive policing



Guide police patrols based on past crime rates

Technology  
Review

Featured

Topics

Newsletters

Events

Podcasts

Sign in

Subscribe

TECH POLICY

## Predictive policing is still racist—whatever data it uses

Training algorithms on crime reports from victims rather than arrest data is said to make predictive tools less biased. It doesn't look like it does.

By Will Douglas Heaven

February 5, 2021

<https://www.technologyreview.com/2021/02/05/1017560/predictive-policing-racist-algorithmic-bias-data-crime-predpol/>

... really bad idea ...

# ML predictions have consequences

- Positive

- Assistance
- Productivity
- Creativity



- Negative

- Manipulation
- Polarization
- Discrimination



**AI Engineering must be responsible!**

# Safety is a system property

- Code or models do not harm people directly
- The systems in which they are used might
- Example: Computer Vision



<https://www.youtube.com/watch?v=fKXztwtXaGo>

Always safe if you download the data and train/test models offline

→ Not embedded in a system

Was involved in traffic incidents when used to make decision on the road

→ Safety is an issue of the use in the system



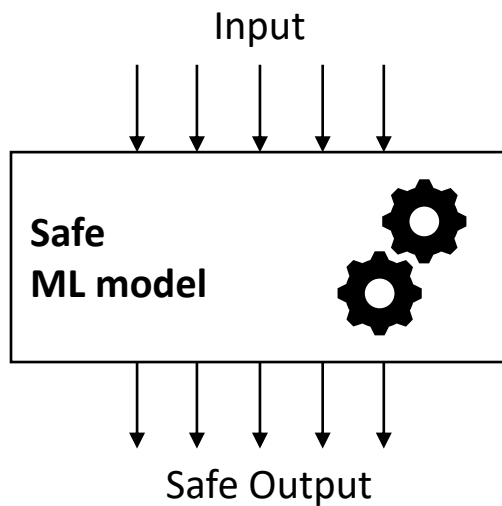
# Live Exercise: A smart toaster

- Use machine learning to regulate toasting for optimal result
- Safety goal:
  - Ensure that the smart toaster does not burn the kitchen
- How can this be achieved?

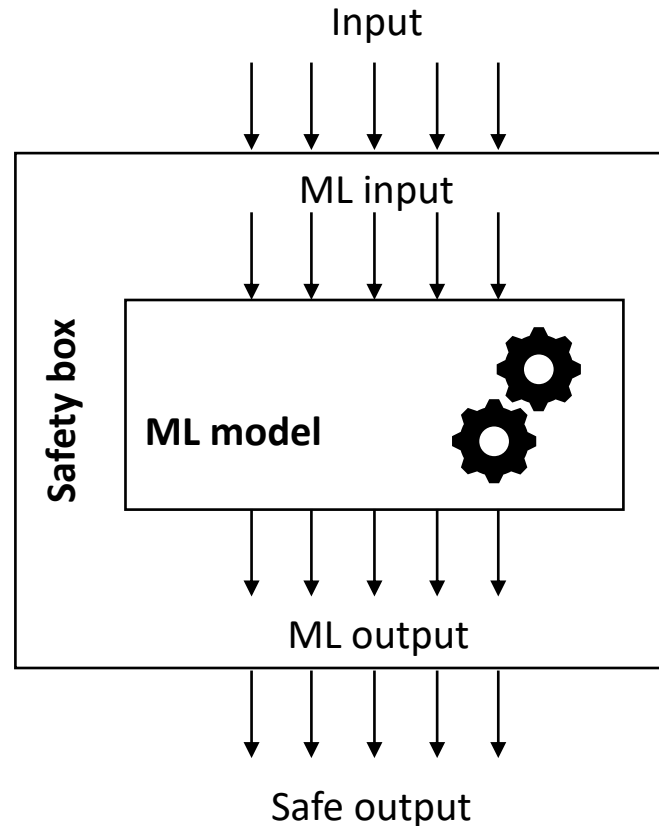




# Safety Assurance in the model vs. outside the model



Model must ensure that the output is safe for **every** possible input  
→ Model verification, hard and (currently) unsolved for most practical problems



May restrict input space  
→ Avoid inputs from unknown regions

May transform outputs  
→ Replace unsafe ML outputs with safe outputs

Safety box must ensure that the output is safe for every possible input  
→ Allows lightweight solutions

# Safety box for the smart toaster

- Software solution:
  - Post-process output and ensure a maximum toasting time
- Hardware solution:
  - Install a thermal fuse
- HW/SW solution
  - Install a temperatur sensor
  - Software initiates shutdown if temperature too high



# Model vs system properties

- Similar to safety, many other qualities should be discussed at model and system level
  - Security
  - Privacy
  - Transparency and accountability
  - Maintainability
  - Scalability
  - Energy consumption
  - Impact on system goals
  - ...

# Think about systems!

- Holistic approach
  - Look at the big picture
  - Involve all stakeholders
- Consider relationships and interactions between components and environment
  - Everything is interconnected
  - Combining parts creates something new with emergent behavior
  - Developers must understand the dynamics, be aware of feedback loops and the effect of actions
- Understand how humans interact with the system

# System-level challenges for ML-enabled systems

- Getting (and updating) data and concept drift
- Handling massive amounts of data
- Interactions with the real world and feedback loops
- Lack of modularity of AI components
- Lack of specifications
- Deployment and maintenance
- Versioning, debugging, and incremental improvements
- Keeping training and operation costs manageable
- Interdisciplinary teams
- Balancing system and model goals
- ...

# Human-AI interaction

# Predictions influence the world

- Product or music recommendations
- Feed curation in social media and news
- Image search engines
- ...
- ...
- Smart homes
- ...
- ...
- Self-driving cars
- Robotic workers

Today!

Today?

Tomorrow?

# Model outputs and user interactions

- Design considerations
  - How much information should be revealed?
    - Suggestion with reason and likely consequence (do X because Y to achieve Z with a certain probability)
    - Suggestion with reason (do X because Y)
    - Suggestion (do X)
    - Automatically take action
  - How to influence the user towards the system's goal?
  - How to minimize the consequences of flawed predictions?
  - How to collect data to continue to learn from users and mistakes?
- Requires balancing of (at least!) three system-level outcomes:
  - Achieving objectives
  - Protection from mistakes
  - Collecting data for training



# Types of result presentations

- Automation
  - Take action on user's behalf
- Prompt
  - Ask the user if an action should be taken
- Organize
  - Display a set of items in an order
- Annotate
  - Add information to a display
- Hybrids of these

# Live Exercise: Self-driving car

- What should the system achieve?
- What are trade-offs?
  - Objectives?
- Protection from mistakes?
- Collecting data?



<https://www.youtube.com/watch?v=fKXztwtXaGo>

# Operating production ML systems

# Things change ...

- Newer, better models are released
  - Better model architectures
  - More training data

- Goal and scope change
  - Support for new use cases
- Model re-training due to drift
  - Changes in the environment

→ Should be supported by online experimentation!

Paper's published in 2021:

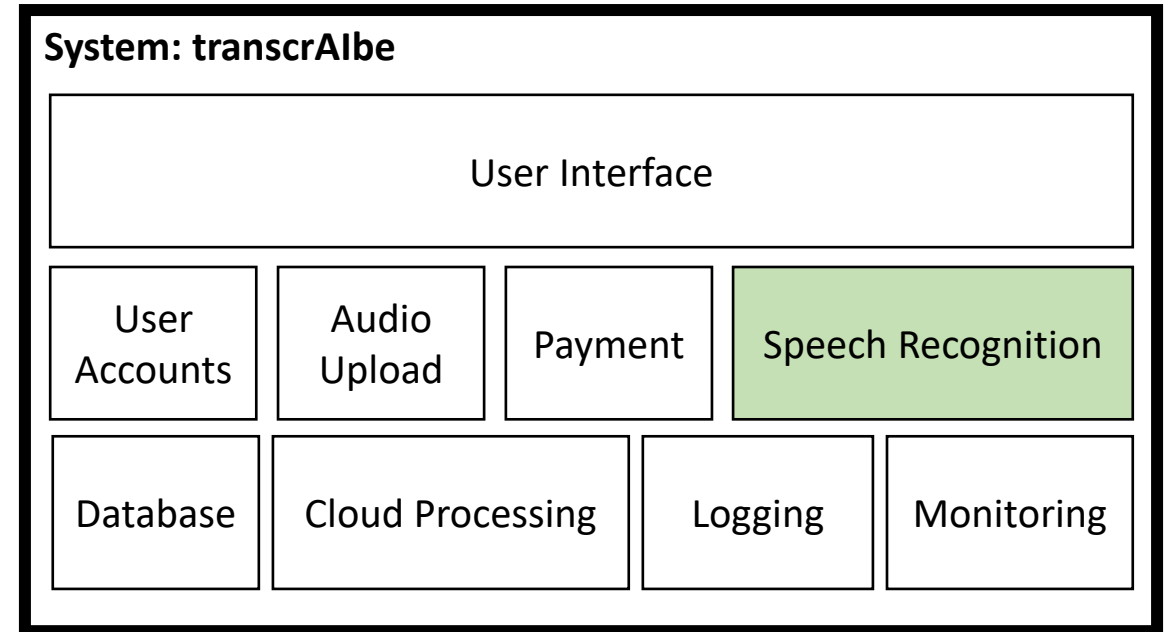
|              |      |
|--------------|------|
| NeuroIPS:    | 2334 |
| AAAI:        | 1692 |
| ICLR:        | 860  |
| ICML:        | 1184 |
| CVPR:        | 1661 |
| ACL:         | 1167 |
| ICASSP:      | 1734 |
| INTERSPEECH: | 963  |

**→ 11595 paper's at these venues in \*one year\*!**

# Example: Speech recognition

- New speech-to-text models
- New words in language
  - Social developments
  - New concepts or products
  - Slang
- Support for new domains
  - E.g., new research domains
  - New languages
  - Regional dialects

→ Observe how each change affects the system



# Monitoring in production

- Design systems for telemetry
- Example
  - Request active feedback on wrong predictions


## Report Incorrect Phishing Warning

If you received a phishing warning but believe that this is actually a legitimate page, please complete the form below to report the error to Google.


When you submit sites to us, some account and system information will be sent to Google. We will use the information you submit to protect Google products, infrastructure, and users from potentially harmful content. If we determine that a site violates Google's policies, we may update the site's status in our Transparency Report and share the URL and its status with third parties. You may find out more information about the Transparency Report [here](#). Information about your report will be maintained in accordance with Google's [Privacy Policy](#) and [Terms of Service](#).

URL:

☐ I'm not a robot

  
reCAPTCHA  
[Privacy](#) · [Terms](#)

Additional details:  
(Optional)

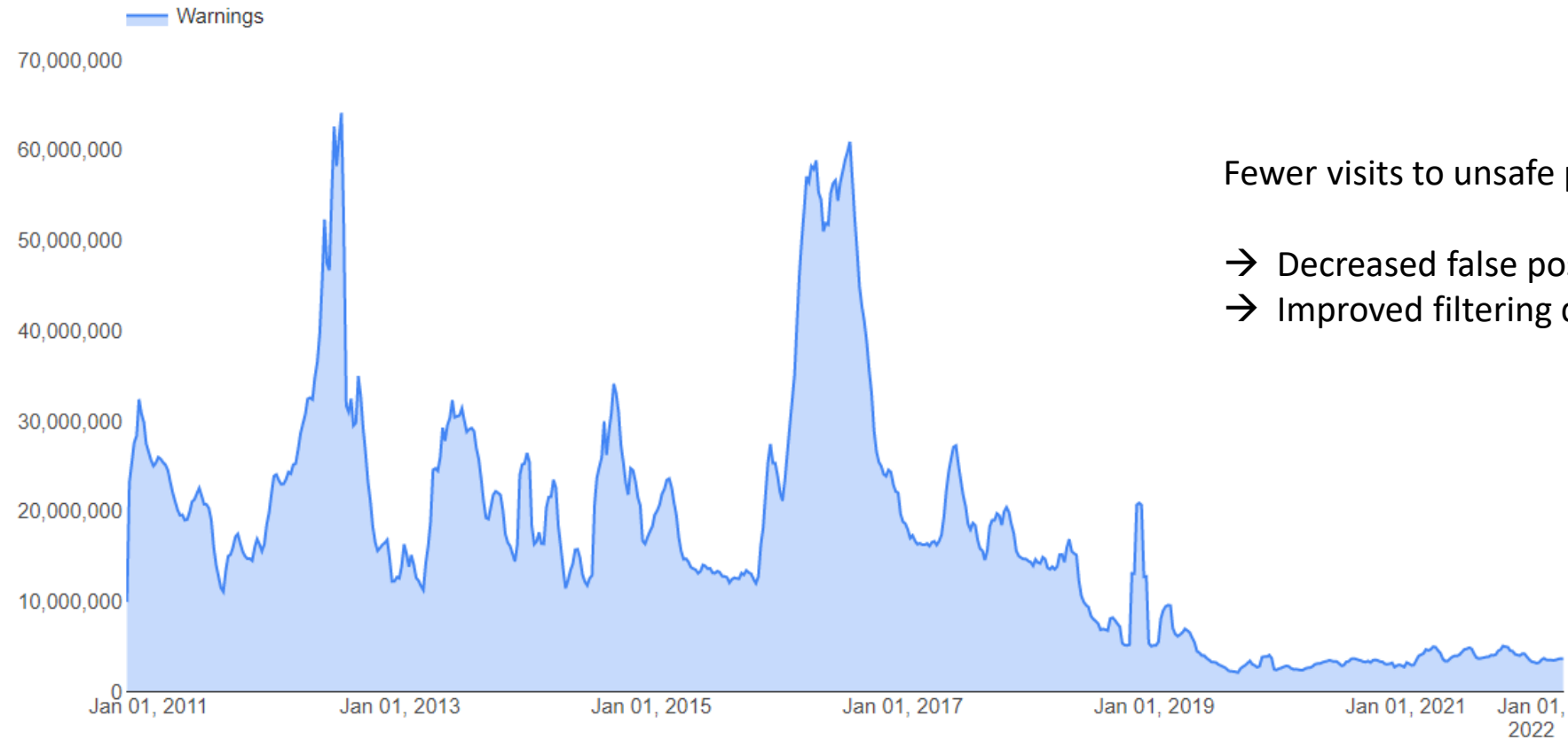


[https://safebrowsing.google.com/safebrowsing/report\\_error/?hl=en](https://safebrowsing.google.com/safebrowsing/report_error/?hl=en)

# Using data improves the system!

Warnings displayed per week

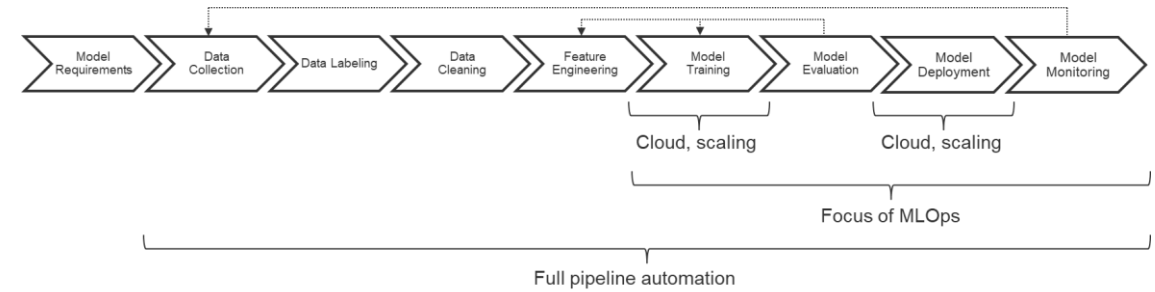
Start 📅 10/31/2010 End 📅 4/3/2022



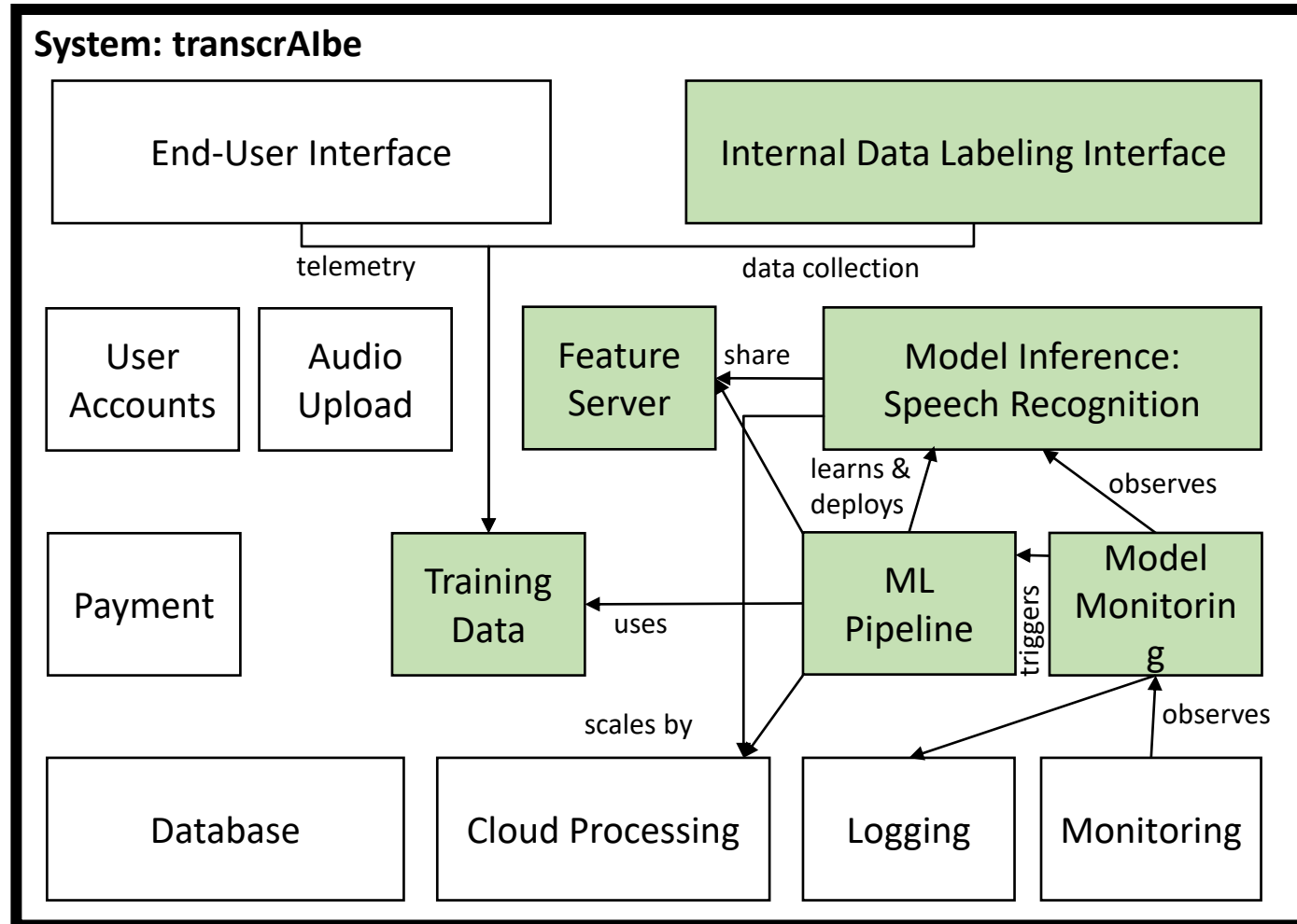
Fewer visits to unsafe pages!

- Decreased false positive rate?
- Improved filtering during search?

# „Pipeline thinking“



Design the system with the pipeline in mind



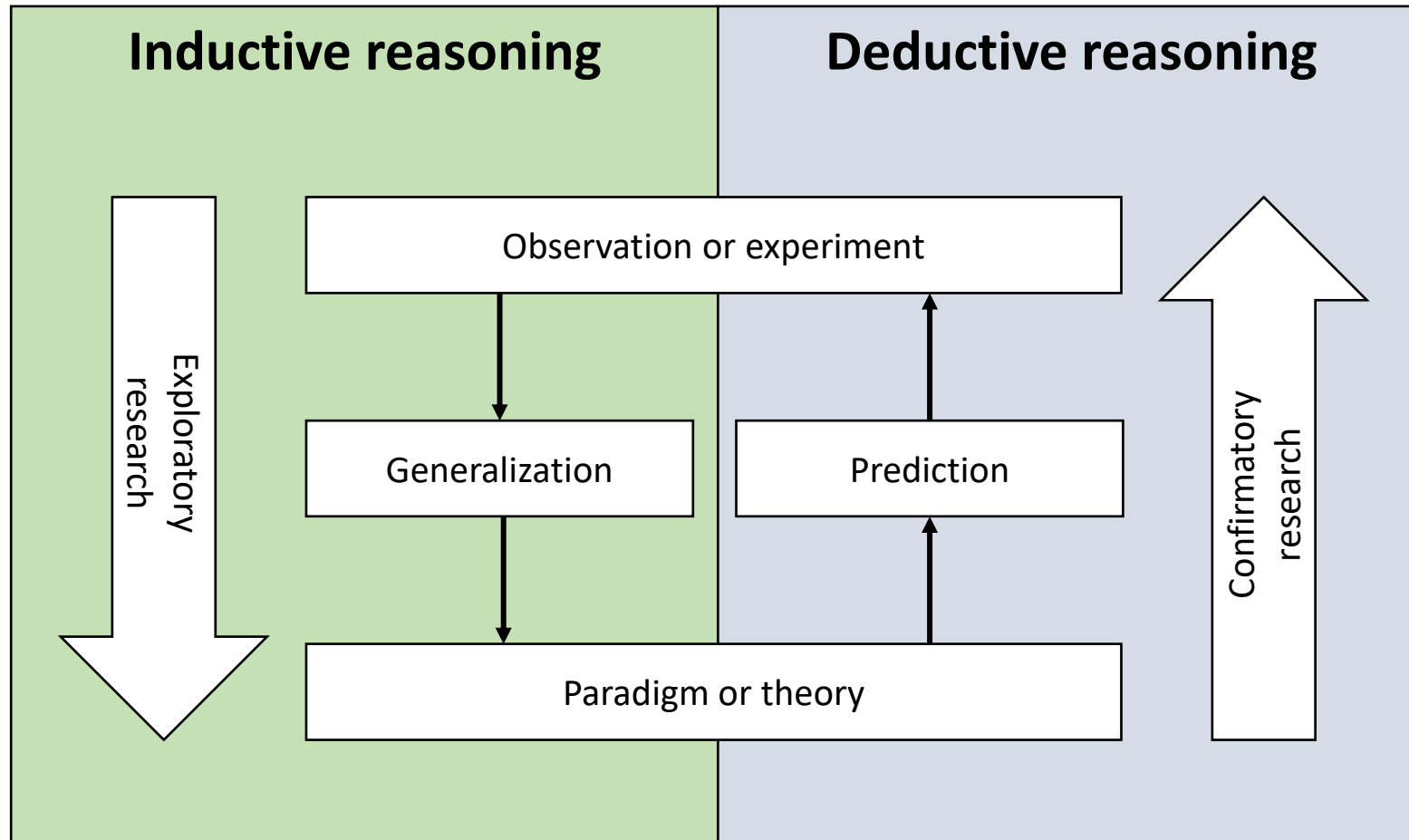


# Shift from model focus to pipeline focus

- Modeling in a local environment
  - Model-centric workflow that allows rapid prototypical model development
  - Rarely robust infrastructure
  - Often monolithic and tangled
- Shift to pipeline
  - Robust programs
    - Input data, infrastructure, nothing “hard coded”
  - Standardized and modular infrastructure
    - Cloud services, Kubernetes, libraries
- Big conceptual leap that requires a different skill-set

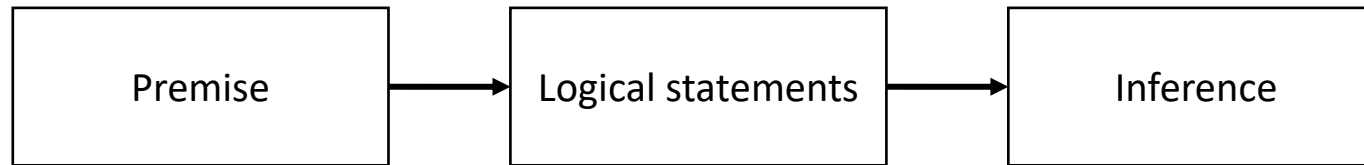
# Deductive and inductive reasoning

# Observations, paradigms, theories, and experiments



# Deductive reasoning

- Combining logical statements following agreed upon rules to form new statements



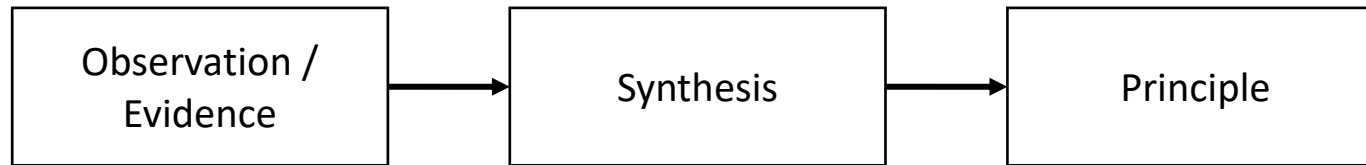
- Examples
  - Proving theorems from axioms
  - From general to the particular
  - Sherlock Holmes solving cases

**If the premise is true, the  
inference is also true.**

- Formal methods, classic rule-based AI systems

# Inductive reasoning

- Constructing axioms from observations



- Examples
  - The score increases after the ball went into the goal → number of goals equal score
  - From particular to the general

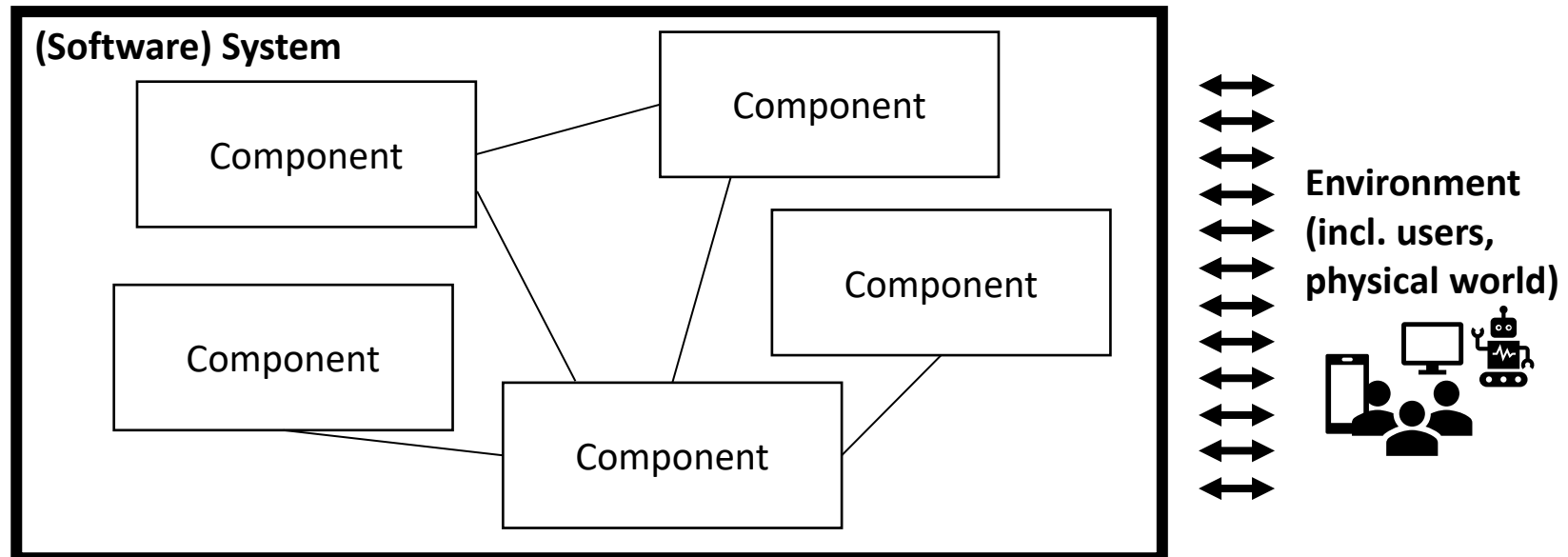
**Principle is probably true based  
on the evidence**

- Most modern machine learning systems, statistical learning

# Managing software complexity

# Methods to manage complexity

- Abstraction
  - Hide details and focus on high-level behaviors
- Reuse
  - Package into reusable libraries and APIs with well-defined contracts
- Composition
  - Build large components out of smaller ones



# Example: Manage complexity within code

Composition:

Small building block of a large system (e.g., audit risk assessment)

```
/**
 * compute deductions based on provided adjusted
 * gross income and expenses in customer data.
 *
 * see tax code 26 U.S. Code A.1.B, PART VI
 *
 * adjusted gross income must be positive;
 * returned deductions are not negative.
 */
float computeDeductions(float agi, Expenses expenses) {
    ...
}
```

Reuse:

Comment defines API contract that specifies how this function can be reused

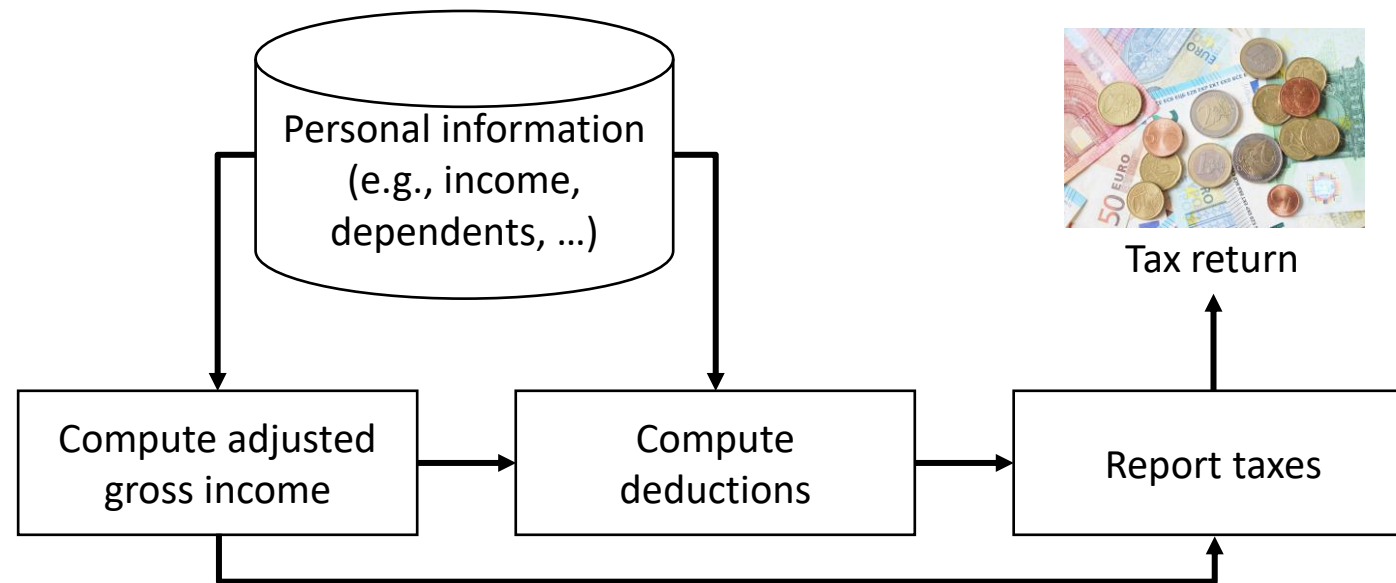
Abstraction:

No need to know how the deductions are computed to use this in a larger system



# Divide and conquer

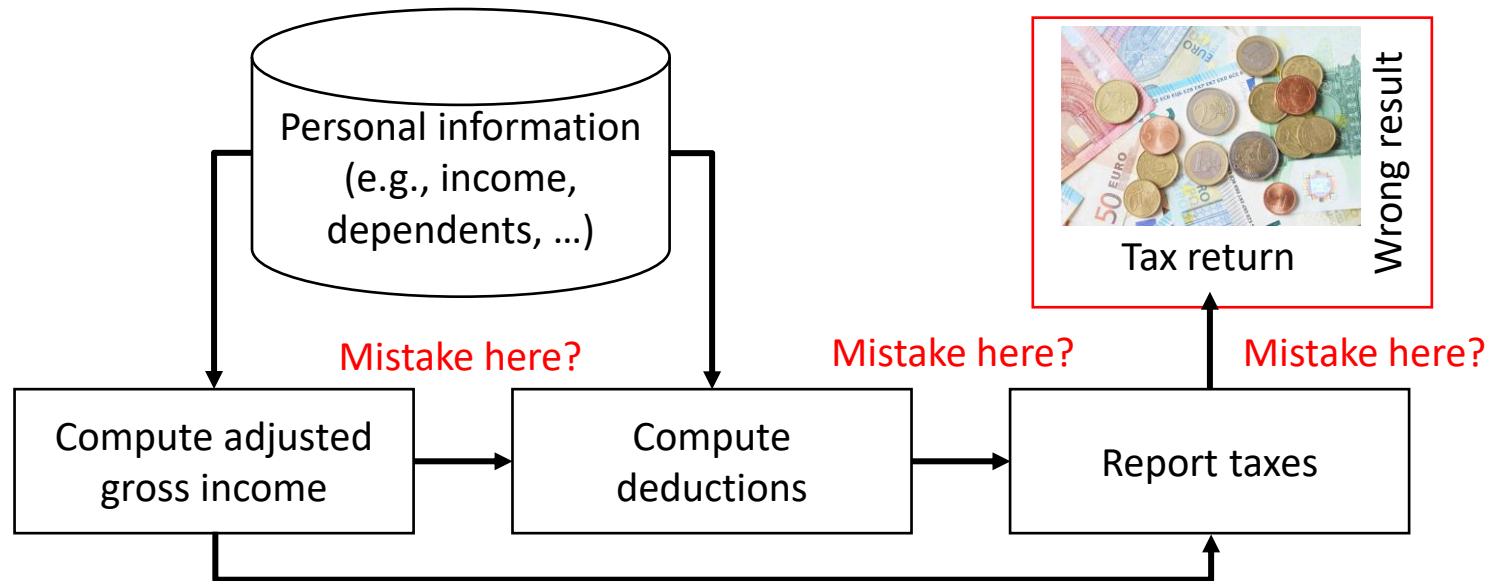
- Human cognitive ability limited
- Decomposition is necessary to handle complexity
- Use deductive reasoning to understand how groups of smaller units of complexity work together as a system



- Additional advantage:
  - Allows division of labor

# Debugging and assigning blame

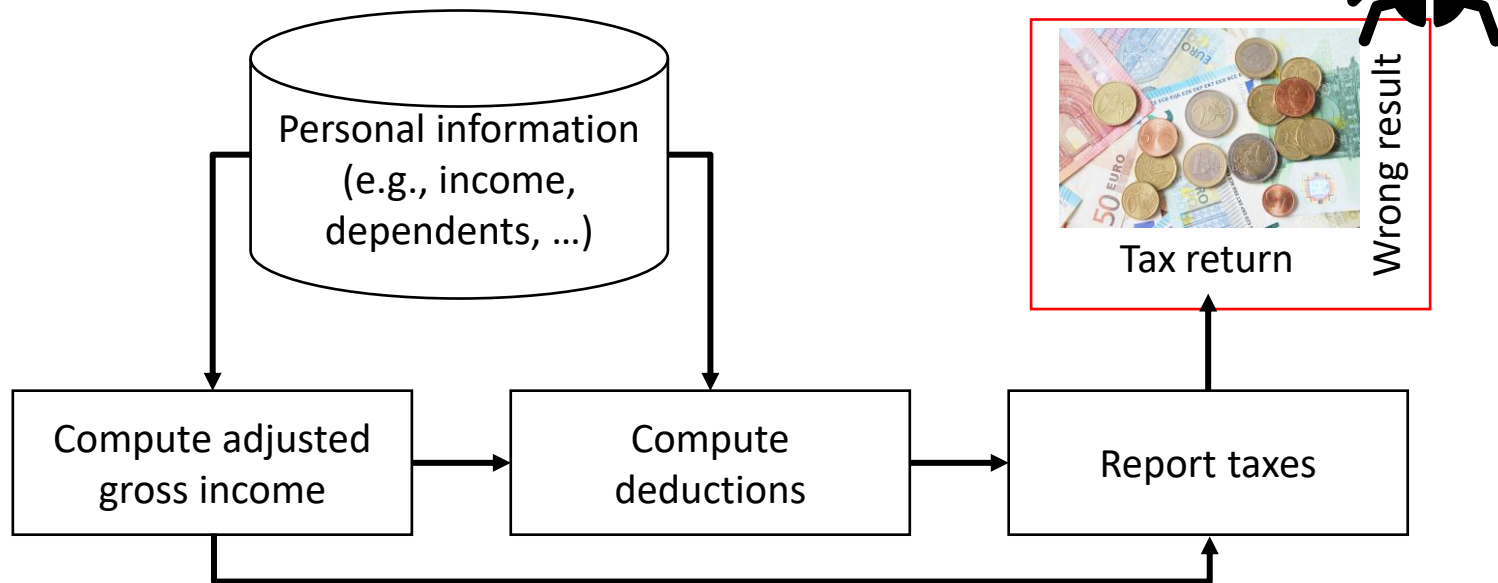
- Each component has a specification
  - Defines for each input the expected output
- Important for debugging!



# Strict correctness assumption

- Specification determines which outputs are correct/wrong
- No concept of “pretty good:”
  - Even 99,9% accuracy not acceptable

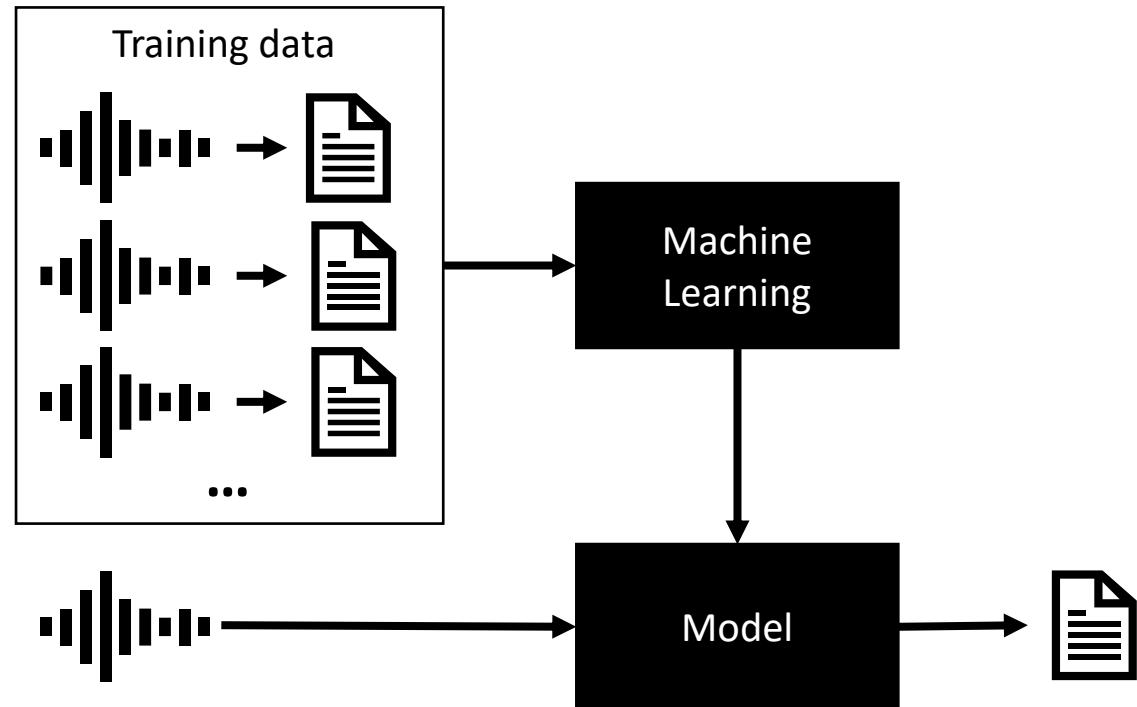
→ A single wrong result indicates a bug in the system



# Lack of specifications

```
/**
 * Returns the text spoken within the audio file
 * @param audioFile handle of a file with a supported audio format (mp3, wav, ogg)
 * @return String with the extracted text
 */
public String transcript(File audioFile) {
    ...
}
```

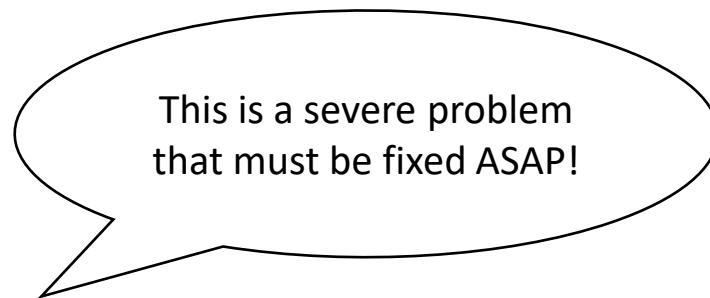
- We cannot specify this!
- Instead:
  - Inductive reasoning, i.e., machine learning



→ No (automated) rules that specify the correct output, other than our learned model!

# Weak correctness assumption

- Often no reliable ground truth
  - Usually human judgment
  - Expensive
- Accepting that mistakes will happen, but hopefully not too frequently
  - “95% accuracy” may be pretty good
- More confident for data similar to training data

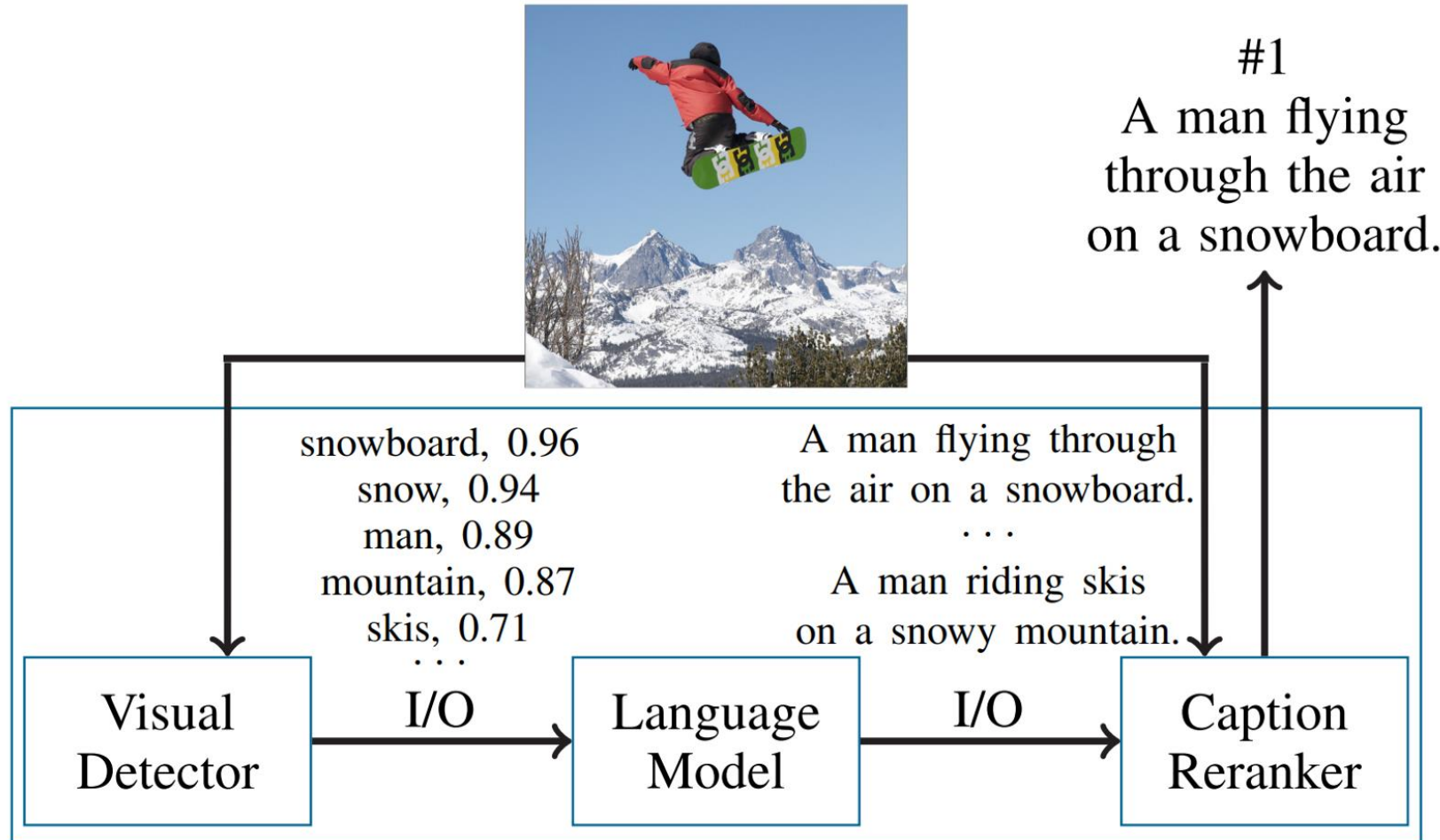


# Further consequences

- In general, no specifications for learning tasks
  - Too complex
  - Rules unknown
- ML will learn rules/specifications (inductive reasoning)
  - Unclear if the rules are correct
  - Rules often opaque for humans (e.g., deep neural networks)
- We can define correctness for some data, but not general rules
- Sometimes we can only determine correctness after the fact
  - E.g., multiple translations may be correct, we cannot specify all of them

**Goals instead of specifications**  
**Objective to maximize, instead of rules to fulfill**

# Example: Decomposition for image capturing



# Example: Blaming assignment?!



| Visual Detector    |      |
|--------------------|------|
| 1. teddy           | 0.92 |
| 2. on              | 0.92 |
| 3. cake            | 0.90 |
| 4. bear            | 0.87 |
| 5. stuffed         | 0.85 |
| ...                |      |
| 15. <b>blender</b> | 0.57 |

## Language Model

1. A teddy bear.

2. A stuffed bear.

...

108. A **blender** sitting on top of a cake.

## Caption Reranker

1. A **blender** sitting on top of a cake.

2. A teddy bear **in front of** a birthday cake.

3. A cake sitting on top of a **blender**.



# Fixing mistakes may introduce mistakes



## Visual Detector

|          |      |
|----------|------|
| teddy    | 0.92 |
| computer | 0.91 |
| bear     | 0.90 |
| wearing  | 0.87 |
| keyboard | 0.84 |
| glasses  | 0.63 |

1. A teddy bear  
sitting on top  
of a computer.

## Fixed Visual Detector

|          |     |
|----------|-----|
| teddy    | 1.0 |
| bear     | 1.0 |
| wearing  | 1.0 |
| keyboard | 1.0 |
| glasses  | 1.0 |

1. a person wearing  
glasses and holding  
a teddy bear sitting  
on top of a keyboard.

Questions?

