AI in the Open World: Discovering Blind Spots of AI

Ece Kamar

eckamar@microsoft.com
Recent AI Successes

Object recognition
Human parity
2016

Speech recognition
Human parity
2017

Machine reading
comprehension
Human parity
Jan 2018

Machine translation
Human parity
March 2018
What Happens in the Real World

**AI in the court: When algorithms rule on jail time**

By MATT O'BRIEN AND OAKIE KANG, ASSOCIATED PRESS

CLEVELAND — Jan 31, 2018, 5:04 PM ET

**Machine Bias**

There's software used across the country to predict future criminals. And it’s biased against blacks.

By Joel Christians, Jeff Carter, Asia Matti and Lauren Heiler, theStack

May 19, 2018

**A.I. Shows Promise Assisting Physicians**

Doctors competed against AI computers to recognize dementia on magnetic resonance images of a human brain during a competition in May. Last year, the human doctors lost.

**Self-driving car timeline for 11 top automakers**

**Tesla’s Autopilot keeps crashing into parked cars. Here’s why.**

IBM’s Watson gave unsafe recommendations for treating cancer

Doctors ran 11 hypothetical scenarios, not real patient data

By Angela Chen | TheStack | Jan 25, 2018, 3:04 PM ET
Principles of Responsible AI

- Fairness
- Reliability & Safety
- Privacy & Security
- Inclusiveness

- Transparency
- Accountability
Aether Committee

Microsoft

AI, Ethics and Effects in Engineering & Research

BIAS AND FAIRNESS
INTELLIGIBILITY & TRANSPARENCY
RELIABILITY AND SAFETY
HUMAN-AI INTERACTION & COLLABORATION
Insight #1: Responsible AI emerges from understanding **AI limitations** in the **open-world**.
Many Reasons for Failures

- Incomplete/incorrect models
- Misspecified goals and objectives
- Attacks against AI systems
- Human error

Eykholt, Evtimov, et al. 2018
Insight #2:
There is no silver bullet, but variety of emerging techniques
• Robust optimization and training
• Outlier detection approaches
• Incorporating causal learning
• Model expansion
• Risk-sensitive optimization …
In the lab

training data → development → evaluation → accuracy

test data

In the real-world

training data → development → evaluation → Complex success criteria

real-world
Insight #3:
Aligning efforts with AI development cycle is key.
Software Engineering for Machine Learning: A Case Study

Sarena Anserhi
Microsoft Research
Redmond, WA, USA
sarena@microsoft.com

Andrew Begel
Microsoft Research
Redmond, WA, USA
awbegel@microsoft.com

Christian Bird
Microsoft Research
Redmond, WA, USA
christian@microsoft.com

Robert DeLine
Microsoft Research
Redmond, WA, USA
rde@microsoft.com

Harold Gall
University of Zurich
Zurich, Switzerland
gall01@ah.ch

Eesha Kumar
Microsoft Research
Redmond, WA, USA
eesha@microsoft.com

Nachappan Nagarajan
Microsoft Research
Redmond, WA, USA
nachappan@microsoft.com

Boumaza Nourou
Microsoft Research
Redmond, WA, USA
boumaza.nourou@microsoft.com

Thomas Zimmermann
Microsoft Research
Redmond, WA, USA
thomas.zimmermann@microsoft.com

Abstract—Recent advances in machine learning have stimu-
lated widespread interest within the Information Technology
sector on integrating AI capabilities into software and services.
This trend has forced organizations to evolve their development
processes. We report on a study that we conducted on-of-the-shelf
software tools at Microsoft as they develop AI-based applica-
tions. We explore their experiences in developing, deploying,
and running the tools on end-to-end projects. We describe
prior experience developing AI applications (e.g., search and
NLP) and data science tools (e.g., application diagnostics and bug
e Reporting). We find that various Microsoft teams have added
their own tools into production, well-valued, higher-order
software engineering processes, providing insights about several
essential challenges that organizations may face in creating
AI-driven software tooling. We identify three best practices from
Microsoft teams to address these challenges.

In addition, we have identified three aspects of the AI domain
that make it fundamentally different from prior software application
domains: (1) discovering, managing, and revising the data
collections and infrastructure is much more rapid and
difficult than other types of software engineering. It is difficult
to quantify, monetize, or even accept the value of AI
software tools and to accurately measure their effectiveness.
We identify (2) potential for improved software development
efficiency. We find that the AI domain presents an opportunity
to improve software efficiency by increasing the productivity
to other organizations.

Index Terms—AI, Software engineering, process, data

I. INTRODUCTION

Personal computing. The Internet. The Web. Mobile com-
ing. Cloud computing. Now a thread goes by without a
disruptive shift in the dominant application domain of the
software industry. Each shift brings with it new software
engineering goals that spur software organizations to evolve
t heir development practices in order to address the novel
aspects of the domain.

The latest trend to the software industry is around
integrating artificial intelligence (AI) capabilities based on
advances in machine learning. AI broadly includes technol-
gies for reasoning, problem solving, planning, and learning,
among others. Machine learning refers to statistical modeling
techniques that have powered recent excitement in the software
and services marketplace. Microsoft product teams have used
machine learning to create application suites such as Bing
Search for the Consumer virtual assistant, as well as platforms
such as Microsoft Translator for real-time translation of text,
voice, and video. Cognitive Services for text, speech,
language understanding for building interactive conversational
agents for Cortana and Q&A products. Microsoft’s teams build
their own machine learning applications [1]. To ensure these
products, Microsoft has formed in-house tooling capabilities
in AI and developed new areas of expertise across the
company.

In this paper, we describe a study in which we learned how
various Microsoft software teams build software applications
with cognitively-focused AI features. For that, Microsoft has
integrated existing Agile software engineering processes with
AI-specific workflows informed by prior experiences in devel-
oping early AI and data science applications. In our study, we
asked Microsoft employees about how they worked through the
growing challenges of full software development lifecycle
for AI as well as the larger, more essential issues inherent in
the development of large-scale AI infrastructure and applications.

With teams across the company having different amounts of
work experience in AI, we observed that many issues reported
by newer teams dramatically drop in importance as the teams
mature, while some issues remain critical to the practice of large-
scale AI. We have made a first attempt to create a process
maturity metric to help teams identify how far they have come
on their journey to building AI applications.

As a key finding of our analysis, we discovered three funda-
mental differences in building applications and platforms for
training and deploying machine-learning models that we have
seen in prior application domains. First, machine learning is all
about data. The amount of effort and rigor it takes is decisive,
source, manage, and version data is inherently more complex
and different than doing the same with software code. Second,
building for customizability and extensibility of models require
teams to not only have software engineering skills but almost

A study with 565 ML engineers
Tools make ML too difficult
Data collection and cleaning is hard
Education and lack of expertise
Debugging is hard
Model evaluation and deployment
Compliance and ethics
Outline

- Blind spot detection for supervised learning  
  [with Himabindu Lakkaraju, Rich Caruana and Eric Horvitz, AAAI 2016]

- Blind spots of reinforcement learning  
  [with Ramya Ramakrishnan, Debadeepta Dey, Julie Shah and Eric Horvitz, AAMAS 2018; AAAI 2019]

- Backward-compatibility for AI systems  
  [with Gagan Bansal, Besmira Nushi, Dan Weld and Eric Horvitz, AAAI 2019]
Outline

- **Blind spot detection for supervised learning** [with Himabindu Lakkaraju, Rich Caruana and Eric Horvitz, AAAI 2016]

- **Blind spots of reinforcement learning** [with Ramya Ramakrishnan, Debadeepta Dey, Julie Shah and Eric Horvitz, AAMAS 2018; AAAI 2019]

- **Backward-compatibility for AI systems** [with Gagan Bansal, Besmira Nushi, Dan Weld and Eric Horvitz, AAAI 2019]
Blind Spots in Data

OPEN-WORLD DATA

TRAINING DATA

M

cat (conf = 0.96)

unknown unknown
Focus: Blind-spot Detection

- **Unknown unknowns:** Data points with confident but incorrect predictions.
- **Blind-spots:** Feature spaces with high concentration of unknown unknowns
- **Cause:** Mismatch between training and execution data
- Common active learning techniques do not help!
Problem Definition

Problem statement: Find $x_t$ s.t. $o(x_t)$ is maximized.

Two Main Challenges:
• Searching for errors
• Interpretability of error regions
Our Framework

**Input:** Execution data points with high confidence

**Step 1:** Descriptive Space Partitioning

- **White Dogs**
- **White Cats**
- **Brown Cats**
- **Brown Dogs**

**Step 2:** Multi-armed bandits for unknown unknowns
Comparison with Alternative Methods
(more in the paper)
Outline

- Blind spot detection for supervised learning
  [with Himabindu Lakkaraju, Rich Caruana and Eric Horvitz, AAAI 2016]

- Blind spots of reinforcement learning
  [with Ramya Ramakrishnan, Debadeepta Dey, Julie Shah and Eric Horvitz, AAMAS 2018; AAAI 2019]

- Backward-compatibility for AI systems
  [with Gagan Bansal, Besmira Nushi, Dan Weld and Eric Horvitz, AAAI 2019]
Simulation
Real world
Simulation $\rightarrow$ Real world

Blind spots in RL: Mismatch between simulation and real world can cause costly mistakes

Approach: Use human (oracle) guidance to transfer from sim to real in a safe way
Type of blind spots: Limited State Representation

$S_{sim}$
What agent sees

$S_{real}$
What oracle sees
Type of blind spots: Limited State Representation

- $S_{sim}$: What agent sees
- $S_{real}$: What oracle sees

Blind spot

Agent action unacceptable

Agent action acceptable
Learning blind spots

One Approach
Active learning and testing in the world
- Dangerous

Alternative Approach
Use oracle to safely obtain data in the real world

Demonstrations
- Easy to obtain
- Only provides oracle’s action, not input on agent’s action

Corrections
- Harder to obtain
+ Gives direct input on agent’s policy
Problem

$\pi_{sim}$

$O \{ A(s, a), \pi_{real} \}$

Our approach

$M = \text{Blind spot model}$
Our Approach

Data collection

\[
\begin{array}{c}
\pi_{sim} \\
O \\
B \\
\end{array}
\]
Noise and Bias in Data

State representation (SR) noise

Many states look identical

Action mismatch (AM) noise

Observing oracle only shows behavior deviation

Oracle

Agent

Deviation ok or not?

Bias in data

Trajectory-based feedback adds bias to the data
Our Approach

Data collection

Noisy aggregation

Model learning

\[ \pi_{sim} \rightarrow O \rightarrow B \rightarrow M \]

<table>
<thead>
<tr>
<th>( s^1_{sim} )</th>
<th>( s^2_{sim} )</th>
<th>( s^3_{sim} )</th>
<th>( \ldots )</th>
<th>( s^l_{sim} )</th>
<th>( s^r_{sim} )</th>
<th>( s^n_{sim} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \sim \sim \sim \sim )</td>
<td>( \times \times )</td>
<td>( \times )</td>
<td>( \sim \sim \sim \sim )</td>
<td>( \sim \sim )</td>
<td>( \times )</td>
<td>( \sim \sim )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( s^1_{sim} )</th>
<th>( s^2_{sim} )</th>
<th>( s^3_{sim} )</th>
<th>( \ldots )</th>
<th>( s^l_{sim} )</th>
<th>( s^{l+1}_{sim} )</th>
<th>( s^n_{sim} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \times )</td>
<td>( \times )</td>
<td>( \sim )</td>
<td>( 0.6 )</td>
<td>( 0.3 )</td>
<td>( 0.96 )</td>
<td>( 0.4 )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( s^1_{sim} )</th>
<th>( s^2_{sim} )</th>
<th>( s^3_{sim} )</th>
<th>( \ldots )</th>
<th>( s^l_{sim} )</th>
<th>( s^{l+1}_{sim} )</th>
<th>( s^n_{sim} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 0.55 )</td>
<td>( 0.4 )</td>
<td>( 0.7 )</td>
<td>( 0.98 )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Evaluation

**Catcher**

- **Sim**
- **Real**

**FlappyBird**

- **Sim**
- **Real**
Example Result – more on paper

Our blind spot model can be effectively used in the real world to get 80% of the reward with less than 30% queries.
Mini autonomous car demo
Human Blind Spots
Relaxing Assumptions: Towards Joint Execution

Simulation data

Disagreement analysis

Behavior cloning

Demonstration data

Real world

[with Ramya Ramakrishnan, Debadeepta Dey, Julie Shah and Eric Horvitz, AAAI 2019]
A study with 565 ML engineers

End-to-end tool fragmentation

Tools make ML too difficult

Data collection and cleaning is hard

Education and lack of expertise

Debugging is hard

Model evaluation and deployment

Compliance and ethics
Pandora: Error Analysis for ML systems

[with Besmira Nushi and Eric Horvitz, HCOMP 2018]
Current Practices of Error Detection

Benchmark
ML Model

Aggregate metrics
hide important failures

73.8%
Key: Empowering AI developers

Different regions fail for different reasons

100% 74% 67%
75% 59%
66% 42%

[VISION]
“Build tools that help engineers **accelerate** the development iterations by identifying errors **faster**, **systematically**, and **rigorously**.”
Tool for Error Analysis

Benchmark Workload -> Feature Augmentation -> Interpretable Error Prediction Models

ML Model -> Error labels

Global Views
Cluster Views
Instance Views

Interpretable Error Prediction Models
Outline

- Blind spot detection for supervised learning [with Himabindu Lakkaraju, Rich Caruana and Eric Horvitz, AAAI 2016]

- Blind spots of reinforcement learning [with Ramya Ramakrishnan, Debadeepta Dey, Julie Shah and Eric Horvitz, AAMAS 2018; AAAI 2019]

- Backward-compatibility for AI systems [with Gagan Bansal, Besmira Nushi, Dan Weld and Eric Horvitz, AAAI 2019]
Putting models into a system perspective

**Sociotechnical System:** Human-AI collaboration

**Software System:** component-component collaboration
Tesla can change so much with over-the-air updates that it’s messing with some owners’ heads

Praise for a recent software fix to the Model 3’s braking is met with worry that different update slowed some customers’ cars

By Sean O’Kane | @sokane1 | Jun 2, 2018, 1:00pm EDT

This week was different, though, because it showed just how far the company can go with those updates. With a swift change in the software, the company showed it can reach as deep as the systems that control the brakes. It creates the feeling that you could get out of your car one night, and by the time you get back in the next morning, the car could do some things — maybe everything — in a totally different way.

Rinesi says it’s also hard to define “software” in the first place since much of what modern technology does relies on things that live outside the physical object — in this case, the car. “You don’t buy a car, or a phone, or soon enough a house or a medical implant or whatever: you buy an interface to, or an aspect of, a huge platform-company-ecosystem-whatever that changes by the minute,” he says.
What can go wrong?

Software System: component-component collaboration

ML Model A \( \xrightarrow{I/O} \) ML Model B \( \xrightarrow{I/O} \) ML Model C

- Preserving Robustness Checks
- Error Handling and Chain Effects
- Maintenance

[Sculley et al. NeurIPS 2015] [Nushi et al. AAAI 2017] [Andrist et al. ICSR 2017] [Wang et al. SIGIR 2012]
Compatibility Score

\[ C(v_1, v_2) = \frac{\#(v_1=\text{Right} \cap v_2=\text{Right})}{\#(v_1=\text{Right})} \]

**Goal**: v2 should not introduce any new errors during retraining.
Compatibility is not in-built

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Dataset</th>
<th>Perf. v1</th>
<th>Perf. v2</th>
<th>Compatibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logistic Regression</td>
<td>Recidivism</td>
<td>0.68</td>
<td>0.72</td>
<td>72%</td>
</tr>
<tr>
<td></td>
<td>Credit Risk</td>
<td>0.72</td>
<td>0.77</td>
<td>66%</td>
</tr>
<tr>
<td></td>
<td>Mortality</td>
<td>0.68</td>
<td>0.77</td>
<td>40%</td>
</tr>
<tr>
<td>Multi-layered Perceptron</td>
<td>Recidivism</td>
<td>0.59</td>
<td>0.73</td>
<td>53%</td>
</tr>
<tr>
<td></td>
<td>Credit Risk</td>
<td>0.70</td>
<td>0.80</td>
<td>63%</td>
</tr>
<tr>
<td></td>
<td>Mortality</td>
<td>0.71</td>
<td>0.84</td>
<td>76%</td>
</tr>
</tbody>
</table>

High-stake decision-making

Low compatibility
Training Compatible Models

Reformulated loss function
\[ L_c = L + \lambda_c \cdot \mathcal{D}(v_1, v_2) \]
Dissonance

New-error dissonance
\[ \mathcal{D}(x, y, v_1, v_2) = 1 (v_1(x) = y) \cdot L(x, y, v_2) \]

Imitation dissonance
\[ \mathcal{D}(x, y, v_1, v_2) = L(x, v_1, v_2) \]

Strict imitation dissonance
\[ \mathcal{D}(x, y, v_1, v_2) = 1 (v_1(x) = y) \cdot L(x, v_1, v_2) \]
FUTURE DIRECTIONS
Where to invest development effort for the largest impact?

[Integrative ML Systems]

[with Besmira Nushi and Eric Horvitz, AAAI 2017]
Adversarial investigations of reliability and safety

- Data poisoning
- Backdoor attacks
- Model inversion
- Model stealing
- ML supply chain attacks
- Membership inference
Directions around AI, People and Society

- Blindspots
- Biases
- Gaps

Human intellect → Machine Intellect → Human intellect
Takeaways

- Reliability & safety risks carry societal implications.
- Understanding AI limitations is key.
- Need for algorithmic advances as well as tools, design guidelines and frameworks for empowering developers and users.
Questions

eckamar@microsoft.com