

21st Century Grand Challenges in Computing Education

Amy J. Ko, Ph.D. *she/her/hers*

W UNIVERSITY *of* WASHINGTON



This is *not* a scientific talk, but a **political** talk *about* science.



I want to talk about which **research questions** we choose to answer.

The computing education research (CER) community is awesome!



Our mission from 1970-present...

How do we teach **programming** *effectively, equitably, and at scale* to post-secondary students?



Our missions from 2000-present...

- How can we equitably and effectively teach CS to **every child** in primary and secondary education?
- How can we ensure **everyone** feels welcome in computing, not just white and Asian boys and men?



We're not done with these missions.

But, I believe **bigger problems**
deserve our scarce attention.

Automation, amplified by computing, may cause **global economic disruption**.



Climate change is causing mass migration, natural disasters, and political upheaval.



Disinformation is **amplifying** these trends,
destabilizing democracies.



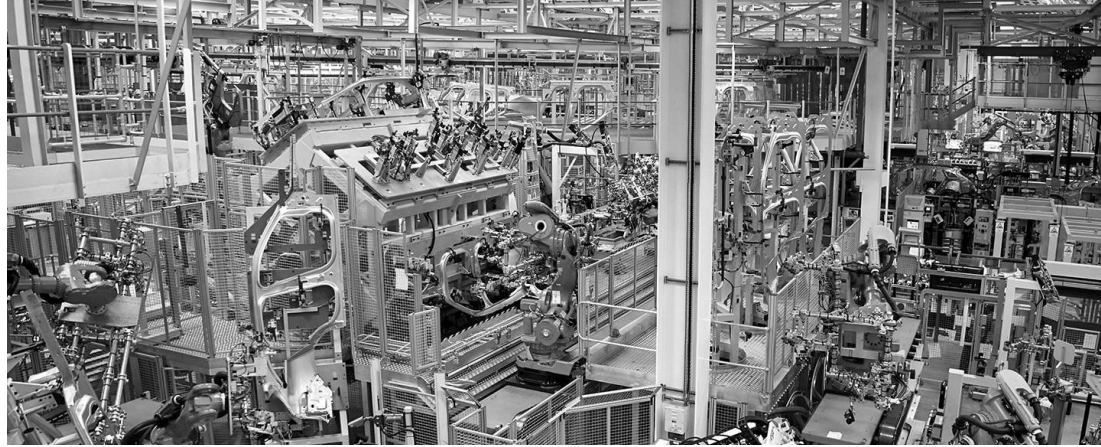
The consequences of these problems are falling upon our most **marginalized people.**



What do these have to do with **computing**?



As a CS major, I learned that...



CS values **efficiency** over **humanity**.
(As if automation is always good. It's not.)



*But I learned later about the **limits** of computing,
and how focusing on efficiency can harm humanity.*

As a CS major, I learned that...



CS values **power** over **sustainability**.
(As if computing resources are free. They're not.)



*I later learned that the choices I make in software have consequences, and I'm **responsible** for those consequences.*

As a CS major, I learned that...



CS values **neutrality** over **truth**.
(As if software doesn't have an opinion. It does.)



*I later learned that software isn't neutral because the **data** that moves through it isn't neutral, nor are the **algorithms** that process it.*

As a CS major, I learned that...

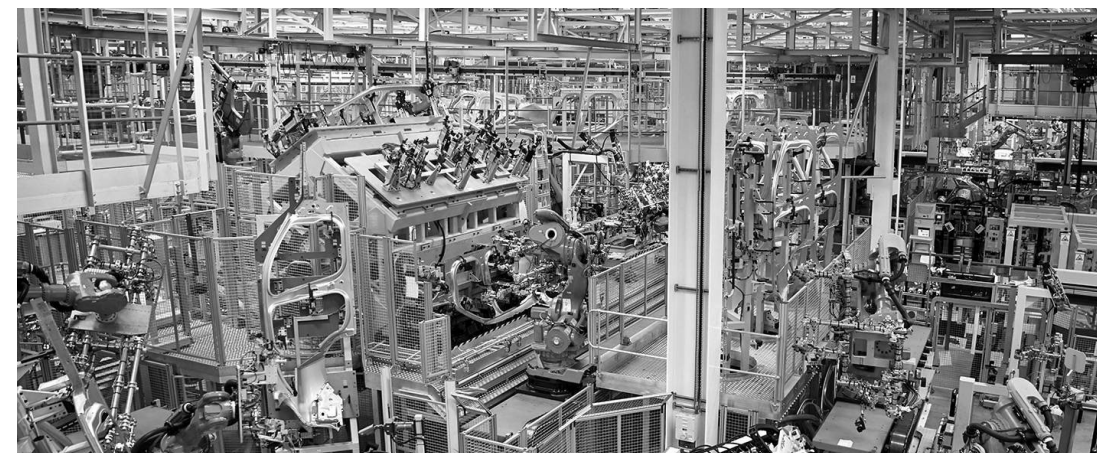
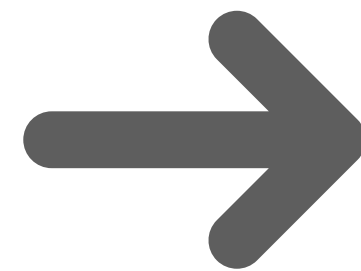


CS values **competition** over **justice**.
(As if society is a pure meritocracy. It's not.)



*I later learned that the world isn't a meritocracy, but one that often reinforces power and ignores **diversity**.*

The values in CS reinforce, amplify, and contribute to the world's 21st century grand challenges.



But we can **change** the values of CS...

- Prioritize **efficiency** → *Accept the **limits** of computing*
- Maximize **convenience** → *Accept **responsibility***
- Maintain **neutrality** → *Accept that **data** is non-neutral*
- Maintain **meritocracy** → *Design for **diversity***

My proposal

Our most urgent global problems demand that we integrate **four ideas** into CS curricula:

The limits of computing

Social responsibility

Data literacy

Designing for diversity

The rest of this talk...

- Why the limits of computing?
- Why social responsibility?
- Why data literacy?
- Why diversity?
- Call to action

The **limits** of computing



Computing *is* **powerful**.



Many people believe that the power of computing *always* makes things **better**.



Judges in the U.S. are automating sentencing because they believe that machine learning can't be wrong.



Faith in technology to **solve any problem** is used as a justification for inaction on climate change policy.

COP21/CMP11

Paris, France



Belief that **algorithms are neutral** has concentrated power in Facebook, Apple, Amazon, Netflix and Google.



Developers are spending their time making apps to "**solve**" homelessness, rather than addressing housing costs.



PROPERTY OWNERS

SERVICE PROVIDERS

NEED A HOME?

CONTACT OUR TEAM

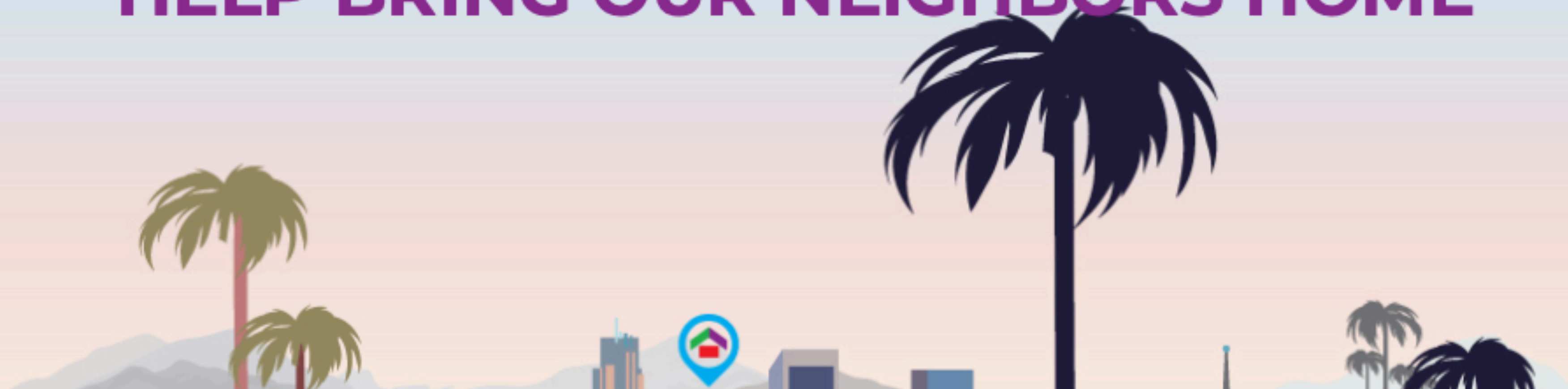


323.428.4742

JOIN US!

PARTNER WITH LEASEUP

HELP BRING OUR NEIGHBORS HOME



Everyone in the world needs to know that...

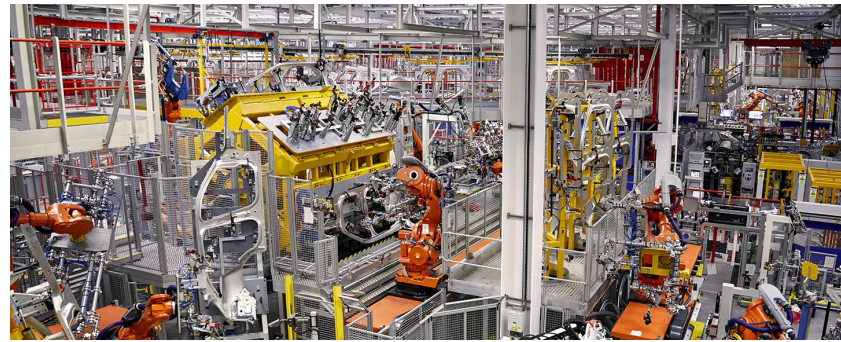
Software is often wrong. Garbage in, garbage out.

Software isn't neutral. It expresses the values encoded by its makers.

Software isn't free. People have to make it, energy has to power it.

Software can't solve every problem. No software will ever solve poverty.

If everyone knew these limits...



Judges and driverless car users might have less faith in algorithms to make autonomous decisions, preventing death.



The public might be more skeptical that technology alone will save us from climate change, promoting policy change.



Consumers might be more mindful about the price that they and others are paying for a "free" internet, catalyzing regulation.



Developers might spend less time making unhelpful apps and more time advocating for meaningful social change.

Teaching everyone to **code** only *strengthens* belief in the power of computing.

When people learn to program, they come to believe that:

- 1) Programming is **powerful**.
- 2) *Programmers* are **powerful wizards** who can do anything.

They learn nothing about what it can't do, only that “smart people” can make computers do anything.

Research opportunities

- How do we teach the limits of computing in a way that transfers to **everyday life**?
- How do we integrate these ideas into primary, secondary, and post-secondary education?
- How do we **prepare teachers** at all levels to teach these the limits of computing in transferable ways?

Social responsibility



Most CS education is agnostic to **what** students choose to make.

- We focus on **programming**
- We focus on whatever students' **interests** happen to be (*games, apps, jobs, money*)

Of course, what students choose to make **matters**.



When students become developers, what ethical stands against automation should they fight for?

We are Google employees. Google must drop Dragonfly.



Google Employees Against Dragonfly

[Follow](#)

Nov 27, 2018 · 15 min read

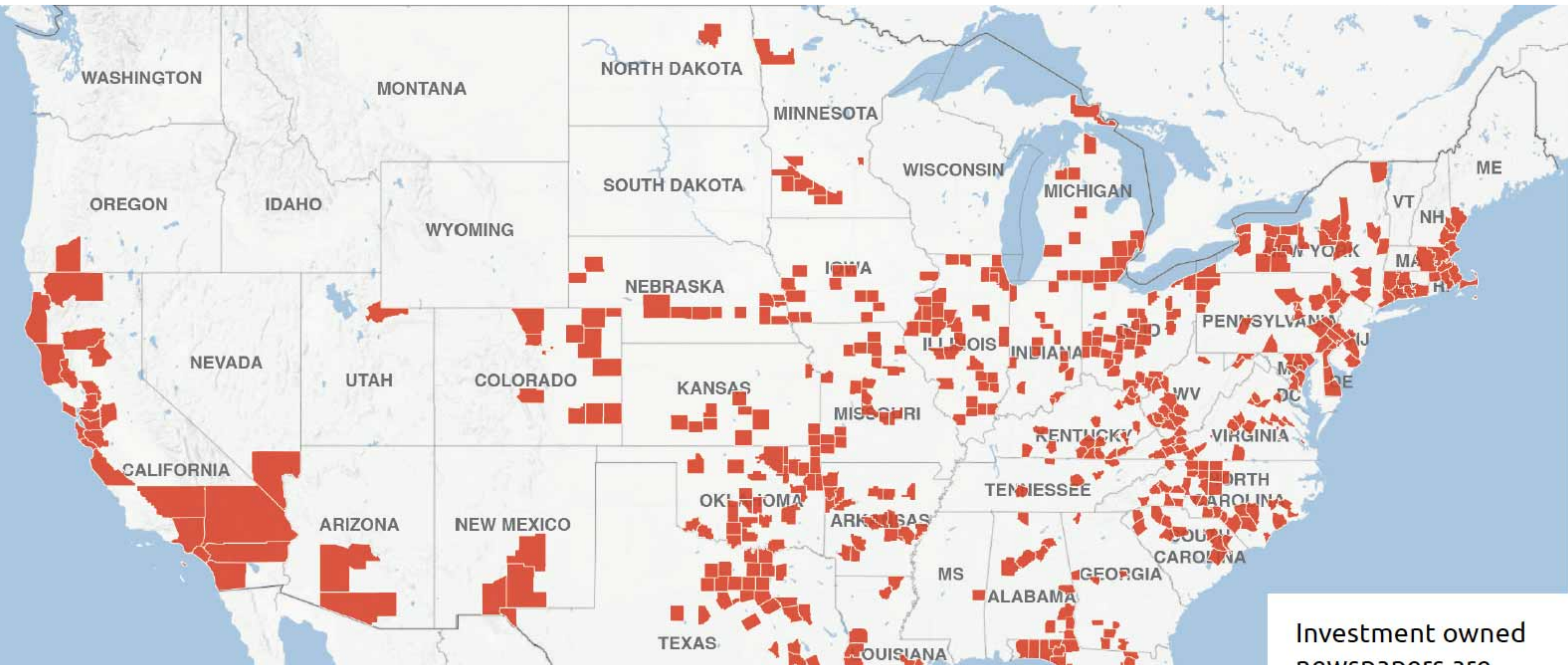
We are Google employees and we join Amnesty International in calling on Google to cancel project Dragonfly, Google's effort to create a censored search engine for the Chinese market that enables state surveillance.

We are among thousands of employees who have raised our voices for

Should students' first jobs be e-commerce at Amazon or sustainable energy?



Should they innovate in social media or in sustainable business models for **ethical journalism**?



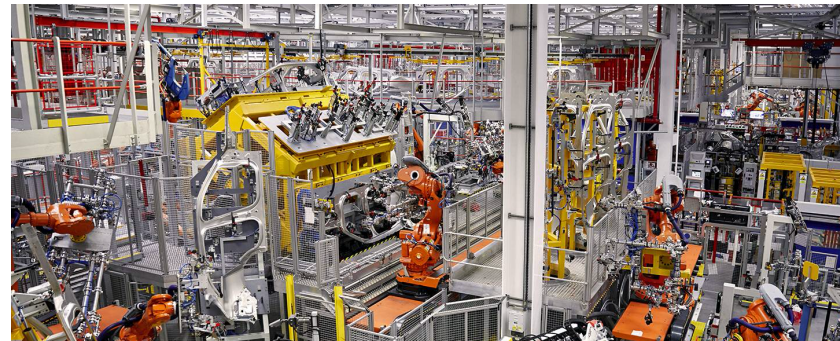
Why students ignore big problems

- Wealthy companies are better at recruiting, **but we do little to show them the alternatives**
- Students want to work on new technology, **but we rarely talk about applications outside of consumer technology**
- Students value wealth and status, **but we rarely attempt to change their values**

Every developer needs to know that...

- The **problems they choose to work on** will shape the world.
- **Where they choose to work** will determine problems they work on.
- To make these choices, we need to help them develop their **values and ethics**, and their entrepreneurial skills

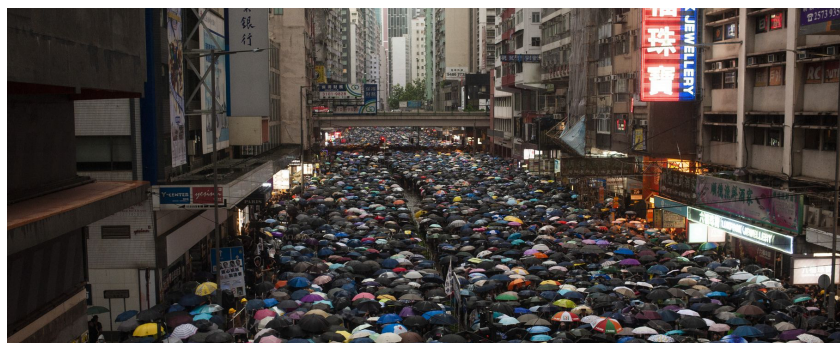
If developers had different values...



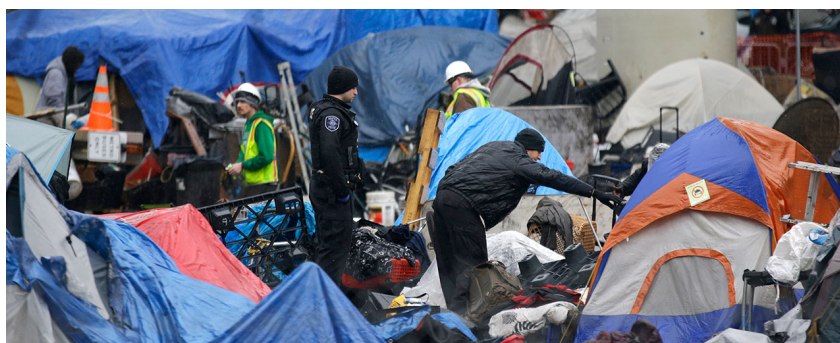
We might have products and services that **augment** human ability rather than try to replace it.



We might have more products and services **addressing climate change.**



We might have a more robust **free press and stable democracy.**



We might have **more effective public and private services** for people with the greatest needs in society.

Educating great programmers will *not* teach them social responsibility.

When people learn to program, they come to believe that:

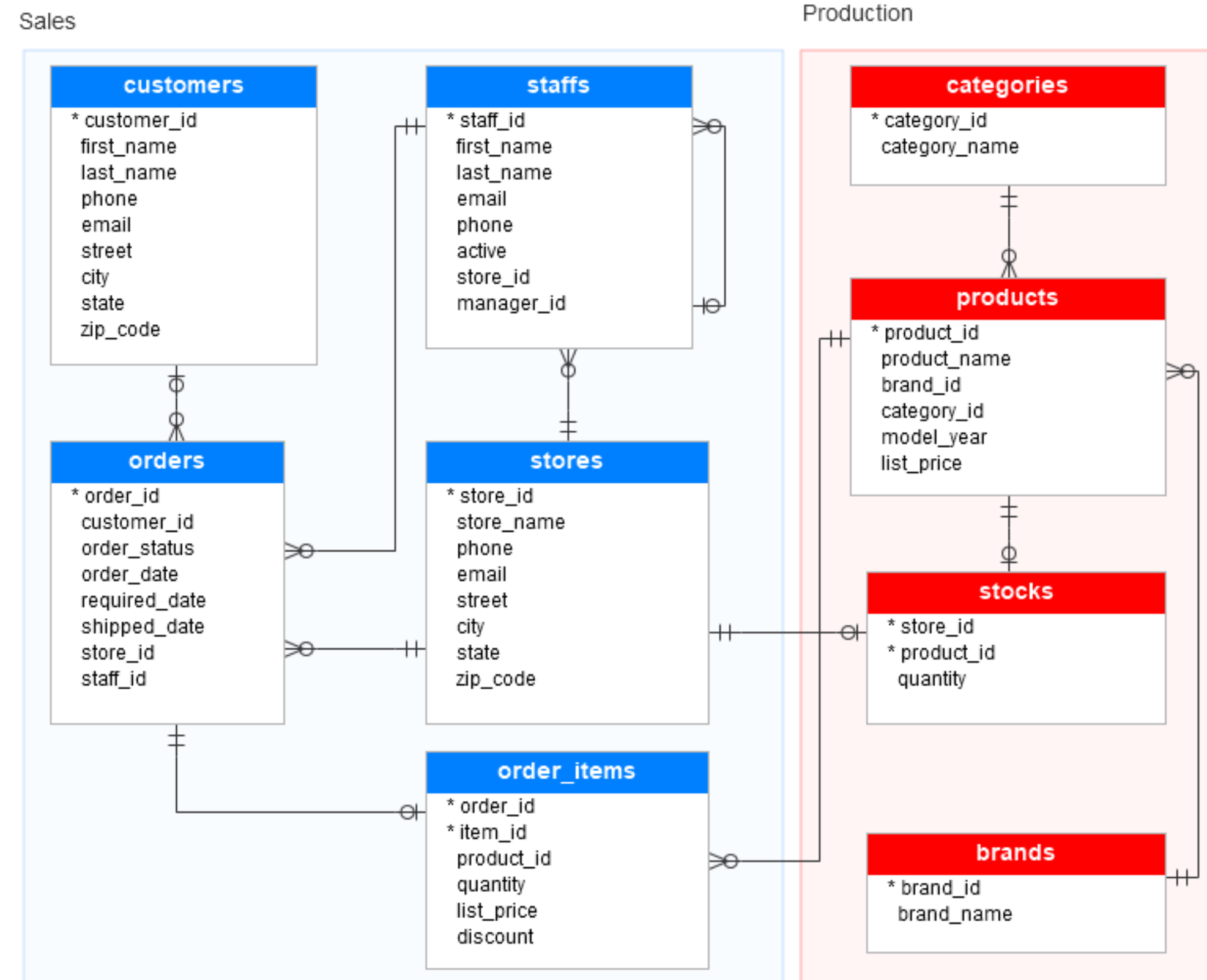
- 1) Programming is about **puzzle solving, not social change**
- 2) Developers' responsibility is purely **technical, not social**

They learn nothing about the role that software developers play in shaping our products, services, governments, and communities.

Research opportunities

- How do we convince students that they are **responsible** for their technical and career choices?
- How do we prepare students to make socially responsible career decisions **across their lifespan**?
- How do we empower students to start socially responsible **software companies**?

Data literacy



Google with and without data

The image displays two side-by-side screenshots of a Google search for "huskies".

Left Screenshot (with data):

- Search bar: "huskies"
- Navigation: All, Images, Shopping, News, Videos, More, Settings, Tools
- Results: About 116,000,000 results (0.61 seconds)
- Featured Snippet: **Washington Huskies | University of Washington Athletics**
gohuskies.com/ ▾
The Official Athletics Site for the University of Washington. Watch game highlights of Washington Huskies games online, get tickets to Huskies athletic events, ...
Schedule · Staff Directory · Football · Washington Huskies CWS ...
- Top stories:
 - For Huskies baseball, this short stay just might just lead to a long future in Omaha**
The Seattle Times
15 hours ago
 - Huskies get commitment from Idaho offensive lineman Gaard Mommelaar**
The Seattle Times
15 hours ago
 - Woof: Huskies Land 2020 Offensive Lineman From Idaho**
UW Dawg Pound
16 hours ago
- Map: Shows the location of Husky Stadium and Husky Ballpark in Portage Bay, Seattle.

Right Screenshot (without data):








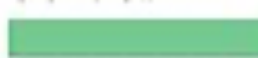
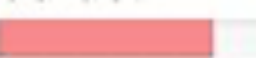
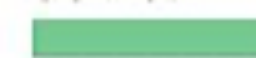
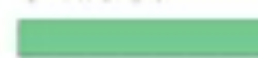


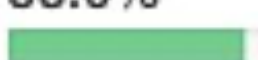
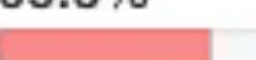

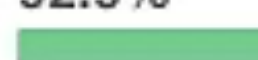

- Search bar: "huskies"
- Navigation: All, Images, Shopping, News, Videos, More, Settings, Tools
- Results: Only the "Top stories" section is visible, identical to the left screenshot.

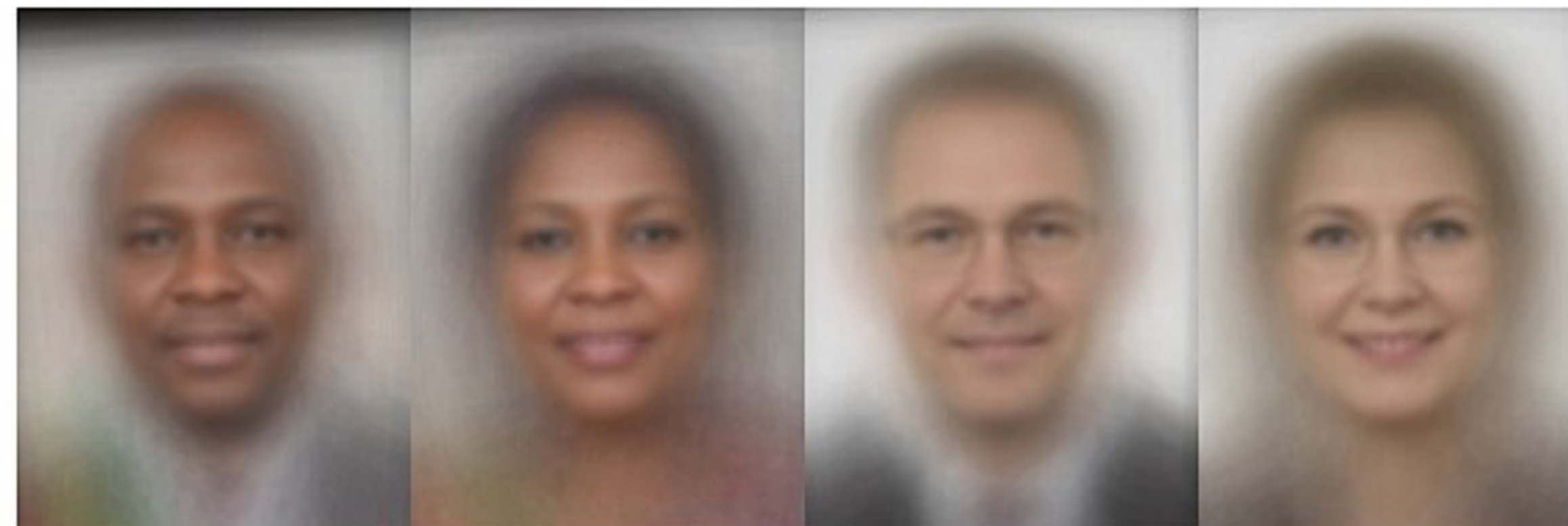
CS largely ignores the power of data

- When people think of Facebook, they think of apps, the Internet, and news feed algorithms, **not the content that billions of people give Facebook for free and make Facebook valuable.**
- When people learn about machine learning, they learn of algorithms, **not the data required to power it, and what it takes to produce, maintain, and repair that data.**

Of course, data is *central* to computing, and is often the source of the **harms** caused by computing

Facial recognition datasets have severe race and gender biases, but law enforcement uses it anyway.

Gender Classifier	Darker Male	Darker Female	Lighter Male	Lighter Female	Largest Gap
 Microsoft	94.0% 	79.2% 	100% 	98.3% 	20.8% 
 FACE++	99.3% 	65.5% 	99.2% 	94.0% 	33.8% 
 IBM	88.0% 	65.3% 	99.7% 	92.9% 	34.4% 



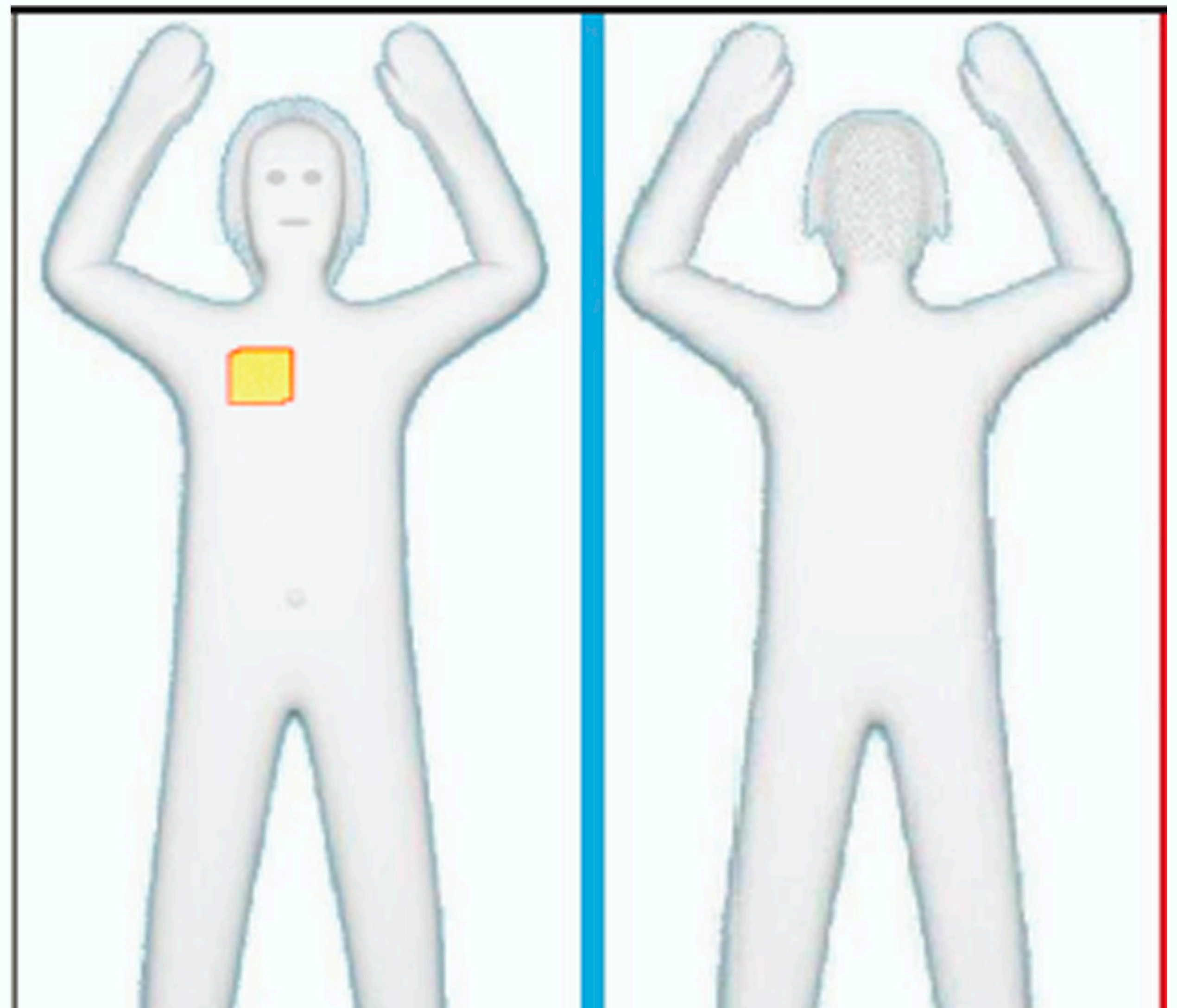
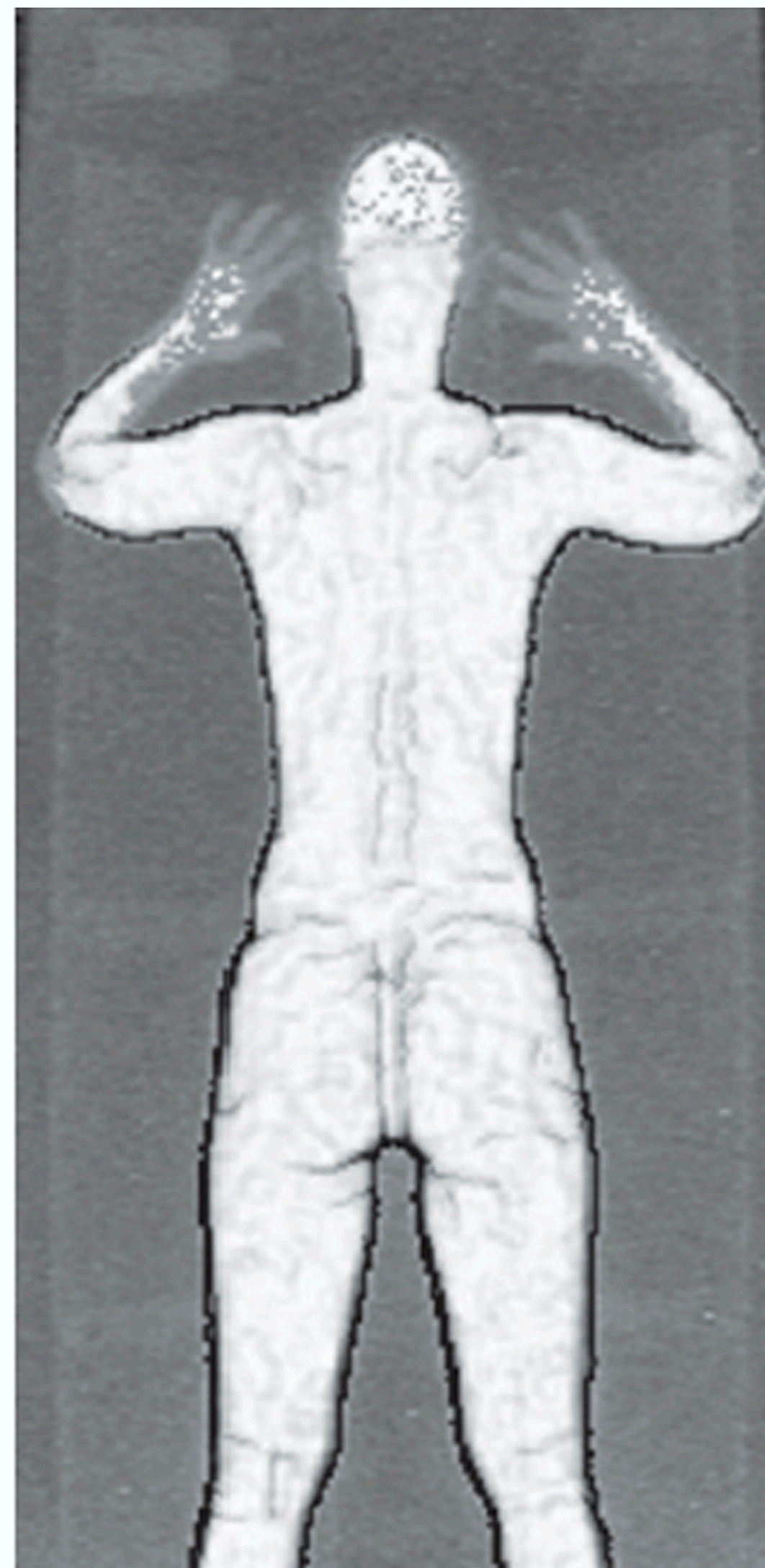
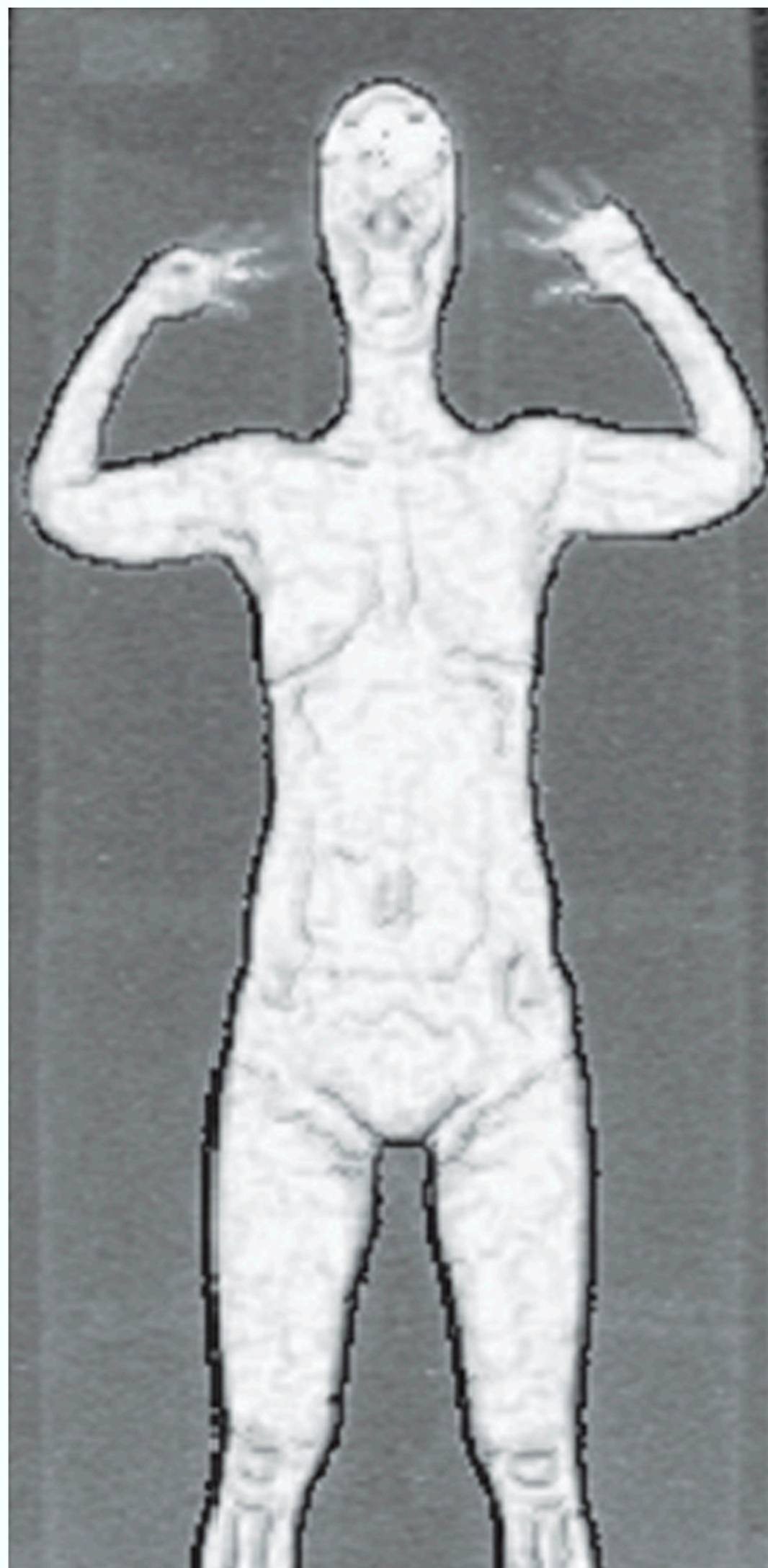
It's humanity's **hunger for data, not algorithms** that drives the massive power consumption in data centers.



Social media companies have quickly learned that disinformation can **kill, bully, violate, and destabilize.**



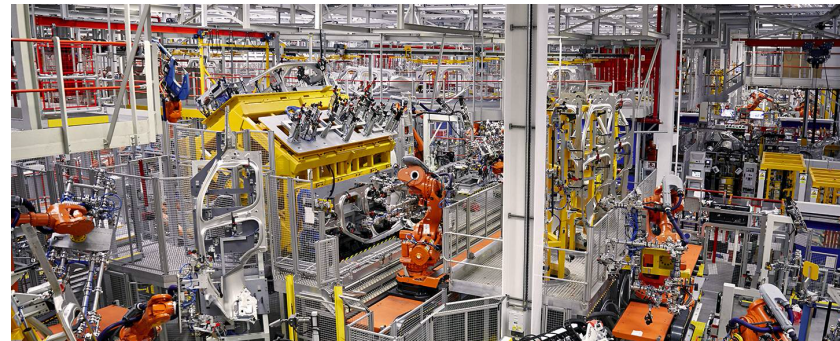
Metadata used in machine learned models reinforce invalid models of gender, leading to harassment



Everyone in the world needs to understand that...

- **Data is an imperfect record**, and can often be entirely wrong (e.g., my car title still says “Andrew”; do I still own my car?)
- **Data is always about the past**, not the future (e.g., CS is predominantly male, but won't always be)
- **Data is always biased**, encoding the values of the people who captured it (e.g., my medical record says I'm female; am I?)

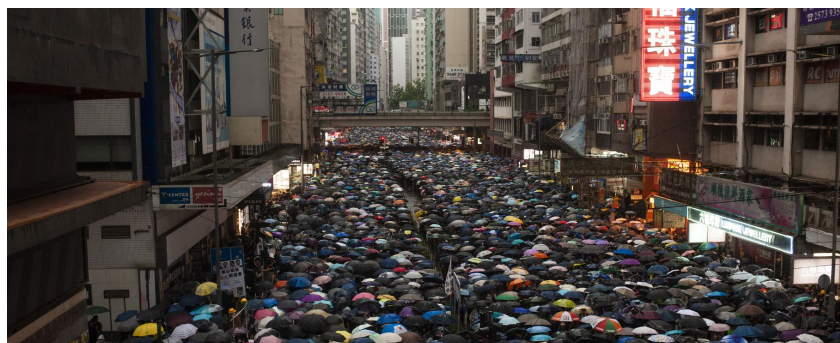
If everyone knew these facts...



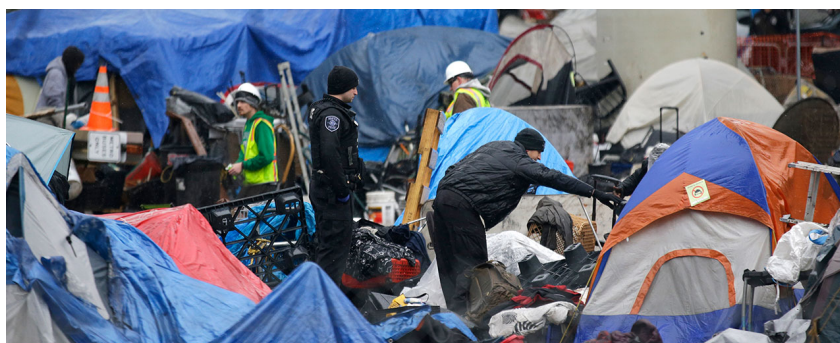
We might stop developers from using **biased, incorrect, unrepresentative** data to automate the world.



We might **reduce people's faith in machine learning** to solve every problem, nudging toward policy solutions.



We might grow demand for **better data privacy rights** (e.g., GDPR++)



We might prevent computing from **amplifying bias** in the world.

Teaching everyone to **code** will *not* teach data literacy.

When people learn to program, they come to believe that:

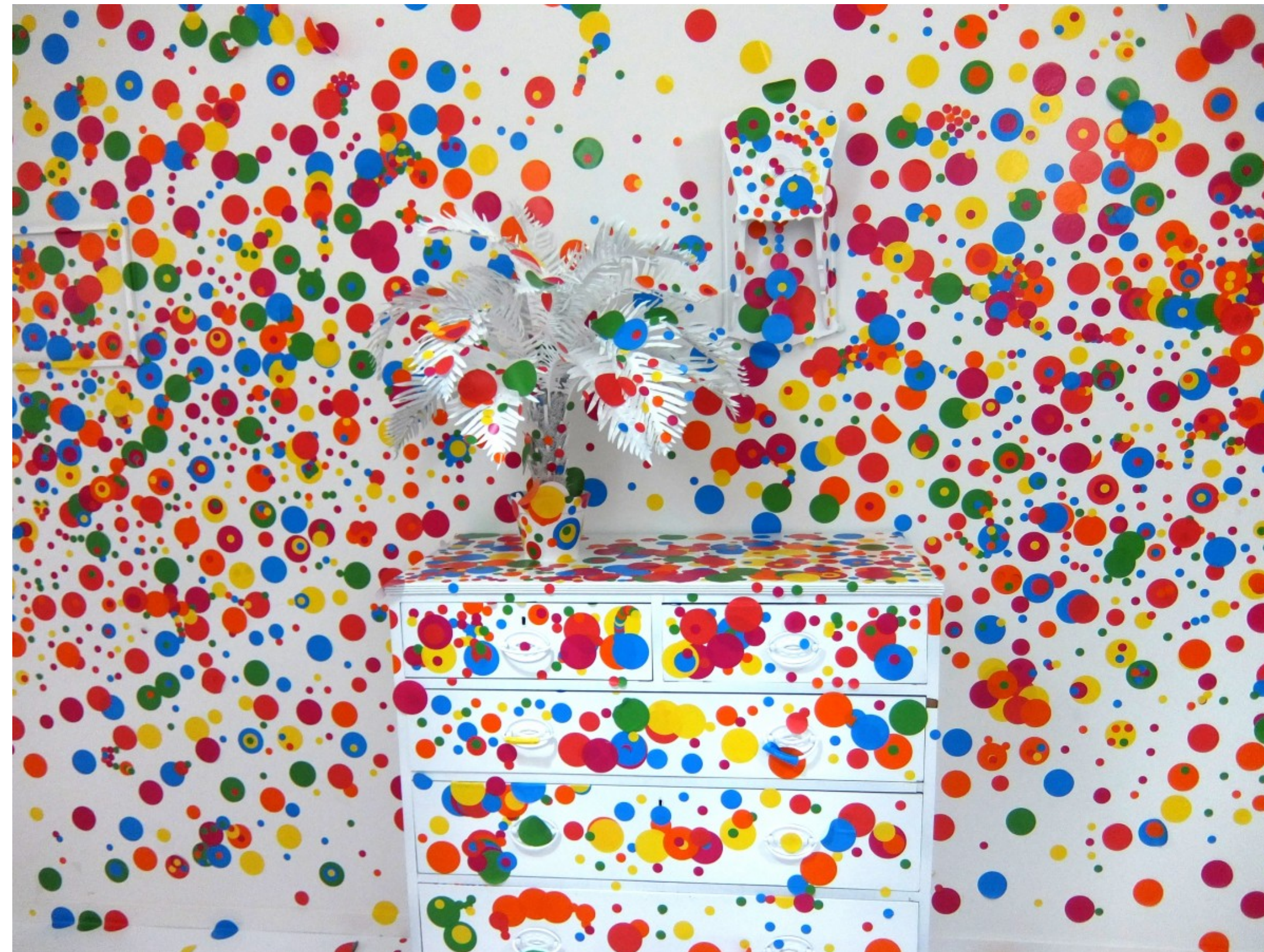
- 1) Data is abstract, free of values, free of meaning, and free in general.
- 2) Algorithms are the primary lever of computation, not data.

They learn nothing about what data is, how it is limited, and how algorithms can amplify the harm of data.

Research opportunities

- How do we teach the limits of data in a way that transfers to decisions in **everyday life** and **software engineering**?
- How do we integrate ideas about data into primary, secondary, and post-secondary education?
- How do we **prepare teachers** at all levels to teach these the limits of data in transferable ways?

Designing for diversity



Diversity

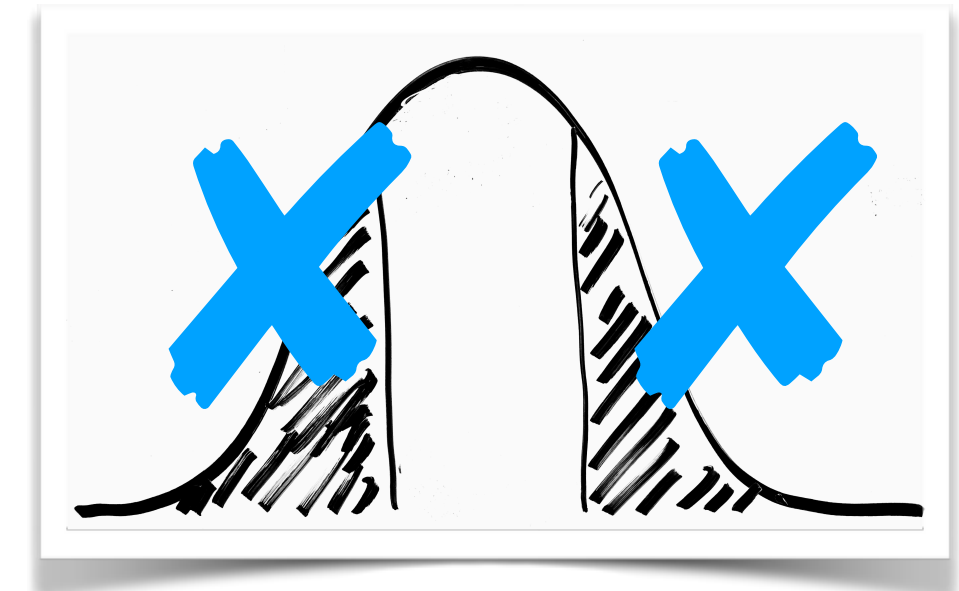
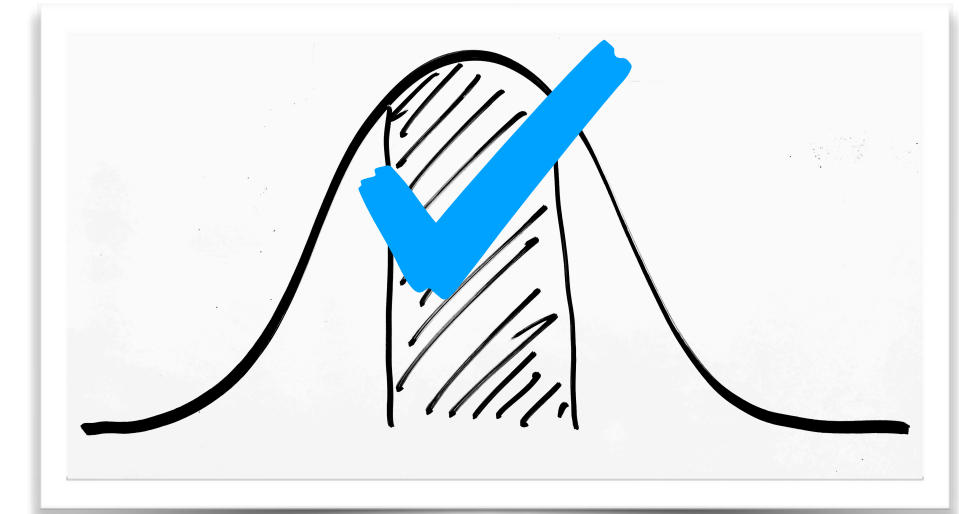
It is a fact of nature and society.

But it's a fact often ignored in the design of software.

It's also a fact at the foundation of the three areas of knowledge I've proposed.

Diversity limits the power of computing

- Automation can actually work *great* for “normal”, “typical”, “average” cases.
- But diversity *guarantees* that non-normality.
- *Whether deterministic algorithms or data-driven probabilistic reasoning, software struggles with diversity, and our students don't know this.*



Diversity increases social responsibility

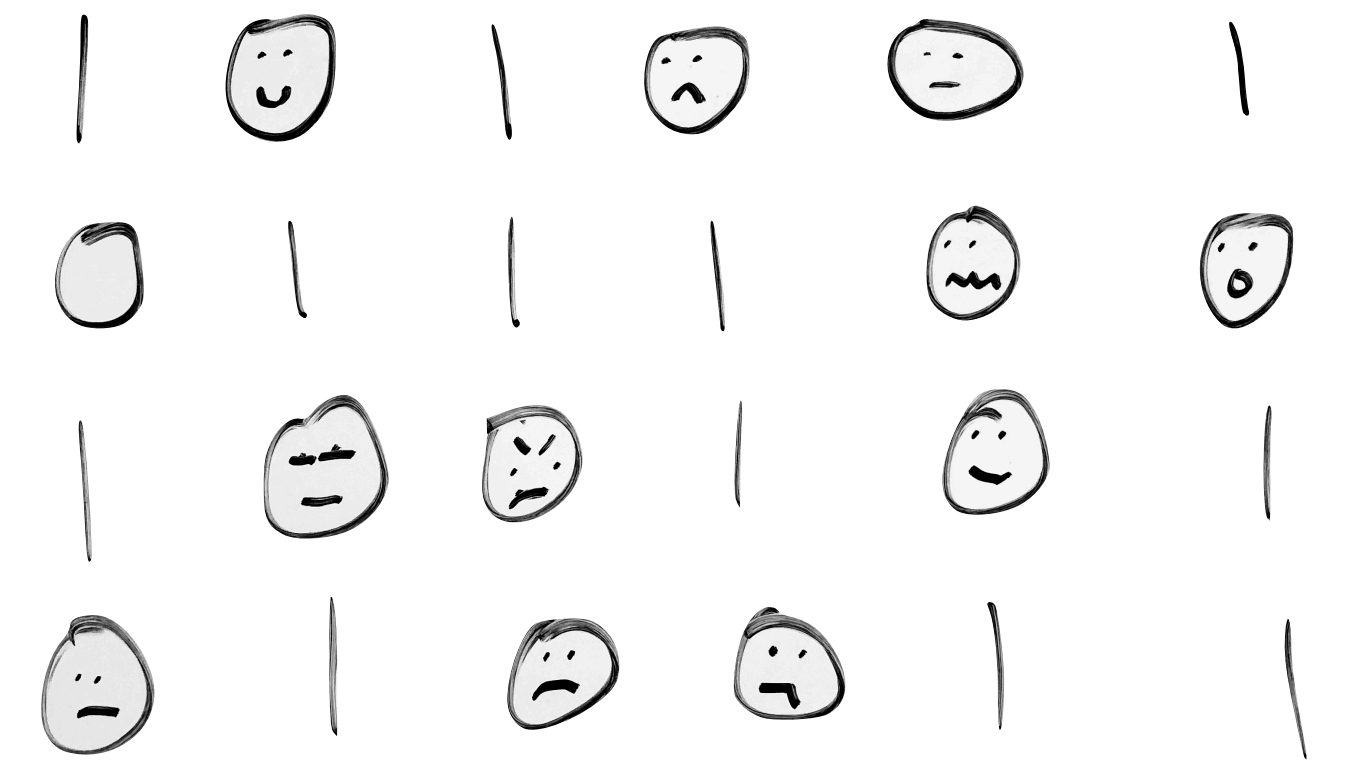
- Responsibility comes with power.
- Disrupting every facet of society, CS has become responsible for *everything* and *everyone*.
- *Our students don't recognize the power we don't educate them to use it responsibly.*



Gary Waters/Getty Images

Data literacy is diversity literacy

- Data can do harm precisely *because* of diversity.
- Data bias, disinformation, data privacy are fundamentally about *who* is represented by data.
- *Our students don't comprehend the true meaning of a bit and it's ability to entrench power, warp society, and destroy lives.*

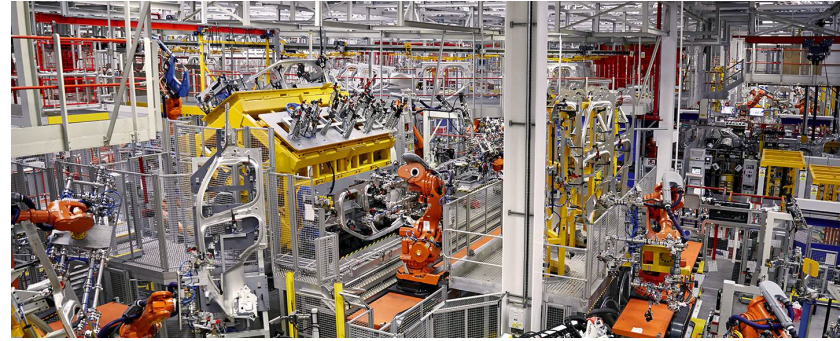


Greene, D., Hoffmann, A. L., & Stark, L. Better, nicer, clearer, fairer: A critical assessment of the movement for ethical artificial intelligence and machine learning. HICSS 2019.

Every developer needs to know that...

- **It's because of diversity** that *computing is limited*.
- **It's because of diversity** that *they are responsible*.
- **It's because of diversity** that *data does harm*.
- And so if developers ignore diversity, they're ignoring a central property of software as a medium and a tool for change.

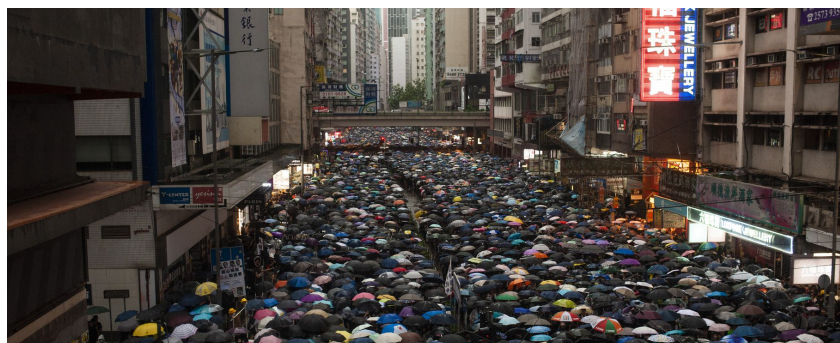
If developers knew these ideas...



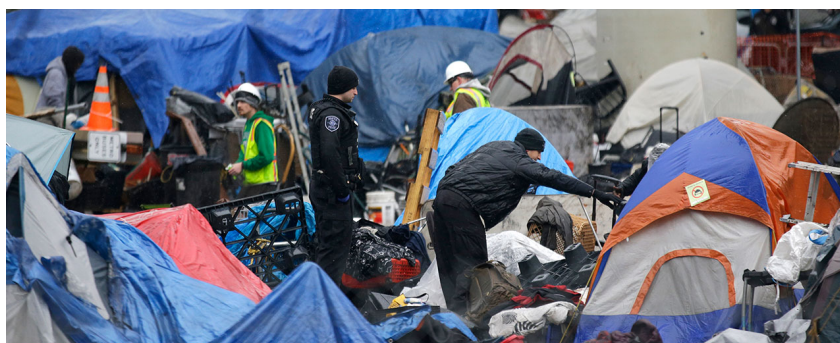
Developers might seriously consider **not creating something** because of the harm it might do.



Developers might choose projects that **help everyone** rather than just people like them.



Developers might acknowledge the power that they have to **make or break democracy**.



Developers might recognize that the world is **far more complicated than they think**, humbling them.

Teaching everyone to **code** will *not* teach how to design for diversity.

When people learn to program, we teach:

- 1) The simple, average cases for highly abstracted problems
- 2) That good design is a computational matter, not a social one.

They learn nothing about the diversity of humanity or society, or how their software design choices interact with this diversity. Today, most of academic CS doesn't view these as part of computer science.

Research opportunities

- How do we teach diversity in the context of computing in a way that transfers into **software design contexts**?
- How do we prepare developers to design for **outliers instead of averages**?
- How do we **prepare teachers** to teach these ideas?

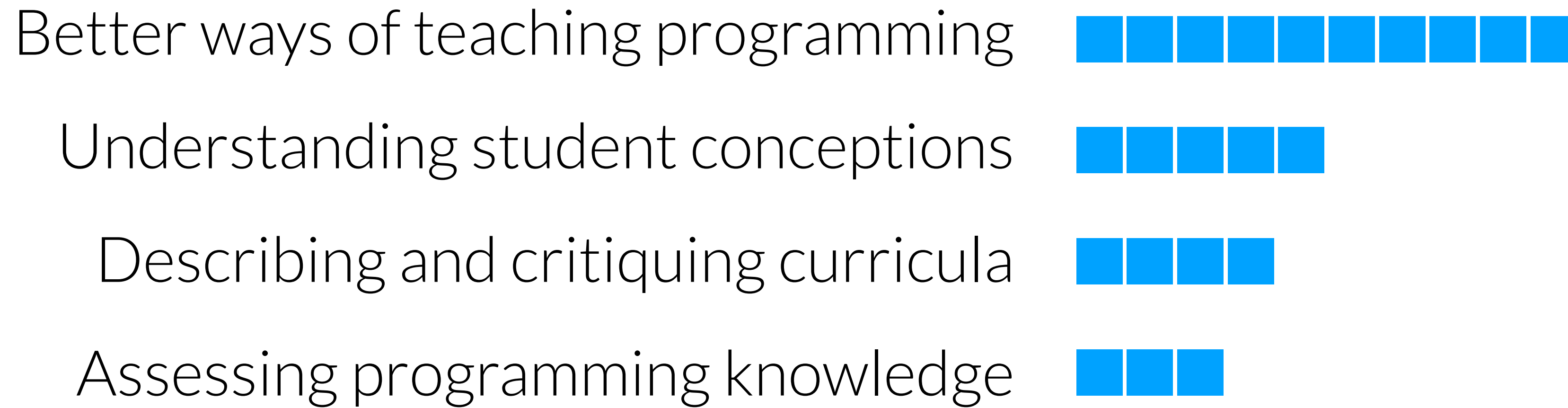
Call to action



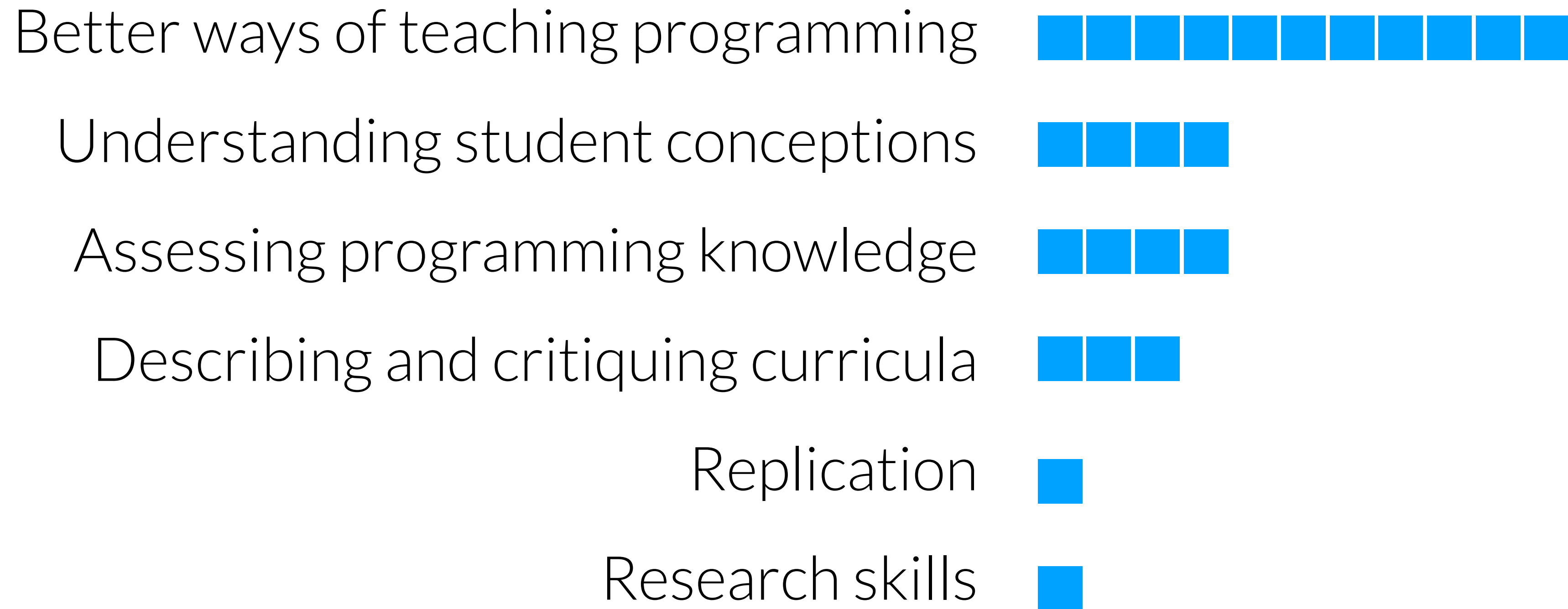
Remember, our community is awesome



Here's what we talked about at Koli 2018



Here's what we're talking about at Koli 2019



What we *should* talk about at Koli 2020

Effective ways of teaching the limits of computing 

Effective ways of teaching social responsibility 

Effective ways of teaching data literacy 

How to integrate these topics into curricula 

How to assess this knowledge 

How teaching programming affects the above 

We are the *only* community positioned to answer these questions and change CS education. It is our responsibility.

Thank you!

Amy J. Ko, Ph.D. *she/her/hers*

W UNIVERSITY *of* WASHINGTON

Teach and research the learning of:

- The limits of computing
- Social responsibility
- Data literacy
- Diversity

