Lecture 18: Identifying Social Threats

11:00 AM, Mar 3, 2021

Contents

Objectives

By the end of these notes, you will know:

- Dimensions to social threats from programs and algorithms

By the end of these notes, you will be able to:

- Systematically think through potential social threats

1 Analyzing Algorithms and Programs

What criteria have we learned for evaluating the quality of programs or algorithms? So far, we've mainly talked about running time and space usage. In your prior CS course, however, you also touched in some fashion on how programs can have impacts on individuals or organizations. The potential influence of a program on people, organizations, or population—whether that influence is positive or negative—s is called *social impact*.

Over the last few years, there has been growing societal recognition that technologists should recognize and design around the social impacts of their products. Technologists should be trained to consider not only the performance of their solutions, but also their social impacts. We are thus putting social impacts on a similar footing (and placement!) within CS18: one more of the many analyses you should be able to do on a program or algorithm.

In this lecture, we introduce you to some of the core dimensions of thinking about social impacts. In particular, we will work with some guidelines for identifying the *social threats* that could arise from a system.

2 Getting Started: What is Fairness?

In the course background survey, we asked you to tell us what you thought it would mean for an app like Yelp or Dazhong Dianping (an analogous site in China) to be fair. Sample (snippets of) responses from your classmates are at the following URL:

`https://docs.google.com/document/d/12FkAbgLChEU5JdX5GHNDNIPdUHiy6NXkZJGBH7bGRfg`

We asked you to talk through two questions (in breakouts) about each snippet:

1. Who is potentially harmed by this unfairness
2. Which aspects of the application (model, view, controller, or other) play a role in enabling that unfairness

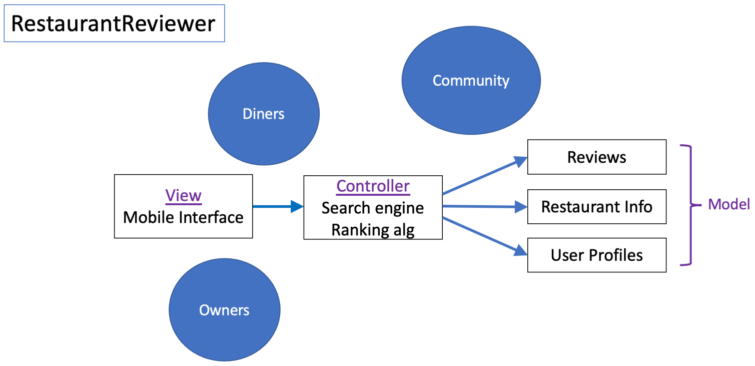
Ideas from each breakout are in the slides at the following URL:

`https://docs.google.com/presentation/d/1JK8hrSqUla9jBPLCOOc7F7LW0B7oSoUDPNd_xck0P2E/edit?usp=sharing`

3 Impacts Start with Stakeholders

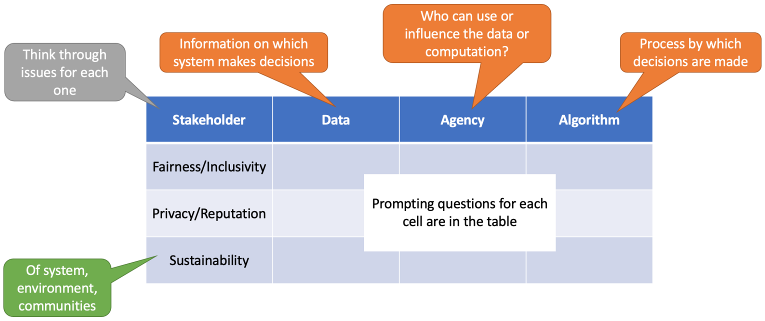
One goal of the previous exercise was to have everyone recognize that there are many stakeholders who contribute or who can be influenced by an application. In restaurant review apps, for example, there are potential impacts on each of diners, restaurant owners, restaurant employees, owners of other restaurants, and surrounding neighborhoods.

Taking a systematic look at social impacts starts from trying to identify the stakeholders who *influence* or *are influenced by* the application or algorithm. Notice that our original model-view-controller diagram does not include stakeholders, even though they interact with the system and its data in significant ways. Therefore, one place to start is by adding the stakeholders to the MVC diagram for a program:



4 Impacts Fall Into General Categories

The next step is to think about how these components of the system architecture might interact with key social concerns, such as fairness, inclusivity, privacy, reputations, and sustainability. We mapped these out into a 3x3 grid, with these concerns on the vertical axis and different aspects of a system on the horizontal:



Think through issues for each one

Information on which system makes decisions

Who can use or influence the data or computation?

Process by which decisions are made

Stakeholder	Data	Agency	Algorithm
Fairness/Inclusivity			
Privacy/Reputation			
Sustainability			

Prompting questions for each cell are in the table

Of system, environment, communities

Roughly, the *data* component relates to the model and data parts of the system, the *algorithm* component relates to the controller and data parts of the system, and the *agency* component relates to the stakeholders and view parts of the system.

We gave you a collection of questions for each cell (see page 2 at the following URL):

`https://docs.google.com/document/d/1PDYOcwgQmfSLS7YAAzoUlwoytvRr7_gDwM-BRkPa548/edit?usp=sharing`

5 Case Study: Recommendations in YouTube

To practice applying these ideas, let's consider the recommender algorithm in YouTube. danah boyd [sic] is a researcher at Microsoft (and Brown CS alum) who studies social impacts of technologies (often with a focus on teens). In 2019, she gave a terrific talk on aspects of the YouTube recommender algorithm that lead to adverse social impacts. The talk was titled "The Fragmentation of Truth". You can watch/read the full transcript at the following URL:

`https://points.datasociety.net/the-fragmentation-of-truth-3c766ebb74cf`

The talk centers around a theme: "Search doesn't return high-quality content when high-quality content doesn't exist". We worked with a high-level summary of the article in class, based on 5 points:

- If people search for content that doesn't exist, YouTube waits for users to upload something to the platform (rather than see what other quick-response sites like Twitter might be reporting)
- People find content by searching on keywords. If the search algorithm uses simple keyword match, groups can agree on keywords/phrases to promote content. Then if journalists or influencers promote the phrases ...
- Searching on outdated terms returns outdated results, devoid of context (including newer thinking on the terms)
- If content largely takes a one-sided opinion, all search results reinforce that opinion
- YouTube autoplays a "next" video, based on what previous viewers watched next or left comments on

Based on these points, we asked the breakouts to pick one cell of the table and think through its prompting questions (from the table) in the context of YouTube recommendations.

6 Key Takeaways

The goal of this lecture was to start showing you how to think systematically about social threats to systems. We've already been working on testing systems for correctness and analyzing systems for performance. This is the next step: exploring system designs for impact.

This lecture lands nicely between our first two projects, one on recommender systems and one on algorithms for search engines (meaning – we'll come back to the ideas in this lecture!).

6.1 What should you be able to do now?

We expect that you could work from the table from class and start to think ways that a system might impact others. You aren't expected to memorize anything from this lecture. Rather, we are helping you start to build a mental muscle around considering social impacts. We'll revisit this topic again later in the course.

Please let us know if you find any mistakes, inconsistencies, or confusing language in this or any other CS18 document by filling out the anonymous feedback form: `https://cs.brown.edu/courses/cs018/feedback`.