Toby Jia-Jun Li — Research Statement

Artificial intelligence (AI) technologies are emerging and affecting our lives in many aspects. However, the majority of individuals are merely *users* of AI with little capability to adapt AI to their own long-tail tasks, preferences, and interests that are not covered by the existing AI solutions. **To fulfill this need, I design, build, and study interactive systems to empower end-users and novice programmers to create, configure, and extend AI-powered systems.**

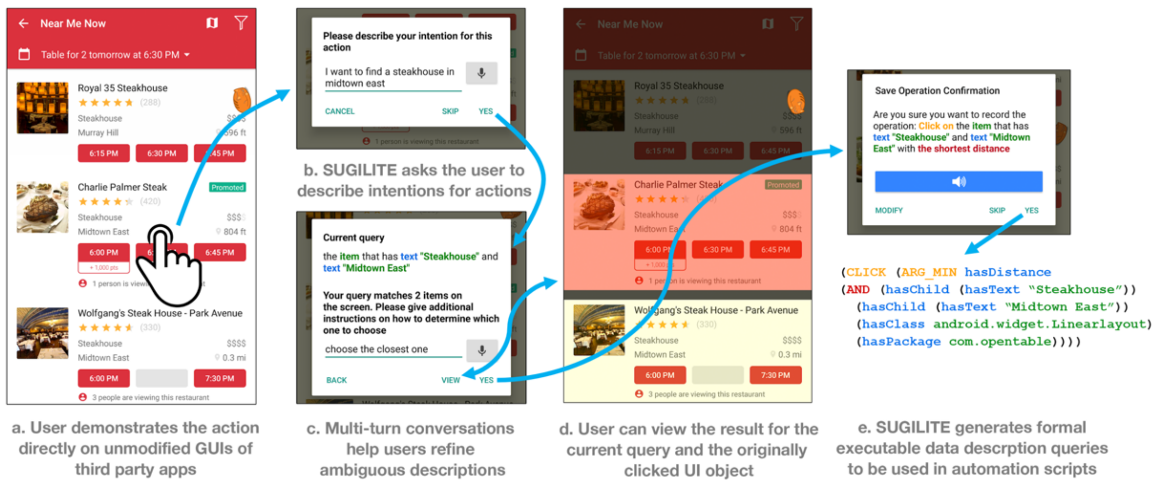
I am a human-computer interaction (HCI) researcher with a background in natural language processing (NLP) and end-user software engineering. In my dissertation, I developed and deployed a multi-modal interactive-task-learning agent that allows end-users to teach new task procedures, concepts, automation rules, and personal preferences using a combination of natural language instructions and demonstrations on the graphical user interfaces (GUIs) of existing apps [1–8]. I also designed, implemented, and studied developer tools that enable novice developers to build AI-powered systems [9,11,13,14].

The goal of my research is to bridge human intelligence and machine intelligence to solve real-world problems. My typical research paradigm starts with human-centered approaches [12] for studying how users naturally understand, think about, and express their tasks. Based on the insights, I develop new interfaces and interaction techniques that enable users to provide more effective inputs to the AI system. Meanwhile, I extend the capability of the AI system to understand, generalize from, and interact with user inputs. Another particular focus of my work is to support users to detect, understand, and recover from errors in such interactive AI systems. During this process, I apply a wide range of quantitative and qualitative research methodologies.

FACILITATING HUMAN-AI COLLABORATION IN INTERACTIVE TASK LEARNING

Enabling end-users to automate their tasks using intelligent agents has been a long-standing objective in both the HCI and the AI communities. A key research problem is enabling users to teach the agents new tasks. Despite the wide-adoption of agents like Siri, Google Assistant and Alexa, their capabilities are limited to domains that are either built-in or programmed by third-party developers [5]. My formative study showed that users' automation needs are highly diverse and personalized, with a "long-tail" that is not currently supported by the prevailing agents [1].

In summary, I built SUGILITE, a new multi-modal interactive task learning agent that enables end users to teach new tasks, concepts, personal preferences, and automation rules using a combination of natural language instructions and demonstrations on existing app GUIs. Compared to prior systems, SUGILITE made significant progress in usability, applicability, generalizability, expressiveness, robustness, and shareability. SUGILITE has been open-sourced on GitHub, deployed in-situ through Google Play Store, and evaluated in several user studies. My work in SUGILITE has resulted in 8 publications at top academic venues [1–8], including two award-winning ones, and I helped my advisors raise over $1 million in research grants to support this project.


Near Me Now
Table for 2 tomorrow at 6:30 PM
Royal 35 Steakhouse
Charlie Palmer Steak
Wolfgang's Steak House - Park Avenue
a. User demonstrates the action directly on unmodified GUIs of third party apps
b. SUGILITE asks the user to describe intentions for actions
c. Multi-turn conversations help users refine ambiguous descriptions
d. User can view the result for the current query and the originally clicked UI object
e. SUGILITE generates formal executable data descrption queries to be used in automation scripts

Please describe your intention for this action

I want to find a steakhouse in midtown east

CANCEL SKIP YES

Current query

the item that has text "Steakhouse" and text "Midtown East"

Your query matches 2 items on the screen. Please give additional instructions on how to determine which one to choose

choose the closest one

BACK VIEW YES

Save Operation Confirmation

Are you sure you want to record the operation: Click on the item that has text "Steakhouse" and text "Midtown East" with the shortest distance

MODIFY SKIP YES

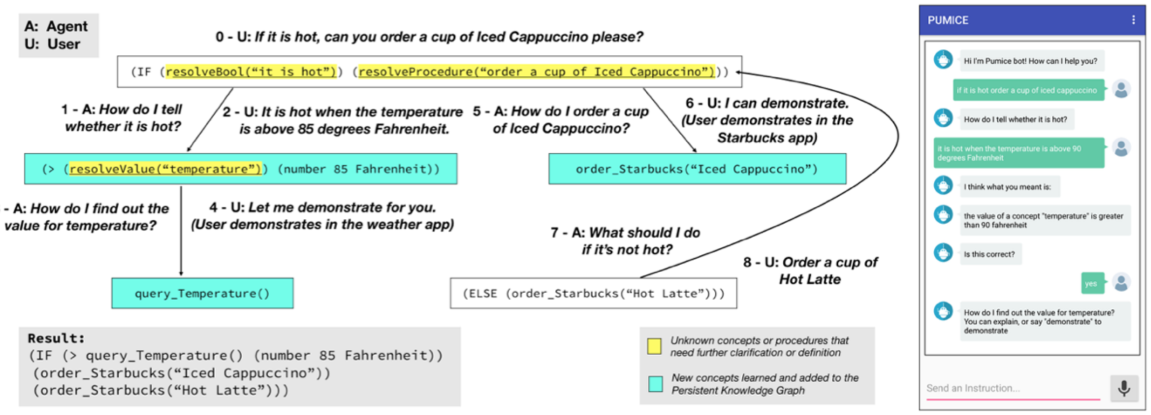
```
(CLICK (ARG_MIN hasDistance
(AND (hasChild (hasText "Steakhouse"))
  (hasChild (hasText "Midtown East"))
  (hasClass android.widget.Linearlayout)
  (hasPackage com.opentable))))
```

Figure 1. The screenshots of SUGILITE's demonstration mechanism and its multi-modal mixed-initiative intent clarification process for the demonstrated actions.

I designed, implemented, and studied novel interfaces and interaction techniques to facilitate the collaboration between the user and the agent as a part of SUGILITE. The multi-modal approach allows the user to first describe their intended task in natural language, and demonstrate new task procedures and concept on existing app GUIs. The utterances were grounded to the GUI demonstrations, allowing *mutual disambiguation* of the inputs from different modalities. The GUIs provide access to a large variety of computing services, information sources, and IoT devices [6] and encapsulate rich knowledge about the flows of the underlying tasks and the properties of relevant entities, so they can be used to bootstrap the domain-specific knowledge needed by the agent.

I created a mixed-initiative approach for users to disambiguate the intents of demonstrated actions [4]. After the user demonstrates an ambiguous action for generalization, the agent asks the user to clarify their actions in natural language in multiple turns of conversations. The system provides an interactive GUI-grounded visualization that highlights the agent's understanding of the user's utterances and prompts the user to explain specific aspects in their descriptions (Figure 1).

My work in SUGILITE introduced a new top-down lazy-evaluation dialog framework [8] for supporting concept and conditional instructions. The framework allows the user to start with saying an automation rule at a high level with ambiguous and vague concepts, procedures, and conditions. The agent then recursively resolves the ambiguities and vagueness in the initial instruction with the user through conversation, seeks clarifications from the user, and guides the user to define new concepts and procedures by demonstration if needed (Figure 2). I also designed a new multi-modal interface [3] that helps users discover, identify the causes of, and recover from conversational breakdowns using existing mobile app GUIs for grounding. This interface displays the system's understanding of user intents using GUI screenshots, allows the user to refer to third-party apps and their GUI screens in conversations as inputs for intent disambiguation, and enables the user to repair breakdowns using direct manipulation on these screenshots (Figure 3).


A: Agent
U: User
0 - U: If it is hot, can you order a cup of Iced Cappuccino please?
(IF (resolveBool("it is hot")) (resolveProcedure("order a cup of Iced Cappuccino")))
1 - A: How do I tell whether it is hot?
2 - U: It is hot when the temperature is above 85 degrees Fahrenheit.
5 - A: How do I order a cup of Iced Cappuccino?
6 - U: I can demonstrate. (User demonstrates in the Starbucks app)
(> (resolveValue("temperature")) (number 85 Fahrenheit))
order_Starbucks("Iced Cappuccino")
3 - A: How do I find out the value for temperature?
4 - U: Let me demonstrate for you. (User demonstrates in the weather app)
7 - A: What should I do if it's not hot?
8 - U: Order a cup of Hot Latte
query_Temperature()
(ELSE (order_Starbucks("Hot Latte")))
Unknown concepts or procedures that need further clarification or definition
New concepts learned and added to the Persistent Knowledge Graph
PUMICE
Send an Instruction...

```
Result:
(IF (> query_Temperature() (number 85 Fahrenheit))
 (order_Starbucks("Iced Cappuccino"))
 (order_Starbucks("Hot Latte")))
```

Hi I'm Pumice bot! How can I help you?

if it is hot order a cup of iced cappuccino

How do I tell whether it is hot?

it is hot when the temperature is above 90 degrees Fahrenheit

I think what you meant is:

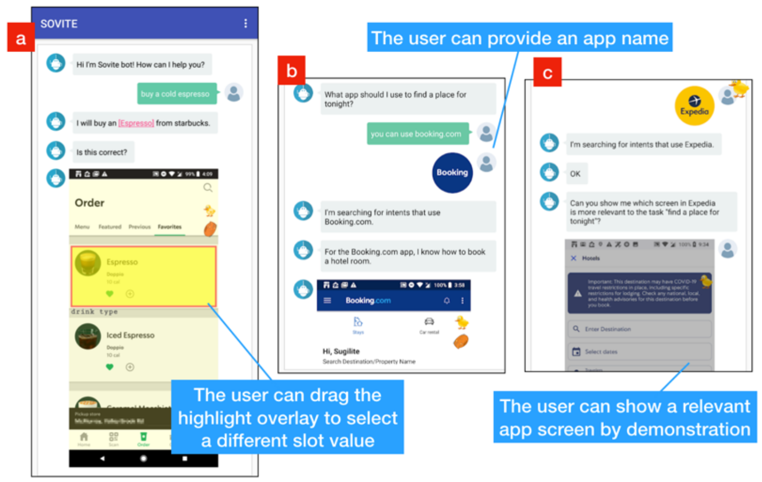
the value of a concept "temperature" is greater than 90 fahrenheit

Is this correct?

yes

How do I find out the value for temperature? You can explain, or say "demonstrate" to demonstrate

Figure 2. An example dialog structure when SUGILITE learns a new task that contains a conditional and new concepts. The numbers indicate the sequence of the utterances. The screenshot on the right shows the corresponding conversational interface.


a
SOVITE
Order
Menu
Featured
Previous
Favorites
Espresso
drink type
Iced Espresso
b
The user can provide an app name
Booking
Hi, Sugilite
Search Destination/Property Name
c
Expedia
Hotels
Enter Destination
Select dates
The user can drag the highlight overlay to select a different slot value
The user can show a relevant app screen by demonstration

Hi I'm Sovite bot! How can I help you?

buy a cold espresso

I will buy an [Espresso] from starbucks.

Is this correct?

What app should I use to find a place for tonight?

you can use booking.com

I'm searching for intents that use Booking.com.

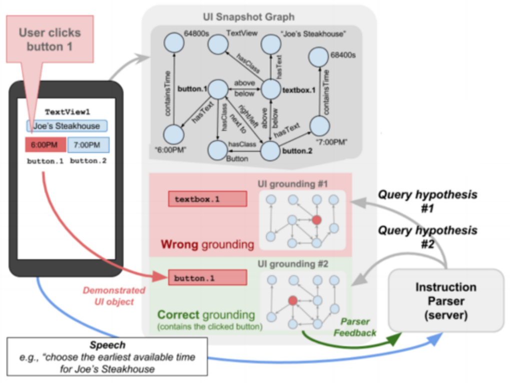
For the Booking.com app, I know how to book a hotel room.

I'm searching for intents that use Expedia.

OK

Can you show me which screen in Expedia is more relevant to the task "find a place for tonight"?

Figure 3. The interface for handling conversational breakdowns



```
(CLICK (ARG_MIN numeric_time (AND
  (hasClass Button)
  (below (hasText "Joe's Steakhouse")))))
```

Figure 4. The UI Snapshot Graph for grounding natural language instructions to GUIs

I created new AI-powered system capabilities that support the aforementioned interfaces. A key challenge in supporting the multi-modal task learning is to effectively ground the user's natural language explanations to the corresponding GUI elements. I developed a new technique in SUGILITE that represents each GUI screen as a *UI Snapshot Graph* [4,7]. A floating semantic parser can parse the user's natural language instruction into graph queries on this graph in a simple but flexible LISP-like functional query language. An iterative human-in-the-loop pipeline executes the query on the graph, verifies if the result matches the correct entity that the user originally demonstrated, and seeks additional input from the user if the result does not match, or matches more than one entity (Figure 4). The goal of this process is to generate a query that uniquely matches the target UI element and also reflects the user's underlying intent.

This multi-modal GUI grounding approach allows SUGILITE to learn generalized procedures (e.g., to order *any* kind of beverage from Starbucks) from a demonstration of a specific instance of the task (e.g., ordering an iced cappuccino) [1]. SUGILITE achieves this by grounding the utterance (e.g., "order a cup of iced cappuccino") to the data descriptions of the target UI elements (e.g., click on the menu item that has the text "Iced Cappuccino") and the arguments of the demonstrated actions. It analyzes the hierarchical structure of GUI to extract the potential alternative values of the parameter (Figure 5). A similar GUI-grounding approach is also used for disambiguating the procedures to invoke [1] and for learning generalized concepts [8].

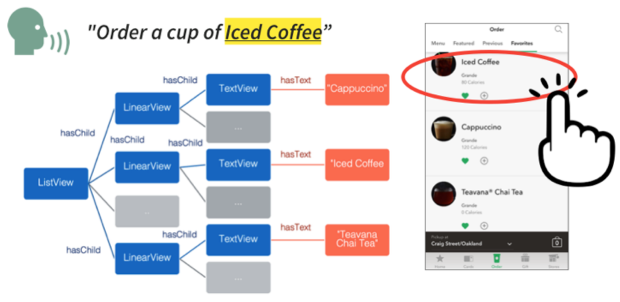


Figure 5. SUGILITE parameterizes the script using the hierarchical GUI structure

Another major barrier to the adoption of GUI-based interactive task learning agents has been the privacy concern in sharing the user-taught scripts, which may contain personal information like passwords, account numbers and balances. To address this, I developed a new mechanism [2] in SUGILITE for the privacy-preserving sharing of GUI-based demonstrational scripts. This mechanism can identify and obfuscate personal information at the time of sharing with minimal user intervention. It collects and aggregates hashed texts from app GUIs on users' phones in everyday use. This aggregated data allows the system to identify fields in scripts with personal information, and obfuscate their values. At runtime, the obfuscated fields can be rebuilt locally using the script consumer's apps, which helps preserve the shared scripts' transparency, readability, robustness, and generalizability.

I developed a new technique for computationally modeling app GUIs and user GUI interactions, providing the distributed representations of them for use in neural networks [10]. With the rise of data-driven computational methods for modeling GUI interactions, GUI screens have become not only interfaces for users to interact with the underlying computing services, but also valuable data sources that encode the underlying task flow, the interactions, and the design patterns of the corresponding apps. My new self-supervised SCREEN2VEC technique can generate distributed vector semantic representations for GUI Screens and components using their textual contents, layouts, and meta-data without requiring manual human annotation efforts, addressing an important gap in the prior literature. The result vectors are shown to be effective and useful in many downstream tasks for modeling user tasks, querying GUI screens with natural language commands, and supporting embedding composability.

LOWERING THE BARRIER TO DEVELOPING AI-POWERED APPLICATIONS

I also created tools to support non-expert developers to build AI-powered applications. There are more than 27 millions of software developers in the world. Most of them do not have specialized expertise in AI domains, which prevents them from adopting the recent AI advances in their own application. My work seeks to lower the barrier to developing AI-powered applications through new interactive developer tools.

KITE [9] is a practical system I created that enables developers to bootstrap task-oriented conversational bots from existing mobile apps. Using traces of users using an app, KITE can automatically derive a task model, a graph of actions, and the associated inputs representing different task execution paths. On top of them, KITE generates a natural language interface for the task model with questions and answers using a Transformer-based neural

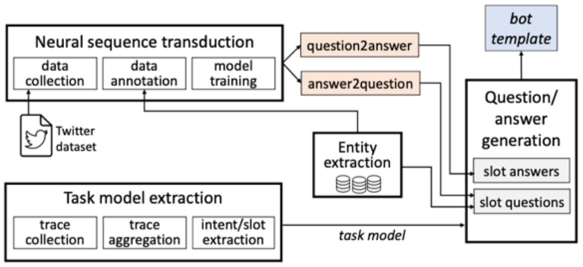
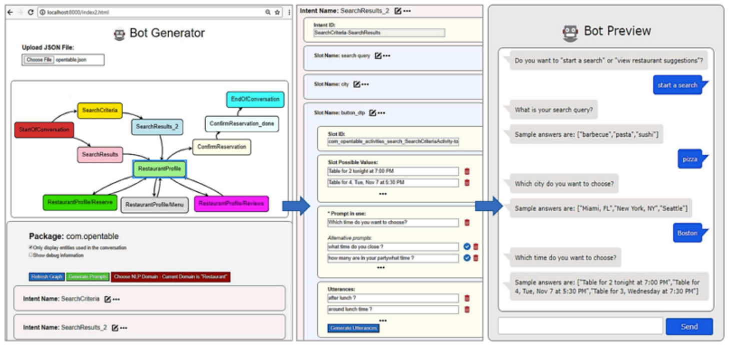


Figure 6. The processing pipeline of KITE's approach



Do you want to "start a search" or "view restaurant suggestions"?

start a search

What is your search query?

Sample answers are: ["barbecue","pasta","sushi"]

pizza

Which city do you want to choose?

Sample answers are: ["Miami, FL","New York, NY","Seattle"]

Boston

Which time do you want to choose?

Sample answers are: ["Table for 2 tonight at 7:00 PM","Table for 4, Tue, Nov 7 at 5:30 PM","Table for 3, Wednesday at 7:30 PM"]

Figure 7. The interactive bot developing and testing interfaces of KITE

network (Figure 6). The developer can then iteratively test and revise the auto-generated bot in KITE 's interactive interface (Figure 7). KITE was not only evaluated in a lab study and published at MobiSys 2018, but also patented and is in the process of being incorporated into commercial products.

In collaboration with other teams of researchers, we developed three additional developer tools for AI-powered applications: GENO, PRIVACYSTREAM, and WIKIBRAIN. GENO [13] is a tool for adding the support for coordinated multi-modal voice inputs (e.g., moving *this event* to the next Monday) to existing web apps without requiring NLP and multi-modal interaction expertise. PRIVACYSTREAM [11] introduces a new Android developer framework for accessing and processing personal data (e.g., sensory inputs, location, contacts, photos) as a stream in a functional programming model, making it easier for developers to use personal data while simultaneously making it easier to analyze how that personal data is processed. WIKIBRAIN [14] democratized computation with Wikipedia knowledge for developers by providing simple access to the Wikipedia data and the state-of-art Wikipedia-based algorithms. All these tools have been open-sourced on GitHub and used by the communities of developers and researchers.

RESEARCH AGENDA

In the future, I plan to continue my line of research in **democratizing AI to empower individuals**. In this section, I outline two research themes that I am the most excited about.

End-User-Programmable AI Systems for the Future of Work

SUGILITE presents an example of supporting end user programming for task automation. For the next step, I am particularly interested in developing end-user-programmable AI systems to specifically address the challenges faced by the workforce in the era of "the fourth industrial revolution". While the adoption of AI technologies have significantly increased the overall productivity of society, the gains have mostly gone to a small group at the top while many workers have been stagnated in wages or displaced by AI. I attribute much of this phenomenon to the *top-down* approach used in developing and deploying AI. While many workers, especially the rapidly expanding group of "gig workers", participate in the corporate-provided AI infrastructures, they often act as interchangeable and replaceable components (e.g., Uber drivers, Instacart shoppers, data annotators) in large algorithm-directed workflows with little agency. The AI systems are often developed from a corporate's perspective with its best interest in mind, with little support for individual worker's personal preferences, motivation, and creativity.

I propose to explore a *bottom-up* approach that helps individual workers automate and augment their work with AI systems. Instead of replacing workers with AI, this approach can emancipate workers from mundane repetitive tasks, enabling them to focus on the creative and social aspects of work they love that AI systems cannot feasibly replace in near future. In our formative study with SUGILITE, many users stated they would love to automate their work tasks. The current SUGILITE is already able to learn work-related tasks that involve basic lookups, simple conditional rules, and well-defined actions through app GUIs. In the future, I plan to conduct contextual user research to investigate the automation needs of workers in different disciplines, study the societal impacts of end-user-programmable AI systems for the future-of-work, and develop more effective and more personalized interactive AI systems to assist individual users with their work-related tasks in their best interests.

A significant barrier to the bottom-up approach is the data ownership problem. An effective AI system usually requires a large amount of data to train, which is often only feasible for large organizations. I consider the recent advances in general-purpose language (e.g., BERT, GPT-3), reasoning (e.g., COMET, Atomic), and task completion (e.g., Almond) models an important step towards end-user programmable AI. These models are great fits for *few-shot learning* and *transfer learning*, allowing effective incorporation of general knowledge with small amounts of data and instructions from end users. For my future research, I plan to focus on designing the interaction between the user and the AI system in these scenarios, bridging the transparency and explainability of AI systems with the user's natural cognitive models for the task, and enabling the user to provide useful inputs that boost the generalizability, expressiveness, robustness of the AI system through effective human-AI collaboration.

Natural Interaction Modalities in Developer Tools

In my past work of designing different interactive developer tools, I have used several distinct input modalities and interaction methods for program specification, such as conventional source code, natural language spoken instructions and conversations, program synthesis with demonstrated examples, and direct manipulation in visual programming and GUIs. For the next step, **I plan to further explore the design space of interactive tools for diverse types of programming activities in the full end-to-end lifecycle of developing AI systems.** such as data labeling, data wrangling, exploratory analysis, model tuning, error analysis, bias mitigation, system deployment and monitoring, system extension, and performance optimization.

The advances in fields such as natural language processing, sensing for gesture and gaze tracking, interactive data visualization, program synthesis, and computational modeling of programming activities through software repository mining have laid the technical foundation for my next breakthrough in multi-modal interaction for software development. In my design process, I will specifically focus on the needs, capabilities, and routines of different types of developers, creating dynamic developer interfaces that are adaptive to these changing factors.

REFERENCES

[1] **Toby Jia-Jun Li**, Amos Azaria, and Brad A. Myers. SUGILITE: Creating Multimodal Smartphone Automation by Demonstration. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems* (*CHI '17*) , ACM. **Best Paper Honorable Mention.**

[2] **Toby Jia-Jun Li**, Jingya Chen, Brandon Canfield, and Brad A. Myers. Privacy-Preserving Script Sharing in GUI-Based Programming-by-Demonstration Systems. In *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW1 (May 2020) , ACM.

[3] **Toby Jia-Jun Li**, Jingya Chen, Haijun Xia, Tom M. Mitchell, and Brad A. Myers. Multi-Modal Repairs of Conversational Breakdowns in Task-Oriented Dialogs. *In Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (*UIST '20*) , ACM. **Best Paper.**

[4] **Toby Jia-Jun Li**, Igor Labutov, Xiaohan Nancy Li, Xiaoyi Zhang, Wenze Shi, Wanling Ding, Tom M. Mitchell, and Brad A. Myers. APPINITE: A Multi-Modal Interface for Specifying Data Descriptions in Programming by Demonstration Using Natural Language Instructions. In *2018 IEEE Symposium on Visual Languages and Human-Centric Computing* (*VL/HCC '18*), IEEE.

[5] **Toby Jia-Jun Li**, Igor Labutov, Brad A. Myers, Amos Azaria, Alexander I. Rudnicky, and Tom M. Mitchell. Teaching Agents When They Fail: End User Development in Goal-oriented Conversational Agents. Chapter of *Studies in Conversational UX Design*, Springer, 2018.

[6] **Toby Jia-Jun Li**, Yuanchun Li, Fanglin Chen, and Brad A. Myers. Programming IoT Devices by Demonstration Using Mobile Apps. In End-User Development, *Proceedings of 6th International Symposium on End User Development* (*IS-EUD '17*), Springer. **Best Paper.**

[7] **Toby Jia-Jun Li**, Tom Mitchell, and Brad Myers. Interactive Task Learning from GUI-Grounded Natural Language Instructions and Demonstrations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (*ACL '20*)*: System Demonstrations*..

[8] **Toby Jia-Jun Li**, Marissa Radensky, Justin Jia, Kirielle Singarajah, Tom M. Mitchell, and Brad A. Myers. PUMICE: A Multi-Modal Agent that Learns Concepts and Conditionals from Natural Language and Demonstrations. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology* (*UIST '19*) , ACM.

[9] **Toby Jia-Jun Li** and Oriana Riva. KITE: Building conversational bots from mobile apps. In *Proceedings of the 16th ACM International Conference on Mobile Systems, Applications, and Services* (*MobiSys '18*) , ACM.

[10] **Toby Jia-Jun Li**, Lindsay Popowski, Tom M. Mitchell, and Brad A. Myers. Screen2Vec: Distributed Representations of GUI Screens and GUI Components. *In Preparation*.

[11] Yuanchun Li, Fanglin Chen, **Toby Jia-Jun Li**, Yao Guo, Gang Huang, Matthew Fredrikson, Yuvraj Agarwal, and Jason I. Hong. PrivacyStreams: Enabling Transparency in Personal Data Processing for Mobile Apps. In *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* (IMWUT) 1, 3 (September 2017), 76:1–76:26, ACM.

[12] Brad A. Myers, Andrew J. Ko, Chris Scaffidi, Stephen Oney, YoungSeok Yoon, Kerry Chang, Mary Beth Kery, and **Toby Jia-Jun Li**. Making End User Development More Natural. Chapter of *New Perspectives in End-User Development*, Springer, 2017.

[13] Ritam Sarmah, Yunpeng Ding, Di Wang, Cheuk Yin Phipson Lee, **Toby Jia-Jun Li**, and Xiang "Anthony" Chen. Geno: A Developer Tool for Authoring Multimodal Interaction on Existing Web Applications. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology* (*UIST '20*), ACM.

[14] Shilad Sen, **Toby Jia-Jun Li**, WikiBrain Team, and Brent Hecht. WikiBrain: Democratizing computation on Wikipedia. In *Proceedings of the 10th International Symposium on Open Collaboration* (*WikiSym + OpenSym 2014*), ACM.