Autonomy and Privacy with Open Federated Virtual Assistants

Monica S. Lam
Computer Science Department
Stanford University

Summary: Autonomy and Privacy with Open Federated Virtual Assistants

Large platforms pose threats to individual privacy not only because of the information that they collect and control, but also due to their market dominance, resulting in no meaningful alternatives. We propose an open, federated virtual assistant, called Almond, to decentralize our computing ecosystem and offer users autonomy and the protection of privacy. This proposal gathers together experts in natural language processing, machine learning, distributed languages and systems, knowledge engineering, security, human-computer interaction, and crowdsourcing to create the world's best virtual assistant and a thriving open distributed computing ecosystem by 2023.

Keywords. [Distributed Systems, Compilers and Programming Languages], [Application Services], [Security], [Machine learning], privacy, human-computer interaction, open competition, virtual assistants.

Intellectual Merit. We propose to connect all the online services and IoT devices into an inter-operable programmable web. This is achieved with: (1) An open, crowdsourced, universal encyclopedia containing all the public APIs in the world called Thingpedia. These APIs are invocable both at programming and natural language levels. (2) A high-level personal-web language called ThingTalk that lets users connect their accounts on multiple internet services by leveraging Thingpedia, shares any internet resource with anybody with fine-grain control, and controls how a resource can share data with third parties.

We propose to let consumers control their personal web using natural language. We will create a deep-learning neural network, called LUInet (Linguistic User Interface Network) to translate natural language into ThinkTalk. LUInet will understand how to operate the world's digital interfaces.

Privacy with fine-grain access control is provided through federated virtual assistants. Our virtual assistants will be open-source, so users can own or choose a host for their personal data. Sharing with privacy is implemented as secure distributed execution of ThingTalk programs. Control of sharing between services and third parties is provided by (1) smart contracts, written in ThingTalk for human-interpretability and (2) an auditable and indelible log of contracts and transactions by using an efficient blockchain based on federated byzantine agreements.

We will apply and validate our technology in (1) a smart building research project that collects personal information to study how architectural designs affect our wellbeing, and (2) a deployed health care sharing system with hospitals. We will create the knowledge bases of the open virtual assistant by organizing the largest-scale computer science research project using our Crowdresearch platform. This will broaden participation in computer science to diverse students.

Broader Impact. Virtual assistants, such as Amazon Alexa and Google Home, threaten to become the largest platform monopoly that sees all personal data and controls access to the web. Almond offers not just an open-source alternative, but a more powerful one as it is natural-language programmable and protects privacy through federation. Natural-language programming expands the utility of computing to ordinary people, reducing the programming bottleneck. Service providers can use our open technology to create linguistic interfaces to their products, build inter-operable virtual assistants and skills. Our broadening participation program teaches the younger generation technology, research, and the importance of privacy.

1 Introduction

There is growing awareness in both the U.S. and in the E.U. that large platforms pose threats to individual privacy not only because of the information that they collect and control, but also due to their market dominance. For example, the social networking market is dominated globally by Facebook, which has no meaningful competition today in most countries. The implementation of the General Data Protection Regulation in the EU mandates that "[t]he data subject shall have the right to receive the personal data concerning him or her, which he or she has provided to a controller, in a structured, commonly used and machine-readable format and have the right to transmit those data to another controller without hindrance from the controller to which the personal data have been provided." This provision represents the first systematic attempt to address by structural means the existing dominance large platforms have over personal information. Relatedly, during the Fall of 2018, the U.S. Federal Trade Commission is seeking public comment and holding hearings on questions of whether market dominance in digital marketplaces is posing a threat to consumer privacy.

Recent advances in Internet of Things (IoT) devices and advanced machine learning will be generating even more valuable and private information about individuals. In particular, the health care industry is among the fastest to adopt the Internet of Things. According to a report by McKinsey, spending on the health care IoT solutions will reach a staggering one trillion by 2025. Collecting data from IoTs and converting the data into actionable insights can greatly improve the quality and effectiveness of service, bringing especially high value for the elderly, patients with chronic conditions, and those requiring constant supervision. But managing IoT data in the healthcare domain must be HIPAA compliant to ensure both system security (no erroneous actions triggered) and data privacy (no personal health information leaked). Virtual assistants were introduced recently to help users manage their IoT device and services. Would Alexa or Google Home evolve to become the dominant platform that collects and controls all the IoT devices and services? This proposal aims to create a new technological foundation that shifts the power from dominant, centralized services to individuals and protects privacy with a decentralized architecture. Our goal is to research the fundamental technology, to develop open software, and to work with industry to create a commercially viable ecosystem built on a decentralized infrastructure foundation.

1.1 Vision of a Personal Natural-Language Programmable Web

The key to a platform's success is to attract consumers with more capability and a better user experience. Our vision is to unify all the disparate digital assets into one inter-operable web. We create an open-source virtual assistant that lets consumers use *natural language* to independently *control, connect and use* all their IoT services and devices easily, and *share any of their information* with *fine-grain control*, *without disclosing information to a third party.* This is far better than today's services where consumers are typically forced into take-it-or-leave-it relationships with companies where their only option is to consent to a company's terms or opt to not use a product or service.

We have built a prototype virtual assistant that demonstrates the feasibility of our approach called Almond[11, 12]. Let us illustrate our vision with the example of an asthma patient, called Bob. Here, Bob is using the open-source Almond that runs on his home server. Bob is using an IoT inhaler; to find out the locales that cause him breathing

problems, he can tell Almond "when I use my inhaler, record my GPS location on Box in InhalerLog". In case of emergency, he tells his assistant "let my Dad know if I am taken to the hospital". Bob's assistant is constantly monitoring his location, and his Dad is informed of the location only when necessary. In the meantime, Bob's doctor, Dr. Smith, tells Bob's Almond to "let him know if Bob's peak flow-meter drops below 180L/min". Bob grants the permission, and Dr. Smith can monitor Bob's health continuously and efficiently. He can also tell Bob's Almond to "warn Bob when the Air Quality Index is above 500 and he is running." All these sentences are directly translated into code and run on Bob's home server without a third party seeing any of the data.

We use this example to illustrate that individuals have many different needs and interests, and there are many sources of data that can be used to improve their lives. With the architecture that we propose, individuals can control digital assets and have custom code generated from their natural language commands. This reduces our dependence on commercial software, which typically has only been developed for the most profitable use cases, often with the goal of collecting and controlling user data.

1.2 Intellectual Merits

This research lays down a new distributed computing infrastructure to give control of personal data back to individuals. Whereas systems research in the past focuses mainly on single services, this research is about the interplay between all the different internet services and how to help consumers manage all the different resources. This project creates an inter-operable web of internet services, and lets consumers program them in natural language. To protect privacy, we deliver this technology to consumers as open-source virtual assistants that can be run on local devices. Easy sharing, while protecting privacy, is supported via decentralized assistants and decentralized block chains.

Too many decentralized systems in the past lack the usability to attract users. To make a real impact in practice, we need to deliver to users new capabilities, a better user experience, and protection of personal data all at the same time. To satisfy all these considerations at the same time, we need cross-disciplinary research in the areas of distributed systems and programming languages, natural language and machine learning, human computer interaction and crowdsourcing, knowledge engineering, security, as well as consumer privacy law.

1. *Inter-operable web of IoT devices and services.* Today consumers are inundated with data from numerous online accounts and IoT devices. Especially with IoT devices generating high volumes of real-time data, we need a programmatical way to consume the data and react accordingly. This research connects all internet services into an *inter-operable programmable web*. We will create an open, crowdsourced, universal encyclopedia containing all the public APIs in the world called *Thingpedia*. The representation of this knowledge base is designed to support inter-operability both at the traditional programming level as well as at the linguistic level.

2. *ThingTalk: a personal-web programming language.* To give individuals autonomy over their own assets on the web, we introduce a new class of formal languages for programming the personal web. We propose the language *ThingTalk* to let users take full advantage of all their assets described in Thingpedia. Using this language, the user can succinctly connect internet resources into event-driven processes with just one line of code. They can control who, what, when, where, and how their data is

to be shared. In addition, they can specify the circumstances under which a service holding users' data can share with a third party.

3. *Natural language (NL) for the personal web.* We wish to empower non-developers to use natural language to control their personal web. This research develops deep-learning neural models to automatically translate natural language into ThingTalk. Our model, called LUInet, needs to understand the world's of digital interfaces, as collected in Thingpedia.

4. *Sharing with privacy through federation.* Sharing between consumers is supported mainly by centralized services that claim ownership of users' data. This research develops a federated virtual assistant architecture that lets users share everything they own on the personal web easily, while not losing data ownership. Sharing is supported by securely executing ThingTalk programs over a secure Distributed ThingTalk Protocol. In addition, we allow users to control how a service provider can share their personal information to third parties. The smart contracts are written in ThingTalk, because of its natural language interpretability, and are kept on an efficient blockchain, based on federated byzantine agreement, for auditability.

5. *Crowdresearch for social good.* Because of the need of an expansive knowledge base of APIs and natural language utterances, virtual assistants require a research and development effort that few companies can afford. Thus, to create a decentralized computing ecosystem, the world also needs to know how to decentralize the research and development effort. We hypothesize that for virtual assistants, once we determine the right APIs and data representations, we can crowdsource the contributions from students. We will research how to organize students into the largest-scale research team to create the most knowledgeable virtual assistant. We will use this opportunity to broaden participation among diverse high-school and college students and teach students about technology, research, and the importance of privacy.

1.3 Project Validation

To fully understand if our technology can serve a real need for consumers, we will perform three end-to-end experiments with real scenarios:

1. We will validate our proposal that federated virtual assistants can address consumers need to share their internet resources with access control while preserving privacy. We will use the crowdresearch platform to conduct this experiment, with the help of thousands of students.

2. Many research projects require the acquisition and analysis of highly-sensitive personal data, and it is paramount to obtain user permissions and to safeguard the privacy of information. To study Almond for such use cases, we will work with a cross-departmental research team at Stanford to evaluate the effects of building design on occupants' wellbeing.

3. Big data analytics is poised to revolutionize health care; to fully reach its potential, however, we must give users the ability to dictate how their personal health data are to be shared. We will apply Almond in helping patients share their EHR in emergencies, and to manage the IoT pillbox device data to improve medical adherence.

1.4 Broader Impact

All the technologies developed in this project will be made available as open-source, allowing the industry to create a wide variety of competitive products. They include: Thingpedia, which contains all the public programming and linguistic interfaces of public APIs; a new personal-web language called ThingTalk, which can be used to develop apps that protect privacy; and LUInet, deep neural models for translating natural language into formal languages using Thingpedia. We will make available to the public a fully working open-source virtual assistant that protects privacy.

Together with the ThingTalk dataset, we expect that Thingpedia will make an important contribution to research natural language processing. Just like Wikipedia is the go-to knowledge base for open question answering [89, 69, 65, 93], and Freebase used to be the preferred knowledge base for structured query generation [8, 9, 50], we envision Thingpedia as the best open knowledge repository to study what users want to "do" with their devices and on the Internet.

These technologies can be used to create competitive general-purpose or special-purpose virtual assistants, such as ones for hotels or cars. They can be used to build linguistic user interfaces for their apps and skills. The virtual assistant architecture can also be used in corporations to let employees automate their jobs.

The distributed virtual assistant infrastructure we develop in this project can support a wide variety of interesting research projects. We will run the largest-scale computer science research project that engages not just college students, but high-school students as well. Students will learn and experiment with the inter-operable web of IoT devices and services, natural language processing, and machine learning, decentralized systems, and the importance of privacy.

1.5 Team

To reach the goals of creating a useful system and a thriving ecosystem, we need a team whose expertise covers all the different disciplines and can work closely together. Thus, our core team consists of faculty all from the same institution (Stanford): Michael Bernstein (human-computer interaction, crowdsourcing), Dan Boneh (security, cryptography), Monica Lam (programming languages, distributed systems), James Landay (human-computer interaction), David Mazières (distributed systems, blockchain technology), Chris Manning (natural language processing, machine learning), and Chris Re (knowledge engineering, machine learning). Many aspects of this project requires making advances in multiple disciplines at the same time, we expect many of the students in the project will be jointly advised.

2 Research Rationale

2.1 Commercial Virtual Assistants

The virtual assistant will likely become our biggest threat to privacy and open competition in the future. The fast adoption of virtual assistants in recent years is a testimony that it addresses an important consumer pain point: it is tedious to manage or operate all the different digital services. Virtual assistants offer a uniform, language-based interface to the large number of accounts to web services and IoTs that we all have. As an intermediary between consumers and the internet, as well as having access to all our accounts, they can

offer the most frictionless services. They are in the best position to promote a wide variety of services, such as offering banking services to users with the best credit ratings. Virtual assistants, when mature, are a threat to all companies selling to consumers, from music, grocery, retail, travel, health, finance, and all IoT manufacturers.

The virtual assistant is our interface to the *linguistic web*, just like the browser is our interface to the *graphical* web. Today, Amazon Alexa has a repository of 40,000 skills and Google Home has 1 million functions. We are witnessing the creation of a *closed, proprietary* web, reminiscent of the walled garden of AOL.

Today's virtual assistants let us use voice to perform the same operations as we can with GUIs. They use a repository of skills, or intents, that specify how a verbal command is to be dispatched to a particular service. For example, "ask Weather.com to get the weather" is matched to an API call to Weather.com which can read out weather report. These assistants play the role of dispatch, similar to how a browser would take users to the page of interest. Intents are not adequate to create a programmable, inter-operable web. The repository is missing the information on the result parameters, so it cannot inter-operate across functions. For example, it cannot be used to accomplish tasks that connect multiple servics, such as reminding us to bring an umbrella whenever rain is forecast the next day.

We hope to leapfrog over existing virtual assistant technology by letting users automate higher level tasks. Especially with IoTs generating real-time data continuously, it is important that consumers do not have to be in the loop to read every result. We here propose to develop this technology in the open domain to help promote an open, competitive ecosystem.

2.2 Inter-Operability with Open, Crowdsourced Thingpedia

We have an opportunity right now to prevent the linguistic web from being owned and controlled as proprietary platforms.

First, we take advantage of this paradigm shift to unify all the individual commercial services into an inter-operable web. The Thingpedia representation captures the entire signature of APIs, not just the intents with input parameters as in Alexa and Google Home, but also the result parameters. It captures it both at the coding level, as well as the natural language level. Having the right data representation makes a big difference, because our representation creates an inter-operable web, which users can program in natural language, thanks to our personal web languages and natural language translation.

Second, because of GDPR regulations, companies are required to make all their data available to the owners, and we are no longer limited by the public APIs supported. By entering information on how all released data can be accessed into the publicly available Thingpedia, this opens up access to everybody and every company.

Third, we need an open repository of APIs. Consumers are not well served with proprietary platforms. Just like how Wikipedia attracts contributions from every corner of the world, we expect Thingpedia to attract contributions for every big and small online services that wish to be accessible linguistically. Just as Wikipedia has become larger than any proprietary knowledge base, it is possible for Thingpedia to do the same. Many companies are threatened by the proprietary virtual assistants, Thingpedia can potentially attract contributions from companies who see value in an open linguistic web.

Fourth, because Thingpedia is open, and our representation subsumes that of Alexa and Google Home, we offer the skill writers a practical advantage. They just have to enter the information once, and their skills can be made available on Alexa, Google Home, as

well as Almond. If run on Almond, their skills can inter-operate with others, which offers a great benefit to users. This portability, in addition to more natural language processing support, also entices contributions to Thingpedia.

2.3 ThingTalk: an NL-Interpretable Personal Web Language

The demand for consumers to connect online services and IoT devices together was first demonstrated by IFTTT [1], which is a website that lets users describe IF-This-Then-That recipes. For example, "if my checking account has less than $100, send me an email message", or "if I change my profile on Facebook, change my profile to the same picture in Twitter". So far, 75 millions of recipes have been created using a wide variety of internet services, demonstrating the wide applicability of this simple construct. However, IFTTT does not have a formal language, nor does it have a natural language interface. Users have to configure the recipes using a menu-driven interface. Furthermore, IFTTT yet another proprietary centralized service; all the contributed APIs are proprietary, and users have to give IFTTT their credentials to use the recipes. The recently released Apple shortcut packages up the IFTTT graphical user interface.

The IFTTT service suggests a powerful abstraction. Users register all the devices and services and give their credentials to a smart agent. Users can simply specify the functionality of interest, and the smart agent handles authetication, management of passwords, internet connections, callbacks, polling, notifications, and routing all the data. This abstraction inspires the design of ThingTalk.

ThingTalk is a new class of language designed for individuals to manage their personal web. All the commands are assumed to be applied to their personal devices that users have logged into. This is a great departure from typical scripting languages designed to handle many different users accounts on the servers. It is designed to work with Thingpedia. It allows users to create event-driven code involving multiple internet services, control fine-grain access, and grant the rights to share with third parties. We need the power of languages to cope with the large number of publically available APIs, as a GUI interface is too clumsy. ThingTalk is easy to learn because of its high-level semantics. It has a very simple control construct but it is powerful because its large library of Thingpedia, which contains functions familiar to consumers, such as tweeting, posting on Facebook, or playing YouTube videos.

Virtual assistants are all based on natural language inputs. ThingTalk provides an important abstraction in natural language understanding for virtual assistants:

1. Carrying out natural language commands can now be separated into two steps: translate the natural language into a program in a formal personal-web language, and execute the translated program. This separates the concern of natural language processing and implementation of the language.
2. The high-level semantics of ThingTalk makes translation from natural language feasible. It also gives a well-defined semantics to natural language utterances.
3. It is possible to translate the program back to natural language. Natural language is ambiguous. We can ensure that the natural language is translated correctly, by reading the program back to the user in natural language, and asking the user to confirm. This makes these personal-web languages suitable for writing smart contracts; the code can be translated back into natural language to ensure that all the signees have the same interpretation.

4. It also lets incentivized users to learn to write in the formal language before natural language translation is perfected.

2.4 Natural Language Programming

Programming is a powerful tool, which previously is available only to developers. Non-developers are dependent on commercial products or company's in-house developers for automation. Developers have to code in all the supported options, often limited by the need to keep the menu-driven interfaces manageable. As a result, software today is developed to support only the majority and most profitable use cases.

Previous work in natural language programming focuses on translating natural language into code of existing programming languages, such as Python [63, 90, 68] or SQL [93, 88]. It is unclear why a developer would find coding Python in natural language is useful, since they are not bogged down by the syntax of Python. The research on SQL focuses mainly on answering natural language queries. Given a corpus of questions and translations, batch training can be performed end-to-end by testing if the answers are correct, using reinforcement learning. This is not possible for our work because our language performs side effects affecting the real world.

At the same time, unlike previous work in semantic parsing with abstract meaning representations [4, 66], our goal is not to understand general natural language. Our scope is limited to families of sentences such as commands and access control restrictions, in the domain of what is controllable with a computer. Additionally, users are incentivized to use understandable language with the virtual assistant, like how we have adapted to Alexa's limitations and learned to write in Graffiti before handwriting recognition became feasible.

Unlike other programming languages, ThingTalk has a simple construct, but it utilizes all the known APIs in Thingpedia. The translator must understand how the web works as represented by the APIs. Thus, our goal is to create a deep-learning neural network we call LUInet. As Thingpedia gets populated with the world's public APIs, LUInet will learn how humans operate the internet of things and services. The considerable overlap between various services presents opportunities to improve learning and robustness with multi-task learning. The pre-trained LUInet will be made openly available, like VGG, Alexnet for image understanding. This resource is likely to accelerate our technology in natural language understanding with respect to computer interfaces.

2.5 Federated Virtual Assistants and Blockchain

Despite years of research in distributed systems, the only mainstream distributed communication system used today is Email, invented before the Arpanet. Most consumers value convenience more than privacy; a successful distributed system must offer a better user experience than the centralized counterparts. Here we propose to use a federated virtual assistant with natural language programmability to give users unprecedented control over how they share their internet resources.

Sharing is broken today. People wishing to share must have accounts with the same services, and are limited by what is supported by the provider. Or, we need to give out our account credentials, and hence full access to our accounts. If we use Facebook or Google credentials to sign on to an account, sharing the credential to one service will means sharing many others.

Our solution is to let our virtual assistant shares data for us. Requests for data are sent

Figure 1: The open, distributed Almond virtual assistant with Thingpedia (an inter-operable skill repository), LUInet (a neural network to translate natural language into ThingTalk code), and Distributed ThingTalk Protocol for inter-virtual assistant communication.

as ThingTalk programs to the owner's assistant, which executes them, upon the owner's approval. We use remote Thingtalk program execution to implement sharing between consumers. Requests for data are executed by the owner's virtual assistant, subject to the owner's access control, and only the need-to-know results are shared. The owner can share anything they can access via the virtual assistant without the need to disclose their credentials. The Distributed ThingTalk Protocol (DTP) supports inter-operability between virtual assistants, just like SMTP supports sharing by email servers. As Almond is open-source, we make it easy to create inter-operable virtual assistants.

Consumers data today are kept by service providers in the cloud, and we would like to control how such are shared with third parties. One important application is health care, where data sharing is retricted by HIPAA. It is useful to give patients flexibility, while maintaining control, so they can contribute select personal health data for the sake of advancing health care research, possibly with renumeration from pharmaceutical companies. As discussed, being natural-language interpretable, ThingTalk is an effective high-level language for describing these contracts. We need to keep these contracts, as well as data-sharing transactions, in a trusted, indelible, auditable log. We propose the use of blockchain, implemented with federated byzantine agreements, as an efficient, trusted, decentralized storage for this purpose.

2.6 Open Systems

Academia and open-source community have a tradition in creating open technology that disrupts or prevents monopolies. Unix, first released in 1971, and subsequently BSD and Linux gave rise to server and mobile OSes used widely today. The NCSA Mosaic browser, first released in 1993, and subsequently Netscape and Firefox offer an open-source solution to the Internet Explorer. We note that timeliness is important; e.g. Linux could not disrupt the desktop because Windows was well entrenched on the PCs. Our research so far gave us a lead in NL-programmable virtual assistants; it is important to popularize the open system as soon as possible.

Being open-source, Almond can be offered as services by different vendors. We envision that there will be free services supported by ads, paid services that respect privacy, and home servers that let users keep their own data. This achieves the goal of giving users choice and encouraging competition, rather than having just a monopoly or duopoly platform provider, leaving no meaningful alternatives.

3 A Natural-Language Programmable Personal Web

In this section, we first present the technical details of the Almond prototype, then describe the research challenges that we will address in this project.

3.1 Feasibility Study: the Almond Prototype

To study the feasibility of this approach, we have prototyped Almond, and made it available as a web service and also an Android app; users can run Almond on their own devices, thus owning all their credentials and data, if desired. Its skill respository, Thingpedia, currently contains the APIs of over 60 devices and 200 functions.

The high-level architecture of federated virtual assistants [11] is shown in Figure 1. Users issue virtual assistant commands in natural language; the commands are translated by the deep-learning neural network called LUINet into ThingTalk. ThingTalk has a simple construct that can connect multiple functions in the Thingpedia knowledge base. Thingpedia is a crowdsourced repository containing natural language interfaces and open APIs for any device or service. For privacy, all the personal data and credentials are stored in the Almond virtual assistant, whose code is open-source and can be run on personal phones or home servers. The virtual assistants communicate using the Distributed ThingTalk Protocol to support sharing between users.

3.1.1 Thingpedia and ThingTalk

Let us first present the syntax for the basic ThingTalk construct:

$$when, p_{\text{WHEN}} \Rightarrow get, p_{\text{GET}} \Rightarrow do$$

The program consists of one, two, or three of the clauses, separated by $\Rightarrow$: a WHEN clause, a GET clause, and a DO clause. The WHEN clause specifies when the command will be triggered. In the syntax, *when* can refer to "`now`", a calendar or interval timer, or the monitoring of a *retrieval* function f with the syntax "`monitor`$f(...)$"; in the latter case, the command is executed automatically whenever the result of $f(...)$ changes. *get* in the GET clause calls a retrieval function, while the DO clause calls the *action* function indicated by *do*. If unspecified, the DO clause defaults to "`notify`", meaning that the results will be presented to the user.

For example, the following program continually monitors the user's Instagram profile for new pictures that have hashtag #cat, and copies them on the user's Twitter account:

$$\texttt{monitor}\ \text{@instagram.get\_pictures}(), \texttt{contains}(\mathit{hashtags}, \#\text{cat})$$
$$\Rightarrow \text{@twitter.post\_picture}(\mathit{url} = \mathit{instagram.url}, \mathit{caption} = \text{"cat"})$$

Input parameters can be passed as keywords to each function, and output parameters can be referred to by name in later functions. Both the WHEN and the GET clause can include predicates, denoted by p_{WHEN} and p_{GET} respectively, which operate on the output parameters of the current and the previous functions. Information flows from the WHEN clause to the GET clause and then finally, to the DO clause.

The power of ThingTalk derives from the large number of APIs in Thingpedia. Each function entry includes the API, with its list of input and output parameters and type information, its implementation, example utterances for natural language training, and a canonical confirmation sentence to ensure that the input command was parsed correctly. An example Thingpedia entry for a security camera is shown in Fig. 2. The represented function "@security_camera.event" has no input parameter and 3 output parameters.

3.1.2 Access control

We generalize ThingTalk to specify access control policies with the following syntax: [12]:

Retrieval function: @security_camera.event	
Parameters	`out` *picture_url* : URL `out` *has_person* : Boolean `out` *has_motion* : Boolean
Example utterances	"get a snapshot of my security camera" "show me my security camera" "when there is a new event detected on my security camera" "when my security camera detects a person" "when my security camera detects motion"
Confirmation	"the current event detected on your security camera"
Returns a list	no
Can be monitored	yes

Figure 2: The security camera entry in Thingpedia.

$$\hat{p}_\sigma, \epsilon = \text{SELF} : \mathit{when}, \hat{p}_{\text{WHEN}} \Rightarrow \mathit{get}, \hat{p}_{\text{GET}} \Rightarrow \mathit{do}, \hat{p}_{\text{DO}}$$

This syntax lets the owner specify who, $\hat{p}_\sigma$, can make the request; the request can be an entire ThingTalk program with WHEN, GET and DO clauses, augmented wildcards for generalization. The owner can impose predicates $\hat{p}_{\text{WHEN}}, \hat{p}_{\text{GET}}, \hat{p}_{\text{DO}}$ on the parameter values and results returned from any retrieval function in Thingpedia to limit disclosure. Thus, the language lets us express who, what, when, where, and how the data is to be shared, including the flow of information between multiple services.

For example, the constraint "*allow students to post any ACM articles with tag 'access control' to our lab's Facebook account*" is expressed as follows:

$$\begin{aligned}\texttt{group}(\sigma, \text{@students}), \epsilon = \text{SELF} : \texttt{now} &\Rightarrow \text{@acm.search}, \texttt{contains}(\mathit{tags}, \text{"access control"}) \\ &\Rightarrow \text{@facebook.post}, \mathit{url} = \mathit{acm.url}\end{aligned}$$

With this extension, Thingtalk policies is a form of *attribute-based access control* [92], a classic method of access control. Yet, ThingTalk differs from existing ABAC policy languages, like XACML [58], because it constrains the execution of the whole program, rather than individual sub-requests. It is easy to understand what programs are allowed by a ThingTalk policy: they are all the programs that replace wildcards in the policy with concrete value and that respect the constraints. ThingTalk policies are also easy to compose, as they use only positive assertions (a program is always allowed, never explicilty disallowed), hence there is no confusion or conflict between the policies being composed.

ThingTalk also differs from program policy languages, such as Polymer [5] and Resin [91]. Those policy languages are written by developers, often the same authors of the program being checked, to enforce high level constraints such as security or data integrity. ThingTalk is instead written by end users to permit limited execution programs that are otherwise not allowed; it must be understandable at a high level, without regard to the actual implementation.

We have formalized the meaning of ThingTalk with denotational semantics, and proved that constraint conformance can be formulated and enforced using SMT (Satisfiability Modulo Theories). Our experiments show that the SMT solver is fast enough to handle conformance in real time.

3.1.3 Natural Language Translation

The first challenge we faced was that there is no high-quality training data set for compound virtual assistant commands. Inspired by the Sempre project, we developed the following methodology for collecting training data. We sample the formal grammar, and use the natural language templates to create a synthetic training set, consisting of pairs of natural language utterances and its equivalent in code. We crowdsource paraphrases of the synthetic training set. We extract more templates based on the inputs, and repeat the process of generating more synthetic data and getting paraphrases until the training set contains a representative sample. We then design and train a semantic parser with the synthetic and paraphrased data. We analyze the errors and refine the language, library representation, and the semantic parser.

We experimented with two neural models: sequence to sequence with attention [76, 3, 21] and an updated version of Transformer [80] which supports multi-modality generation. As in previous neural semantic parsing work [87, 90, 45], the target program is encoded as a sequence of productions in the formal grammar of ThingTalk, augmented with explicit type annotations. This sequence of productions is generated by a deterministic parser. The decoder learns a conditional probability for the next production given the history. By using a copying mechanism [82, 56, 39], we are able to retrieve free-form parameters such as "Good Morning!" and "take you pills" from anywhere in the sentence and insert them in the output.

On the ThingTalk dataset we created, which includes a set of 24,566 paraphrased commands, corresponding to 3,410 distinct programs in ThingTalk, the Transformer model achieves 77% accuracy. This suggests that it is feasible to create an accurate semantic parser for initial system experimentation, and the amount of data needed is scalable with the growth of Thingpedia.

3.2 Research Agenda

The results reported above were obtained after about two years of work. We started by defining and implementing Thingpedia and ThingTalk, we generated synthetic data and paraphrased them with Amazon Mechanical Turk workers. Our first attempt to create a semantic parser failed. We discovered that it was very hard to get quality data because our workers are unfamiliar with the concept of event-driven commands. The paraphrase would be erroneous and there is little variability in the sentences. This prompted us to develop the methodology to amplify variability observed by incorporating them in synthetic data through templates.

After getting a good quality data set, we discovered an even more fundamental problem. The accuracy of the semantic parser depends on the design of Thingpedia and Thingtalk. This means that the data representation, ThingTalk, and the neural model need to be co-designed. In the original design of Thingpedia, it had three kinds of APIs, triggers of events, retrievals, and actions, to match the WHEN, GET, DO clauses in the ThingTalk program. For example, Twitter provides a callback primitive which triggers on tweets, say, with certain hashtags; it also provides a retrieval primitive to retrieve existing tweets with certain hashtags. The availability of triggers and retrieval functions depends on what is supported by specific devices. The confusion between triggers and retrievals makes it impossible to create an accurate semantic parser.

Our solution is to unify triggers and retrievals as one set, so we can use either in the

WHEN and GET clauses. This requires us to change the implementation of ThingTalk, for example, to poll retrievals to turn them into triggers. This design simplifies the users' mental model, makes the semantics more uniform, and improves the accuracy of the semantic parser.

The interaction between Thingpedia, ThingTalk, and LUInet present new challenges that require joint research across different disciplines in computer science. We now describe our research agenda to scale our prototype from 200 functions to the world of public APIs, and from paraphrased sentences to natural sentences in the wild.

Expressiveness of ThingTalk. Is ThingTalk expressive enough to cover what consumers want to say? Because ThingTalk is a generalization of IFTTT recipes, we know it is express enough to cover the 75 million recipes created at IFTTT. What about the expressiveness of the ThingTalk extension as an access control policy language? We conducted a preliminary user study with 60 AMT workers, and found that 85% of their proposed access control use cases can be expressed as ThingTalk, provided that APIs they need are implemented in Thingpedia.

Next, we will study the expressiveness of ThingTalk for controlling how a service provider can share our data with third parties. Examples of scenarios include health care institutions sharing data with CDC (Centers for Disease Control and Prevention) and pharmaceutical companies, or social networks sharing user behavioral data with advertising networks, or schools sharing transcripts with potential employers or other schools, or the OS sharing hardware access (GPS, accelerometers) with apps. It has been shown, for example, the access of gyrometers can let applications identify the speaker and even parse speech[**?**]. ThingTalk can be used to constrain the permission given to apps.

Representing the World's IoT and Services. As Thingpedia grows, we envision it will host many services in the same domain and with similar functionality, but different APIs. Every entry in Thingpedia today contains the raw APIs as supplied by each service provider and the terminology is not standardized. This is problematic for services because users may not use the expected word for a specific service, or might not know what specific APIs and parameters are available. We envision eventually Thingpedia will accumulate all the possible data and actions associated with each category of devices. It will be able to respond intelligently even if a particular device is missing expected functionality. For example, if a user asks to dim a light-bulb that does not support dimming, the virtual assistant can indicate as such instead of saying it does not know what the user wants or behaving erratically.

Data programming is a technique to collect information using heuristics that may be erroneous, and uses statistical methods to extract the signals from the noise [70]. We plan to use data programming to acquire Thingpedia entries automatically and to regularize the information. We will investigate how to automatically derive linguistic interfaces from existing websites, and to identify common primitives across similar services. The consistency and comprehensiveness of Thingpedia are key to the success of our virtual assistant.

End-to-End Learning of Natural Language. Today, APIs in Thingpedia must be annotated with natural language metadata. This metadata includes a canonical sentence for each API that can be used to confirm or ask permission for a command, it includes the name of each parameter and how it is used in filters and projections, how the agent should ask for missing parameters, and how each result is presented to the user. It also allows us to reason deterministically about the correctness of confirmation sentences, which is key to the security of federated assistants. The use of this hard-coded table of annotations makes the dialog agent ungrammatical and unnatural. It also limits the ability to use datasets or

conversation traces collected from other systems. We will study how all natural language commands, questions and responses can be learned from end-to-end, crowdsourced user interactions. We will investigate whether transfer learning, multi-task learning, as well as the use of compositional understanding and generation architectures can be applied, so that the knowledge acquired on one service can be transferred to new services.

One of the biggest opportunities to improve generalization and performance is to exploit multi-task learning, including doing multi-domain learning across domains or dialects and transfer learning to reuse things learned on one task for similar tasks. Neural model in general have a strong record in transfer learning [13], and in particular we have recently shown strong results for multi-task learning on NL tasks [17] using a technique that allows semi-supervised learning from a combination of labeled and unlabeled training data. Almond provides an ideal setting for such an approach to perform well: ThingPedia provides a common core ontology that can be shared, and although tasks are disparate, the share many elements (opening things, sending things, playing things, etc.) that will allow effective cross-task generalization.

Compositionality. A natural language understanding system, in any domain, must understand compositionality, because natural language in inherently compositional: complex sentences and textbfs are composed of small words and phrases. The ThingTalk dataset is the first moderately large, high-quality dataset of compositional commands from an open domain, in which compositionality for semantic parsing models can be investigated. Previous work in studying this important problem used toy datasets [46] or IFTTT [67, 6, 21, 90, 51]; the latter suffers from such a high amount of noise examples (unintellegible, non-English, spam, etc.) that the best reported program accuracy is 3.7%.

Using the ThingTalk dataset, we will investigate how different models can generalize to unseen combinations, including combinations involving wholly new devices trained with limited data. An emerging research question in deep learning is precisely how to model compositional reasoning [46]. We have done some initial work in this area in the domain of Visual Question Answering [33], where we treated the image as a "knowledge base." Here, we can take advantage of a richer, larger knowledge base and aim to apply extensions of this architecture to the composition of command sequences.

Understanding High-Level Concepts. Much of the power of natural language actually comes from the fact that it "chunks" useful high-level concepts. One way to achieve this is to allow introduction of higher-level concepts by explicit instruction from users. Current virtual assistants are frustrating to users because they do not learn on the job; it is the user that needs to learn "what works". In the alternative of "on the job learning" [84, 83], a system can be explicitly instructed. For example a user can say "When I say 'Remind Mary of her appointment.' that means look up Mary's next appointment and send a text with content 'Mary, here are the details of your upcoming appointment' followed by details of the next appointment." Remaining challenges here include learning how to abstract variables (here, "Mary") and learning the implicit contextual restrictions and generalizations that are appropriate ("Remind Mary to be polite" is very different).

Robust Performance on True Data. Machine learning models are commonly trained and tested on data that is very similar, yielding a model with high results in theory, but poor performance on real data. The ThingTalk prototype is no different: we trained and evaluated our model on paraphrased data – a corpus with a different distribution than real users' commands.

Can we build a robust model that, while trained on skewed, artificial or even wholly synthetic data, can achieve good performance in the real world task? Acquiring a large

Figure 3: Dad's request for Alice's security camera in natural language is translated into a ThingTalk program, which is executed by Alice's virtual assistant, according to Alice's access control, specified in natural language and translated into ThingTalk.

training set of real user data, even unannotated, is a chicken-and-egg problem, until the system is performing well enough to attract users. Our preliminary results on ThingTalk suggest that the combination of paraphrased and synthetic data can improve the quality on paraphrased data, but the effect depends heavily on the structure of the model and the training strategy. We plan to investigate the use of robust optimization and robust statistics, off-policy learning [53, 59], curriculum learning [7, 40, 41] and possibly other techniques. This will be made possible by having a baseline system that is deployed in a real user setting, acquiring enough data to evaluate our approach.

Full System Considerations. Our goal is to create a deep neural LUInet that can understand all the commands related to the open internet APIs. We expect the biggest challenge to be compositionality as Thingpedia scales. We need to extend the current testing with paraphrased data to data in practice. We need also to complement the LUInet with techniques to personalize the parser for higher accuracy, automatically generate precise, yet natural, sentences to confirm the commands, generate dialogs to refine users' commands, suggest useful commands, and help users discover new capabilities.

4 Federated Architectures

Federated virtual assistants, with natural language programmability, lets users share anything their assistant can access easily. Our proposal is to have all sharing requests be sent to the owners virtual assistant which, upon the approval of the owner, will perform the requested commands on behalf of the requester and return the results [12].

4.1 Federated Virtual Assistants

In this design, the owner's assistant performs the task on behalf of the requester and only returns precisely the allowed result. A access control compliant ThingTalk program is translated into distributed ThingTalk programs, executing on the requester and the owner's virtual assistant, and they communicate via a secure remote execution protocol, called DTP (Distributed ThingTalk Protocol). We illustrate this protocol with an example in Fig. 3. The example shows a dad accessing his daughter's security camera through Almond. (a) Dad first makes a request to his virtual assistant in natural language, (b) Dad's virtual assistant translates the request into a ThingTalk program and sends it to Alice's assistant. (c) Alice's assistant checks if the program conforms to Alice's previously specified ThingTalk policies, possibly with the addition of run-time checks. If so, Alice is notified that the program will be run. If not, (d) Alice's assistant translates the program into natural language and as Alice for approval; upon which Alice can provide additional constraints (e). If permission is given, the assistant (g) executes the request, potentially with Alice's additional constraints, and returns the detected events to Dad's assistant; (h) Dad's assistant then notifies Dad of the result.

This example illustrates how this protocol supports a simple, intuitive user experience, while providing security. The requester's assistant abstracts away the details of communication. The owner can specify access control policies for future requests, so they do not

have to approve the requests one by one. The owner can give approval on demand; this is important because it is hard to anticipate possible requests and specify access controls a priori. The owner can impose additional constraints, instead of denying a request flat out. The owner is given a full description of the program in natural language, to prevent the vulnerability of executing unknown remote programs. This is made possible because ThingTalk is an NL-target language. Only the need to know requested data is returned to the requester.

Research Challenges, Experimentation and Validation. This research shows a promising approach that shifts the control of sharing from service providers to the individuals owning the data. This design is general and extensible: the requester can potentially access anything the owners virtual assistant can, which is defined by the open-world encyclopedia of virtual assistant skills. Sharing does not require the requester to get an account with the service provider; nor is the sharing limited by what is supported by the service itself. The design is secure and supports privacy. Only need-to-know information is disclosed: a requester receives precisely the approved shared results, and does not gain access to any credentials or additional information.

In a preliminary study with 20 users, 18 liked the proposed concept, and 14 liked the prototype we have developed. In a survey where 200 people are asked if they would share 20 different personal resources, the number of people willing to share doubles if they can impose access controls like the ones allowed in ThingTalk. This suggests that there is interest and need among consumers for access control. However, as one of the users in our study notes, the concept is "totally new and great" but it has a "high barrier to entry".

This project aims to develop and validate all the technologies necessary to make this concept a reality. Besides improving on previously noted research topics in Thingpedia, ThingTalk, LUInet, we have to address the complexity of access control. Access control is known to be complex and error-prone [54, 71, 75]. The topics to address include the interactions betwen different access control rules, corner cases that users may not be aware of, and potential unintended consequences when the enable a rule. As discussed in Section 6.1, we plan to perform an end-to-end test to evaluate the feasibility of giving users autonomy in internet resource sharing.

4.2 Blockchain

A key requirement for users to trust Almond is knowing not just that it will enforce access control and protect their data, but that they can learn what has happened to their data. We will use blockchain technology to implement a highly-available global directory and indelible audit trail, ensuring both that people can find and get access to data when needed, and that they can be held accountable for doing so.

Consider a person whose desired policy is to allow any medical personnel to access his or her medical records in case of emergency, and to empower a healthcare proxy to take decisions in case of incapacity. Several pieces are required to implement such a policy: Medical personnel need to know what medical institution is storing the patient's medical records. They need to declare an emergency, so as to gain access to the records. Finally, they need to be held accountable for declaring an emergency.

A blockchain offers several properties that are well suited to such tasks. First, a blockchain's data is widely replicated, ensuring very high read availability and concurrent read throughput in almost any scenario. This is ideal for low-update, critical, global databases (such as a directory of users' primary medical institutions) that aren't owned by

Figure 4: Automatic generation of GUIs from natural language, before and after styling with ImagineNet

any particular institution and hence must be updatable in a decentralized way. Second, blockchains provide a very strong append-only property ideal for "break-glass" procedures such as declaring a medical emergency. Finally, blockchains provide a robust means of disseminating security critical data, such as credential revocations when someone is fired.

Unfortunately, despite these appealing properties, traditional mined blockchains such as Bitcoin [61] are not a good fit for the applications we envisage. In particular, proof-of-work [23] blockchains (the dominant paradigm) require long latencies to publish data, and more to publish it securely. Bitcoin has an average block latency of 10 *minutes*, with multiple blocks required for increased security. Though other blockchains have tuned this lower, under reasonable assumptions 10 minutes may be the right value to accommodate maximum network delay [64]. Second, the security of mined blockchains is inherently tied to cryptocurrency-denominated mining fees, which may be incomparable to the extrinsic damage caused by tampering with audit logs, policy logic, or credential revocations.

PI Mazières has proposed Federated Byzantine Agreement (FBA) [55] as an alternative to mining-based consensus. His Stellar Consensus Protocol (SCP) construction has been used by the Stellar and Mobilecoin [57] blockchains, and, because it relies only on digital signatures, can reach global consensus in just a few seconds. SCP's key innovation is allowing quorums to be defined in a decentralized way based on each node's idea of sets of organizations important enough in aggregate to speak for the whole network. Because these decisions are based on organizations' reputations, this model invalidates one of the key assumptions required for a successful Sybil attack [22], making it possible to have both open membership and secure Byzantine agreement.

We will conduct research on the following questions:

- Improve on the SCP algorithm. By randomizing the algorithm between ballots, we will achieve provably efficient termination even in the presence of malicious nodes.
- Develop a simple and efficient credential and policy replicated state machine on top of SCP to be used in ThinkTalk. Unlike turing-complete smart contracts, whose computational complexity must be metered in cryptocurrency, we will factor functionality such that the bulk of policy is implemented outside the blockchain, and only very simple abstractions like flags, audit records, and delegation need be created on-chain.
- Incorporate non-cryptocurrency related blockchain topics into Stanford's blockchain class, CS251.

5 Graphical Virtual Assistants

After testing Almond with users, we found that a dialog based interface is limited for some types of user interactions. Users wanted a GUI to be able to see notifications, interact with UI elements such as sliders, and view images. We can give users much better access to their personal web if their frequently used high-level ThingTalk commands are turned into widgets with GUIs automatically. We have created such a prototype system, called Brassau [24], and the results of five automatically generated widgets are shown in Figure 4.

Figure 5: Crowd research [78] is a crowdsourcing technique that has enabled access to research experiences for over 1,500 people from 62 countries. Participants achieve upward educational mobility while creating research systems and co-authoring papers at top-tier ACM venues.

We want our automatically generated GUI to be not just functional, but aesthetically pleasing as well. We propose to create ImagineNet, a neural network to generate aesthetically pleasing GUIs. Our technique is based on neural style transfer, which has previously been proposed to copy the style of a painting onto a natural scene. When applied to GUIs, however, style transfer renders GUIs illegible as the graphical elements lack definition. We found that the style as captured by the original formulation is missing a key component, which is the way structure is conveyed in the original painting. By including this component in the style transfer, the objects in the resulting transformation remain well defined. This technique allows us to leverage the large collection of masterpieces to create aesthetically pleasing GUI.

The results show that when a user specifies their programs using natural language, we are able to automatically produce a graphical user interface for them to interact with their program. Without the user having to specify any extra additional information, the produced GUI is well designed and aesthetically interesting. We plan to continue to use a deep-learning approach to improve our GUI layout, and to create an end-to-end tool for generating great designs. Aesthetics is seldom considered in the creation of decentralized systems, but we believe that this needs to be changed if we wish to attract users to this platform.

6 Experimentation and Project Validation Plan

We propose three experiments to test if this technology can be used by (1) consumers to share their internet resources; (2) researchers to collect and protect personal data in big data analytics projects; (3) healthcare to share EHR (Electronic Health Records).

6.1 Access Control for Consumers

It is challenging to perform virtual assistant experiments because they are predicated on having a lot of data: a well populated Thingpedia, real-life virtual assistant commands, and natural-language utterances for training the semantic parser. Amazon has reportedly hired over 1000 engineers to develop Alexa. This problem is representative of the difficulty increating, developing and deploying distributed systems in general. Research topics such as cryptographic private aggregation techniques cannot be tested in the wild because companies are not interested in helping users share, say, their locations without learning about where they are.

A distributed computing ecosystem needs a distributed experimental testbed. We propose to reach out to high-school and college students to create a large-scale research platform for advancing social good with distributed computing. In this way, we give students a chance to get excited and learn about technology, participate in cutting-edge research, and learn about the importance of privacy. Our first research topic is to validate if federated virtual assistants can be used to support fine-grain sharing across all the internet services, while preserving privacy.

To achieve this, we will build upon our previous work on crowd research [78]. Crowd research (Figure 5) is a crowdsourcing technique for conducting high-quality remote research with distributed participants with varied backgrounds. So far, crowd research has engaged over 1,500 volunteers from 62 countries worldwide in the development of top-tier research. 74% of crowd research participants thus far have been from institutions with low access to research opportunities, and many have gone on to top-tier institutions afterward.

The technique utilizes an iterative cycle of open contribution, synchronous collaboration, and peer assessment to coordinate distributed efforts. Crowd research projects proceed under the leadership of a PI; projects thus far have spanned human-computer interaction, computer vision, and computational social science. Our goal in this project is to increase the scale by an order of magnitude by reaching out to historically black colleges in the United States and other under-represented groups, recruiting them with virtual assistant development as the subject of research. We hope this will help create a much needed testbed for distributed, privacy-oriented algorithms and software.

From our preliminary user study in access control with our prototype, we hypothesize that we can use natural language to support many sharing use cases of interest to consumers. We will share with students the vision, the technology, and keep them informed of the progress they help to make. It is a project where students of different technical levels can participate. They can be a test subject, they can suggest use cases, contribute natural language utterances, ThingTalk commands, and Thingpedia entries, improve the natural language semantic parser, etc. They can help recruit participants, design and carry out surveys, and disseminate information about the project.

6.2 Research on Hybrid Physical+Digital Spaces

To test if Almond can help researchers protect and collect personal data in research projects, we are collaborating with a cross-disciplinary team in Civil Engineering, Computer Science, Education, Psychology, Law in their Hybrid Physical+Digital Spaces (HPDS) project.

The HPDS project aims to develop a building information platform that promotes both environmental sustainability and occupant wellness and that can be easily employed in building design and management. This platform will use information technology and readily modifiable, environmentally sustainable building features to adapt buildings for occupant performance and wellbeing while encouraging sustainable behaviors, both automatically and by encouraging occupant intervention.

The proposal is to develop pattern detection sensors and software that measure occupant behavior, cognition, emotion, relations, and physiology, as well as determine associations between building qualities and occupant wellbeing and behavior. Leveraging these insights, the plan is to create a set of digital and physical adaptations and rigorously test their effects on occupants.

This research calls for collecting information from sensors on personal devices (smartphones, smartwatches, and fitness bands) to measure step counts and other physical activity, heart rate variability (an indicator of stress), and vocal prosody (indicators of mood). Further, smartphone usage logs will be used to detect sleep quality and alertness; emails and SMS messages will be analyzed to detect cognitive performance and mood.

Almond provides an infrastructure that makes it easy to run different experiments and provides privacy and transparency. We propose using an *admin* virtual assistant that is responsible for running the experiment and collecting data, and that each user will run a personal virtual assistant to gather data. All the primitive operations for collecting data

and submitting data will be entered in Thingpedia. The admin can issue data requests to all the participant's virtual assistants such as the daily phone usage or steps taken; they can ask the assistants to report the sentiment analysis results of the SMS messages to assess the user's mood rather than demanding the raw highly personal communication data. In addition, the admin can monitor the progress of self reports and issue commands like "*Remind the participant for their mood at 8pm every day, if they have not submitted the information*" to prompt users for input.

This architecture lets the experiment designer add experiments easily by entering the new APIs into Thingpedia, and sending additional requests to the participants. We will experiment with automatic GUI generation to improve the interface for self reports and to give users visual reminders.

6.3 Data Sharing in Health Care

To advance our understanding of how decentralized ThingTalk contracts can be used in important domains like health care, we are partnering with HTC Healthcare and a blockchain team in Computer Science at National Taiwan University in two health data sharing experiments.

Emergency room EHR. In an Emergency Room (ER) setting, there is tremendous pressure to make rapid decisions with complete information. With CDC and local hospitals, HTC is developing an ER-EHR system, through which an ER can access a patients distributed EHRs in real time to perform quality care. Such access consent can be formulated as a smart contract stored on Almonds blockchain. In the contract, a patient makes an agreement with his hospitals to keep his EHRs, and the patient further delegates the access right of his records to any ER services. In the case when the patient is incapacitated, the contract can be located on the blockchain and executed to provide the ER with the EHRs retrieved from the hospitals listed on the contract. This research will be carried in partnership with Taiwan CDC and local hospitals.

Medication Adherence Management Poor medication adherence contributes to inadequate symptom control, increased use of health care resources, higher medical costs, and markedly higher mortality rates. Working with Machay Memorial Hospital, HTC launched a medication management mobile app in 2016. However, the app usage is low as most patients do not routinely enter their data. We plan to integrate with the medication management app a smart pill tracker, an IoT device that tracks when medicine is taken. Actions such as relapse reporting and over-dosage warning can be specified through ThingTalk and a contract written with recipients specified. IoT devices that monitor vital signs such as blood pressure, blood oxygen, heart rate, and temperature may subsequently be added to gauge medication effectiveness and side effects.

This research will be carried in partnership with Machay Memorial Hospital, Taipei.

7 Results from Prior NSF Support

Lam: NSF Expedition on "Programmable Open Mobile Internet (POMI) 2020", NSF Award 0832820, $10,000,000, August 15, 2008–November 30,2014.

Intellectual merit: Pioneered the software-defined network. Developed an open social OS based on a novel distributed semantic file system, where users can keep their data in the repository of their choice.

Broader impacts: The industry is embracing SDN and it is projected to be $150B industry and open source is thriving in SDN. And SDN is also one of the hottest topics in research conferences. The 5G architecture is all based on SDN principles.

Bernstein: NSF CAREER award "CAREER: Enabling expert crowdsourcing via coordination, targeted contribution and education", IIS-1351131, $550,000, January 1, 2014 through December 31, 2018.

Intellectual merit: The CAREER grant is supporting PI Bernstein's work on computationally-guided expert crowdsourcing. The current proposal is complementary in that it focuses on creating an open community of contributors, rather than paid crowd work. The CAREER has led to the development of "flash teams" and "flash organizations" of crowd experts, which won Best Paper awards at UIST 2014 [72] and CHI 2017 [79]. Other contributions include novel microtask crowdsourcing techniques [73, 85, 25, 44, 15, 20, 81, 14, 47, 43], educational support for crowd workers [77], and social scientific studies of crowd work [2, 31, 34, 16]. The award does not overlap with the projects described in this proposal.

Broader impacts: Graduate students who participated in projects on the grant have gone on to become researchers at Adobe Research and Facebook Data Science, as well as in industry at B12. The grant has also been used to launch public tools that support crowd workers, as well as public education materials.

Landay: NSF IIS funded project Interaction Economics: Instruments that Measure Social-Computational Systems (NSF Award #IIS-1110965, $766,000, September 2011-February 2017)

Intellectual merit: Measured utility through online labor markets and utilize this in human-computer interaction and design [51]; developed tools to allow measuring utility via online experiments and use these measured utility values to, for example, summarize the subjective qualities of particular user interfaces and study fundamental labor economics issues.

Broader impacts: Determined the attention needed to eliminate unconscious discrimination in hiring using tools developed in this project.

Mazières: Award 1040190, $ 427,602, 09/01/2010–08/31/2015: FIA: Collaborative Research: NEBULA: A Future Internet That Supports Trustworthy Cloud Computing.

Intellectual merit: The primary result of this work was to demonstrate the feasibility of building a global network in which packets may only be forwarded along a path if all organizations along the path consent to that path [62]. With this approach, changes to fundamental services such as inter-domain routing become analogous to changing the domain name system (DNS) today. Another significant result of the project was EyeQ, a hypervisor-based network fairness mechanism for multi-tenant datacenter networks [37, 38]. Finally, a third result was "tiny packet programs," or TPP–a highly restricted form of active networking that exports network switch ASIC features to smart packets that can be processed at line rate [36, 35]. TPP promises to enable far greater innovation in cloud datacenter networks, because hardware vendors can implement TPP without predicting which network features will be important in the future. Other results included tools for facilitating network debugging [29, 30].

Broader impacts: Ph.D. students supported on this project have gone on to influential positions in industry at Cisco and Andreesen Horrowitz.

Manning: NSF award IIS-1514268 $1,100,000.00. June 1, 2015–May 31, 2018; no cost extension to May 31, 2019. RI: Medium: Deep Understanding: Integrating Neural and Symbolic Models of Meaning. (Dan Jurafsky, PI; Christopher Manning; Percy Liang)

Intellectual merit: The results of our project have had a large impact on our ability to use combinations of neural and symbolic approaches to do semantic understanding of human language, including novel models of discourse coherence, models that combine text and networks, novel models of interpretability of embeddings and neural models, new ways of combining adversarial learning with text processing, and novel neural models for conversational dialogue.

Broader impacts: The project has trained 17 graduate students, postdocs, and undergraduates, including 6 women, who have received weekly mentoring from the PIs in various phases of the project, with training in research methodology, in career development, and in the research content described above. One woman postdoc from this project, Yulia Tsvetkov, has now begun her faculty career at CMU. Three graduate students have graduated: Will Hamilton is starting a faculty job at McGill this year. Raine Hoover received her MS and Jiwei Li received his PhD and both are now at startups. Finally, the undergraduate, Jon Gauthier, graduated and is now a PhD student at MIT.

We have described and distributed our algorithms in publications, talks, and open source software. Neural approaches to NLP are finding many commercial applications in task like translation, chatbots/dialogue, and summarization, and our research actively informs and is informed by this broader industry work.

Publications: Understanding neural models: [48, 42]; coreference: [18, 19]; summarization: [74]; neural language understanding: [10, 27, 28]; events: [32]; dialogue: [49, 60, 26]; neural sequence translation models: [52, 86].

Research products: The algorithms have been described in publications and described in talks to the community and in talks given by the PIs to computer science departments around the country. Several systems have been incorporated into the Stanford CoreNLP open source NLP software, which is widely used by research labs and companies around the country, including in particular Clark's statistical and neural coreference systems: `https://stanfordnlp.github.io/coref.html` and Muzny's quote annotator: `https://stanfordnlp.github.io/quote.html`. Several datasets have been made publicly available, including: The SCONE dataset for learning models of context-dependent executable semantics at: `https://nlp.stanford.edu/projects/scone/`; the network model of Reddit interaction at: `http://snap.stanford.edu/data/web-RedditNetworks.html`; and quote attribution data: `https://stanfordnlp.github.io/quote.html`.

8 Conclusion

Our goal is to transform mobile and ubiquitous computing by letting individuals program their virtual assistant in natural language. This research will create an open and interoperable web of services, IoT devices, and virtual assistants. This proposal aims to establish a commercially viable distributed virtual assistant ecosystem by crowdsourcing data and creating a starter user community. It will also have the impact of producing a generation

of students who understand the importance of privacy and new technologies for mobile and ubiquitious computing.

References

[1] If This Then That. `http://ifttt.com`, 2011.

[2] Ali Alkhatib, Michael S. Bernstein, and Margaret Levi. Examining crowd work and gig work through the historical lens of piecework. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 4599–4616, New York, NY, USA, 2017. ACM.

[3] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[4] Laura Banarescu, Claire Bonial, Shu Cai, Madalina Georgescu, Kira Griffitt, Ulf Hermjakob, Kevin Knight, Philipp Koehn, Martha Palmer, and Nathan Schneider. Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, 2013.

[5] Lujo Bauer, Jay Ligatti, and David Walker. Composing security policies with polymer. In *Proceedings of the 2005 ACM SIGPLAN Conference on Programming Language Design and Implementation*, PLDI '05, pages 305–314, New York, NY, USA, 2005. ACM.

[6] Ian Beltagy and Chris Quirk. Improved semantic parsers for if-then statements. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL-16)*, pages 726–736, 2016.

[7] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[8] Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. Semantic parsing on freebase from question-answer pairs. In *EMNLP*, pages 1533–1544, 2013.

[9] Jonathan Berant and Percy Liang. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL-14)*, pages 1415–1425, 2014.

[10] Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. A fast unified model for parsing and sentence understanding. In *Association for Computational Linguistics (ACL)*, 2016.

[11] Giovanni Campagna, Rakesh Ramesh, Silei Xu, Michael Fischer, and Monica S. Lam. Almond: The architecture of an open, crowdsourced, privacy-preserving, programmable virtual assistant. In *Proceedings of the 26th International Conference on World Wide Web - WWW '17*, pages 341–350, New York, New York, USA, 2017. ACM Press.

[12] Giovanni Campagna, Silei Xu, Rakesh Ramesh, Michael Fischer, and Monica S. Lam. Controlling fine-grain sharing in natural language with a virtual assistant. *Proceedings of the ACM on Interactive Mobile, Wearable and Ubiquitous Technologies (IMWUT)*, 2(3), 2018.

[13] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[14] Justin Cheng and Michael S Bernstein. Flock: Hybrid crowd-machine learning classifiers. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work & Social Computing*, pages 600–611. ACM, 2015.

[15] Justin Cheng, Jaime Teevan, and Michael S Bernstein. Measuring crowdsourcing effort with error-time curves. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1365–1374. ACM, 2015.

[16] Justin Cheng, Jaime Teevan, Shamsi T Iqbal, and Michael S Bernstein. Break it down: A comparison of macro-and microtasks. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 4061–4064. ACM, 2015.

[17] Kevin Clark, Minh-Thang Luong, Christopher D. Manning, and Quoc V. Le. Semi-supervised sequence modeling with cross-view training. In *EMNLP*, 2018.

[18] Kevin Clark and Christopher D. Manning. Deep reinforcement learning for mention-ranking coreference models. In *Empirical Methods on Natural Language Processing*, 2016.

[19] Kevin Clark and Christopher D. Manning. Improving coreference resolution by learning entity-level distributed representations. In *Association for Computational Linguistics (ACL)*, 2016.

[20] Çağatay Demiralp, Michael S Bernstein, and Jeffrey Heer. Learning perceptual kernels for visualization design. *IEEE transactions on visualization and computer graphics*, 20(12):1933–1942, 2014.

[21] Li Dong and Mirella Lapata. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL-16)*, pages 33–34, 2016.

[22] John R. Douceur. The Sybil attack. In *Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 251–260, March 2002.

[23] Cynthia Dwork and Moni Naor. Pricing via processing or combatting junk mail. In *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, pages 139–147, August 1992.

[24] Michael Fischer, Giovanni Campagna, Silei Xu, and Monica S. Lam. Brassau: Automatically generating graphical user interfaces for virtual assistants. In *Proceedings of the 20th International Conference on Human-Computer Interaction with Mobile Devices and Services (MobileHCI 2018)*, 2018.

[25] S S Gaikwad, D Morina, A Ginzberg, C Mullings, S Goyal, D Gamage, C Diemert, M Burton, S Zhou, M Whiting, K Ziulkoski, A Ballav, A Gilbee, S S Niranga, V Sehgal, J Lin, L Kristianto, J Regino, N Chhibber, D Majeti, S Sharma, K Mananova, D Dhakal, W Dai, V Purynova, S Sandeep, V Chandrakanthan, T Sarma, S Matin, A Nassar, R Nistala, A Stolzoff, K Milland, V Mathur, R Vaish, and M S Bernstein.

Boomerang: Rebounding the Consequences of Reputation Feedback on Crowdsourcing Platforms. In *Proceedings of the 29th Annual ACM Symposium on User Interface Software and Technology*, UIST '16, New York, NY, USA, 2016. ACM.

[26] Kelvin Guu, Panupong Pasupat, Evan Zheran Liu, and Percy Liang. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *Association of Computational Linguistics (ACL)*, 2017.

[27] William L Hamilton, Jure Leskovec, and Dan Jurafsky. Cultural shift or linguistic drift? comparing two computational measures of semantic change. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[28] William L. Hamilton, Jure Leskovec, and Dan Jurafsky. Diachronic Word Embeddings Reveal Statistical Laws of Semantic Change. In *Association for Computational Linguistics (ACL)*, 2016.

[29] Nikhil Handigol, Brandon Heller, Vimalkumar Jeyakumar, David Mazières, and Nick McKeown. Where is the debugger for my software-defined network? In *Workshop on Hot Topics in Software Defined Networking*, August 2012.

[30] Nikhil Handigol, Brandon Heller, Vimalkumar Jeyakumar, David Mazières, and Nick McKeown. I know what your packet did last hop: Using packet histories to troubleshoot networks. In *11th Symposium on Networked Systems Design and Implementation*, pages 71–85, Seattle, WA, April 2014.

[31] Kenji Hata, Ranjay Krishna, Li Fei-Fei, and Michael S Bernstein. A glimpse far into the future: Understanding long-term crowd worker quality. 2017.

[32] Ruihong Huang, Ignacio Cases, Dan Jurafsky, Cleo Condoravdi, and Ellen Riloff. Distinguishing past, on-going, and future events: The eventstatus corpus. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2016.

[33] Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. In *International Conference on Learning Representations (ICLR)*, 2018.

[34] Kazushi Ikeda and Michael S Bernstein. Pay it backward: Per-task payments on crowdsourcing platforms reduce productivity. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 4111–4121. ACM, 2016.

[35] Vimalkumar Jeyakumar, Mohammad Alizadeh, Yilong Geng, Changhoon Kim, and David Mazières. Millions of little minions: Using packets for low latency network programming and visibility. In *ACM SIGCOMM Conference*, pages 3–14, August 2014.

[36] Vimalkumar Jeyakumar, Mohammad Alizadeh, Changhoon Kim, and David Mazières. Tiny packet programs for low-latency network control and monitoring. In *12th ACM Workshop on Hot Topics in Networks*, College Park, MD, November 2013.

[37] Vimalkumar Jeyakumar, Mohammad Alizadeh, David Mazières, Balaji Prabhakar, and Changhoon Kim. EyeQ: Practical network performance isolation for the multi-tenant cloud. In *4th USENIX Workshop on Hot Topics in Cloud Computing*, June 2012.

[38] Vimalkumar Jeyakumar, Mohammad Alizadeh, David Mazières, Balaji Prabhakar, Changhoon Kim, and Albert Greenberg. EyeQ: Practical network performance isolation at the edge. In *Symposium on Networked Systems Design and Implementation*, April 2013.

[39] Robin Jia and Percy Liang. Data recombination for neural semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 12–22, 2016.

[40] Lu Jiang, Deyu Meng, Qian Zhao, Shiguang Shan, and Alexander G Hauptmann. Self-paced curriculum learning. In *AAAI*, volume 2, page 6, 2015.

[41] Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Regularizing very deep neural networks on corrupted labels. *arXiv preprint arXiv:1712.05055*, 2017.

[42] Urvashi Khandelwal, He He, Peng Qi, and Dan Jurafsky. Sharp nearby, fuzzy far away: How neural language models use context. In *Association for Computational Linguistics (ACL)*, 2018.

[43] Joy Kim, Sarah Sterman, Allegra Argent Beal Cohen, and Michael S Bernstein. Mechanical Novel: Crowdsourcing Complex Work through Revision. In *Proceedings of the 20th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '17, New York, NY, USA, 2017. ACM.

[44] Ranjay A Krishna, Kenji Hata, Stephanie Chen, Joshua Kravitz, David A Shamma, Li Fei-Fei, and Michael S Bernstein. Embracing Error to Enable Rapid Crowdsourcing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pages 3167–3179. ACM, 2016.

[45] Jayant Krishnamurthy, Pradeep Dasigi, and Matt Gardner. Neural semantic parsing with type constraints for semi-structured tables. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1516–1526, 2017.

[46] Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In *International Conference on Machine Learning*, pages 2879–2888, 2018.

[47] Walter S Lasecki, Juho Kim, Nick Rafter, Onkur Sen, Jeffrey P Bigham, and Michael S Bernstein. Apparition: Crowdsourced user interfaces that come to life as you sketch them. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 1925–1934. ACM, 2015.

[48] Jiwei Li, Minh-Thang Luong, Dan Jurafsky, and Eudard Hovy. When are tree structures necessary for deep learning of representations? In *Empirical Methods in Natural Language Processing (EMNLP)*, 2015.

[49] Jiwei Li, Will Monroe, Tianlin Shi, Sébastien Jean, Alan Ritter, and Dan Jurafsky. Adversarial learning for neural dialogue generation. In *Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[50] Chen Liang, Jonathan Berant, Quoc Le, Kenneth D. Forbus, and Ni Lao. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. *CoRR*, abs/1611.00020, 2016.

[51] Chang Liu, Xinyun Chen, Eui Chul Shin, Mingcheng Chen, and Dawn Song. Latent attention for if-then program synthesis. In *Advances in Neural Information Processing Systems*, pages 4574–4582, 2016.

[52] Minh-Thang Luong and Christopher D. Manning. Achieving open vocabulary neural machine translation with hybrid word-character models. In *Association for Computational Linguistics (ACL)*, Berlin, Germany, August 2016.

[53] Hamid Reza Maei, Csaba Szepesvári, Shalabh Bhatnagar, and Richard S Sutton. Toward off-policy learning control with function approximation. In *ICML*, pages 719–726, 2010.

[54] Roy A. Maxion and Robert W. Reeder. Improving user-interface dependability through mitigation of human error. *International Journal of Human-Computer Studies*, 63(1-2):25–50, jul 2005.

[55] David Mazières. The Stellar consensus protocol: A federated model for internet-level consensus. `https://www.stellar.org/papers/stellar-consensus-protocol.pdf`, February 2016.

[56] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843*, 2016.

[57] MobileCoin. Mobilecoin. https://www.mobilecoin.com/whitepaper-en.pdf, 2017.

[58] Tim Moses. Extensible access control markup language (xacml) version 2.0. *Oasis Standard*, 2005.

[59] Rémi Munos, Tom Stepleton, Anna Harutyunyan, and Marc Bellemare. Safe and efficient off-policy reinforcement learning. In *Advances in Neural Information Processing Systems*, pages 1054–1062, 2016.

[60] Grace Muzny, Michael Fang, Angel X. Chang, and Dan Jurafsky. A two-stage sieve approach for quote attribution. In *Proceedings of the European Chapter of the Association for Computational Linguistics (EACL)*, 2017.

[61] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system, 2008. `http://bitcoin.org/bitcoin.pdf`.

[62] Jad Naous, Michael Walfish, Antonio Nicolosi, David Mazières, Michael Miller, and Arun Seehra. Verifying and enforcing network paths with ICING. In *7th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*, December 2011.

[63] Yusuke Oda, Hiroyuki Fudaba, Graham Neubig, Hideaki Hata, Sakriani Sakti, Tomoki Toda, and Satoshi Nakamura. Learning to generate pseudo-code from source code using statistical machine translation (t). In *Automated Software Engineering (ASE), 2015 30th IEEE/ACM International Conference on*, pages 574–584. IEEE, 2015.

[64] Rafael Pass, Lior Seeman, and Abhi Shelat. Analysis of the blockchain protocol in asynchronous networks. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 643–673. Springer, 2017.

[65] Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53nd Annual Meeting of the Association for Computational Linguistics (ACL-15)*, pages 1470–1480, 2015.

[66] Michael Pust, Ulf Hermjakob, Kevin Knight, Daniel Marcu, and Jonathan May. Parsing english into abstract meaning representation using syntax-based machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1143–1154, 2015.

[67] Chris Quirk, Raymond Mooney, and Michel Galley. Language to code: Learning semantic parsers for if-this-then-that recipes. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics (ACL-15)*, pages 878–888, 2015.

[68] Maxim Rabinovich, Mitchell Stern, and Dan Klein. Abstract syntax networks for code generation and semantic parsing. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1139–1149, 2017.

[69] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.

[70] Alexander Ratner, Stephen H. Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. Snorkel: Rapid training data creation with weak supervision. *Proc. VLDB Endow.*, 11(3):269–282, November 2017.

[71] Robert W. Reeder, Lujo Bauer, Lorrie Faith Cranor, Michael K. Reiter, Kelli Bacon, Keisha How, and Heather Strong. Expandable grids for visualizing and authoring computer security policies. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1473–1482, New York, NY, USA, 2008. ACM.

[72] Daniela Retelny, Sébastien Robaszkiewicz, Alexandra To, Walter S Lasecki, Jay Patel, Negar Rahmati, Tulsee Doshi, Melissa Valentine, and Michael S Bernstein. Expert crowdsourcing with flash teams. In *Proceedings of the 27th annual ACM symposium on User interface software and technology*, pages 75–85. ACM, 2014.

[73] Niloufar Salehi, Andrew McCabe, Melissa Valentine, and Michael Bernstein. Huddler: Convening stable and familiar crowd teams despite unpredictable availability. In *Proceedings of the 2017 ACM Conference on Computer Supported Cooperative Work and Social Computing*, CSCW '17, pages 1700–1713, New York, NY, USA, 2017. ACM.

[74] Abigail See, Minh-Thang Luong, and Christopher D. Manning. Compression of neural machine translation models via pruning. In *Computational Natural Language Learning (CoNLL)*, 2016.

[75] D. K. Smetters and Nathan Good. How users use access control. In *Proceedings of the 5th Symposium on Usable Privacy and Security*, SOUPS '09, pages 15:1–15:12, New York, NY, USA, 2009. ACM.

[76] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[77] Ryo Suzuki, Niloufar Salehi, Michelle S. Lam, Juan C. Marroquin, and Michael S. Bernstein. Atelier: Repurposing Expert Crowdsourcing Tasks as Micro-internships. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, New York, NY, USA, 2016. ACM.

[78] Rajan Vaish, Snehalkumar (Neil) S. Gaikwad, Geza Kovacs, Andreas Veit, Ranjay Krishna, Imanol Arrieta Ibarra, Camelia Simoiu, Michael Wilber, Serge Belongie, Sharad Goel, James Davis, and Michael S. Bernstein. Crowd research: Open and scalable university laboratories. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pages 829–843, New York, NY, USA, 2017. ACM.

[79] Melissa A Valentine, Daniela Retelny, Alexandra To, Negar Rahmati, Tulsee Doshi, and Michael S Bernstein. Flash organizations: Crowdsourcing complex work by structuring crowds as organizations. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, pages 3523–3537. ACM, 2017.

[80] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[81] Vasilis Verroios and Michael S Bernstein. Context trees: Crowdsourcing global understanding from local views. In *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014.

[82] Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700, 2015.

[83] Sida I. Wang, Samuel Ginn, Percy Liang, and Christopher D. Manning. Naturalizing a programming language via interactive learning. *CoRR*, abs/1704.06956, 2017.

[84] K. Werling, A. Chaganty, P. Liang, and C. Manning. On-the-job learning with Bayesian decision theory. In *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[85] Mark E Whiting, Dilrukshi Gamage, Aaron Gilbee, Snehal Gaikwad, Shirish Goyal, Alipta Ballav, Dinesh Majeti, Nalin Chhibber, Freddie Vargus, Teo Moura, Angela Richmond Fuller, Varshine Chandrakanthan, Gabriel Bayomi Tinoco Kalejaiye, Tejas Seshadri Sarma, Yoni Dayan, Adam Ginzberg, Mohammed Hashim Kambal, Kristy Milland, Sayna Parsi, Catherine A Mullings, Henrique Orefice, Sekandar Matin, Vibhor Sehgal, Sharon Zhou, Akshansh Sinha, Jeff Regino, Rajan Vaish, and Michael S Bernstein. Crowd Guilds: Worker-led Reputation and Feedback on Crowdsourcing Platforms. In *Proceedings of the 20th ACM Conference on Computer Supported Cooperative Work {&} Social Computing*, 2016.

[86] Joern Wuebker, Spence Green, Saša Hasan John DeNero, and Minh-Thang Luong. Models and inference for prefix-constrained machine translation. In *Association for Computational Linguistics (ACL)*, Berlin, Germany, August 2016.

[87] Chunyang Xiao, Marc Dymetman, and Claire Gardent. Sequence-based structured prediction for semantic parsing. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1341–1350, 2016.

[88] Xiaojun Xu, Chang Liu, and Dawn Song. Sqlnet: Generating structured queries from natural language without reinforcement learning. *arXiv preprint arXiv:1711.04436*, 2017.

[89] Yi Yang, Wen-tau Yih, and Christopher Meek. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2013–2018, 2015.

[90] Pengcheng Yin and Graham Neubig. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 440–450, 2017.

[91] Alexander Yip, Xi Wang, Nickolai Zeldovich, and M. Frans Kaashoek. Improving application security with data flow assertions. In *Proceedings of the ACM SIGOPS 22Nd Symposium on Operating Systems Principles*, SOSP '09, pages 291–304, New York, NY, USA, 2009. ACM.

[92] Eric Yuan and Jin Tong. Attributed based access control (abac) for web services. In *IEEE International Conference on Web Services (ICWS'05)*, page 569, July 2005.

[93] Victor Zhong, Caiming Xiong, and Richard Socher. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103*, 2017.