Sketching NLP: A Case Study of Exploring the Right Things To Design with Language Intelligence

Qian Yang
Carnegie Mellon University
yangqian@cmu.edu

Justin Cranshaw
Microsoft Research
justincr@microsoft.com

Saleema Amershi
Microsoft Research
samershi@microsoft.com

Shamsi T. Iqbal
Microsoft Research
shamsi@microsoft.com

Jaime Teevan
Microsoft Research
teevan@microsoft.com

ABSTRACT

This paper investigates how to sketch NLP-powered user experiences. Sketching is a cornerstone of design innovation. When sketching designers rapidly experiment with a number of abstract ideas using simple, tangible instruments such as drawings and paper prototypes. Sketching NLP-powered experiences, however, presents unique challenges. It can be hard, for example, to visualize abstract language interaction, or to ideate a broad range of technically feasible intelligent functionalities. Via a first-person account of our sketching process when designing intelligent writing assistance, we detail the challenges we encountered and describe emergent solutions such as a new format of wireframe for sketching language interactions and a new wizard-of-oz-based NLP rapid prototyping method. These findings highlight the importance of abstraction in sketching language interfaces and of desiging within the capabilities and limits of NLP systems.

KEYWORDS

User Experience, Artificial Intelligence, Writing Assistance, Natural Language Processing.

ACM Reference Format:
Qian Yang, Justin Cranshaw, Saleema Amershi, Shamsi T. Iqbal, and Jaime Teevan. 2019. Sketching NLP: A Case Study of Exploring the Right Things To Design with Language Intelligence. In *CHI Conference on Human Factors in Computing Systems Proceedings (CHI 2019), May 4–9, 2019, Glasgow, Scotland UK*. ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3290605.3300415

CHI 2019, May 4–9, 2019, Glasgow, Scotland UK

ACM ISBN 978-1-4503-5970-2/19/05…$15.00
https://doi.org/10.1145/3290605.3300415

1 INTRODUCTION

From conversational agents at home to opinion mining on social media, intelligent language interactions are hitting the mainstream. Powering these systems are Natural Language Processing (NLP) technologies. NLP makes it possible to computationally comprehend, interpret, and even generate human language, and enables novel and otherwise impossible modes of interactions. But little is known about how user experience (UX) design practice might instigate novel products and services that use NLP. Existing practice-focused UX design patterns, books and toolkits do not talk about NLP or language interaction design [12, 21, 32, 38, 40], and HCI research, while offering many valuable and creative NLP applications, typically does not report the ideation or prototyping process.

In this paper we provide a first-person account of our experience adding intelligent language functionality into a Word document editor in collaboration with a group of NLP researchers. We started by attempting to rapidly experiment with many tentative NLP design ideas and broadly explore how NLP might improve the authoring experience. We wanted to use storyboards, UI wireframes, paper prototypes, and other simple, tangible instruments to sketch out early design ideas and probe users' reactions [7, 14]. But we soon encountered unexpected challenges. Common sketching tools and techniques deal with tangible interactions and are ineffective at abstracting the experience of language or a conversation. A number of technical aspects of language intelligence further complicate its UX design. For example, data-driven interactions vary across users, adapt to different contexts, and evolve over time, and it can be difficult for designers to envision such a divergent courses of interactions or to visualize using traditional wireframes and prototypes [2, 13, 20, 43].

These challenges led us to explore how to sketch NLP-powered user experiences, and what sketching actually means in the context of intelligent language interactions. This work serves as a first step in addressing these complex research

questions. We took a Research through Design approach [45] to our project at hand. Our goal was twofold:

(1) Provide a rare, first-person account of the sketching process of a NLP-based product, as well as an articulation of the challenges we encountered. Until we can understand the process and the challenges, we cannot make it easier for other researchers and practitioners;
(2) Ground future investigations of NLP design and innovation support. Our reflection-in-action during this project provides some solutions to these challenges.

The remainder of this paper offers an intentional accounting of the intelligent text editor project. We identify five challenges that are central to sketching NLP in our practice. We also describe a set of instruments that became effective for our sketching: a new form of wireframes that illustrated abstract language-interaction design ideas and became an effective boundary object; a set of NLP technical properties that are closely relevant to UX design; and a new prototyping method that enabled us to rapidly simulate various kinds of NLP errors. We discuss how these findings reveal underexplored research questions and new insights in innovating the UX of NLP systems.

2 RELATED WORK

We first provide an overview of UX sketching of technologies, and then turn to how NLP and machine learning (ML) have started to destabilize these conventions.

Sketching Technologies

In his book *Sketching User Experiences: Getting the Design Right and the Right Design*, Buxton defined sketching as any design process where the output is quick, plentiful, disposable, ambiguous and with minimal detail [7]. Because HCI deals with issues that are difficult to capture using pen and paper (e.g., interactivity, temporality, affection), unlike other design fields, HCI employs a number of additional techniques and tools in sketching, such as visualizations, paper prototypes, and wizard-of-oz experiments [22].

This kind of sketching—the kind that most HCI research, including this paper, focuses on—is different from a traditional view that considers sketching as merely a way to *externalize* images that are already in the mind of the designer. Instead, HCI focuses on sketching as primarily a tool for thinking, a process of design inquiry [14]. Through sketching, designers reconfigure, amplify, or de-contextualize various factors of the technology to see what new design possibilities may emerge. They engage in reflective conversations with the technology as a design material [25, 37].

This creative process happens less readily with new or partially understood technologies. In these cases, researchers "tinker with" the technology to develop a tacit understanding of its capabilities and experiential possibilities, and then transfer their understanding to practitioners via sketches and design exemplars. For example, Moussette created a set of haptic sketches; a set of physical prototypes that embodied his felt understanding of what experiential potentials haptics possess [27]. For technologies that are more difficult to "tinker with," researchers facilitated sketching by providing abstractions of the technology's capabilities and experiential qualities. For example, to help designers sketch taste experiences, Obrist et al. created visualizations representing the temporal and affective characteristics of taste [30].

Sketching Language Technologies

When we speak of NLP technologies, we broadly refer to any computer manipulation of natural language, ranging from simply counting word frequencies, to giving meaningful responses to human utterances [3]. Some examples of modern canonical NLP problems are information retrieval, machine translation, dialogue systems, and question answering.

HCI research on NLP systems does not discuss the sketching process. While offering creative, user-centered systems, researchers in this area typically describe one design solution, followed by its implementation and subsequent user study evaluation [9, 26, 34].

An exception is the work on prototyping conversational AI. Researchers simulated system behaviors with wizard or rule-based simulators so as to rapidly explore many interaction possibilities [8, 11, 23, 39]. For example, prototyping tools for speech interfaces such as Suede [24] enabled designers to quickly test their conversation scripts in Wizard-of-Oz (WoZ) experiments. Unlike the aforementioned, common sketching methods, WoZ does not facilitate designers to experiment a technology's capabilities and limits; Instead, it frees designers from the technical complexities of NLP and facilitate experimentation on interactions. In this light, recent HCI work started to call for demystifying NLP [28], arguing that UX designers need to possess some technical understanding of NLP to be able to design with it [26, 35].

Sketching Machine Learning

Modern NLP technologies are often based on ML techniques, so in some cases existing knowledge about how to sketch ML systems may be applicable to sketching NLP interactions.

Many designers report challenges when working with ML proactively, especially in conceiving of an entirely new way of using ML to improve UX [13, 18, 41, 42]. We identify three themes in these challenges.

First, ML's technical complexity. It requires an unwieldy amount of data and effort to even estimate whether an interaction is technically feasible, or to create a functional prototype [13, 43]. This challenges fundamental UX mantras

like "*fail fast, fail often*" and "*the last thing that you should do when sketching an interactive system is to write code*" [7]. To address these challenges, some researchers speculated that ML would require a new kind of user-centered design process [4, 15].

Second, ML-powered UX adapts to different users and contexts, evolves over time, and can make bizarre errors hard to anticipate. Even if they know generally how ML works, UX practitioners found it difficult to visualize such divergent courses of interaction, or to ideate new experiences with such fluidity [13, 43].

Finally, designers in the industry needed data scientists as a proxy of the technology when sketching. However, data scientists can be a scarce resource for many design teams [18, 42]. Some designers also found it challenging to effectively collaborate with data scientists [13].

Our work attempts to bring these strands of related work together. We draw inspirations from established methods of UX sketching (e.g., abstract representation and playful experimentation of technology materials) and confront the challenges of sketching NLP and ML. To investigate "sketching NLP" is to restore the fast-and-dirty, playful, and iterative aspects of sketching into NLP's design process.

3 METHODOLOGY

We wanted to identify challenges of sketching NLP within the context of one specific project and share our learning. Below we first describe our research through design approach, then introduce our design project.

Research Through Design

We chose an Research through Design (RtD) approach because, in alignment with our goals, RtD underscores that design knowledge arises from, and in response to, concrete problems and situations [19, 36]. We first immersed ourselves in the concrete design problems of the project, and then offered an intentional accounting of the project to allow for objective reflections on procedural, pragmatic, and conceptual insights [17]. To achieve the methodological transparency needed for capturing our own design activities, we followed Bayazit's three-stage process [31]:

(1) *Knowledge elicitation in an unstructured and unanalyzed form.* Throughout the project, we HCI/design researchers in the team wrote project dairies and weekly summaries, documenting our design activities. We documented all regular project meeting and impromptu conversations (n=24), and how they affected our later design activities. The regular meetings took place among all HCI and NLP researchers in the project three times a week. Additionally, in the final weeks of the project, we conducted 14 formal interviews with 9 external NLP researchers. We recorded audio of these meetings, each lasting approximately one hour. This resulted in more than 36 pages of description of our own design thinking and activities, as well as 9 hours of interview recordings, documenting major conversations between design and NLP expertise.

(2) *Data analysis and interpretation.* After the project ended we performed a thematic analysis on the data collected to identify key instances of challenges and reflection-in-action that happened during the design process. We transcribed the meeting recordings, and reflected on whether and how they shaped the later design trajectory. Finally, we sought agreement on interpretations across project members.

(3) *Finding validation.* We presented the findings respectively to all project members as well as to external NLP researchers. They validated our interpretations of our design journey and understandings of NLP's capabilities.

Designing an Intelligent Text Editor

We are collaborating with a group of NLP researchers on designing intelligent functionality offerings in a Word document editor. Prior HCI research have utilized NLP for providing writing assistance in several ways, for example, suggesting next sentences as inspiration [9, 34].

Our design goal was to improve individual users' writing *experience*. This focus on *experience* means that we worked to understand how authors themselves want to be supported by machine intelligence and designed accordingly. Our focus was not on whether or how authors needed to write "better". Relatedly, we did not pre-identify the authors in need of assistance ("target user groups") either, because our very design task was to search for the authors in desperate need of assistance such that they would embrace the likely imperfect machine suggestions.

Research on designing ML has highlighted the importance of collaboration with data scientists [18, 42]. Therefore, we included 4 NLP researchers in our project team. One specializes in computational linguistics; the other three in language modeling and deep learning. Later in the project, as we started to design with techniques that are not typically used in writing assistance, we interviewed other NLP researchers in our organization. Their expertise ranges from conversational agents, search, machine translation and more.

4 FINDINGS

This section first provides an overview of the design process of the intelligent text editor, then details five challenges we encountered when sketching, as well as the solutions emergent in our reflection in action.

Overview

We followed a traditional user-centered design process [10]. First, we conducted a field study of 18 participants to understand their needs and wants in writing. We invited them to

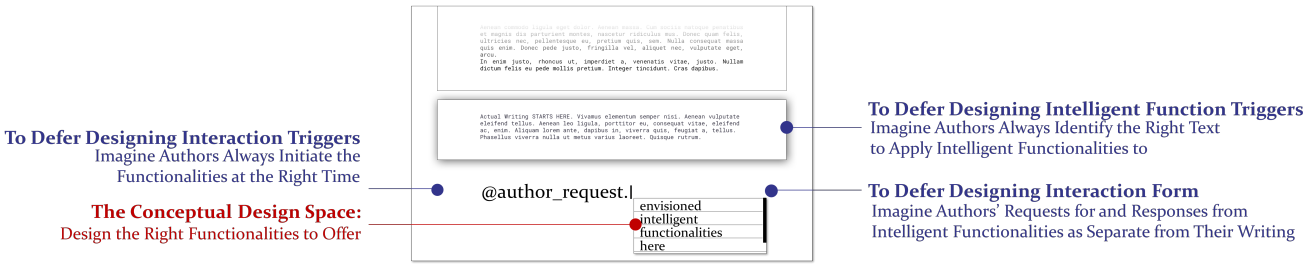

Figure 1: "The notebooks" as a form of wireframe for sketching abstract, language-based interactions. It bounded us to focus on envisioning "the right thing to design" and deferred detailed interaction design tasks.

record their screen for 40 minutes as they were writing one of their own documents. We then conducted an 1-hour interview. Participants walked us through their thought process in writing during the time of screen recording and discussed their unmet needs and wants toward writing assistance.

Next, we envisioned many intelligent functionalities that users would be likely to find valuable. In doing so, we encountered several challenges: (1) How to sketch language interactions abstractly? (2) How to design with data scientists without data at hand? (3) How to understand and stretch NLP's technical limits? (4) Within these limits, how to envision novel, less obvious applications of NLP?

After addressing these challenges, we proceeded with a small set of design ideas. For example, we envisioned an ask-your-reader function that compares an user's writing with their target venues', helping them account for readers' likely expectations. We created prototypes of these early ideas and tested them in a second user study. In this process, we encountered another challenge: (5) How to prototype an intelligently flawed UX?

How to Abstractly Sketch Language Interactions?

Early in the project, we wanted to focus on "designing the right thing" rather than making detailed interaction design choices. Surprisingly, untangling the two turned out to be a challenge.

Traditionally designers address this challenge by drawing storyboards. Storyboards capture the contexts and the holistic experiences of a macroscopic design idea, while dismissing its interface and interaction details. This doesn't work for language interactions; language as a form of interaction carries both the interface and the utility it manifests. We did not know how to sketch one without the other, nor did we know how to sketch language interactions abstractly.

Adding on to this challenge is the transient nature of authors' need for assistance. This entails two complex design tasks: Designing the trigger of the NLP function such that it applies to a right part of the author's writing; and designing the trigger of the interaction so that the authors only interact with the NLP function when they want to. While these two are interaction details, we found them difficult to ignore because they are significant mediators of the perceived value of our designs. As a result, the looming question "*Are we designing another Clippy?*" frequently derailed our discussion of a design idea from its utility to its interaction details.

How to stay abstract when sketching NLP? Our solution to this challenge was a new format of wireframe, namely *the notebook* (Figure 1). It is an abstract representation of the moment when a user requests intelligent assistance: against the backdrop of what has been written in the document, the writer selects a part of the text and requests an intelligent assistance function from a drop-down menu. The user may additionally specify whether the assistance function should overwrite their writing, or display the response elsewhere as a reference. The user may also specify other information as additional inputs into the intelligent function.

Notably, the notebook does *not* depict a design idea. It is highly unlikely that our final design will require users to type their requests via such computer-program-like commands. Rather, it is an instrument that facilitates our sketching.

The notebook bounded our design problem at hand, that is, *assume that users have made the right judgments on what intelligent function to apply to which text, and they are willing to make great efforts to make the function happen, what functions can we offer?* The notebook prompted us to freely imagine valuable functionality offerings while deferring other detailed design choices.

Before we created the notebook, we had drawn many different representations of language interactions; some were literal and visually resembled a text editor, others abstract and conceptual. Only the notebook caught on and was later organically adopted by the whole team.

Upon later reflection, the notebook caught on because it embodied our initial stances in designing intelligent writing assistance: we wanted machine intelligence to support authors' writing as a process, not a resulting product [29]; we refused to assume that authors need or want help in writing,

Axes of NLP Capabilities	What It Takes to Extend the Capabilities
Text Length. Words are easier to computationally process than phrases, than sentences, than paragraphs and finally a document. Knowledge beyond the written texts (e.g., common sense) is the most difficult to process.	Escalating an intelligent functionality, for example, from word level to sentence level, requires building **new models** and "*there is no guarantee how well it will work.*"
Classification - Comprehension - Generation Assessing or classifying a piece of text is easier than comprehending it (i.e. pinpointing the problem in this text), than text generation.	Escalating intelligent functionality along this axes requires building a **new model** and collecting additional or new labeled **datasets** for building it.
[classification only] Efforts Needed to Label Training data. If it is *easy* and *fast* for humans to make an *agreed-upon* judgment of its class, curating a labeled dataset for this intelligent classifier is likely to be practical.	Time, effort and often financial costs. Medicated by the amount of labeled data needed.
Likelihood to find training data that resemble the envisioned input/output pairs. We cannot presume a model that performs well on a benchmark research corpus would naturally perform in other texts.	Transferring or generalizing an existing model to a different corpus requires building **new models** and "*there is no guarantee how well it will work.*"

Table 1: NLP Capabilities, Limits and What It Takes to Extend the Capabilities

hence the intelligent assistance is passive by default and only became proactive upon user request.

Embodying these stances, the notebook became "*a very effective problem framing*" (designer diary, week 3) for us. For the rest of the design process, the notebook framework evolved every time when we reframed the design problem. For instance, after we had discovered in the user study that most participants outlined in the same document what they want to say before they worked to improve on how to say it, we included outlines as part of the notebook framework. These outlines externalize users' communicative goals and can serve as a valuable source for more situated and personalized interactions and functionality. Including the outline in the notebook suggests that 1) we can to imagine new intelligent design possibilities with outlines as a resource; 2) motivating authors to externalize their communicative goals will be one of our later interaction design goals.

How to Design with Data Scientists Without Data?

With the notebook framework, we started to ideate many writing assistance utilities that, based on our user study, users are likely to find useful. The first round of ideation generated 19 design ideas. To our great surprise, according to the NLP researchers, *none* of the ideas were promising from a technical feasibility perspective. These ideas "*need ten more years to make happen,*" they said, only half-joking.

Eager for more insights, we asked the NLP researchers: why are these designs technically unfeasible? Why does this functionality work in this research publication, but doesn't work for our design? What *is* feasible then? However, technical researchers were unable to answer these questions. They could not articulate NLP's technical limits with our abstract design ideas. "*It's difficult to say; It depends on data.*". "*The function you described is too abstract; I need to look at the data*". For example, when we asked *Would these two models you are building work for our users?*:

> Scientist 2: *Both models are sort of data agnostic. So as long as the data [users' writing] is somewhat to analogous to what we have now, it should, theoretically, translate very easily.*
> Scientist 5: *That's true for any model, right? So really, we don't know. We need to look at the data.*

In order to rapidly explore the design space, we could not afford to collect data before sketching. Data collection, prepossessing and exploration take up more than 80% of the total ML effort [44]. The question became: how can we partner with NLP scientists without a text corpus at hand?

After experimenting with numerous ways to explain our design ideas, we arrived at one boundary object [5] that effectively scaffolded our conversation with NLP researchers—that is, a more developed version of the notebook. This version of the notebook is projected on a Text Editor wireframe, resembling a user interface (Figure 2a). When we embedded our design ideas within the notebook framework, NLP researchers no longer asked for data and became able to engage in feasibility discussions about abstract design ideas.

Interpreted as a design problem framing for us, the notebook represents a language model for NLP researchers: when a user triggers an intelligent function to be applied to a selected snippet of text, the content and paratext of the Word document at that moment constitutes potential training data. The selected text is the model's runtime inputs. The envisioned function outputs are modeling goals.

The notebook became a shared representation and a means of translation between the two worlds of UX and NLP. It scaffolded our discussions with NLP researchers for the rest of the project. We described our design ideas by describing what they would look like on the notebook. NLP researchers then gave feedback on whether these are sufficient sources of data for building intelligence. They also proposed additional kinds of data that could boost model performance. Drawing on user study findings, we considered what additional data might be present or attainable through user interactions, and iterated on our designs. This process iterated smoothly and required no data collection or cleaning efforts.

How to Understand and Stretch Technical Limits?

> *You are really good at designing things we cannot build. We are good at making things that users don't use.* (NLP researchers 2 & 9, weeks 3 & 5)

Our first-round sketching produced design ideas that are uniformly beyond the limits of existing technical capabilities or existing datasets. It is worth noting that, when we envisioned these designs, we did not imagine NLP as a crystal ball. We drew our ideas from NLP literature; we intended to innovate writing assistance by amplifying or re-contextualizing these existing techniques. Below are two examples of our initial design ideas:

- Rephrasing the selected text in a more positive tone (seems possible based on existing work on style transfer between different sentiments);
- Identifying whether the selected text is logically coherent with its context (seems possible based on textual entailment analysis techniques [1, 16]);

How can we understand NLP's technical capabilities and limits from an UX perspective? Realistically, to what extent can we push these limits to enable novel designs? A set of technical boundaries became clear to us after many discussions with NLP researchers, through negotiating with them and iterating on our design ideas. We describe these boundaries via four measures of NLP's technical difficulty: text length, text classification-comprehension-generation, effort needed for labeling (for classification problems only), and likelihood to find training data that resemble the envisioned input/output pairs (Table 1). This set of measures enabled us to eventually find the intersection design space between what is valuable to users and what is technically feasible.

Taken together, the four axes depict an algorithmic approach to language processing that is quite different from typical authors'. Authors start writing with a big idea in mind, then scaffold its constituent supporting ideas, structure paragraphs, sentences, and so on. In contrast, language models first parse a sub-word, then a word, then a phrase, a sentence, and so on. Comprehending the big idea underlying the written texts is considered "*the holy grail of NLP research*"; It is so challenging that "*when we figure this out, the whole field of NLP would have become a solved problem*".

Because of this difference, our early design ambition to support the *experience* of writing – the ongoing process of translating the big idea to texts – has unknowingly led us toward technically challenging designs. In order to generate any implementable design ideas, we cannot naively project authors' goals onto intelligent functions. We need to identify technically achievable intermediate steps toward their goals.

By quickly assessing the four measures we were able to "gut-check" the feasibility of new design ideas and weigh their promise of UX gain against the technical effort they required. For example, the "efforts required for labeling" stands as a reminder that we cannot presume many seemingly trivial skills that authors master can be easily automated. This is because "*humans generally don't have a sense of classification or labelling. It's unclear when they used common sense, biases, or their world knowledge. But algorithms require labelling (to learn these)*" (Designer note, week 7).

Take the aforementioned design idea "paraphrasing in a more positive tone" as an another example. It is technically out of research because of its low likelihood to find training data that resemble the envisioned input/output pairs (axis 4), as an NLP researcher explained:

> *We don't have the training data at the scale that we need. [If] Say: here is a mangled version of the sentence, here is a cleaned up, positive version of it, times 50 million pairs. If we had that, it (building the intelligence) will be trivial. But it's very unlikely to have this kind of data.*

How to Envision Less Obvious NLP Applications?

With some understanding of the technical limits, we sketched new design ideas that NLP researchers consider as implementable, or "*at least have a clear direction to work from*". However, we found our own design ideas rather unsatisfying. State-of-art NLP can assess or classify writings, but cannot easily pinpoint causes of the problems or generate suggestions on how to improve. This seemed a textbook recipe of a frustrating user experience. As a result, most of our design ideas are word or phrase-level alternations, "*many variations of auto-complete and auto-correct basically*".

How can we envision technically feasible NLP designs that have not been imagined before? How can we expand this narrow intersection between what is value to users and what can be built?

We addressed these questions with a classic designerly approach: taking designing writing assistance as a wicked problem [6] and seeking reframings. Our initial problem

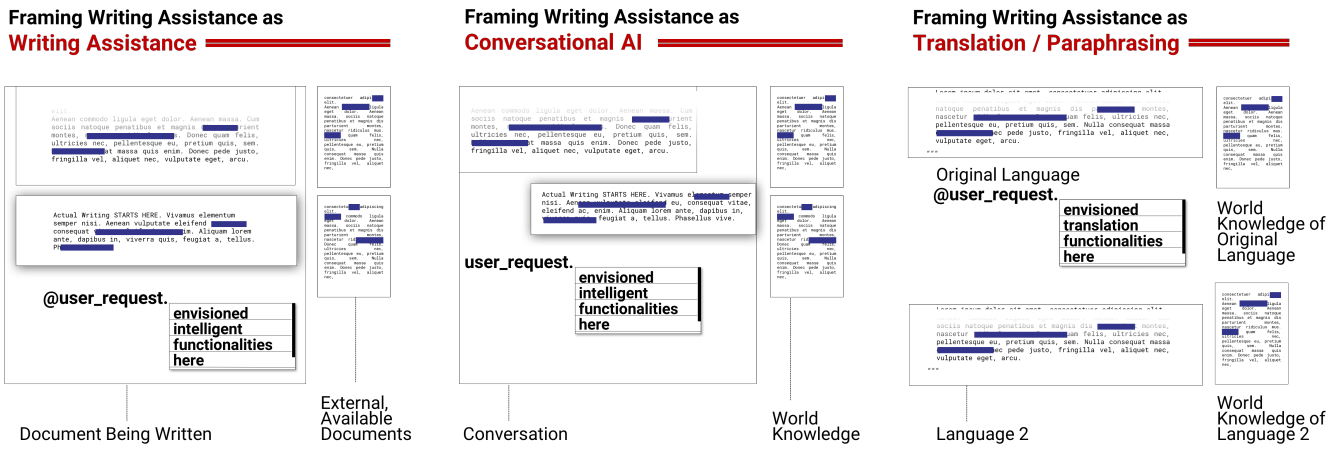

Figure 2: Left: A developed version of the Notebook (Figure1), used as boundary object between HCI and NLP researchers. "Contexts" that can help inform intelligent function outputs are marked blue. Middle and Right: Reframing the problem of designing writing assistance as other canonical NLP technical problems. This expands our design space to the intersection between what authors want and what existing NLP capabilities can do.

framing underlying the original notebook wireframe is that the writing assistance functionality comprehends and generates texts as the author writes. As described in the last section, this framing is prone to technically challenging design solutions. Authors are inherently better than algorithms at comprehending their unfinished writing and at predicting their unformed ideas, which is a wicked problem that cannot be accurately modeled.

We reframed the relationship between authors and writing assistants as other canonical NLP problems, specifically human-AI conversations, information retrieval/search, and question answering. (Figure 2 illustrates how we abstracted users' writing into many text components, and then mapped them onto other NLP problems.) Each of these alternative framings exposed us to a new set of technical capabilities in a different NLP sub-domain. We demonstrate how these new framings broadened our design space through the two design ideas they spurred.

A Context-aware, Rhetorical Search Function. Search is a relatively matured NLP sub-domain. Conceptualizing an intelligent writing as a search experience included many near-future design possibilities into our design space. Instead of algorithmically generating responses to authors, a search function can simply retrieve relevant writings to author requests. It does not require large datasets or to collect labels.

We started to ideate intelligent search tools that users are likely to find useful. We observed and interviewed participants in our user studies how they sought for information during writing. We noticed that, prior to writing, most participants outlined and organized their thoughts in the forms of bullet lists, tables, and even drawings. Yet many struggled with translating these organizations of thought into a linear, natural flow. Authors therefore searched online for rhetorical structures that they could borrow, for example, one participant, P7, Google'd "*[quotation mark][comma] in comparison to [quotation mark]*" to search for examples of connecting ideas of contrastive relationship. However, this carefully constructed search query does not actually work as he expects. Modern search engines expand and rewrite search queries based on similar searches, user search history and so on, optimizing for finding content that is relevant to the query topically rather than rhetorically. Participants like P7 could not find the writing examples he sought.

To support this unmet need we sketched a rhetorical search function. It searches the web for text that is similar in language *structure* and *composition* to the author's query. It takes into considerations the topic and style of the authors' current document to optimize the relevance of the search results. When an author selects a part of their bullet-list outline (e.g., "Issue A: good/bad examples") the writing assistance tool then searches for contents online that contain contrastive examples relevant to Issue A and sorts by different ways of transitioning between them. Rather than optimizing for topic relevance, this search functionality helps users find better ways to organize and connect their thoughts. It can be implemented with readily available search techniques.

An Asking-Your-Reader Function. Another useful reframing is conversational AI, that is, reframing the role of writing assistance as a conversation partner of the author. This reframing asserted new design questions: Whom would authors like

to talk to during writing and for what purpose? What information can conversational assistance offer? These design questions naturally expanded our design space beyond "helping writers verbally what they have in mind", and prompted us to imagine utilities NLP can provide as an outsider to the author's world.

With these questions in mind, we asked participants whom and how they asked for feedback while writing. We found they often picked those who are close to their target readers as their "beta-readers". Participants worked to translate their often egocentric writing into a style that meets the expectations and needs of their target readers. Many read other documents from their target venue to infer the expected length, lexical complexity, or level of detail that they should write in.

We see this as an excellent opportunity for NLP technologies to help authors, as algorithms are good at rapidly summarizing or characterizing a sizable collection of documents. We therefore designed an "ask your reader" function. It mines documents from an author-identified venue. The author can request insights about these documents or make comparisons between their own writing against it. For instance, "Am I writing too formally?" "How long is a typical introduction section in [venue]?" In this design writing assistance does not assist authors in writing *per se*, but supports their communications with their target readers.

Through these two design exemplars, we demonstrated that design problem reframing helped us envision novel forms and functions of existing NLP techniques, expanding the design space of technically feasible writing assistance.

How to Prototype an Intelligently Flawed UX?

We generated a prioritized set of intelligent function offering ideas informed by the our initial user study and bounded by existing NLP capabilities. We then turned to building a low-fidelity prototype to rapidly experiment on these ideas with users. We wanted to test the ideal behavior of our envisioned intelligent assistance with users to see if we were pursing the right design direction; We also wanted to probe users' reactions to a more realistic range of NLP-powered behaviors and errors to account for these reactions and expectations when improving on our design.

But how can we realistically simulate NLP's errors without spending months fully-functioning systems? We experimented with a series of prototyping methods in collaboration with 9 NLP researchers.

Failed Attempts. Wizard-of-Oz is a common way to prototype NLP. However, we learned early in the project that algorithms make errors that are unlike humans'. For example, even state-of-art NLP can fail in text comprehension or generation because of a lack of common sense knowledge. To enable wizards to simulate NLP behaviors we need to prevent them from accessing their common sense, which is extremely difficult.

Beyond unrestricted wizard-of-oz experiments we also considered using a rule-based simulator to prototype intelligent input/outputs. We encoded some rules (e.g., decision trees) into the prototype. However, NLP researchers pointed out that an rule-based simulator at best could behave as good as a rudimentary, rule-based ML system. Their capabilities are far behind state-of-art technology.

We attempted to use publicly available, pre-built NLP models to power the prototype, yet failed for similar reasons. These application-agnostic toolkits only include the most matured kinds of NLP technologies. Their level of sophistication is not close to state-of-art NLP technologies either.

We then experimented building simple ML/NLP models to simulate modern NLP's behaviors, using publicly available datasets and off-the-shelf toolkits such as AllenNLP [1]. This failed for a number of reasons: First, preparing the datasets is itself a daunting task. Integrating NLP toolkits that were built upon different platforms, in different programming languages into one prototype further complicates the prototype building. Finally, after we built a simple model, its performance was just not good enough for an user study. When an algorithm-generated sentence makes sense but reads awkward, the awkwardness washed out all other user "experiences". The sentence reads simply, awkward.

Successful Attempts. We prototyped our design ideas with an alternative WoZ method. For each NLP-powered interaction, we designed a different hybrid of WoZ and off-the-shelf toolkits to best simulate the likely errors. The design of each hybrid mimics the likely architecture of its underlying NLP system. This method highlights that different intelligent features produce different kinds of errors, each of which can have different UX consequences. In order to better capture these consequences, we need to better orchestrate WoZ behaviors to simulate NLP behaviors. Below are few examples:

(1) Simulating context-awareness with machine translators: Most of our designs take authors' writing as an input and provide context-aware, personalized writing suggestions. Our prototype takes in authors' writing in English, translates to a foreign language using existing machine translation services, and translates back to English. The output of this process is used as the "context" detected by intelligent writing assistance.

Language technologies could fail at extracting relevant contexts from authors' writing. There instead of taking their writing into full account, our prototype removes parts of it that algorithms could not easily comprehend through the two round of translations. We simply used online machine

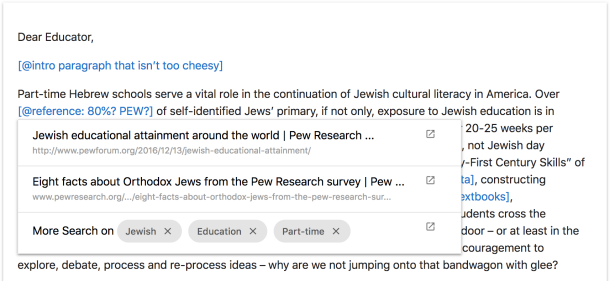

Figure 3: This prototype interface is a simple text editor. At any time of their writing, users type @ to signal the start of an intelligent function request and Enter to end. When they click on a request, intelligent assistance pops out. This prototype probes users' needs and wants for writing assistance, and their reactions to the simulated intelligent responses.

translation services to build this prototype. To simulate context awareness of lower quality, we selected the second, third, fourth ranked translations that the translator provided, such that more context and meaning were lost in the translation.

(2) For intelligent functions that assess or categorize author's writing, simply simulate the results based on what kind of errors is more likely to happen (precision, recall, etc.) and which classes are more error-prone.

(3) Simulating generative writing assistance with a multi-wizard simulator. When an user study participant request a piece of machine-generated text, multiple wizards and a meta classifier work in the background; each produces a response that excels at one aspect of the text generation. For example, one wizard produces a topically relevant response. The second wizard takes charge of the response fluency; the third focuses on the coherence between the generated text and the writers; the forth adds domain knowledge to the response, the fifth generates random words, and so on. The meta classifier assembles all wizard's responses into the final response returned to the user.

We designed these wizards' roles based on common models of generative neural networks. We simulate different kinds/degrees of generative errors by tuning the weights that each wizard carry. As such, we could probe user study participants on their preference among various designs of a generative writing assistance as well as their error tolerance. These user study results can inform our future iterations of sketching and design refinement.

Summary of Emergent Solutions

We have detailed the five challenges we encountered. In the process, three instruments became useful to us.

- The notebooks. The notebooks are a set of wireframes that illustrate abstract language interaction design ideas. The notebooks ended up playing three important roles in our sketching process: It enables us to externalize and communicate early-stage, abstract, language-based design ideas (challenges 1,4). It also served as a boundary object between designers and data scientists (challenge 2) and between designers and users (challenge 5), which enabled conversations among UX, design and NLP expertise;
- A set of NLP properties that are closely relevant to UX design, including "axes of NLP capabilities" and "what it takes to extend them". Understanding these properties helped us frame the design space within current technical limits (challenge 2);
- An alternative WoZ prototyping method. For each NLP-powered interaction, we designed a different hybrid of WoZ and off-the-shelf toolkits to best simulate the likely errors (challenge 5). This method shares the goals of traditional WoZ in enabling fast prototyping of NLP. In addition, our method highlights that different intelligent features produce different kinds of errors; each kind can have very different UX consequences. In order to better capture these consequences, we needed to better orchestrate WoZ to simulate NLP behaviors.

5 DISCUSSION

Sketching and rapid prototyping are cornerstones of HCI's creative activities. They are quick, timely, inexpensive, disposable and iterative. *The last thing to do when sketching is to write code* [7]. However, "expensive" technologies like NLP and ML have started to challenge the fundamental notions of sketching and rapid UX prototyping, as they do not fail fast or fail often. This work detailed what sketching NLP is like in one specific design project. In doing so, we highlight that sketching remains critical to AI's design innovation, therefore merits further study.

As a case study, this project offers a point-of-reference for researchers who aim to support language-interaction design practice. Do the challenges we encountered generalize to other design situations? What central questions can integrate these different challenges and emergent solutions into coherent fields within HCI? Answers to these questions have the potential to radically improve the UX design and innovation of NLP systems at large. To jump start this community discussion, below we describe some of our reflections.

The Importance of Abstraction

Abstraction is essential to any early design ideation, yet a missing perspective in NLP HCI literature. Most assumed that designers start designing by "*writing linear dialog examples*" [24]. WoZ studies often simulated NLP interactions with rule-based systems or crowd intelligence; Deliberations are lacking on whether these are effective abstractions of NLP system outputs.

This case study reminds us of the importance of abstraction in sketching intelligent language-interactions. It was particularly important in the early design stages, the process of exploring many broad ideas before drafting concrete UIs or dialogues. For example, what kind of writing assistance do people even want? It seemed impossible to traverse such a design space by writing concrete dialogue examples, and it was not intuitive how to sketch dialogues abstractly.

NLP systems are difficult to design abstractly because both language interactions and intelligent interactions are difficult to abstract. On one hand, most sketching techniques and tools in designers' tool-belt, such as storyboards and wireframes, have evolved over the last two decades under the dominance of the graphical user interface, not directly applicable to language interactions. On the other , data-driven interactions are divergent, constantly evolving and sometimes make errors incomprehensible and difficult-to-anticipate. It could seem like only building a working NLP system can reveal its likely behaviors.

Upon reflection, all five challenges we encountered, and their solutions, involve some aspect of abstracting NLP interactions: Challenge 1, 2 and 4 dealt with abstracting interaction design ideas into different problem framings effective for design deliberation, communication and innovation respectively. The slightly different forms of the notebooks embodied these abstractions. In response to Challenge 3, Table 1 attempted to abstract NLP's technical capabilities and limits, in order to bound the design space it enables. In response to Challenge 5, our prototyping methods dealt with challenges in abstracting the experiential qualities of NLP interactions, especially its errors.

In this light, we argue that supporting sketching language interactions *abstractly* is an important yet under-engaged issue for HCI/design research. This case study revealed three aspects of abstraction, offering a starting place for this line of research.

(1) Abstracting language *interactions* as ways of framing its design problems
(2) Abstracting NLP *capabilities* to frame its design space realistically
(3) Abstracting NLP's *experiential qualities* to enable rapid UX prototyping

Future work may also take inspirations from previous work on designerly abstractions of difficult technology materials, such as visualizations, taxonomic vocabulary and sensitizing concepts [27, 30, 33]. In doing so, our community can develop a robust family of methods for designing NLP abstractly, enabling its UX design innovation.

Designing with NLP's Capabilities *and Limits*

Let us expand on the issue of "abstracting NLP capabilities".

Like many other designers [13], we found understanding the capabilities and limits of intelligent technologies challenging. Our early sketches of writing assistance revealed a significant gap between how we wanted to support users ideally and what NLP can build realistically.

This gap is hazily assumed yet rarely discussed in HCI research. Much work – exemplified by the many unrestricted WoZ studies – instead has focused on the design *possibilities* NLP aspired. This orientation leads to some NLP researchers' self deprecation that "*HCI people design useful things that we cannot build; we make things that nobody uses.*"

In parallel to works that freely imagine possible futures, there should also be research on creating products that wisely attend to state-of-art NLP's capabilities and limits. Towards this goal, more work needs to investigate respective advantages and disadvantages of human and artificial language intelligence in order to choreograph harmonious interactions in-between.

This provides a glimpse into what a designerly understanding of NLP capabilities might look like. In this case study, three aspects of NLP were relevant to our design, as they emerge naturally in our design activities: (1) High-level understandings of NLP's capabilities and limits, whih oriented our design ideation (Table 1 left); (2) NLP's capabilities given the available data and development resources, which informed our design deliberation and UX-gain-technical-investment negotiation (Table 1 right); (3) Each design's likely errors and other experiential qualities, which enabled rapid prototyping and helped us account for unexpected system behaviors.

Future research should evaluate and improve this set of NLP design properties. Moreover, enabling practitioners to develop their own tacit understanding of NLP opens up new research opportunities and promises real impact on UX practice. For example, what new boundary objects that help designer more effectively collaborate with NLP scientists and understand the technical capabilities and limits applicable to their respective design problems?

6 CONCLUSION AND ACKNOWLEDGEMENT

This paper presented our sketching process when designing intelligent writing assistance. We have articulated the challenges we encountered and the effective instruments that emerged. We encourage fellow researchers to join us, investigating the sketching and rapid prototyping NLP for their importance to NLP's UX practice and innovation.

We thank the Knowledge Technologies and Intelligent Experiences group at Microsoft Research (Ryen White, Michael Gamon, Edaena Salinas Jasso, Sujay Kumar Jauhar, Mark Encarnación, and more) for making this work possible. We thank all contributing NLP researchers for their valuable inputs.

REFERENCES

[1] [n. d.]. The AllenNLP toolkit demo: Text Entailment. http://demo.allennlp.org/textual-entailment.

[2] Jonathan Bean and Daniela Rosner. 2014. Big data, diminished design? *interactions* 21, 3 (2014), 18–19.

[3] Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

[4] Sara Bly and Elizabeth F Churchill. 1999. Design through matchmaking: technology in search of users. *interactions* 6, 2 (1999), 23–31.

[5] Geoffrey C Bowker and Susan Leigh Star. 2000. *Sorting things out: Classification and its consequences.* MIT press.

[6] Richard Buchanan. 1992. Wicked problems in design thinking. *Design issues* 8, 2 (1992), 5–21.

[7] Bill Buxton. 2010. *Sketching user experiences: getting the design right and the right design.* Morgan Kaufmann.

[8] Ana Paula Chaves and Marco Aurelio Gerosa. 2018. Single or Multiple Conversational Agents?: An Interactional Coherence Comparison. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18).* ACM, New York, NY, USA, Article 191, 13 pages. https://doi.org/10.1145/3173574.3173765

[9] Elizabeth Clark, Anne Spencer Ross, Chenhao Tan, Yangfeng Ji, and Noah A. Smith. 2018. Creative Writing with a Machine in the Loop: Case Studies on Slogans and Stories. In *23rd International Conference on Intelligent User Interfaces (IUI '18).* ACM, New York, NY, USA, 329–340. https://doi.org/10.1145/3172944.3172983

[10] Design Council. 2005. The 'double diamond' design process model. *Design Council* (2005).

[11] Justin Cranshaw, Emad Elwany, Todd Newman, Rafal Kocielnik, Bowen Yu, Sandeep Soni, Jaime Teevan, and Andrés Monroy-Hernández. 2017. Calendar. help: Designing a workflow-based scheduling agent with humans in the loop. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems.* ACM, 2382–2393.

[12] C. Crumlish and E. Malone. 2015. *Designing Social Interfaces: Principles, Patterns, and Practices for Improving the User Experience.* O'Reilly Media. https://books.google.com/books?id=dchcCgAAQBAJ

[13] Graham Dove, Kim Halskov, Jodi Forlizzi, and John Zimmerman. 2017. UX Design Innovation: Challenges for Working with Machine Learning as a Design Material. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems - CHI '17.* ACM Press, New York, New York, USA, 278–288. https://doi.org/10.1145/3025453.3025739

[14] Daniel Fallman. 2003. Design-oriented human-computer interaction. In *Proceedings of the SIGCHI conference on Human factors in computing systems.* ACM, 225–232.

[15] Melanie Feinberg. 2017. A Design Perspective on Data. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems.* ACM, 2952–2963.

[16] Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. AllenNLP: A Deep Semantic Natural Language Processing Platform. arXiv:arXiv:1803.07640

[17] William Gaver. 2012. What should we expect from research through design?. In *Proceedings of the SIGCHI conference on human factors in computing systems.* ACM, 937–946.

[18] Fabien Girardin and Neal Lathia. 2017. When User Experience Designers Partner with Data Scientists. In *The AAAI Spring Symposium Series Technical Report: Designing the User Experience of Machine Learning Systems.* The AAAI Press, Palo Alto, California. https://www.aaai.org/ocs/index.php/SSS/SSS17/paper/view/15364

[19] Elizabeth Goodman, Erik Stolterman, and Ron Wakkary. 2011. Understanding Interaction Design Practices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11).* ACM, New York, NY, USA, 1061–1070. https://doi.org/10.1145/1978942.1979100

[20] Patrick Hebron. 2016. *Machine learning for designers.* O'Reilly Media.

[21] S. Hoober and E. Berkman. 2011. *Designing Mobile Interfaces: Patterns for Interaction Design.* O'Reilly Media. https://books.google.com/books?id=j2eYAgAAQBAJ

[22] Stephanie Houde and Charles Hill. 1997. What do prototypes prototype? In *Handbook of Human-Computer Interaction (Second Edition).* Elsevier, 367–381.

[23] Bogyeong Kim, Jaehoon Pyun, and Woohun Lee. 2018. Enhancing Storytelling Experience with Story-Aware Interactive Puppet. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems (CHI EA '18).* ACM, New York, NY, USA, Article LBW076, 6 pages. https://doi.org/10.1145/3170427.3188515

[24] Scott R. Klemmer, Anoop K. Sinha, Jack Chen, James A. Landay, Nadeem Aboobaker, and Annie Wang. 2000. Suede: A Wizard of Oz Prototyping Tool for Speech User Interfaces. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST '00).* ACM, New York, NY, USA, 1–10. https://doi.org/10.1145/354401.354406

[25] Panagiotis Louridas. 1999. Design as bricolage: anthropology meets design thinking. *Design Studies* 20, 6 (1999), 517–535.

[26] Robert J. Moore, Raphael Arar, Guang-Jie Ren, and Margaret H. Szymanski. 2017. Conversational UX Design. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17).* ACM, New York, NY, USA, 492–497. https://doi.org/10.1145/3027063.3027077

[27] Camille Moussette. 2012. *Simple haptics: Sketching perspectives for the design of haptic interactions.* Ph.D. Dissertation. Umeå Universitet.

[28] Cosmin Munteanu and Gerald Penn. 2014. Speech-based Interaction: Myths, Challenges, and Opportunities. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI '14).* ACM, New York, NY, USA, 567–568. https://doi.org/10.1145/2628363.2645671

[29] Donald Murray. 1972. Teach writing as a process not product. *The Leaflet* 71, 3 (1972), 11–14.

[30] Marianna Obrist, Rob Comber, Sriram Subramanian, Betina Piqueras-Fiszman, Carlos Velasco, and Charles Spence. 2014. Temporal, Affective, and Embodied Characteristics of Taste Experiences: A Framework for Design. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems (CHI '14).* ACM, New York, NY, USA, 2853–2862. https://doi.org/10.1145/2556288.2557007

[31] Owain Pedgley. 2007. Capturing and analysing own design activity. *Design studies* 28, 5 (2007), 463–483.

[32] Jenny Preece, Yvonne Rogers, and Helen Sharp. 2001. *Beyond Interaction Design: Beyond Human-Computer Interaction.* John Wiley & Sons, Inc., New York, NY, USA.

[33] Johan Redström, Maria Redström, and Ramia Mazé. 2005. *IT+ textiles.* Edita Publishing Oy.

[34] Rushit Sanghrajka, Daniel Hidalgo, Patrick P Chen, and Mubbasir Kapadia. 2017. Lisa: Lexically intelligent story assistant. In *Proceedings of the 13th Artificial Intelligence and Interactive Digital Entertainment Conference.*

[35] Ari Schlesinger, Kenton P. O'Hara, and Alex S. Taylor. 2018. Let's Talk About Race: Identity, Chatbots, and AI. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18).* ACM, New York, NY, USA, Article 315, 14 pages. https://doi.org/10.1145/3173574.3173889

[36] Donald Schön and John Bennett. 1996. Reflective conversation with materials. In *Bringing design to software.* ACM, 171–189.

[37] Donald A Schön. 1984. *The reflective practitioner: How professionals think in action*. Vol. 5126. Basic books.

[38] A. Seffah and H. Javahery. 2005. *Multiple User Interfaces: Cross-Platform Applications and Context-Aware Interfaces*. Wiley. https://books.google.com/books?id=zRSsGC8ecAwC

[39] Lisa Stifelman, Adam Elman, and Anne Sullivan. 2013. Designing Natural Speech Interactions for the Living Room. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 1215–1220. https://doi.org/10.1145/2468356.2468574

[40] Daniel S. Weld, Corin Anderson, Pedro Domingos, Oren Etzioni, Krzysztof Gajos, Tessa Lau, and Steve Wolfman. 2003. Automatically Personalizing User Interfaces. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI'03)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1613–1619. http://dl.acm.org/citation.cfm?id=1630659.1630944

[41] Qian Yang, Nikola Banovic, and John Zimmerman. 2018. Mapping Machine Learning Advances from HCI Research to Reveal Starting Places for Design Research. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI '18 (CHI '18)*. ACM.

[42] Qian Yang, Alex Scuito, John Zimmerman, Jodi Forlizzi, and Aaron Steinfeld. 2018. Investigating How Experienced UX Designers Effectively Work with Machine Learning. In *Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18)*. ACM, New York, NY, USA, 585–596. https://doi.org/10.1145/3196709.3196730

[43] Qian Yang, John Zimmerman, Aaron Steinfeld, and Anthony Tomasic. 2016. Planning Adaptive Mobile Experiences When Wireframing. In *Proceedings of the 2016 ACM Conference on Designing Interactive Systems - DIS '16*. ACM Press, Brisbane, QLD, Australia, 565–576. https://doi.org/10.1145/2901790.2901858

[44] Shichao Zhang, Chengqi Zhang, and Qiang Yang. 2003. Data preparation for data mining. *Applied Artificial Intelligence* 17, 5-6 (2003), 375–381. https://doi.org/10.1080/713827180

[45] John Zimmerman, Jodi Forlizzi, and Shelley Evenson. 2007. Research through design as a method for interaction design research in HCI. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 493–502.