Developing a Toolkit for Prototyping Machine Learning-Empowered Products: *The Design and Evaluation of ML-Rapid*

Lingyun Sun, Zhibin Zhou, Wenqi Wu, Yuyang Zhang, Rui Zhang, and Wei Xiang *

Key Laboratory of Design Intelligence and Digital Creativity of Zhejiang Province, Hangzhou, China
State Key Lab of CAD&CG, Zhejiang University, Hangzhou, China
Alibaba-Zhejiang University Joint Institute of Frontier Technologies, Hangzhou, China

Machine learning (ML) and design co-support the development of intelligent products, which makes ML an emerging technology that needs to be further understood in design practice. However, the unusual attributes of ML and the transformations in the prototyping process frequently prevent most designers from continuous innovation. Thus, we invited designers to work together in a participatory design process and then developed ML-Rapid, an easy-to-use and flexible ML prototyping toolkit. ML-Rapid helps designers to rapidly empower their physical prototype with ML by invoking simple code while exploring more design possibilities. A method of applying the toolkit within the design process was also proposed to promote meaningful innovation opportunities. We evaluated our work in a project called Design for Information Product. The evaluation results showed that designers who were new to ML programming increased their understanding of ML after participating in the project, and ML-Rapid lowered the barrier to ML for designers by allowing them to explore design possibilities throughout the main steps of the ML process.

Keywords – Design, Machine Learning, Prototyping, Toolkit.

Relevance to Design Practice – In this study, a workflow that enables designers to work with ML was explored and ML-Rapid was developed to facilitate the prototyping process and explore possibilities. A method for applying ML-Rapid in the design practice was provided to help designers successfully design a real-world ML-empowered product.

Citation: Sun, L., Zhou, Z., Wu, W., Zhang, Y. Zhang, R., & Xiang, W. (2020). Developing a toolkit for prototyping machine learning-empowered products: The design and evaluation of ML-Rapid. *International Journal of Design, 14*(2), 35-50.

Introduction

The application of machine learning (ML) provides an exciting opportunity for intelligent product design, and ML also implements new practices under the influence of design. The intelligence of ML, as a type of artificial intelligence (AI), assists designers in extracting features or patterns (Nasrabadi, 2007; Witten et al., 2016) from high-dimensional information, such as information on facial expressions (Yu & Zhang, 2015), image style (Johnson et al., 2016), and body language (Behoora & Tucker, 2015). Simultaneously, demonstrative examples can be provided to enable ML to function without explicitly written rules (Hebron, 2016). Rule-based intelligent products only perform pre-defined responses to specific events in accordance with established rules, but ML-empowered products learn the patterns from the provided data to respond appropriately to various situations, and have the potential to cope with complex situations that could only be previously solved by human intelligence. Furthermore, design thinking, such as human-centered thinking, is gradually changing ML technology so that ML can better serve users (Gillies et al., 2016). For instance, researchers in the field of human-computer interaction (HCI) encourage users to freely edit corrective examples to improve the performance of the ML model (Amershi et al., 2014; Fails & Dan, 2003). Thus, there is an urgent need for designers to learn more about ML and potentially create new ways of empowering products using ML (Dove et al., 2017; Yang, 2017; Yang et al., 2018a).

Unfortunately, most designers do not have adequate ML literacy to ideate creatively and practically (Yang, 2017) because ML, compared with traditional technologies, has undergone dramatic transformation and is thus a difficult technology to understand. ML literacy refers to designers (1) understanding the basic concepts of the ML capability, mechanism, and working process; and (2) leveraging or expanding the ML capability so that they can prototype and evaluate design proposals. ML literacy helps designers to become familiar with design material and thus enhances their design abilities to empower products with ML, for example, knowing how to obtain necessary data and computing resources and construct interaction. Moreover, ML literacy

Received December 11, 2018; **Accepted** July 13, 2020; **Published** August 31, 2020.

***Corresponding Author:** wxiang@zju.edu.cn

enables designers to participate in the main steps of the ML process (e.g., data collection and annotation, ML model training, and updating, etc.) and enables them to explore design possibilities through prototyping. Prototypes mean the representative and manifested forms of design ideas (Lim et al., 2008). Prototypes can lay particular emphasis on different dimensions such as interactivity and functionality. The interactivity dimension helps designers to determine the way in which users interact with the product, while the functionality dimension assists designers in testing and verifying the functions that the final product will have. In our case, the purpose of a prototyping tool is more related to the functionality dimension. However, the intelligent capability of ML is derived from large-scale data (Mitchell et al., 1990; Yu & Zhang, 2015), resulting in a prototyping approach that requires designers to extract patterns from data, which is different from the rule-based approach. Thus, designers need to understand the ML process when extracting features or patterns from data.

The challenges in introducing ML and developing designers' ML literacy call for an easy-to-use and flexible prototyping tool (Dove et al., 2017). Existing tools do not strike a balance in terms of ease-of-use and flexibility (Patel et al., 2008; Yang et al., 2018b). Flexibility means that the toolkit enables designers to freely extend and modify diverse ML capability according to the given design goal, scenario, and data, thus being able to be used to prototype working and diverse ML-empowered products. Ease-of-use means that the toolkit supports designers without skilled programming skills by allowing them to easily modify ML capabilities and produce prototypes in a rapid manner. Current tools are either easy-to-use toolkits for beginners or flexible toolkits designed for experts. Easy-to-use toolkits provide limited access to the entire ML process and diverse ML capabilities while flexible toolkits help designers to prototype various types of ML capabilities but require skilled programming ability. However, an ideal toolkit should help designers to easily engage in the main steps of the ML process. For example, it might allow designers to use their own labeled data to train the ML model, and enable the product to own other ML capabilities.

To support designers to prototype ML functionality, the prototyping tool should assist designers without skilled programming ability to access the entire ML process and build desired ML applications both easily and flexibly, thus inspiring novel ideas. Here, ML applications mean the capabilities of ML that can be used to perform. For example, the ML capabilities, like image-based object recognition and Text-to-Speech, can be considered as different ML applications. Our work focuses on the functionality aspect of ML applications used to empower the functional prototype of products. Thus, we will selectively simplify the ML process through a participatory design process, rather than pursuing the simplest ML process.

With the above goals, we develope the ML-Rapid toolkit and propose an approach of applying ML-Rapid in design practice. ML-Rapid provides an Arduino-similar integrated development environment (IDE) that simplifies the prototyping of ML-empowered products. ML-Rapid reduces the complexity of programming with ML, and allows designers to rapidly create a functional physical prototype using a Raspberry Pi and neural network accelerator by invoking simple code. It also allows designers to train and refine ML-empowered prototypes with their own data and labeling so that they can develop ML literacy through prototyping.

To develop ML-Rapid, we involved undergraduate industrial design students and focused on the challenges that they encountered in working with ML. First, through an investigation, we identified the attractive ML applications, including text-to-speech (TTS) and object recognition, etc. Second, we proposed a possible workflow that enables designers to work with ML via an analysis of the participatory design process. Third, we implemented the workflow and evaluated it by developing the ML-Rapid toolkit. To accomplish the above work, we encapsulated each ML application according to the former workflow as a module and built an embedded hardware platform on which those modules could run. Moreover, we determined how ML-Rapid could be fully used in all stages of the design process. As an evaluation of our work, 30 participants took part in a project titled Design for Information Product (DIP) to learn how

Lingyun Sun is a professor at Zhejiang University. He is the Deputy Director of the International Design Institute, Ng Teng Fong Chaired Professor and the director of ZJU-SUTD Innovation, Design and Entrepreneurship Alliance, and a deputy director of the ZJU-Beidou Joint Innovation Design Engineering Center. His research interests include Design Intelligence, Innovation and Design, and Information and Interaction Design.

Zhibin Zhou is a Ph.D. candidate attached to the International Design Institute of Zhejiang University. With a background in user experience design and artificial intelligence (AI), his PhD research focuses on capturing the interaction between humans and AI and gaining an increased understanding of AI as a design material. His prior work aims to bring together knowledge in the fields of design and AI as well as help the next generation of designers to foster AI literacy to understand its nature and thus creatively ideate and practically prototype ML-enhanced products.

Wenqi Wu is a Doctoral student from the College of Computer Science and Technology, Zhejiang University. He majors in intelligent product design and interactive design. As a designer and researcher, he has done some research to bring artificial intelligence into design work. Meanwhile, he also keeps an eye on the frontier research of developing design principles for AI-empowered product design as well as applying them into real-world design.

Yuyang Zhang is a UX researcher with a background in interaction design. She graduated from Zhejiang University with a bachelor's degree in Industrial Design and is currently a master fellow of the International Design Institute of Zhejiang University. She is interested in AI and Design, exploring how to address the issues of designing AI-empowered products. Her prior work includes the challenges of AI-empowered products as well as the design tools for AI-empowered products.

Rui Zhang received a B.S. degree in mathematics from Zhejiang University, Hangzhou, China, in 2018. He is currently pursuing a Ph.D. degree in the College of Computer Science and Technology, Zhejiang University, China. His Ph.D. project aims to generate graphic design by using ML. His research interests include computer vision, font generation, computer calligraphy, and human-computer interaction.

Wei Xiang is a research assistant in the Modern Industrial Design Institute, College of Computer Science and Technology, Zhejiang University. He received his Ph.D. degree in Digital Art and Design, and M.S. degree in software engineering from Zhejiang University. His research lays in design intelligence, design cognition, experience computing, and multi-route interaction methods that support communication and creativity. Wei Xiang employs psychological and electrophysiological methods to analyze the cognitive process during design, and has proposed several design ideation methods in addition to developing corresponding computer-aided communication and design tools, which have been used in creative painting, design ideation, and online consumption.

to use ML-Rapid to build ML-empowered prototypes. We then discussed where the toolkit could be improved and how to make better use of ML in design practices.

The contributions of this paper are the ML-Rapid toolkit and the insights we obtained from the participatory design process of the toolkit and the DIP projects.

- ML-Rapid is developed for designers to prototype ML functionality, thus, the provided function and modules are concise and clear, and can be used to empower diverse products with ML technology. ML-Rapid reduces the complexity of ML while retaining the steps that allow designers to innovate in the ML process, thereby helping designers to build ML applications and then prototype functional ML-empowered products. Furthermore, ML developers/coders can also follow the workflow of ML-Rapid to transform their ML applications into the toolkit modules so that their applications can be used by designers.
- The insights obtained from the participatory design process of ML-Rapid support the development of future tools. These insights include (1) keeping balance between ease-of-use and flexibility of the tool to help designers complete the prototyping process; (2) keeping familiar design activities, like prototyping, reflection, and collaboration, when designers work with ML. These activities help designers to increase their ML literacy and promote meaningful innovation.

Background

Practice Leveraging Design and ML

Existing studies have shown how ML and design mutually inspire each other. Some studies have attempted to reframe ML workflows according to human practices (Gillies et al., 2016) and the users' purpose (Amershi et al., 2014). The Google user experience (UX) community attempted to obtain user feedback over the entire product lifecycle to improve ML systems (Lovejoy, 2018). Yang et al. (2018b) proposed a design process that included identifying the problem for ML to solve, validating the technical feasibility, and then iterating the ideation process. Additionally, HCI design researchers adopted interactive ML (Amershi et al., 2013), where the ML model updates involved interactions with humans. They endeavored to build a natural interaction system by allowing users to freely edit training data (Amershi et al., 2014; Fails & Dan, 2003) because the corrective examples clearly reflected specific behavior (Fiebrink, 2011). Additionally, Brown et al. (2016) found that the user tended to look for real-time updates of the model, so models should run sufficiently quickly for interaction.

Such practical experience indicates that for designers who use ML in a creative manner, they need to understand its capabilities and the important steps of the ML process. Some creative strategies have been generated to improve the UX and ML workflow, for example, purposefully editing samples, valuing the user's purpose, and speeding up ML models. However, current ML tools do not introduce ML to designers properly or prepare designers to implement the aforementioned strategies.

Introduce Emerging Technologies to Designers

Designers are inspired to research emerging technology information because design practice is more creative when designers are familiar with technology (Şendurur et al., 2016). Thus, a design process that emphasizes reflective learning (Daalhuizen & Schoormans, 2018; O'Connor et al., 2018), making (Martin, 2015; Peppler et al., 2016), and collaboration (McGlashan, 2017; O'Connor et al., 2018) was adopted in design practice. Thus, prototyping activities are used for exploring the design space and reflecting on the design ideas (Lim et al., 2008). Prototyping activities can focus on different dimensions, like interactivity and functionality, that help designers to refine corresponding aspects of a design idea. For example, the functionality dimension includes system functions and users' functionality needs. Designers can use several prototypes with different function combinations for filtering the best functional plan. On the other side, the interactivity dimension includes input behavior, feedback behavior, etc., which is mainly used to select the best solutions for interaction. The prototype tools range from storyboards (Guo & Goh, 2016), and sketching (Lewis et al., 2018) to interactive systems used to make abstract manipulations involved in emerging technologies more manageable. Thus, easy-to-use prototyping tools can help designers with limited engineering knowledge to prototype intelligent artifacts. Such tools have already been applied to introduce abstract concepts that underlie emerging technologies, such as computational fashion material (Genç et al., 2018) and haptics (Vallgårda et al., 2017). McCardle (2002) therefore emphasized the strong need for appropriate tools in design courses in the field of AI.

Several tools have been developed to allow designers to make intelligent prototypes. The Smart-Its project first presented a modular system for prototyping electronic artifacts. Barragán (2004) developed Wiring that is a microcontroller coupled with a friendly IDE. The Arduino system was then developed as an open-source platform with many modules that can be modified freely (Qu et al., 2017). And platforms such as Raspberry Pi have been developed to compensate for Arduino's limited multimedia capabilities (Hodges et al., 2012).

The aforementioned work has showed that prototype tools have evolved into friendly, open-source, and modular tools. ML is now an emerging technology, but to the best of our knowledge, there is no tool that works proficiently, like Arduino, enabling designers to prototype interactions in this area.

Existing ML Tools

Several platforms and tools have been devolved to democratize ML. These tools can be categorized as ML as a service (MLaaS) platforms (Ribeiro et al., 2016), such as IBM Watson; programming toolkits, such as TensorFlow (Abadi et al., 2016); and non-programming tools, such as Yale and Wekinator. However, they do not simultaneously provide essential ease-of-use and flexibility for designers. Table 1 shows the characteristic of existing tools and Figure 1 illustrates their workflows.

Table 1. Characteristics of ML tools and platforms (Hebron, 2016).

Platform	Brief summary	Intended user	Prototype dimension	Ease-of-use	Flexibility
MlaaS (e.g., IBM Watson)	Shares computational resources across multiple users	Mostly developer	Functionality	• Requires skill to deploy network devices; • Requires devices/hardware to always connect to the Internet.	• Provides limited optional easy-to-use ML applications; • Inconvenient to manipulate the hardware.
Programming toolkit (e.g., TensorFlow)	Professional and fully customizable	Developer	Functionality	• Requires programming ability • Requires deep knowledge of the ML algorithm;	• Can access nearly all steps of the ML process and ML applications
Non-programming tool (e.g., Delft AI toolkit)	Solves real-world problems without code or with simple code	Designer	Interactivity	• Requires none programming experience.	• Provides limited ML applications; • Accesses limited steps of ML process;

MLaaS and programming toolkits are designed for professional developers who need to access all the steps of the ML process. Programming toolkits enable its users to control almost every detail of the ML process using various and complex coding functions, which is quite hard for most designers (see Figure 1). Additionally, MLaaS requires users to deploy network devices to connect to an internet platform (see Figure 1). These tools provide high flexibility but require skilled programming ability, focusing more on the functionality of ML technology.

Non-programming tools facilitate the process of prototyping the interaction between user and ML system by providing pre-trained ML models and graphic interfaces. For example, the Delft AI Toolkit, a modularized toolkit for designers, uses drag & drop to prototype interactions between the ML systems and users. It mainly aims to remind designers of the interaction possibilities that ML could provide. The ML applications provided by non-programming tools are typically pre-defined. Therefore, several steps of the ML process (see Figure 1) are eliminated to keep the simplicity, which makes prototyping of the interaction rapid and efficient while keeping ML technology in a black box. This hides the mechanisms of ML and also restricts the prototyping of ML functionality as well as the diversity of ML applications, categories of training data, etc. Other similar toolkits include Wekinator for gesture control (Fiebrink & Cook, 2010), Yale (Mierswa et al., 2006) for data analysis, and ml.lib (Bullock & Momeni, 2015), a library of Max and Pure Data for gesture recognition. Some MLaaS platforms will also provide pre-trained ML applications to ease the prototyping process.

An ideal toolkit would allow designers to not only adapt to complex technologies in a friendly manner but also modify and extend the capabilities of the prototype with sufficient space for exploration (Vandevelde et al., 2015). The current tools (e.g., TensorFlow, IBM Watson) for developers mostly serve the purpose of prototyping functionality while the tools (e.g., Delft AI toolkit) for designers are usually for the purpose of interactivity. Current tools somehow prevent most designers from developing ML literacy because they are either too difficult to use or too focused on particular steps of the ML process or limited application domains.

Summary

The insights from the review can be categorized according to the requirements of the toolkit and the application approach.

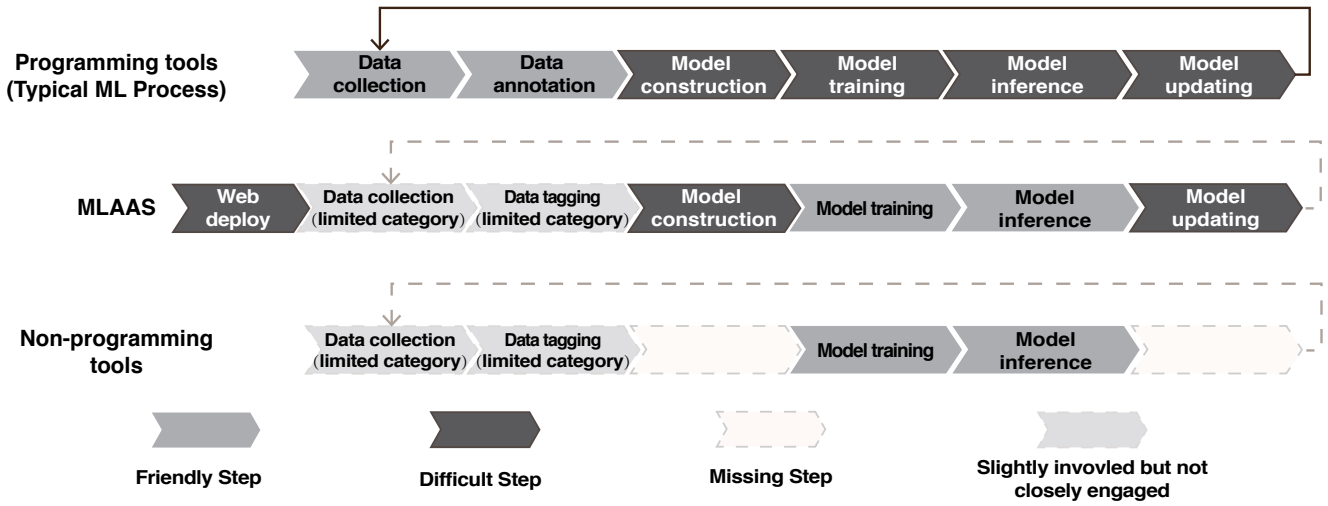


Figure 1. Workflow of current ML prototype tools.

The ideal toolkit requires the following:

- a suitable application domain for exploring a sufficient number of design possibilities;
- a friendly IDE and simple programming language, which is similar to Arduino;
- modular and open-source frameworks to help designers to use the toolkit and encourage developers to contribute modules;
- all the main steps of the ML process should be covered so that designers can improve ML systems over the entire process from the perspective of design; and
- designers and users should be involved in and contribute to the ML process.

The ideal approach requires the following:

- practice, reflection, and collaboration that provide designers with a better understanding of an emerging technology; and
- reasonable tool and technology support that benefit design practice.

Above all, the toolkit needs to be easy to use while providing sufficient modules that can be explored. However, the types of technical ML capabilities in which designers are interested are unclear. Additionally, how designers work with ML also remains unclear, which means we still do not know which steps of the ML process stop designers from prototyping their ideas, not to mention how ML tools can be designed to determine the balance between flexibility and ease-of-use.

Design of ML-Rapid

In our study, we first explored the types of technical ML capabilities that designers are interested in to clarify the application domain of our toolkit. We observed how designers work with ML and developed a toolkit balancing flexibility and ease-of-use for designers. Furthermore, we proposed an approach to applying ML-Rapid to support a design process that encourages prototyping, reflection, and collaboration.

Participatory design (Kensing & Blomberg, 1998), a means to involve participants in the design process to ensure that the result meets their needs, was used in the design of ML-Rapid (see Figure 2). The method was used to investigate not only the technical capabilities of ML but also the workflow that allows designers to work with ML. Additionally, we adopted a similar research procedure to that in the work of Genç et al. (2018). We obtained design insights into ML-Rapid through conducting workshops, the analysis of workshop outcomes, and interviews.

Twenty-five undergraduates (14 males and 11 females) majoring in industrial design were invited to participate in the design process. The participants had mastered open-source hardware, such as Arduino and Raspberry Pi, in previous courses, but they had only limited understanding of ML.

Stage 1: *Identify the Application Domain*

Since this study is an early attempt to verify our idea, we only selected a few technical applications in which designers are most interested. We will improve our research and transform more ML applications into toolkit modules based on our idea. Stage 1 determined the ML-empowered applications that designers hoped to use in their design practice.

We conducted a one-week workshop to investigate the interesting ML applications. The process of defining the application domain consisted of four steps: (1) We asked the participants to independently collect as many cases of ML-empowered products as possible. They were required to summarize the ML technical applications in each case (see Figure 3a). They initially collected 147 ML-empowered products, and each design case was tagged with several ML applications (see Figure 3b). However, some descriptions of the collected cases were ambiguous. (2) In presentations, the participants introduced the applied ML technical applications and how each ML-empowered product worked. Three experts in ML-empowered design helped the participants to modify the descriptions of the ML applications, including unifying different descriptions of the same ML application and removing products that were not empowered by ML. We then concentrated on 42 technical applications that ML can provide. (3) The experts and participants used affinity diagrams to manually cluster these modified 42 applications into 19 application clusters using sticky notes (see Figure 3c). Notes that described similar applications were placed in the same cluster. After an iterative process of reorganizing different clusters and selecting cluster elements, each cluster was assigned a representative name. For example, the TTS cluster contained speech generation, speech synthesis, smart dialogue, etc. (4) The 19 application clusters were described individually to the participants, and through a paper-based survey questionnaire, we asked each participant to select six technical applications that would be most frequently used in their future design, intending to cover as many design domains as possible (see Figure 3d). From the selected technical applications, the final six applications included image-based object recognition, speech-to-text, facial based identity recognition, facial based emotion recognition, TTS, and the generation of images.

Stage 2: *Define the Workflow that Enables Designers to Work with ML*

Stage 2 clarified the designers' difficulties in working with ML and provided insights for improving the workflow that designers use to prototype with ML. The participants were asked to reproduce a cucumber-sorter prototype based on a TensorFlow project. The project provided the cucumber image dataset and instructions for migrating an existing MNIST classification model (handwritten digit model for image processing) to the cucumber classification

Figure 2. Design process of ML-Rapid.

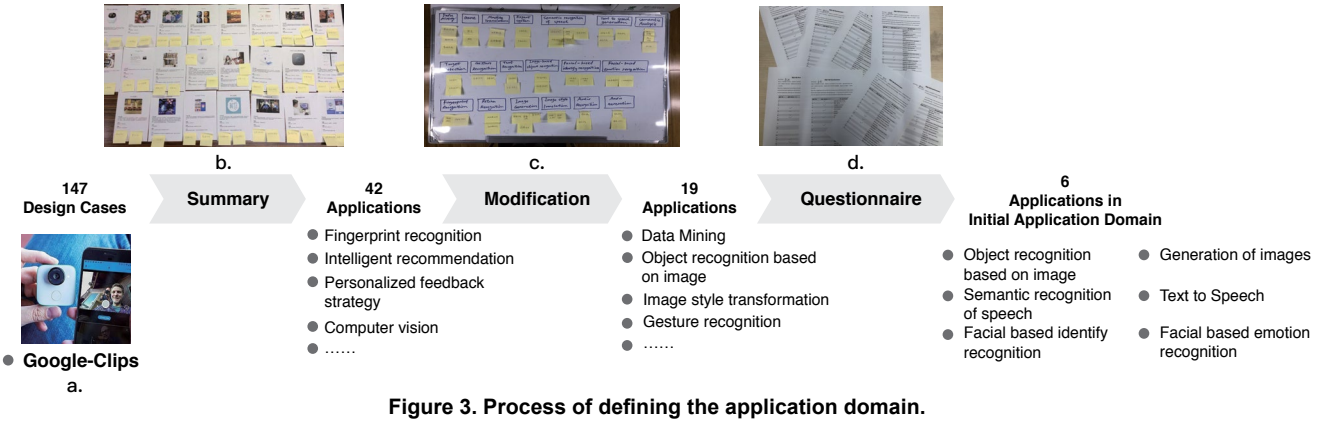


Figure 3. Process of defining the application domain.

model. The challenges in this case are similar to those in real-life prototyping, for example, because of differences in the structures of the datasets and model, some designers may not be able to apply an existing solution. A subsequent questionnaire asked which step of the ML process confused the participants, the reasons for the confusion, and their expectations of the toolkit.

The analysis of the questionnaire revealed that the participants were frustrated in completing this task; in fact, only two participants successfully completed the task in a week. Table 2 shows the details of how the designers failed in the task, the number of participants who became stuck at a certain step and did not progress further in the task, and the reason for failure.

Insights from Table 2 and reported expectations of the toolkit from the questionnaire revealed that (1) most of the participants could not manage concepts that were too abstract and preferred to build ML model in a modular manner to avoid concepts such as activation functions; (2) the participants paid a great deal of attention to the hardware/software integration, which helps to convert ML applications into physical artifacts; (3) most of the participants preferred not to be exposed to detailed engineering problems, such as those related to the dataset format and environment configuration; and (4) the presented function and code needed to be simple and clear.

The basic idea of our workflow is derived from the interactive ML (Amershi et al., 2013) discussed in the background section. It encourages designers to contribute in the ML process through interaction and data accumulation. Besides, our workflow was inspired by the workflow of Arduino that simplifies the confusing steps and attempts to support the essential participation of designers in the ML process. Arduino serves a workflow that does not simply reduce the complexity of code but also retains the steps that allow designers to innovate in the prototyping process, thereby helping designers to rapidly build various kinds of prototypes. Our workflow uses simple function commands to cover necessary ML processes including data collection, data annotation, model training, inference, and model updating (see Figure 4), in which new data will continue to accumulate within the toolkit when inferring on the basis of new data in the step of data collection. Designers calibrate or correct new data through a certain operation, and organize the annotated data using the same format as the original dataset during the data annotation step. The collected dataset is then reframed with the existing dataset. For model construction, our workflow provides models of different sizes, and designers can choose the model according to the platform on which their prototype will be deployed. In the model training step, it is suggested that designers use an initial

Table 2. Specific problems encountered by the participants in making cucumber classifiers.

Stage	Stuck participant Number	Concept	Reason for failure
Environment configuration	2	• Computer environment settings • Basic Python knowledge	• Participants were unfamiliar with the Python language and could not set up an environment that is capable of running TensorFlow.
Preprocessing the dataset	6	• Data and labels • Loading the dataset into the model	• Difficulties were mainly due to different formats of the dataset.
Model reconfiguration	9	• Construction of neural network • Activation functions • Gradient descent • Loss functions among others	• Most problems were related to specific TensorFlow functions and libraries. Additionally, the concepts in this stage were too abstract for participants to master.
Model training	4	• Learning speed • Training and testing dataset among others	• The concepts in this stage are abstract; especially the learning speed, which is difficult to manipulate. Costly hardware is required to train the model.
Model inference	2	• Making a prediction using the trained model • Capturing and reading the images	• Participants encountered a problem in capturing and reading pictures. The hardware, such as cameras, is not so easy for participants to manipulate.

dataset and model with an appropriate structure to start training, or even directly use a model that has been trained for similar scenarios. In the inference step, the inference result can be used to instruct other hardware components, such as motors, speakers, and cameras. Designers can obtain a new retrained model based on retraining the reframed dataset in the model updating stage. Models before and after retraining are compared by adopting a certain strategy, and the model that performs best in the current scenario is used continuously.

The workflow satisfies the main needs of designers in that only the essential steps of the ML process are covered, each function presented to designers is easy to understand, all details and concepts that are not friendly are hidden, the format of the dataset remains the same throughout the process, designers are assisted with the integration of hardware and software, and a simple framework helps to transform other open-source projects into easy-to-use modules.

Stage 3: *Development and Iteration of ML-Rapid*

According to the results of the preliminary study, we chose reasonable open-source platforms to develop both the software and hardware of the ML-Rapid toolkit. Then we asked the participants to produce a prototype using ML-Rapid under our observation, guiding us to develop the toolkit.

Inspired by the huge appeal of open-source platforms to ML application developers, we developed ML-Rapid based on TensorFlow, a popular framework within the ML technology community. To meet the expectations for the integration of hardware and software, we chose Raspberry Pi as the platform with which to develop ML-Rapid and run other hardware components, such as motors and cameras. In accordance with the six technical capabilities of ML identified in previous research, we selected six TensorFlow-based ML applications that apply open-source protocols. Then we converted them into modules based on the identified workflow (see Figure 4).

By asking the participants to apply ML-Rapid to prototype and collecting their feedback, several challenges were identified and overcome in the process of developing an ML programming environment as friendly as the Arduino: (1) To solve the inference delay caused by the insufficient computing performance of the Raspberry Pi, the Movidius neural compute stick (NCS; Intel®, 2018), a tiny ML device with low-power architecture, was loaded to deploy ML inference applications. (2) An attempt was made to use transfer learning (Quattoni et al., 2008) so that the collected dataset played an important role, even if the participants gathered much less data than that in the original dataset. (3) The six ML application projects were completed using different versions of TensorFlow and Python. These applications had to be rebuilt following the aforementioned workflow in a stable Raspberry Pi Operating System. (4) To enable the ML application to work on the NCS accelerator, it was necessary to compile the trained TensorFlow models into files that could be calculated in the NCS. Meanwhile, several necessary drivers and libraries, such as OpenCV for supporting TensorFlow and NCS, were also installed.

Furthermore, we developed an IDE (see Figure 5) inspired by the Arduino IDE, which included the following: (1) it uses simple icons to help designers to write the code, and compile and upload it to the hardware; and (2) the balance between ease-of-use and flexibility within the Arduino inspired us to package the code in a friendly manner but leaves sufficient room for designers to explore.

Table 3 presents the functions of the TTS module as an example. The other five modules were packaged in the same manner to the TTS module, so designers can use the same function name even though they are using different modules. Meanwhile, we prepared the pre-trained model in advance in all the modules so that designers can use the toolkit directly to empower their prototype if they want to save training time. The modules of ML-Rapid are similar to libraries with different features in Arduino. Designers can build the prototype by invoking simple code and complete tasks in all the main steps of the ML process.

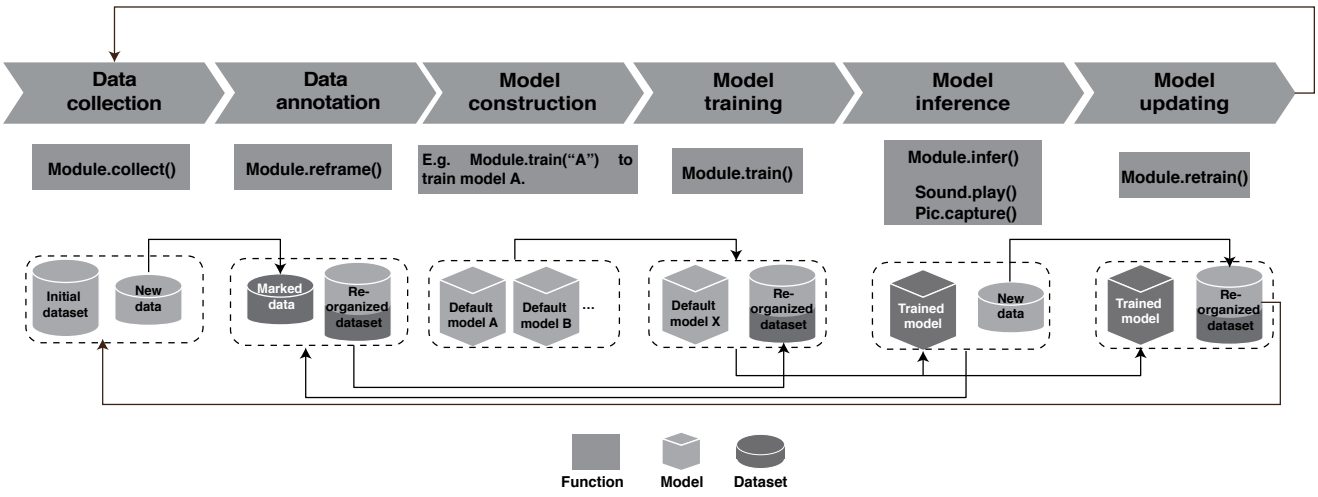
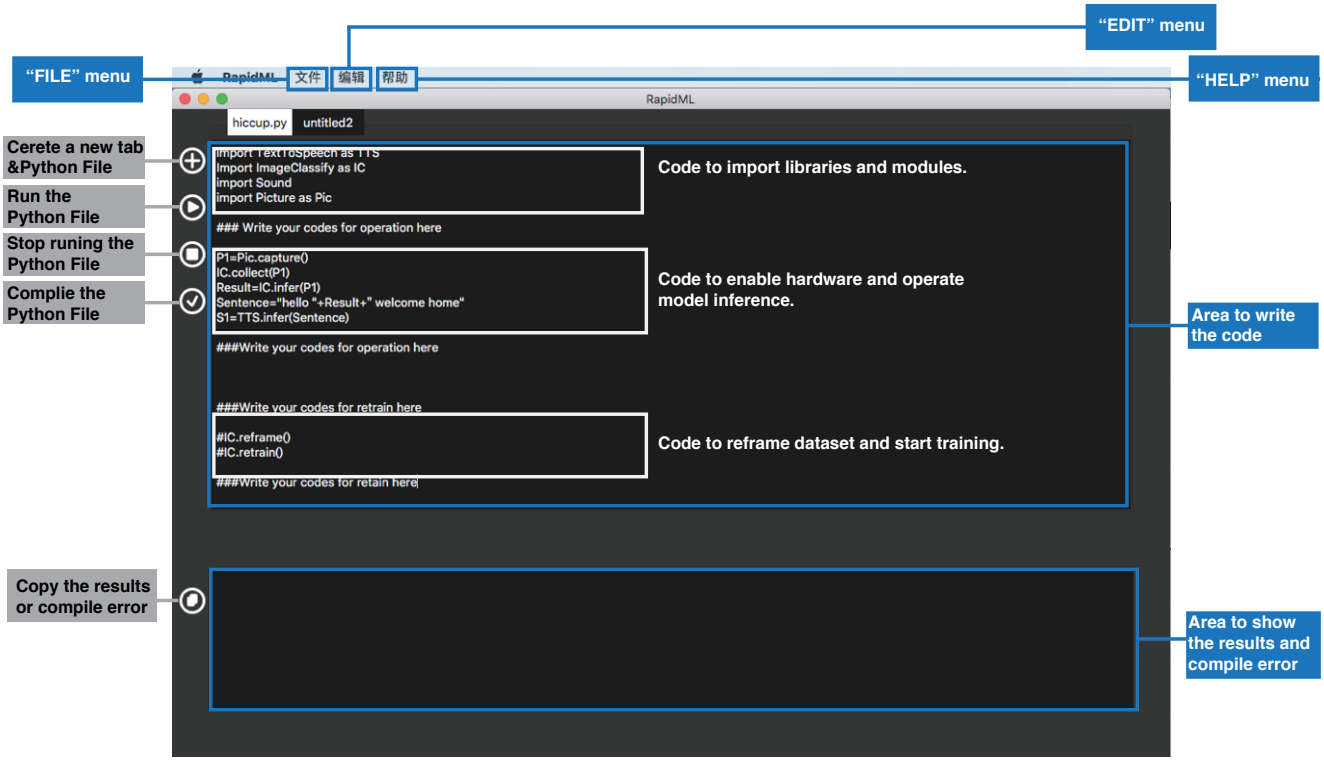


Figure 4. Workflow that enables designers to work with ML.



The codes shown in the screenshot of the IDE and the corresponding comments are presented as below.

```
import TextToSpeech as TTS
#import different ML modules
import ImageClassify as IC
import Sound
#import other necessary libraries
import Pictures as Pic

### Write your codes for operation here

P1=Pic.capture()
#Capture a picture and store it as "P1"
IC.construct("Middle_size")
#Construct a middle size ML model
IC.collect(P1)
#The ImageClassify module collect the picture and store it into the dataset
Result=IC.infer(P1)
#The ImageClassify module make prediction (object types) based on the P1
Sentence="hello"+Result+"welcome home"
#Make a full sentence with the prediction result
S1=TTS.infer(Sentence)
#Transfer the text to the speech
Sound.play(S1)
#Play the generated speech

###Write your codes for operation here


###Write your codes for retrain here

#IC.reframe("Application/ML-Rapid/TTS/Dataset/reframe_data")
#This function is used to reframe the dataset located in the given path
#IC.retrain("Middle_size")
#This function is used to retrain the middle size ML model

###Write your codes for retrain here
```

Figure 5. IDE of ML-Rapid.

Table 3. Functions of the text-to-speech module.

Function	Description
TTS.train()	Train the default model with initial dataset
TTS.retrain()	Retrain the model using a new dataset with TTS.retrain(collected) or a reframed dataset with TTS.retrain(reframe)
TTS.collect()	Organize the annotated data based on the format of the initial dataset
TTS.reframe()	Reframe the existing dataset with collected data
TTS.infer()	Make inference based on the trained model and new data (string)

We compared ML-Rapid with existing tools through the example of redesigning an Arduino-based device that encourages children to store toys. Without the ML technology, this device used to apply a pressure sensor to determine whether a toy is stored and send a snoring tone as feedback. However, it cannot determine whether the stored object is a toy, which can lead to an undesirable outcome. The ML-empowered object recognition application could solve this problem, whereas the TTS application provides more diverse feedback.

Figure 5 shows the IDE of ML-Rapid and the specific code content to complete the aforementioned task with the help of the IC module and TTS module. Figure 6 shows the working process of the ML-empowered storage box. When the user puts objects in the box, the prototype identifies the type of object using a function called IC.infer() in the IC module. The sentence that is generated by the TTS.infer() function in the TTS module depends on whether the classification is a toy and, if so, the type of toy. In addition to inference tasks, designers can still be involved in other steps of the ML process with only a few lines of code, including the collection of new toy images [IC.collect()], reframing of the dataset [IC.reframe()], and model retraining [IC.retrain()]. Moreover, when enabling hardware to complete tasks, such as sound play and image capture, it is convenient to use Sound.play() and Pic.capture().

We used Watson, TensorFlow, Delft AI toolkit, and ML-Rapid separately to complete the prototype of the intelligent toy storage box. We compared the ease-of-use and flexibility of these ML tools and showed detailed comparisons in Table 4.

Stage 4: *Applying ML-Rapid in Design Practice*

Based on the developed ML-Rapid and the principles mentioned in the background section, various methods are used to apply ML-Rapid in design practice. By drawing on the work of Yang et al. (2018b), the approach of applying ML-Rapid does not make radical changes to the design activities familiar to most designers.

First, the ML concepts introduced to designers should be carefully selected. Concepts that appear in the ML-Rapid workflow should be emphasized, whereas the overly abstract concepts can be ignored. Second, designers are required to keep the unique attributes of ML in mind and think about how to overcome the uncertainty generated by ML, and how to obtain the dataset and computing resources. ML-Rapid is used to verify design ideas and thus ensure that designers' proposals are practical. Third, prototyping has been placed in a critical position to help designers to innovate in the main steps of the ML process by using ML-Rapid to explore new ways of using ML.

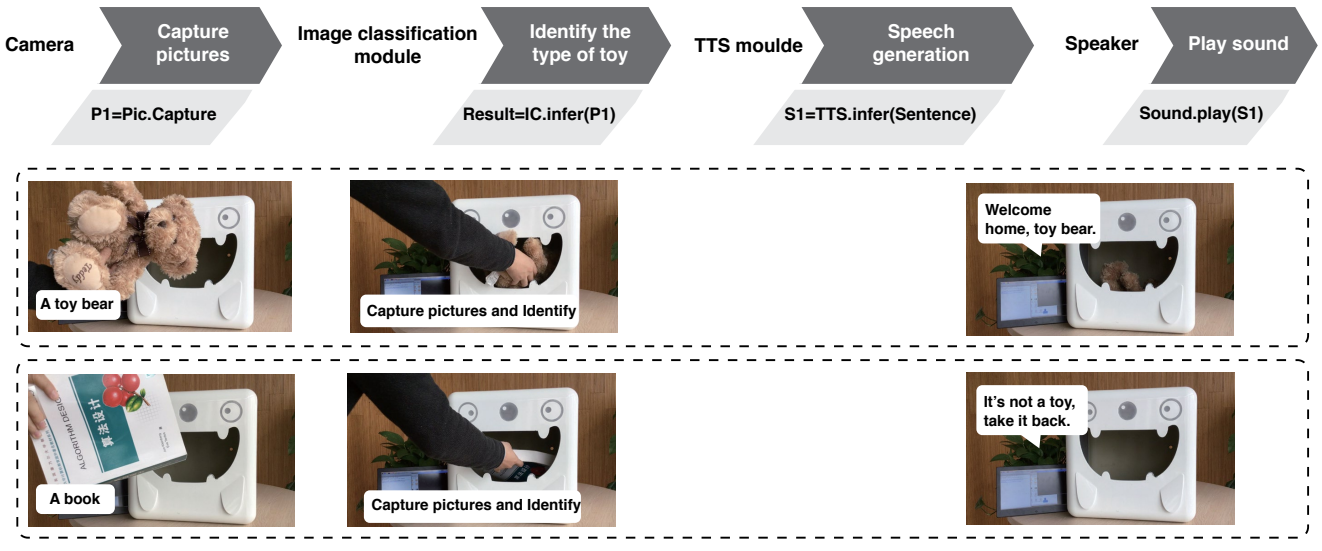


Figure 6. Working process of the ML-empowered storage box.

Table 4. Characteristics of ML tools when completing the intelligent toy storage box.

Platform	Ease-of-use	Flexibility
MLaaS: Watson	• Uses graphical user interface to construct prototypes; • Requires code to connect a prototype to the Internet.	• The operation to drive the hardware needs to be set locally through extra operations and configuration; • Requires skilled programming skill to construct a specific model and train it with the toy image dataset since it is typically for developers.
Programming tool: TensorFlow	• Professional IDE to help developers man-age projects; • Needs large numbers of lines of code and multi-level files to achieve a typical applica-tion.	• Could accomplish nearly all ML applications and support all the main steps for the ML process; • It is compatible with lots of Python libraries that could empower physical prototypes. • It is a software framework and requires extra hardware components.
Non-programming tool: Delft toolkit	• Uses drag& drop to construct prototypes; • No programming skills required; • Hardware components are easy to assemble.	• Focuses on specified applications/ capabilities and dataset for designers to choose; • Has a hardware platform that is compatible with specified sensors and actuators.
ML-Rapid	• Arduino-similar IDE is easy-to-use; • Requires only a few lines of simple code; • The hardware is easy to manipulate.	• Provides access to essential steps of the ML process including data annotation and model training; • Provides diverse ML applications/capabilities; • Provides a hardware platform that is easy to be extended.

The six-step design process outlined below fully reflects the approach of applying ML-Rapid in design practice.

- Introduction: The working mechanisms of ML and ML-Rapid are introduced to designers. The limitations of ML technology, such as the possible errors and the requirements for huge data and computing resources, are also shown at this stage.
- Tutorial of ML-Rapid: Using ML-Rapid as a demonstration tool, the basic ML concepts and corresponding functions of the toolkit are demonstrated simultaneously. This stage aims at helping designers to establish an understanding of the overall ML process and master the use of ML-Rapid.
- Practice with ML-Rapid: Designers should learn ML in-depth and learn how to use ML-Rapid in detail by completing a prototype. Designers are fully involved in the ML process in practice.
- Ideation: Designers conduct the necessary design research to define the problem that they are attempting to solve. They are also encouraged to determine new ways of using ML or new scenarios to apply ML; their ideas do not need to be practical or detailed at this stage.
- Iteration: Designers propose several potential ideas by creating sketches and providing details on how to apply ML, such as how to obtain a suitable dataset and how to ensure the correct rate of results. ML-Rapid can be used as a means of verification at this stage.
- Mutual evaluation: The designers then critique each other's proposals, and draw on the feedback to develop revisions. The designers present their detailed proposal and reflect on their method of applying ML through demonstrating their ideas to others.

Evaluation of ML-Rapid: *Design Project*

To evaluate our work from the viewpoint of designers, we organized a 14-week project called DIP using the design process proposed in Stage 4. The project also simultaneously incorporated design case collection, discussion, product launch, and reflection in the interest of project organization. The 30 participants (17 males and 13 females) in the DIP project had almost the same knowledge as those in the participatory design process and were all junior students of industrial design. During the project, we collected records made by a guidance team on the participants' usage of ML-Rapid and design outcomes.

From the records made by the guidance team, we found that the way the participants operated ML-Rapid in the project was almost always guided by the abovementioned design process. The participants first learned about ML and ML-Rapid through the introduction, case collection, and tutorial, and used ML-Rapid for the first time in the practice phase. In all the subsequent processes, participants used ML-Rapid more or less in design practice for verification. After the construction of the cucumber-sorter prototype in the practice phase, participants were randomly divided into 10 groups and conceived design proposals under the theme of ML-empowered design. The participants' vision was based on the six technical applications that the initial toolkit can support. After four cycles of evaluation and iteration (see Figure 7), they finally settled on six topics according to the evaluation results (see Figure 8). During the ideation phase and iteration phase, the guidance team conducted regular discussions to offer guidance on both the design and ML technology. The guidance team also answered the participants' questions about ML-Rapid and made small adjustments to ML-Rapid according to the feedback. The project ended with a product launch held at the university's theatre.

Figure 7. Images of the DIP project.

Case Studies

Finally, six prototypes were developed and presented at the launch event. Group 1 completed a modular product called QianLi for geographically separated families to stay connected (see Figure 8a). The product combines infrared sensors, microphones, cameras, and other modules for people to share their lives with separated relatives through ML. Group 2 created Cabe (see Figure 8b), which is an AI robot that senses changes in human emotions and presents reactions. Its non-obstructing responses help users to better manage their emotions when studying, thereby resulting in a better learning experience. Group 3 created a piggy bank for children called Honeyjar to correctly understand virtual currency (see Figure 8c). Allowances can be paid by parents into a mobile wallet (removable QR code display) in a jar while a mobile device application monitors the spending habits of the child. ML allows voice interaction between children and the devices. Group 4 continued the idea of a toy storage box. Their design, called Mito (see Figure 8d), can identify different toys and explore preferences of different children, and thus bestow character and life on toys with the voice of a specific character. Group 5 developed a personal meditation assistant called FLOW (see Figure 8e), which generates white noise based on image recognition. Concurrently, it uses light and scent to immerse users in a meditative state. The final work, from Group 6, is called Childsel (see Figure 8f), which is a mobile device application that allows children to create scrapbooks of the worlds that they dream of with the help of ML.

a. Qianli | b. Cabe | c. Honeyjar

d. Mito | e. FLOW | f. Childsel

Figure 8. Final prototypes created in the DIP: (a) Qianli, (b) Cabe, (c), Honeyjar, (d) Mito, (e) FLOW, and (f) Childsel.

We further illustrated how concerns regarding ease-of-use and flexibility were addressed by comparing the design outcomes resulted from ML-Rapid with the outcomes in Stage 2.

Ease-of-use:

First, all the participants learned to use ML-Rapid by following a tutorial, so that they were able to follow the workflow of the ML-Rapid and to call the functions within the ML-Rapid, after which the participants applied the learned usage method to build a working cucumber-sorter prototype. By contrast, in Stage 2, almost all the participants failed to work with TensorFlow after the workshop. Second, the required code to complete a prototype was simplified in the case of ML-Rapid compared with TensorFlow in Stage 2, which shortened the time to produce a working prototype. Third, participants rarely encountered the hardware issues described in Table 2 when working with ML-Rapid because it provides a friendly hardware platform consisting of Raspberry Pi and an NCS accelerator. ML-Rapid facilitated the process of building up the physical artifacts that connect different sensors and actuators.

Flexibility:

(1) ML-Rapid helped participants to freely edit or combine the six ML applications. For example, QianLi consists of several modules that cover different ML applications, including object recognition and voice generation (see Figure 9d). Honeyjar (see Figure 9c) and FLOW (see Figure 9a) also use more than one ML application. (2) ML-Rapid enabled the designers to participate in the ML process, including data annotation and training. Consider Cabe (see Figure 9b) and Mito (see Figure 9e) as examples: the participants created offline scenarios, thereby encouraging users to share the data that helps products to perform better. Furthermore, participants integrated the data annotation and training process into interactive games to attract children. For instance, the virtual characters of Mito ask children about the type of their current stored toy when it can't tell the category of this toy (see Figure 9e). Similarly, Childsel accumulates and shares excellent painting results for model updating (see Figure 9f) when users agree to share this information.

Interview Results

A semi-structured interview that lasted for approximately 15 minutes was conducted in the reflection stage of DIP to assess the 30 participants' perceptions of the ML-Rapid toolkit and the approach to applying the toolkit (P1-P30). The following questions were asked in the in-person interview: (1) Is it easy or difficult for you to use ML-Rapid when prototyping? Which part of ML-Rapid is good or hard to use? Why? (2) Do you think that ML-Rapid is useful or useless in the prototyping process? Which part of ML-Rapid enables or disables you to prototype your design

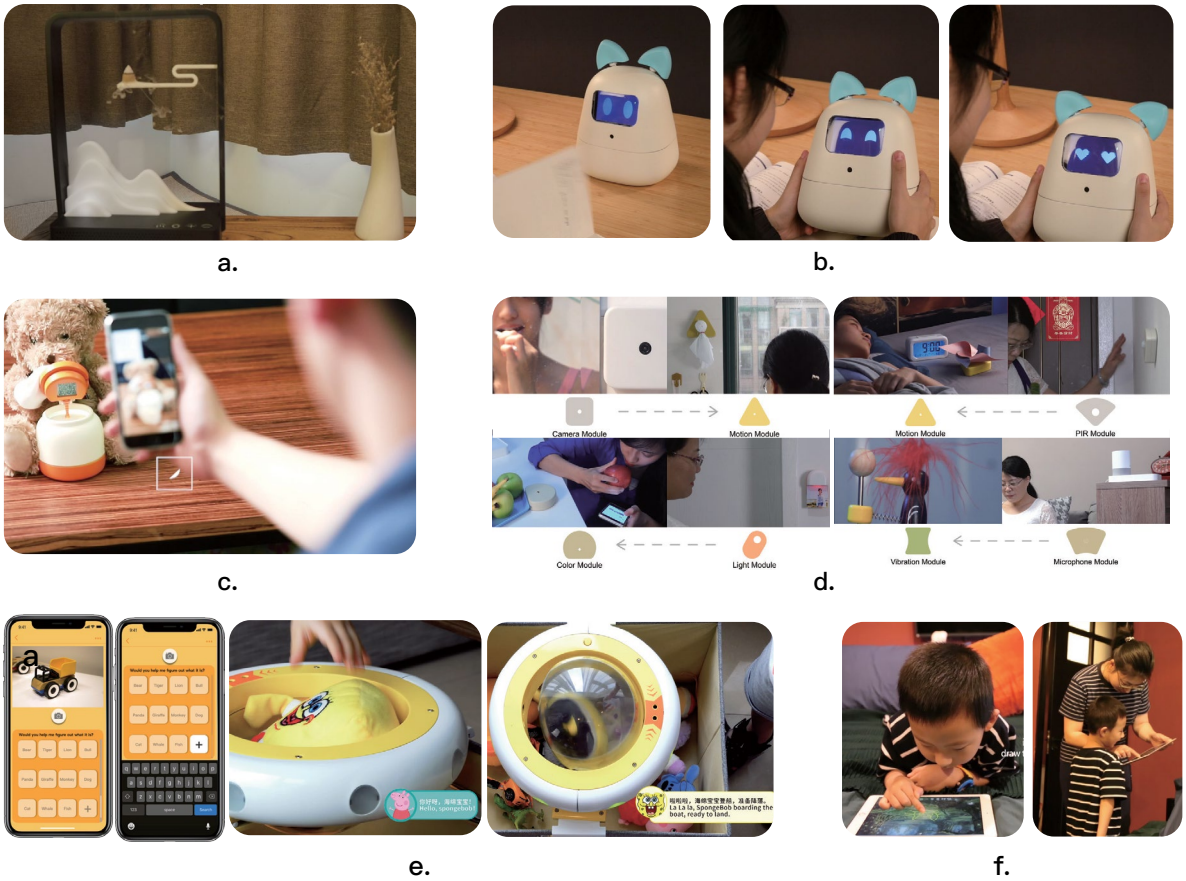


Figure 9. Case studies of the design outcomes that resulted from ML-Rapid.

idea? Why? (3) Do you think that ML-Rapid and its approach help you understand ML technology? Which part of them is helpful or helpless? Why? (4) What do you think are the shortcomings of the current ML-Rapid and its approach? We also collected feedback from the guidance team (G1 and G2) on the participants' understanding of ML at different stages of the project. We applied a thematic analysis process (Howitt & Cramer, 2010) to summarize the semi-structured interview results into three sub-themes. The analysis started with text transcription, familiarization with the data, initial coding generation, and contained an iterative process of searching for potential themes based on the coding and asked questions, review of the themes and codes, theme labeling, etc.

Ease-of-Use of Prototyping Using ML-Rapid

Most of the participants were positive regarding the ease of development using ML-Rapid. For example, "Comparing with other solutions like TensorFlow, I (P6) was greatly surprised that ML-Rapid can simplify the whole working process, and I do not need to do the dirty work." P1, P8, P12, and P17 were pleased with the development speed, for example, "The function within the ML-Rapid was clear so that it saved me (P8) much time writing code and I could try more alternatives using the toolkit." "The IDE of ML-Rapid looks familiar to me (P17), I effortlessly learned how to use this IDE and debug my code" A few participants (P1, P13, and P19) also appreciated the NCS because it "decreases the time consumed in the inference phase" (P1). Some participants (P1, P5, P16, and P29) thought that, with the support of Raspberry Pi and other components, they could make a physical prototype effortlessly. P7 and P18 thought that a more integrated IDE was required because the configuration of the environment remained an annoying problem.

Flexibility of Prototyping with ML-Rapid

Nearly all participants appreciated that ML-Rapid provided the core necessary functionality and flexibility for the prototyping of an ML-empowered product. Some participants (P9, P20, and P25) reported that, although ML-Rapid was not as simple to use as no-programming toolkits, it accomplished different types of applications, which provided sufficient space for exploration. P11 and P19 suggested that more modules containing different applications may need to be developed as soon as possible because having only six modules limited "the problems designers can solve" (P11) and "somehow limited the designers' imagination" (P19). P2 and P4 found the toolkit to be a little constrained, precisely because it is tied to specific models. They wanted to build a fully customizable model that provides a solution that is more suited to their projects.

Improved ML Literacy

All participants stated that they gained a better understanding of ML during the project. Five participants most appreciated the tutorial on the toolkit, and seven more participants appreciated the toolkit practice phase and iteration phase working with ML-Rapid. For example, "Trying to prototype with ML-Rapid let me (P29) know if a solution is feasible and what kind of feeling it will ultimately give users." "I realized how the data is labeled and used for training of ML models (P8)." "I know why ML is strongly dependent on labeled data and computing resources (P12)." Both G1 and G2 believed that the practice parts were the stages in which participants most effectively learned about ML because "participants' novel practice inspire other participants even though they are involved in different projects" (G1) and "the participants often come up with unexpected ideas during aimless attempts" (G2). Eight participants thought that their main gain was from the regular discussion. Some believed that team members helped them understand the overall ML process by "solving detailed problems, such as the environmental configuration" (P9) and "addressing specific concepts" (P10). Another five participants mentioned the design case collection, mainly because it allowed them to "develop a detailed understanding of the entire industry" (P3).

Discussions

The multi-stage study produced a range of specific outcomes: a toolkit that comprises six modules and its workflow, a design project based on the approach of applying the toolkit in design practice, and six ML-empowered product prototypes. The evaluation results showed that designers who are new to ML programming can work on a real-world ML-empowered design project and increase their ML literacy with the help of ML-Rapid and the design process.

With regard to the functioning of ML-Rapid, the feedback revealed that almost all participants successfully prototyped innovative ideas following the main steps of the ML process. To compromise between flexibility and ease-of-use, ML-Rapid simplifies the codes and manipulation, and encourages designers to participate in the prototyping process of ML functionality. Those steps that are deemed to be unnecessary for designers to know are hidden, such as building network models layer by layer. However, ML-Rapid has some limitations compared with existing tools. The comparisons between prior work and ML-Rapid are discussed as below.

Compared with Tools for Designers

Existing tools for designers (non-programming tools, e.g., the Delft AI toolkit) encourage designers without programming skills to capture the ML-user interaction through a non-programming interface. However, the main goal of ML-Rapid is functionality instead of interactivity. Thus, ML-Rapid reveals the learning process of ML through clear functions and limited lines of coding. We illustrate the advantages and limitations of ML-Rapid as below when compared against the tools for designers.

The advantages of ML-Rapid include 1). The workflow of ML-Rapid does not simply hide the working mechanism but provides a workflow that enables designers to participate in essential steps of the ML process via data annotation and model

training/retraining. It provides sufficient space for exploring design possibilities, thus promoting the ML literacy and design innovation. 2) The application of NCS reduces the computation time, thereby allowing a smooth interaction between users and ML prototypes. The NCS would assist designers in building a real-time prototype. 3) The sensors and actuators of ML-Rapid are easy to be modified and extended since these hardware components of Raspberry Pi can be directly enabled through existing open-source Python modules. The flexibility of manipulating hardware components would help designers test the functionality of prototypes.

The limitations of ML-Rapid include 1). ML-Rapid would disappoint the users without any programming skills because ML-Rapid requires users to input codes to participate in the ML process. Meanwhile, the provided hardware needs to be enabled by writing a few lines of codes rather than directly plug-and-play. 2). The coding interface of ML-Rapid is not as intuitive as the visual graph interface of non-programming tools like the Delft AI toolkit.

Compared with Tools for Developers

Tools for developers (programming tools and MLaaS) are designed to support the development of various ML applications via intensive coding. Different from these tools, ML-Rapid serves the purpose of assisting designers in building the physical prototype at the early stage, instead of the professional development of ML application. To achieve such a purpose, the ML-Rapid provides a workflow and IDE that strikes a balance between flexibility and ease-of-use. We illustrate the advantages and limitations of ML-Rapid as below when compared against the tools for developers.

The advantages of ML-Rapid include 1). For designers, ML-Rapid is relatively easy to use while the programming tools and MLaaS tools have high requirements for programming ability. This is mainly because the code is compactly packaged in ML-Rapid so that designers can invoke a function within packaged modules to quickly test their ideas. 2). Compared with the programming tools and MLaaS that are software frameworks and require extra hardware components, ML-Rapid itself provides an extendable hardware platform on integrating the hardware components to build physical artifacts.

The limitations of ML-Rapid include: To guarantee the ease-of-use, ML-Rapid is not as flexible as programming tools. With the provided different optional ML models to choose from, designers can train or retrain the models with their own data. However, they would not be completely able to customize the structure of ML models, which might restrict the potential innovation and the further understanding of ML.

Furthermore, the opinions of participants also encouraged us to improve ML-Rapid in future development: 1) To solve the problems caused by the environment configuration, a more integrated IDE is in need; 2) Since the modules tied to specific ML applications might limit the designers' imagination, the creation of an open-source community is worthwhile in terms of attracting developers to transform more modules; 3) More friendly manipulation methods could be applied in the future ML-Rapid to make it much more intuitive to use.

In terms of the approach, the six prototypes demonstrate that our design process is an appropriate approach to inspiring designers to innovate in ML-empowered product design. The interviews revealed that the practice-oriented parts of the design process were popular among participants and helped participants to gain an increased ML literacy through verifying their own ideas and getting feedback. The communication-oriented and reflection-oriented parts of the design process can inspire design opportunities because they make it easier for participants to identify blind spots and receive objective comments.

Conclusions

In this paper, we proposed an ML prototyping toolkit called ML-Rapid and an approach to applying the toolkit in design practice. Our study identified the ML technical applications that are attractive to designers, explored the difficulties that designers may encounter when participating in ML development, and concluded with an appropriate workflow to enable designers to work with ML. We then developed ML-Rapid, which transforms various types of attractive ML applications into easy-to-use modules. The insights acquired from our study were then summarized as a set of approaches to applying ML-Rapid in DIP project. The evaluation of the DIP showed that our reasonable design process and toolkit can help designers to learn more about ML and innovate in the design of ML-empowered products. ML-Rapid provides a workflow that helps designers to participate in the main ML process, explore possibilities, and cultivate ML literacy.

Although ML-Rapid and its inner workflow are at a relatively rudimentary stage, they take us one step closer to the goal of lowering the barrier to the prototyping of ML-empowered products for designers. Moreover, the evaluation results might inspire continued investigations. In future work, we will improve ML-Rapid and create an open-source community to attract more interesting and promising applications.

Acknowledgment

We would like to thank all participants attending the DIP project and the experiments, as well as the reviewers and editors who provided valuable comments. Besides, we appreciate the efforts of Li Zhuoshu and Fan Yitao in helping us conduct the research. This project was supported by the National Natural Science Foundation of China (No. 61672451), the Provincial Key Research and Development Plan of Zhejiang Province, China (No. 2019C03137), the China Postdoctoral Science Foundation (No. 2018M630658), the Application of Public Welfare Technology in Zhejiang Province (LGF18F020005), and the Ng Teng Fong Charitable Foundation in the form of ZJU-SUTD IDEA Grant.

References

1. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., & Isard, M. (2016). Tensorflow: A system for large-scale machine learning. In *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation* (pp. 265-283). Berkeley, CA: USENIX Association.
2. Amershi, S., Cakmak, M., Knox, W. B., & Kulesza, T. (2014). Power to the people: The role of humans in interactive machine learning. *AI Magazine, 35*(4), 105-120.
3. Amershi, S., Cakmak, M., Knox, W. B., Kulesza, T., & Lau, T. (2013). IUI workshop on interactive machine learning. In *Proceedings of the International Conference on Intelligent User Interfaces Companion* (pp. 121-124). New York, NY: ACM.
4. Barragán, H. (2004). *Wiring: Prototyping physical interaction design.* Ivrea, Italy: Interaction Design Institute.
5. Behoora, I., & Tucker, C. S. (2015). Machine learning classification of design team members' body language patterns for real time emotional state detection. *Design Studies, 39*(6), 100-127.
6. Bullock, J., & Momeni, A. (2015). Ml.lib: Robust, cross-platform, open-source machine learning for max and pure data. In *Proceedings of the International Conference on New Interfaces for Musical Expression* (pp. 265-270). Baton Rouge, LA: dblp. http://doi.org/10.5281/zenodo.1179038
7. Daalhuizen, J., & Schoormans, J. (2018). Pioneering online design teaching in a MOOC format: Tools for facilitating experiential learning. *International Journal of Design, 12*(2), 1-14.
8. Dove, G., Halskov, K., Forlizzi, J., & Zimmerman, J. (2017). UX design innovation: Challenges for working with machine learning as a design material. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (pp. 278-288). New York, NY: ACM.
9. Fails, J. A., & Dan, R. O. (2003). Interactive machine learning. In *Proceedings of the International Conference on Intelligent User Interfaces* (pp. 39-45). New York, NY: ACM.
10. Fiebrink, R., & Cook, P. R. (2010). *The Wekinator: A system for real-time, interactive machine learning in music.* Retrieved from http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.294.6659&rep=rep1&type=pdf
11. Fiebrink, R. A. (2011). *Real-time human interaction with supervised learning algorithms for music composition and performance.* Princeton, NJ: Princeton University.
12. Genç, Ç., Buruk, O. T., Yılmaz, S. İ., Can, K., & Özcan, O. (2018). Exploring computational materials for fashion: Recommendations for designing fashionable wearables. *International Journal of Design, 12*(3), 1-19.
13. Gillies, M., Fiebrink, R., Tanaka, A., Garcia, J., Bevilacqua, F., Heloir, A., & Lee, B. (2016). Human-centred machine learning. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Ext. Abs., pp. 3558-3565). New York, NY: ACM.
14. Guo, Y. R., & Goh, D. H.-L. (2016). From storyboard to software: User evaluation of an information literacy game. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing* (pp. 199-201). New York, NY: ACM.
15. Hebron, P. (2016). *Machine learning for designers.* Newton, MA: O'Reilly Media.
16. Hodges, S., Villar, N., Scott, J., & Schmidt, A. (2012). A new era for ubicomp development. *Pervasive Computing IEEE, 11*(1), 5-9.
17. Howitt, D., & Cramer, D. (2010). *Introduction to qualitative methods in psychology* (Vol. 1). London, UK: Pearson Education.
18. Intel. (2018). *Intel neural compute stick 2.* Retrieved from https://software.intel.com/en-us/neural-compute-stick
19. Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual losses for real-time style ransfer and super-resolution. In *Proceedings of the European Conference on Computer Vision* (pp. 694-711). Berlin, Germany: Springer.
20. Kensing, F., & Blomberg, J. (1998). Participatory design: Issues and concerns. *Computer Supported Cooperative Work, 7*(3), 167-185.
21. Lewis, M., Sturdee, M., & Marquardt, N. (2018). Applied sketching in HCI: Hands-on course of sketching techniques. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Ext. Abs., No. C08). New York, NY: ACM.
22. Lim, Y. K., Stolterman, E., & Tenenberg, J. (2008). The anatomy of prototypes: Prototypes as filters, prototypes as manifestations of design ideas. *ACM Transactions on Computer-Human Interaction, 15*(2), No. 7.
23. Lovejoy, J. (2018). *The UX of AI.* Retrieved from https://design.google/library/ux-ai/
24. Martin, L. (2015). The promise of the maker movement for education. *Journal of Pre-College Engineering Education Research, 5*(1), 30-39.
25. McCardle, J. R. (2002). The challenge of integrating AI & smart technology in design education. *International Journal of Technology & Design Education, 12*(1), 59-76.
26. McGlashan, A. (2017). A pedagogic approach to enhance creative ideation in classroom practice. *International Journal of Technology and Design Education, 28*(2), 377-393.
27. Mierswa, I., Wurst, M., Klinkenberg, R., Scholz, M., & Euler, T. (2006). YALE: Rapid prototyping for complex data mining tasks. In *Proceedings of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 935-940). New York, NY: ACM.
28. Mitchell, T., Buchanan, B., DeJong, G., Dietterich, T., Rosenbloom, P., & Waibel, A. (1990). *Machine learning. Annual Review of Computer Science, 4,* 417-433.
29. Nasrabadi, N. M. (2007). Pattern recognition and machine learning. *Journal of Electronic Imaging, 16*(4), 049901. https://doi.org/10.1117/1.2819119
30. O'Connor, A., Seery, N., & Canty, D. (2018). The experiential domain: Developing a model for enhancing practice in D&T education. *International Journal of Technology & Design Education, 28*(1), 85-99.

31. Patel, K., Fogarty, J., Landay, J. A., & Harrison, B. (2008). Investigating statistical machine learning as a tool for software development. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (pp. 667-676). New York, NY: ACM.
32. Peppler, K., Halverson, E., & Kafai, Y. B. (2016). *Makeology: Makerspaces as learning environments* (Vol. 1). London, UK: Routledge.
33. Qu, Y., Jia, Y., Qu, T., Chen, Z., Li, H., & Li, W. (2017). Research on interactive prototype design and experience method based on open source. In *Proceedings of the International Conference of Design, User Experience, and Usability* (pp. 633-651). Berline, Germany: Springer.
34. Quattoni, A., Collins, M., & Darrell, T. (2008). Transfer learning for image classification with sparse prototype representations. In *Proceedings of the Conference on Computer Vision and Pattern Recognition* (pp. 1-8). Piscataway Township, NJ: IEEE.
35. Ribeiro, M., Grolinger, K., & Capretz, M. A. M. (2016). MLaaS: Machine learning as a service. In *Proceedings of IEEE International Conference on Machine Learning and Applications* (pp. 896-902). Piscataway Township, NJ: IEEE. http://doi.org/10.1109/ICMLA.2015.152
36. Şendurur, E., Ersoy, E., & Çetin, İ. (2016). The design and development of creative instructional materials: The role of domain familiarity for creative solutions. *International Journal of Technology & Design Education, 28*, 507-522.
37. Vallgårda, A., Boer, L., & Cahill, B. (2017). The hedonic haptic player. *International Journal of Design, 11*(3), 17-33.
38. Van Allen, P. (2018). Prototyping ways of prototyping AI. *Interactions, 25*(6), 46-51.
39. Vandevelde, C., Wyffels, F., Ciocci, M. C., Vanderborght, B., & Saldien, J. (2015). Design and evaluation of a DIY construction system for educational robot kits. *International Journal of Technology & Design Education, 26,* 521-540.
40. Witten, I. H., Frank, E., Hall, M. A., & Hall, M. A. (2011). *Data mining: Practical machine learning tools and techniques.* Burlington, MA: Morgan Kaufmann.
41. Yang, Q. (2017). *The role of design in creating machine-learning-enhanced user experience* (Tech. Rep. No. SS-17-04). Palo Alto, USA: Association for the Advancement of Artificial Intelligence.
42. Yang, Q., Banovic, N., & Zimmerman, J. (2018a). Mapping machine learning advances from HCI research to reveal starting places for design innovation. In *Proceedings of the CHI Conference on Human Factors in Computing Systems* (Paper No. 130). New York, NY: ACM.
43. Yang, Q., Scuito, A., Zimmerman, J., Forlizzi, J., & Steinfeld, A. (2018b). Investigating how experienced UX designers effectively work with machine learning. In *Proceedings of the Conference on Designing Interactive Systems* (pp. 585-596). New York, NY: ACM.
44. Yu, Z., & Zhang, C. (2015). Image based static facial expression recognition with multiple deep network learning. In *Proceedings of the ACM International Conference on Multimodal Interaction* (pp. 435-442). New York, NY: ACM.