Data Structures for Designing Interactions with Contextual Task Support

Minsuk Chang
School of Computing
KAIST
minsuk@kaist.ac.kr

ABSTRACT

The diversity and the scale of available online instructions introduce opportunities but also user challenges in currently used software interfaces; Users have limited computational resources, and thus often make strategic decisions when browsing, navigating, and understanding instructions to accomplish a task. These strategic user interactions possess nuanced semantics such as users' interpretations, intents, and contexts in which the task is carried out. My dissertation research introduces techniques in constructing data structures that capture the diverse strategies users employ in which the collective nuanced semantics across multiple strategies are preserved. These computational representations are then used as building blocks for designing novel interactions that allow users to effectively browse and navigate instructions, and provide contextual task guidance. Specifically, I investigate 1) structure of instructions for task analysis at scale, 2) structure of collective user task demonstrations, and 3) structure of object uses in how-to videos for tracking, guiding and searching task states. My research demonstrates that the user-centered organization of information extracted from interaction traces enables novel interfaces with contextual task support.
structure of objects in how-to videos

Author Keywords

computational representations; data-driven interface design; task support; semantic search; learning in the wild

INTRODUCTION

From cooking recipes to software tutorials, people are increasingly turning to online instructions as guides for learning new skills or accomplishing unfamiliar tasks in everyday lives. However, the number of available instructional materials even for a single task is easily in the magnitude of thousands. The increasing availability of instructional material online does not automatically translate into more learning opportunities or better accessibility of instructions, but quite the opposite in current interfaces. Interfaces we use everyday for browsing, navigating, and following instructions are designed with little consideration of usersâĂŹ limited computational resources like cognitive capacity and time constraint, users have trouble finding the contextually useful information to accomplish their tasks.

Users often make strategic decisions under these interface bounds. For example, in traditional search interfaces like recipe search websites, users often go back and forth between search results of thousand instructions and individual pages of one specific instruction. In traditional task-specific software like 3D modeling tools, users have to decide whether to get the task done using already acquired workflows or invest time to learn how to do the task more efficiently. Also, in a traditional video interface like YouTube, users have to use hands for both controlling the playback of tutorial on the player interface and the task at hand. As a result, not only the task outcome is jeopardized but the task experience is also compromised.

I develop techniques in data-driven interface design to amplify people's ability to browse, navigate, understand instructions to accomplish tasks. My research is motivated by the observation that strategic user interactions with hundred thousands of naturally crowdsourced instructions possess user semantics such as users' interpretations, intents, and contexts in which learning happens.

My dissertation research introduces techniques in constructing data structures that capture the diverse strategies users employ in which the collective nuanced semantics across multiple strategies are preserved. These computational representations are then used as building blocks for designing novel interactions that allow users to effectively browse, navigate instructions, and provide contextual task guidance.

In my work, I introduce techniques in modeling the following three data structures for designing data-driven interfaces that enable contextual task support:

1. structures of learning (instruction) materials to support contextual and nuanced search for instructional materials.

2. structures of individual and collective learner workflow to support informal learning opportunities.

3. structures of objects that appear in instructional material to support semantic navigation of tutorials.

UIST'19, October 20–23, 2019, New Orleans, LA, USA

 ISBN 123-4567-24-567/08/06. . . 15.00

DOI: **10.475/123_4**

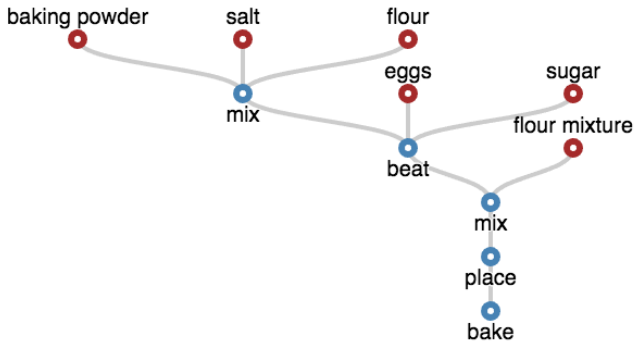

Figure 1: An example of a recipe modeled into a tree structure.

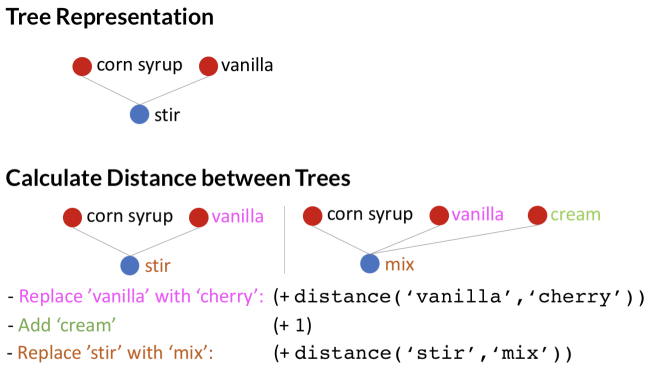

Figure 2: RecipeScape pipeline which uses weighted tree edit distance to calculate structural and semantic similarities between the recipes.

RECIPE TREE - CAPTURING STRUCTURAL AND SEMANTIC SIMILARITIES BETWEEN RECIPES

RecipeScape (https://recipescape.kixlab.org) [1, 2] is an analytics dashboard for analyzing and mining hundreds of instructions for a single dish. The graphical interface was designed to solve a critical problem for cooking professionals, the need to understand not only the composition of ingredients but more importantly the diverse cooking processes.

The computational pipeline (Figure 2) that collects, annotates, computes structural and semantic similarities, and visualizes hundreds of recipes for a single dish uses a tree structure for each recipe (Figure 1).

Representing each recipe as a tree is a deliberate modeling decision, because a weighted tree edit distance can be used to calculate structural and semantic similarities between the recipes.

Cooking professionals and culinary students found that RecipeScape 1) empowers users with stronger analytical capabilities by enabling at-scale exploration of instructions, 2) supports user-centered queries like "what are recipes with more decorations?", and 3) supports creativity by allowing comparative analysis like "where would my recipe stand against the rest?". Moreover, visualization of the computational models allow users to reason and provide their own interpretations and explanations of how the recipes are grouped together. The RecipeScape pipeline illustrates how a graphical representation that captures domain-specific semantics (i.e. cooking) and the structural semantics (procedural instruction) of learning materials enable interpretable interactions for at-scale analysis and learning.

As an immediate future work, I'm exploring characteristics of an ideal data structure for causal inferences on a large set of instructions, so the representation enables interfaces with adaptive support. For example, if a configuration changes like substituting an ingredient, the interface can adapt and update the following steps. Sometimes the adaptation might not be substantial like changing the entire cooking method, sometimes it can be rather minimal like increasing the temperature for cooking. In either cases, it would require the structure to fully capture the causal effects of each steps. Identifying the confounding factors from web-scale instruction data and building a structural representation in causal relationships would be an exciting direction for both causal inference community and HCI.

WORKFLOW GRAPHS - CAPTURING MULTIPLE TASK DEMONSTRATION OF A SHARED GOAL

Workflow can support the generation of tutorials and other learning content, building on past work exploring approaches for generating tutorials by demonstration [6, 5]. The motivation for developing Workflow graphs came from informally observing online communities for 3D modeling software, where it is common for users to discuss many different methods of completing a single modeling task.

Workflow graph representation is a synthesis of many demonstrations of a given task (i.e., re-creating the same 3D model) such that the commonalities and differences between the approaches taken by users are apparent. Moreover, to ensure the technique can scale, the goal is to automate the construction process, based on recordings of demonstrations of the task (which may include screen recordings, command log data, content snapshots, etc.)

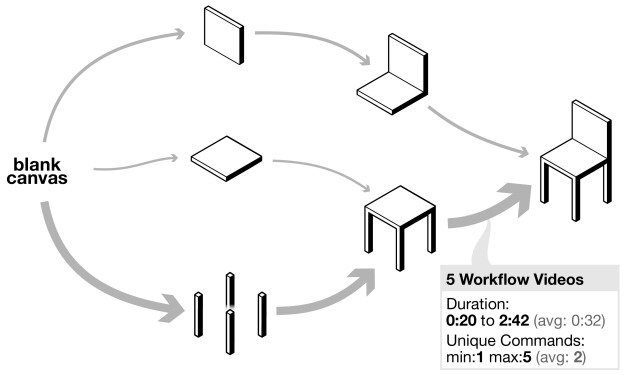

Figure 3: Workflow Graphs encode multiple demonstrations of a fixed task, based on commonalities in the workflows employed by users. Nodes represent semantically similar states across demonstrations. Edges between states represent alternative workflows for sub-tasks, in which the widths represent the number of distinct workflows between the state.

Graph Nodes

The vertices of the graph represent semantically-meaningful states in the demonstrations (Figure 3). These states can be thought of as sub-goals in the workflow— we want them to capture the points where a user has completed a given sub-task, and has yet to start the next sub-task. Detecting these states automatically from unlabeled demonstrations is a challenge, but by leveraging the demonstrations of multiple users we can discover common states that occur across their respective methods for completing the task. If a new demonstration is completely different from those already represented in the graph, it might not share any nodes with those already in the graph, apart from the start and final nodes, which are shared by all demonstrations.

Graph Edges

The directed edges of the graph represent workflows used by a user to move between semantically-similar states. There may be multiple directed edges between a given pair of states, if multiple demonstrations include a segment from one state to another.

The interaction trace data associated with edges enables a great deal of flexibility in how the Workflow graph is used. For example, this data could be used to retrieve snippets of screen recordings of the demonstrations associated with the segment of the task between two states, or it could be used to define metrics on the different workflows used for that segment of the task (e.g., the number of unique commands used, or the average time it takes to perform the workflow). As another example, analyzing the interaction traces along many different paths between states can reveal the average time for sub-tasks or the variance across users. By representing the structure of how to perform a task, workflow graph can serve as a back-end for a number of novel interface mechanisms in addition to suggestion of alternative approaches for sub-tasks.

TASK-OBJECT GRAPHS - CAPTURING THE RELATIONSHIPS OF OBJECTS IN HOW-TO VIDEOS

YouTube searches for how-to videos are growing 70% year to year, people watched 100 Million Hours in just the year of 2015 in North America. Instead of watching these how-to videos passively, viewers actively control the video to pause, replay, and skip backwards or forwards while following along with the video instructions.

Using traditional mouse and keyboard interfaces, viewers commonly navigate tutorial videos by either sequentially scrubbing through the video timeline to examine the preview thumbnails click-guessing on the timeline until the resulting video frame matches the desired point of interest.

However, many how-to videos teach physical tasks that involve interaction with real world objects. When following along with such videos, viewers need their hands both to execute the task and to control the video. Since they can't do both at once, frequently they must stop concentrating on the task itself to instead concentrate on controlling the video, and vice versa. In having to alternate between these two operations, viewers incur a costly context switch.

Voice-based user interfaces (e.g., Apple Siri, Amazon Alexa, and Google Assistant) are becoming increasingly ubiquitous in commercial devices and provide a potential alternative for controlling how-to videos of physical tasks. While these systems provide some help in the context of how-to videos, they are not specifically designed for learning, rather, they directly translate of low-level remote control operations (pause, play, etc.) into voice commands.

An important question is whether this low-level remote-control-like interface is suitable for voice-driven video navigation interfaces for how-to videos, and if not, how should a useful voice interface for navigating how-to videos be designed?

Navigational Needs

Two specific patterns have emerged from our studies [3], pause interactions and jump interactions

The most common type of pause was a pause to gain more time. This happens when the user understands the video content but fails to match the pace of the video. The second type of pause is a pause to compare what's in the video with what's in the hands of the user. This pause precedes the user checking to make sure that their state is similar to that of the video. The final type of pause we observed is a pause for further video control. In this case, the user pauses the video and searches for the next navigation target point on the timeline by either guess-clicking, or scrubbing.

The first type of jump we observed is a reference jump. In this case, the user jumps backwards in the video to remind themselves of something they saw in the past. A replay jump is a different form of backward jump, where the user wants to re-watch a segment of the video again. This jump happens when the user needs to get a better understanding, clarify a possible mistake, or to assure that the current understanding is correct. This jump if often followed by a play or a pause interaction. A skip jump is a type of forward jump where the user wants to skip content that is less interesting, like the introduction of the channel or the personal life of the YouTuber. The second type of forward jump is a peek jump, where the user wants to skip ahead to see what the user should expect after performing one or a number of steps. This happens when users want to check the intermediate or the final result in order to prepare and also check if the user is on the right track.

Design Goals as Task-Object Graph Modeling Objectives

Designing voice interaction for how-to videos is not about supporting a single command, but understanding the higher level user intent behind the utterance is crucial. One possible solution in distinguishing the 7 intents, and to set up the command vocabulary such that each intent has its unique keyword.

Users often need multiple tries to find the intended navigation target, because 1) what users remember can be different from the part they are looking for or vice versa, 2) sometimes users don't remember, 3) sometimes users remember but don't know the exact vocabulary like the names of knitting techniques and tools, and 4) users just do not know how to reference unseen parts of the video.

A data structure that can support these interactions is a graph where nodes are objects appear in the tutorial, and the edges are the operational relationships like actions. I am exploring

techniques for constructing such data structure from multiple videos of the shared task. This is a promising direction as structural representation of the how-to video can be used as an inductive bias for processing users' natural language commands, as shown to be successful in other domains [7, 4].

Since the graph captures the role of the objects via its edges, it will also allow features like comparing the progress or the state of the user and those of the video by supporting various examination features like zoom or taking screenshots.

A promising extension would be to have an internal representation of a user utterance, and to map components of the sequence of tokens in user utterance to the the Task-object graph. It would allow features like keeping a pointer to the origin of a jump and providing a comeback, suggesting updates or narrowing down of the interval of jump to make the subsequent search processes easier.

If we construct Task-object graph for multiple videos, then users navigate across videos by populating snippets that contain the object and action of interest.

The user utterances for navigation within an How-to video can inform where objects appear in what context, it provides another channel for constructing (or fine-tuning) the Task-object graph. With this vision, I am currently collecting more natural conversational utterances for references users make while watching how-to videos.

FUTURE DIRECTIONS AND CONCLUSION

With my research, I aim to investigate techniques for organizing user interactions in data structures, such that algorithms and computational capabilities for the structures enable and maximize contextual task support in user interfaces. My research will continue to lower barriers for users to consume, and produce instructional materials.

I am eager to bring the techniques and systems I build into the real world, and also to make these techniques generalizable and accessible to other application areas. With scalability in mind, I also aim to explore algorithmic approaches in "learning" the structures from large-scale examples using probabilistic graphical models.

In a long term, I wish to be able to provide a computational understanding of the usability of an interface by analyzing the resulting data structures and how they evolve as more users come to the systems. Examining and comparing the topological characteristics of different tasks under an interface will allow us understand the relationship between user task and the interface bounds. I believe it will be a powerful tool for complementing interface design practice and research. I would love to discuss the overall framing that ties together the main themes and goals of my diverse projects into a valuable dissertation, and the feasibility and the potential of future directions at the UIST Doctoral Symposium.

ACKNOWLEDGEMENTS

I am grateful to my advisor Juho Kim, as well as my collaborators and mentors, Maneesh Agrawala, Oliver Wang, Mira Dontcheva, Ben Lafreniere, Tovi Grossman their continual support and feedback. My research is supported in part by KAIST, Stanford University, Adobe, and Autodesk.

REFERENCES

1. Minsuk Chang, Léonore V Guillain, Hyeungshik Jung, Vivian M Hare, Juho Kim, and Maneesh Agrawala. 2018. RecipeScape: An Interactive Tool for Analyzing Cooking Instructions at Scale. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. ACM, 451.
2. Minsuk Chang, Vivian M Hare, Juho Kim, and Maneesh Agrawala. 2017. Recipescape: Mining and analyzing diverse processes in cooking recipes. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems*. ACM, 1524–1531.
3. Minsuk Chang, Anh Truong, Oliver Wang, Maneesh Agrawala, and Juho Kim. 2019. How to design voice based navigation for how-to videos. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, 701.
4. Qian Chen, Xiaodan Zhu, Zhenhua Ling, Si Wei, and Hui Jiang. 2016. Enhancing and combining sequential and tree lstm for natural language inference. *arXiv preprint arXiv:1609.06038* (2016).
5. Pei-Yu Chi, Sally Ahn, Amanda Ren, Mira Dontcheva, Wilmot Li, and BjÃűrn Hartmann. 2012. MixT: Automatic generation of step-by-step mixed media tutorials. In *Proceedings of the 25th annual ACM symposium on User interface software and technology (UIST '12)*. ACM, 93–102.
6. Floraine Grabler, Maneesh Agrawala, Wilmot Li, Mira Dontcheva, and Takeo Igarashi. 2009. Generating photo manipulation tutorials by demonstration. *ACM Trans. Graph.* 28, 3 (July 2009), 66:1–66:9. DOI: http://dx.doi.org/10.1145/1531326.1531372
7. Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* (2015).