Learnersourcing: Improving Learning with Collective Learner Activity

by

Juho Kim

Submitted to the Department of
Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2015

Author . . . .
Department of
Electrical Engineering and Computer Science
August 7, 2015

Certified by . . . .
Robert C. Miller
Professor
Thesis Supervisor

Accepted by . . . .
Leslie A. Kolodziejski
Chair, Department Committee on Graduate Students

Learnersourcing: Improving Learning with Collective Learner Activity

by

Juho Kim

Submitted to the Department of
Electrical Engineering and Computer Science
on August 7, 2015, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy

Abstract

Millions of learners today are watching videos on online platforms, such as Khan Academy, YouTube, Coursera, and edX, to take courses and master new skills. But existing video interfaces are not designed to support learning, with limited interactivity and lack of information about learners' engagement and content. Making these improvements requires deep semantic information about video that even state-of-the-art AI techniques cannot fully extract. I take a data-driven approach to address this challenge, using large-scale learning interaction data to dynamically improve video content and interfaces. Specifically, this thesis introduces learnersourcing, a form of crowdsourcing in which learners collectively contribute novel content for future learners while engaging in a meaningful learning experience themselves. I present learnersourcing applications designed for massive open online course videos and how-to tutorial videos, where learners' collective activities 1) highlight points of confusion or importance in a video, 2) extract a solution structure from a tutorial, and 3) improve the navigation experience for future learners. This thesis demonstrates how learnersourcing can enable more interactive, collaborative, and data-driven learning.

Thesis Supervisor: Robert C. Miller
Title: Professor

Acknowledgments

This thesis and my graduate school career would not have been possible without support from advisors, mentors, colleagues, friends, and family. Specifically, I would like to thank:

- My co-advisor Rob for inspiring me with his passion, energy, intellectual curiosity, and reliability. His low-bar, rapid, and constant experimentation with "Stupid" ideas has taught me how to make iterative improvements not only in research, but also in life.
- My co-advisor Krzysztof for the great conversations and the hard questions. He encouraged me to aim for what is more meaningful and exciting, rather than settle down with what is familiar and easy to accomplish.
- Frédo for being on my thesis committee. His insightful and sharp comments have significantly improved the way I frame my research.
- Friends and colleagues in the UID Group at MIT, a community of wonderful folks building interactive systems that bridge humans and computers: Eirik Bekke, Michael Bernstein, Carrie Cai, Tsung-Hsiang Chang, Elena Glassman, Max Goldman, Philip Guo, Abby Klein, Geza Kovacs, Rebecca Krosnick, Tom Lieber, Greg Little, Adam Marcus, Kyle Murray, Katrina Panovich, Hubert Pham, Michele Pratusevich, Chen-Hsiang (Jones) Yu, Amy Zhang, and Haoqi Zhang
- Friends and colleagues in the IIS Group at Harvard: Elena Agapie, Ofra Amir, Ken Arnold, Anna Huang, Na Li, Sunyoung Kim, Steven Komarov, Katharina Reinecke, Kanya (Pao) Siangliulue, and Joseph Williams
- Undergrad advisees Sarah Weir, Phu Nguyen, and Peter Githaiga for giving me the joy of mentoring
- Internship mentors at Microsoft Research, edX, Adobe Research, and IBM Research for helping me expand the boundaries of my research: Merrie Morris, Andrés Monroy-Hernández, Piotr Mitros, Joel Brandt, Mira Dontcheva, Eser Kandogan, and Tom Moran

- Collaborators for joining me in exploring exciting research ideas: Daniel Li, Steven Dow, Paul André, Anant Bhardwaj, Lydia Chilton, Nam Wook Kim, Eun-Young Ko, Jonghyuk Jung, Chang Won Lee, and Blair Han
- Joon Lee for introducing me to this wonderful field
- Thousands of Turkers for helping me with my research and also for their crowd-sourced opinions and feedback
- The Samsung Scholarship for the opportunity to make respectable friends and for seven years of financial support
- Friends in EECS KGSA for sharing beer with me
- MIT Chamber Chorus for helping me stay sane and get a Ph.D. minor in music performance
- Aeropress for keeping me awake before paper deadlines
- My friends and family for their support
- Jaehun and Yunjin for sharing many lifelong memories with me
- My parents Jung Hye Jeh and Jae Geun Kim for their unconditional care and love
- My wife Jihee for always being there for me. I'm extremely grateful to have her in my life. She makes me believe we can change the world together.

The research projects presented in this thesis were supported in part by Quanta Computer and NSF award SOCS-1111124.

Contents

Cover page **1**

Abstract **3**

Acknowledgments **4**

Contents **7**

List of Figures **15**

List of Tables **23**

1 Introduction to Learnersourcing **25**

1.1 Background 27

1.2 Learnersourcing: Crowdsourcing with Learners as a Crowd 29

1.3 Passive and Active Learnersourcing 30

1.3.1 Passive Learnersourcing: Natural learner interactions improve video learning 31

1.3.2 Active Learnersourcing: Learner prompts contribute to new learning materials 33

1.4 Contributions 37

1.5 Thesis Overview 37

2 Related Work **39**

2.1 Crowdsourcing and Human Computation 39

2.1.1 Crowdsourcing Workflows 41
2.1.2 Crowdsourced Data Annotation 41
2.2 Interaction Data Processing 42
2.2.1 Modeling Collective User Data 43
2.2.2 Tools for temporal pattern analysis 43
2.3 Learning Science 44
2.3.1 Video Learning Theories 45
2.3.2 Learning Procedural Knowledge 45
2.4 Video Applications 47
2.4.1 Video Interfaces for Learning 47
2.4.2 Video Content Analysis 48
2.4.3 Video Navigation Techniques 48
2.4.4 Video Annotation 50
2.4.5 Interactive Tutorials 51

3 In-Video Dropouts and Interaction Peaks **53**

3.1 Motivation and Contributions 54
3.2 Video Interaction Dataset 55
3.2.1 Data Processing Pipeline 56
3.3 Analysis 1. In-Video Dropout 57
3.3.1 Method 57
3.3.2 Results 57
3.4 Analysis 2. Interaction Peaks 60
3.4.1 Method 61
3.4.2 Peak Profile Comparison 63
3.5 Analysis 3. Five causes for peaks 65
3.5.1 Method 67
3.5.2 Results 67
3.6 Design Implications for MOOC Video Interfaces 74
3.6.1 MOOC video analytics platform 76

3.7 Limitations . . . 76
3.8 Conclusion . . . 77

4 LectureScape: Data-Driven Navigation of Educational Videos 79
4.1 Motivation and Contributions . . . 80
4.2 Design Goals . . . 83
4.3 Data-Driven Video Navigation Techniques . . . 84
4.3.1 The Rollercoaster Timeline: 2D, Non-Linear Timeline . . . 85
4.3.2 Keyword Search and Visualization . . . 87
4.3.3 Video Summarization with Highlights . . . 91
4.4 LectureScape: Web-Based Prototype . . . 93
4.5 Evaluation . . . 94
4.5.1 Study Design . . . 95
4.5.2 Participants . . . 96
4.5.3 Procedure . . . 97
4.6 Results . . . 97
4.6.1 RQ1. Navigation Patterns for Search and Summarization . . . 97
4.6.2 RQ2. Perception of Interaction Data . . . 99
4.6.3 RQ3. Perceived Usability of LectureScape . . . 99
4.6.4 Navigation Pattern Analysis . . . 100
4.7 Discussion . . . 102
4.8 Conclusion . . . 103

5 ToolScape: Crowdsourcing Step Labels for How-to Videos 105
5.1 Motivation and Contributions . . . 106
5.2 Efficacy and Challenges of Video Annotation . . . 108
5.2.1 Annotations on How-To Videos . . . 109
5.2.2 ToolScape: Step-Aware Video Player . . . 110
5.2.3 Formative Study Design . . . 110
5.2.4 Formative Study Results . . . 113
5.2.5 Lessons for Video Browsing Interfaces . . . 115

5.2.6 Lessons for How-To Video Annotation 116
5.3 Crowdsourcing Workflow: Find-Verify-Expand 116
5.3.1 Stage 1: Find Candidate Steps 118
5.3.2 Stage 2: Verify Steps 120
5.3.3 Stage 3: Expand with Before and After Images for Steps 123
5.4 Evaluation 123
5.4.1 Methodology 124
5.4.2 Results 124
5.5 Discussion and Limitations 128
5.6 Conclusion 129

6 Crowdy: Learnersourcing Subgoal Labels for How-to Videos 131
6.1 Motivation and Contributions 132
6.2 Learnersourcing Subgoal Labels 134
6.3 Workflow Design 134
6.3.1 Stage 1: Generate 136
6.3.2 Stage 2: Evaluate 137
6.3.3 Stage 3: Proofread 138
6.3.4 Pedagogical Benefits in Learner Prompts 139
6.4 Pilot Study 140
6.4.1 Design Dimensions Refined through Pilot Study 141
6.5 Evaluation Overview 143
6.6 Study 1: Pedagogical Benefits of Learnersourcing 143
6.6.1 Participants 146
6.6.2 Procedure 147
6.6.3 Results 147
6.7 Study 2: Quality of Learnersourced Subgoals 151
6.7.1 Study Structure 151
6.7.2 Deployment Strategy and Participation 152
6.7.3 Evaluation Method 153

6.7.4 Results 155
6.7.5 Interviews with Users 158
6.7.6 Interview with Video Creator 160
6.8 Discussion 160
6.8.1 Crowdy's Video Learning Model 160
6.8.2 Pedagogical Benefits of Learnersourcing 161
6.8.3 Improving the Workflow for Better Subgoal Labels 162
6.8.4 Limitations 165
6.9 Conclusion 165

7 Discussion **167**

7.1 Learnersourcing vs Crowdsourcing 167
7.1.1 Purpose 167
7.1.2 Task Property and Context 168
7.1.3 Design Challenge 169
7.1.4 Goal Visibility 170
7.2 Design Guidelines for Learnersourcing Applications 171
7.2.1 Passive Learnersourcing 171
7.2.2 Active Learnersourcing 172
7.3 Possible Applications Using Learnersourced Artifacts 173
7.3.1 Learner Engagement 173
7.3.2 Video Annotation 174
7.4 Generalizing Learnersourcing 175
7.4.1 Community-driven conference scheduling 175
7.4.2 Budgetary discussion support for taxpayers 175
7.5 Limitations of Learnersourcing 176
7.5.1 Complexity of Tasks and Outputs 177
7.5.2 Data Blindness 177
7.5.3 Privacy 177
7.5.4 Is This a Temporary Solution? 178

8 Conclusion **179**

8.1 Summary of Contributions 179

8.2 Future Directions 180

8.2.1 Innovative Learnersourcing Applications 180

8.2.2 Learning Platforms of the Future 181

8.2.3 Science of Learning at Scale 181

8.2.4 Sociotechnical Application Design 181

Appendix **183**

A Materials for the LectureScape Study **183**

A.1 Study Task Part 1. 183

A.1.1 Task A. 183

A.1.2 Task B. 184

A.1.3 Task C. 184

A.1.4 Task D. 184

A.2 Study Task Part 2. 184

A.2.1 Task A. 185

A.2.2 Task B. 185

A.3 Study Task Part 3. 185

A.3.1 Task 7. (your answer below) 186

A.3.2 Task 8. (your answer below) 186

A.4 Post-Task Questionnaire 186

A.4.1 Workload Assessment 187

A.5 Post-Study Questionnaire 187

A.5.1 Questionnaire for the baseline interface 188

A.5.2 Questionnaire for LectureScape 189

B Materials for the ToolScape Study **193**

B.1 Post-Task Questionnaire 193

B.1.1 Overall Experience 193

B.1.2 Video Browsing Interface 193

C Materials for the Crowdy Study **195**

Bibliography **199**

List of Figures

1-1 Lecture videos on the web show different styles, including a.) classroom lecture, b.) 'talking head' shot of an instructor at a desk, c.) digital tablet drawing format popularized by Khan Academy, and d.) PowerPoint slide presentations. 28

1-2 How-to videos include step-by-step instructions, spanning various domains including cooking, graphical design, home improvement, and applying makeup. 29

1-3 An example interaction peak near a scene transition in a lecture video. . . . 32

1-4 This diagram describes how LectureScape implements passive learnersourcing. LectureScape is a video player that adapts to collective learner engagement. As learners watch a video, the system analyzes collective interaction traces to mine for specific positions that may indicate points of confusion or importance. Based on this information, the video player dynamically changes its behavior to recommend popular frames visited frequently by other learners. 33

1-5 LectureScape: a lecture video player powered by collective interaction data. 34

1-6 ToolScape: a step-aware how-to video player. 34

1-7 This diagram describes how Crowdy implements active learnersourcing. Crowdy is a video player that coordinates learners' collective efforts to generate subgoal labels, which are summarized, abstract descriptions for a group of low-level steps in a procedural task. As learners watch a video, the video player pauses and prompts learners to summarize video sections. The system coordinates learner tasks in multiple stages to reach a final set of summary labels for the clip. The video player dynamically displays the collectively generated video outline as future learners visit and watch the video. 35

1-8 Crowdy: a learnersourcing workflow for summarizing steps in a how-to video. 36

2-1 An example of the breakdown between goal, subgoals, and individual steps for a procedural task. Creating subgoals from individual steps requires self-explanation and summarization. 46

3-1 Longer videos exhibit higher dropout rates. Our linear regression model uses the log-transformed video length (x-axis) to predict the dropout rate (y-axis). The model fits the data well with r=0.55 with 95% CI = [0.50, 0.59]. 58

3-2 Re-watching students tend to drop out more, which might mean that re-watching students watch videos more selectively with a more specific need. 59

3-3 Even after aggregating data into bins of one second, the data is noisy (green curve). Kernel-based smoothing reduces noise in the data and helps salient patterns stand out (black curve). 61

3-4 The location of a peak is determined by three time points (start, peak, and end). Width, height, and area determine the shape, sharpness, and intensity of the peak. 62

3-5 We visualize three streams of data to analyze interaction peaks in MOOC videos. The top graph shows play events, the middle graph shows re-watching sessions, and the bottom graph shows pixel differences over time. Detected peaks are marked with a gray point. In this example, the detected peaks coincide with a spike in pixel differences, which indicate a visual transition in video. 68

3-6 This peak represents the start of a new concept. The instructor started presenting a formal definition of a concept (admissibility) after changing the slide. The peak occurred when this concept explanation started. 69

3-7 This peak represents students returning to see the code snippet slide that disappeared after transitioning into the talking head. An abrupt transition might not give students enough time to comprehend what's presented. . . . 70

3-8 This peak represents students returning to a procedural step demonstrating how to run a command inside a statistics package. Students are more interested in following along the steps than the result afterward, probably because they can see the same result in their own application as well. . . . 71

3-9 This peak represents a short range of interesting segment surrounded by visual transitions before and after. The instructor launched a game application that demonstrates the concept discussed. This engaging demo might have encouraged students to return to it. 72

3-10 This peak represents important remarks from an instructor, without any visual transitions in the video. In this example the instructor was making an important point about random actions in reinforcement learning, the key topic of this AI lecture. 73

3-11 Our prototype video analytics dashboard supports synchronized video playback for various interaction measures. 78

4-1 This chapter presents three sets of novel interaction techniques to improve navigation of educational videos. 1) Dynamic timelines (Rollercoaster Timeline, Interaction Peaks, and Personal Watching Trace), 2) Enhanced in-video search (Keyword Search and Interactive Transcript), 3) Highlights (Word Cloud, Personal Bookmarks, Highlight Storyboard). All techniques are powered by interaction data aggregated over all video watchers. 81

4-2 The 2D Rollercoaster timeline that appears below each video instead of a traditional 1D timeline. The height of the timeline at each point shows the amount of navigation activity by learners at that point. The magenta sections are automatically-detected interaction peaks. 84

4-3 Non-linear scrubbing in the Rollercoaster timeline. To draw the learner's attention to content around interaction peaks, the phantom cursor decelerates scrubbing speed when the cursor enters a peak range. 86

4-4 The real-time personal watching traces visualize segments of the video that the learner has watched. 87

4-5 Our interaction data-driven keyword search brings in more context for the learner to decide where to focus on when searching for a keyword. For instance, the learner can visually check the distribution of when the lecturer said the keyword in the current video, which is useful in seeing where it was heavily discussed versus simply introduced. 88

4-6 The search timeline appears below the video after each search and replaces the Rollercoaster timeline when a search term is entered. It visualizes the positions of search results, as well as the relative importance of each. Here the high peak in the middle indicates both that it contains the search term and that lots of learners watched that part. 89

4-7 Hovering on a search result displays a tooltip with the transcript sentence that contains the search term. Clicking on the result plays the video starting at the beginning of the sentence to assist the learner with comprehending the surrounding context. 89

4-8 The word cloud displays automatically-extracted topics for the currently visible section of the video, providing a keyword-based summarization of the video. Clicking on any word triggers a search for it. 90

4-9 Our pinning algorithm analyzes interaction peaks and visual transitions in the video to display a smaller static frame (on the left) next to the video (on the right). Learners can manually pin any frame as well. 91

4-10 The learner can add a labeled bookmark, which is added to the highlights stream below the video for visual preview and revisitation. 93

5-1 Our crowdsourcing workflow extracts step-by-step information from a how-to video with their descriptions and before/after images. It features the Find-Verify-Expand design pattern, time-based clustering, and text/visual analysis techniques. Extracted step information can be used to help learners navigate how-to videos with higher interactivity. 107

5-2 Progress in many how-to videos is visually trackable, as shown in screenshots from this Photoshop how-to video. Adding step annotations to videos enables learners to quickly scan through the procedure. 109

5-3 How-to videos often contain a series of task steps with visually distinct before and after states. Here the author applied the “Gradient map” tool in Photoshop to desaturate the image colors. 110

5-4 ToolScape augments a web-based video player with an interactive timeline. Annotations are shown above the timeline (a), screenshots of intermediate states are shown below the timeline (c), and the gray regions at both ends (b) show “dead times” with no meaningful progress (e.g., waiting for Photoshop to launch). 111

5-5 These task instructions are shown before the participant starts working on their image manipulation task. It includes a description of the effect to be implemented and a before-and-after example image pair. 113

5-6 Rank-ordered designs created by the participants in the formative study. For each image, the top label displays the rank of the image and the condition it was created in (either ToolScape or Baseline). The mean rankings were lower in ToolScape (5.7) than in Baseline (7.3), which maps to higher quality designs. 114

5-7 In the Find stage, the crowd worker adds new steps to the timeline by clicking on the "New Instruction" button. 117

5-8 Upon clicking on the "New Instruction" button, a popup window asks the worker to describe what the step is about in free-form text. 118

5-9 Our clustering algorithm groups adjacent time points into a candidate step. It further adds a potential cluster as a candidate, which might turn out to be a proper step once checked in the Verify stage. This inclusive strategy mitigates the effect of clustering errors. 119

5-10 The Verify stage asks the worker to choose the best description of a candidate step. The options come from workers in the Find stage. Additional default options allow the worker to either suggest a better description or mark the step as invalid. 121

5-11 The Expand stage asks the worker to choose the best before and after images for a step. The worker visually reviews the thumbnail options and clicks on images to decide. 122

5-12 Our evaluation uses the Hungarian method to match extracted and ground truth steps with closest possible timestamp within a 10-second window size. Then we compute precision and recall, which indicate if our workflow over- or under-extracted steps from ground truth. 124

6-1 We present a three-stage workflow to generate labels for groups of steps (subgoal labels) for how-to videos. This workflow is designed to engage people actively trying to learn from the video to contribute the information. 132

6-2 For each video on Crowdy, we show the video player as well as a side panel with the individual steps. After learners answer a question, the subgoal they added or picked appears in the outline. Learners can click on steps or subgoals to jump to various parts in the video. The question block (Figure 6-3, 6-4, 6-5) temporarily replaces the video player when it appears. The "Print" button enables exporting the outline in a text format for learners to review later even without the video. 135

6-3 In Stage 1, learners are asked to generate a new subgoal label after watching a video segment. 136

6-4 In Stage 2, learners vote on a subgoal they find most accurate. 137

6-5 In Stage 3, learners proofread the most popular subgoal for language and scope. 138

6-6 The baseline condition without displaying subgoals or prompting learners. 144

6-7 The expert condition with expert-generated subgoals displayed but without prompting. 144

6-8 The Crowdy condition with learner prompts and display of learners' own responses. 144

6-9 The number of attempted subgoals in the retention test. Crowdy = Expert > Baseline. Error bar: standard error. 148

6-10 The number of correct subgoals in the retention test. Crowdy = Expert > Baseline. Error bar: standard error. 149

6-11 The pretest and posttest scores. No significant difference between the interfaces. Error bar: standard error. 150

6-12 The Crowdy website features how-to videos with the learnersourcing workflow embedded. 152

6-13 An excerpt from the worksheet given to domain experts to evaluate learner-generated subgoals. The subgoals were randomly placed on the left and right so experts had no notion of which was which. Evaluators were asked to choose whether the subgoals matched, or which was better, and provide an explanation. 155

6-14 The distribution of subgoal label scores. The scoring scheme assigned -1 if an expert label is better, 1 if a learner label is better, and 0 if they match. While a majority of the subgoals were within the -2 and 2 range, three subgoals had unanimous votes for the expert label, and one subgoal had unanimous votes for the learner label. 156

6-15 The subgoals that were added in each of the stages for the introduction to HTML video. The green numbers refer to upvotes, and the red numbers refer to downvotes. In stage 3, we expected small iterations, but found that one learner drastically changed the tone of the original subgoal. 163

6-16 The subgoals that were added in each of the stages for one portion of the introduction to jQuery selectors video. The green numbers refer to upvotes, and the red numbers refer to downvotes. The subgoal added in stage 2 is a combination of the subgoals added in stage 1, which suggests that the learner who created this subgoal may have used the stage 1 subgoals as a baseline. 164

List of Tables

3.1 Overview of the four edX courses in our dataset offered in Fall 2012. *Students* refers to the number of students who watched at least one video, *Videos* is the number of all video clips posted, *Video Length* is the mean duration, and *Processed Events* is the number of total play and pause events captured by the video player. 56

3.2 Peak profile comparison reporting average values across all peaks detected for each video group. Tutorial videos resulted in more peaks than lecture videos. Likewise, re-watching sessions resulted in more peaks than first-time sessions. All differences between lecture and tutorial, and first time and re-watcing were statistically significant. 63

3.3 Five student activity types that lead to a peak are shown, along with their frequency distribution as manually labeled by the authors. We sampled 80 videos and labeled each peak to one of the activity types. Only Type 5 does not involve a visual transition. 66

5.1 The mean number of steps generated by the workflow in each stage. At the end the workflow extracted 15.7 steps per video, which is roughly equivalent to 15.0 from ground truth. Stage 1 clusters the original Turker time points, and Stage 2 merges or removes some. 125

5.2 When considering time information only, recall tends to be higher. When considering both time and text descriptions, incorrect text labels lower both precision and recall, but removing unnecessary steps in Stage 2 recovers precision. 125

5.3 The mean time distance between extracted steps and ground truth steps was only 2.7 seconds on average, for all the steps that had a matching ground truth step. This means that the time-based clustering (Figure 5-9) accurately detected step timings. 125

6.1 How-to videos used in our live deployment. We evaluated the quality of learnersourced subgoal labels for the four bolded videos because they were the most popular videos on Crowdy. 153

6.2 Expert labels compared to Learner subgoals for the four videos selected for evaluation. For each pair, we identify whether a majority of evaluators felt the subgoals matched or that one was better than the other. In some cases there was no clear majority between evaluators, which we have also noted. 154

7.1 Comparison between learnersourcing and paid crowdsourcing, along major design dimensions. 168

Chapter 1

Introduction to Learnersourcing

With an unprecedented scale of users interacting with online platforms, data generated from their interactions can be used to understand collective preferences and opinions, create new content, and improve future users' experience. In several domains, this thesis introduces methods for 1) extracting meaningful patterns from natural interaction traces, 2) eliciting specific information from users by designing microtasks that are inherently meaningful to them, and 3) changing the interface behavior immediately as more data becomes available. These data-driven methods demonstrate how interaction data can be powerful building blocks for enhancing massive-scale learning, planning, discussion, collaboration, and sensemaking online.

Specifically, this thesis focuses on video learning in online education settings. Millions of learners today are using educational videos to master skills and take courses from online platforms such as YouTube, Khan Academy, Coursera, or edX. Video is an efficient and scalable medium for delivering educational content. It can be accessed any time from anywhere, and learners can watch at their own pace. Also, once a video is published, millions of learners can learn from the same material. But learners and instructors encounter many problems in using video to learn and teach online.

For video learners, their diverse goals in navigating through a video are often not met by existing video players. For example, finding information, skimming through the content quickly, skipping familiar sections, and navigating to specific points inside a video are difficult. Because many instructional videos do not have an accompanying outline, learners often have to rely on imprecise estimates to locate their points of interest within a video. Such limited content navigation can be detrimental to learning, because learners cannot easily review or reflect on the content.

Also, many video platforms do not support exercises for active learning. Education literature has long advocated for interactivity while watching a video [46, 61, 156], and a recent study has found that online learners who engage in activities learn more than those who primarily watch videos [94]. More video platforms have started to support in-video quizzes while learners watch a video, but for a long-tail of videos produced by amateurs with limited budget and editing expertise, embedding exercises may be difficult.

For video instructors, they cannot easily understand and measure how learners are using their material. Even if there are editing glitches in the video or most learners are confused at a certain point, instructors often lack useful tools to detect such issues and act on them.

Also, updating content, creating a video outline, or adding exercises requires a significant amount of time and effort. Furthermore, if the platform they are using does not provide built-in support for these actions, many instructors do not realize making such improvements is possible and helpful to learners.

The problems that learners and instructors face with video cannot be easily solved with existing video interfaces and tools. Generating high-level content summaries or in-video quizzes requires deep semantic information about videos. Solutions involving humans require an extensive amount of customization, expertise, or manual effort for each video, but this cannot scale to millions of educational videos found online. Solutions involving automated methods such as computer vision or natural language processing work well for certain constrained environments, but even state-of-the-art AI techniques cannot yet reach the level of video understanding required in this problem. To better understand the problem space, it is important to understand what instructional videos are like, how learners tend to use them, and how they currently support learning.

1.1 Background

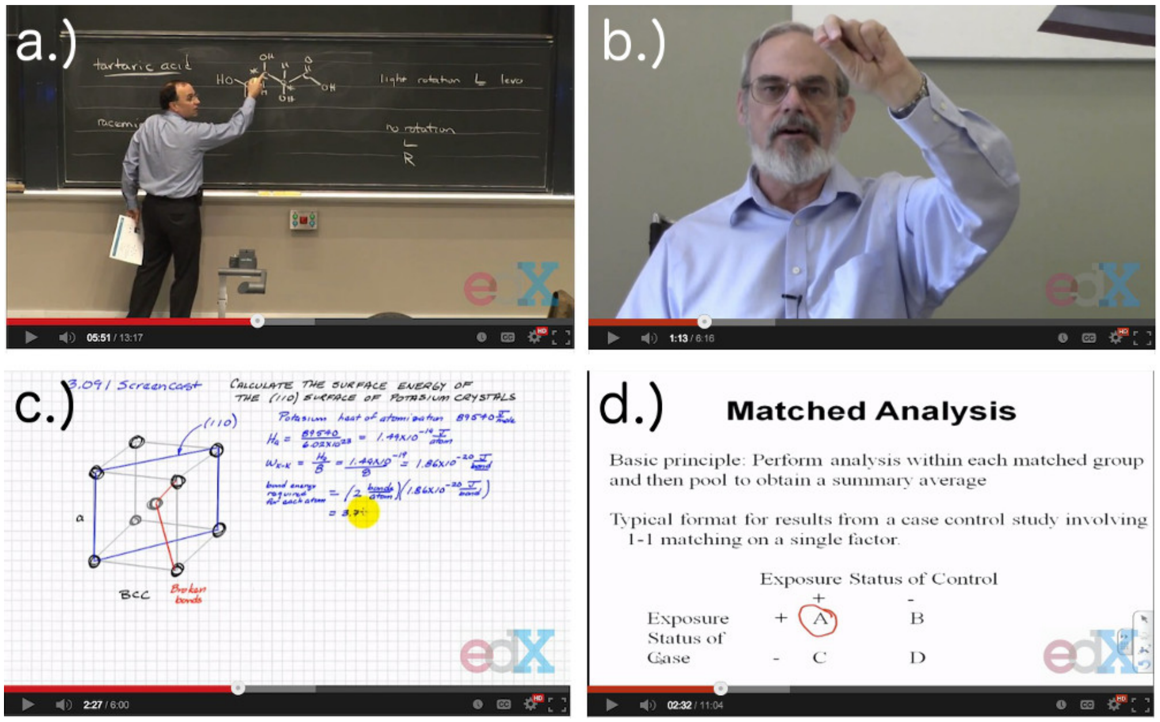
Educational videos contain instructional content for viewers who aim to learn knowledge about a concept or skills about a task. As video has become a popular medium on the Internet, its educational use has also rapidly grown. The YouTube EDU channel has over 10 million subscribers [1], and Khan Academy hosts over 5,000 videos on math and science, with more than 15 million registered students [2]. Massive open online courses (MOOCs) have opened access to quality instructional materials at a global scale, with popular platforms such as Coursera, edX, and Udacity attracting millions of learners, partnering with hundreds of schools, and hosting thousands of courses. Research shows that learners on MOOC platforms spend a majority of their time on the platforms watching videos [18, 149]. For professional skill learning, Lynda.com hosts over 3,500 courses and 140,000 videos [3].

While most web-based video learning platforms primarily target self-directed learners, in-classroom usage of video has increasingly gained popularity. In a recently published report on the use of video in education [73], 66% of survey respondents who are educators answered that their institutions use video for remote teaching in higher education, and 95% mentioned that students create or use video in their school work. Another noteworthy trend is flipped classrooms, an instructional model in which instructional content delivery takes place outside of the classroom and class time is dedicated to deeper exploration and collaborative activities. Often, video lectures are assigned to students to watch at home, as indicated by 46% of the respondents who said they use video for this purpose.

There are different types of educational videos that span a variety of learning objectives, instructional approaches, presentation methods, and editing styles. This thesis mainly covers two types of educational videos, namely lecture videos and how-to videos.

Lecture videos teach conceptual knowledge, which involves understanding principles that govern a domain and relationships between knowledge in a domain [145], and answers *why* things happen in a particular way [67]. These videos can be commonly found in mas-

[1]https://www.youtube.com/education, accessed on August 4, 2015
[2]http://www.telegraph.co.uk/education/educationnews/11379332/A-day-in-the-life-of-Khan-Academy-the-school-with-15-million-students.html
[3]http://www.lynda.com/press, accessed on August 4, 2015



Matched Analysis

Basic principle: Perform analysis within each matched group and then pool to obtain a summary average

Typical format for results from a case control study involving 1-1 matching on a single factor

		Exposure Status of Control +	Exposure Status of Control -
Exposure Status of Case	+	A	B
	-	C	D

Figure 1-1: Lecture videos on the web show different styles, including a.) classroom lecture, b.) ‘talking head’ shot of an instructor at a desk, c.) digital tablet drawing format popularized by Khan Academy, and d.) PowerPoint slide presentations.

sive open online courses (MOOCs) via platforms such as Coursera, edX, or Udacity. While the standard mode of video learning is to watch a clip, instructors and platforms strive to improve the learning experience by curating a series of lecture videos into a course, embedding in-video quizzes, or adding discussion forums. Research shows that video production factors affect learners’ engagement with MOOC videos [59]. For example, with MOOC videos longer than six minutes, learners’ engagement significantly drops. Challenges in supporting learning from lecture videos lie in monitoring learners’ engagement continuously, identifying where and why learners are disengaged during their video watching session, and improving the content or the video interface.

How-to videos teach procedural knowledge, which includes a sequence of actions for solving problems [145], and answers *how* things happen in a particular way [67]. These videos include instructions for how to complete a task in a step-by-step manner, and can be commonly found on tutorial websites and social video platforms. Because many of

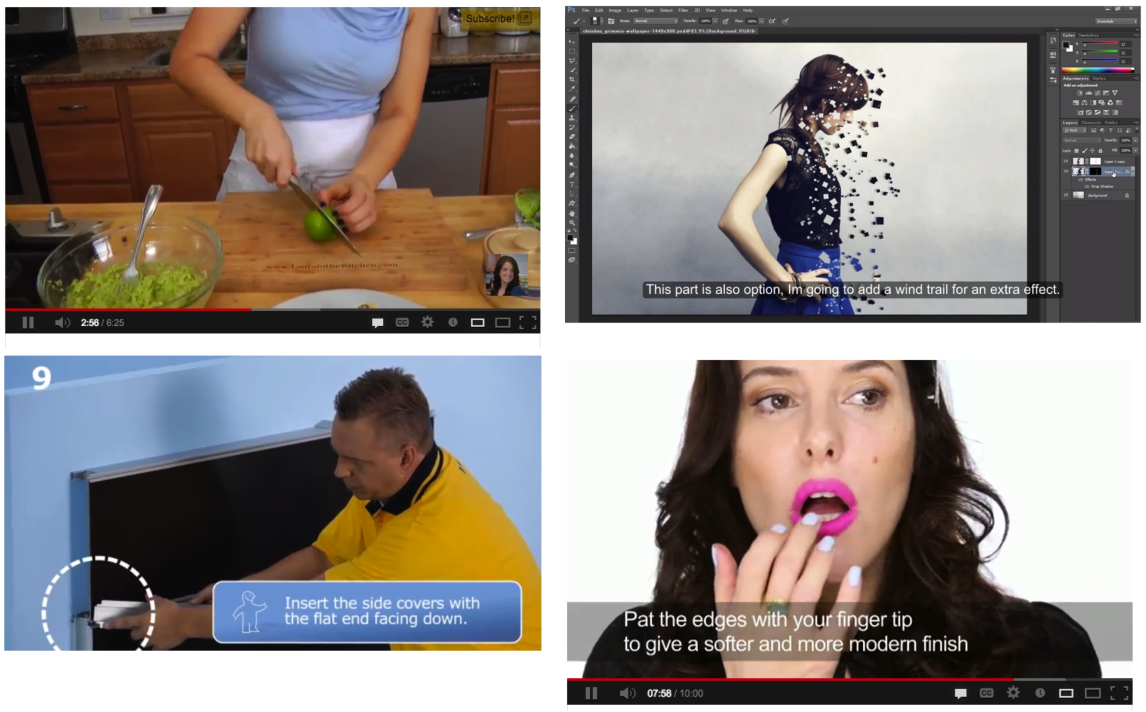

Subscribe!
2:56 / 6:25
This part is also option, Im going to add a wind trail for an extra effect.
9
Insert the side covers with the flat end facing down.
Pat the edges with your finger tip
to give a softer and more modern finish
07:58 / 10:00

Figure 1-2: How-to videos include step-by-step instructions, spanning various domains including cooking, graphical design, home improvement, and applying makeup.

these videos are not labeled with step information, learners often face difficulty navigating the video with more control. Challenges in supporting learning from how-to videos lie in labeling videos with actions or steps, assisting learners with following along the steps in the video, and supporting step-by-step navigation of the video.

1.2 Learnersourcing: Crowdsourcing with Learners as a Crowd

To address the presented challenges in video learning, this thesis takes a data-driven approach. What if we use data generated from learners' interaction with video content to understand and improve learning? Unique design opportunities arise from the fact that a large group of learners watch the same video, and that video platforms can track the learning process at a fine-grained level: second-by-second and click-by-click. Further, data-driven understanding can eventually be used to improve the video content and interfaces.

In designing better video learning environments, this thesis introduces **learnersourcing**, which uses a large group of learners' input to improve video contents and interfaces. As the name implies, learnersourcing is a form of crowdsourcing with learners as a crowd.

Crowdsourcing has offered a new solution to many computationally difficult problems by introducing human intelligence as a building block. Inspired by the concept of crowdsourcing, learnersourcing also attempts to collect small contributions from a large group of users. The fundamental difference is that while crowdsourcing often issues an open call to an undefined crowd, learnersourcing uses a specialized crowd: learners who are inherently motivated and naturally engaged in their learning. This difference leads to a unique set of design considerations, in terms of incentive design, quality control, task scope, and task difficulty. As a result, learnersourcing can tackle problems that computers or general crowdsourcing cannot easily solve.

A crucial element in learnersourcing is to design learner's activity that is pedagogically beneficial, while the system collects useful information as a byproduct of such activity. Learnersourcing attempts to establish a feedback loop between the learner and the system. As the learner watches the video naturally or answers some prompt that helps their learning, the system processes the byproduct of these activities to dynamically improve content and interfaces for future learners.

1.3 Passive and Active Learnersourcing

Depending on how learners' input is collected, this thesis defines two types of learnersourcing: passive learnersourcing and active learnersourcing. In passive learnersourcing, the system collects information passively, just by watching the learner. In active learnersourcing, the system actively interrupts the learner to request information.

Passive learnersourcing uses data generated by learners' natural interaction with the learning platform. Examples includes playing and pausing events from the video player, and browsing patterns between learning modules. On the other hand, **active learnersourcing** prompts learners to engage in specific activities, with the purpose of providing pedagogical benefits (for the current learner) and collecting useful information (for future learners) at the same time. Examples include asking learners to summarize video segments they watched, to answer embedded questions, and to explain concepts discussed in the instructional material in their own words. In both passive and active learnersourcing, the system processes the collected data to further analyze how learners are using the instructional materials and improve future learners' learning experience. This thesis introduces a set of active and passive learnersourcing applications to support video learning at scale.

1.3.1 Passive Learnersourcing: Natural learner interactions improve video learning

In traditional classrooms, teachers adapt their instruction to students based on their level of engagement and confusion. While online videos enable access for a wide audience, instructors and learners are disconnected; it is as if instructors are talking to a wall without feedback from learners watching the video. This thesis introduces a technique that leverages natural learning interaction data to better understand and improve video learning, specifically using thousands of learners' second-by-second video player interaction traces (e.g., clicking the play button in the video player). The thesis will report a post-hoc analysis of this data to guide the improvement of the material, and further introduce an improved video interface in which video presentation dynamically adapts to collective video learning patterns.

Data analysis of 39 million MOOC video clicks

Exploratory data analyses of four massive open online courses (MOOCs) on the edX platform investigated 39 million video events and 6.9 million watching sessions from over 120,000 learners. Analyzing collective in-video interaction traces revealed video interaction patterns, one of which is interaction peaks, a burst of play button clicks around a point in a video indicating points of interest and confusion for many learners. A key observation was that 61% of the peaks accompany visual transitions in the video, e.g., a slide view to an instructor view (Figure 1-3). Extending this observation, it is possible to identify student activity patterns that can explain peaks, including playing from the beginning of new

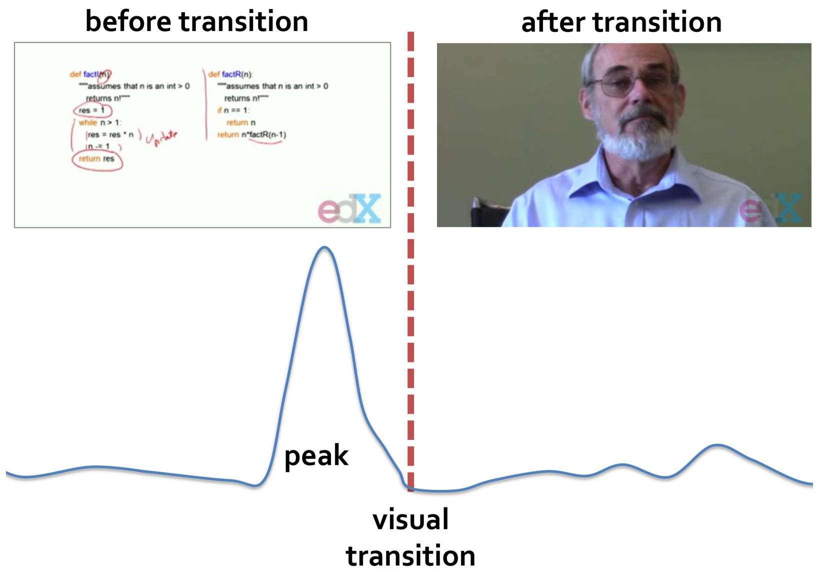


Figure 1-3: An example interaction peak near a scene transition in a lecture video.

material, returning to missed content, and replaying a brief segment [86]. This analysis has implications for video authoring, editing, and interface design, and provide a richer understanding of video learning on MOOCs.

LectureScape: Video interface that evolves with data

Adapting the video presentation to the online classroom can directly improve the video learning experience. LectureScape (Figure 1-5) [85] is an enhanced video player for educational content online, powered by data on learners' collective video watching behavior. LectureScape dynamically adapts to thousands of learners' interaction patterns to make it easier to rewatch, skim, search, and review. By analyzing the viewing data as well as the content itself, LectureScape introduces a set of data-driven interaction techniques that augment existing video interface widgets: a 2D video timeline with an embedded visualization of collective navigation traces; dynamic and non-linear timeline dragging; data-enhanced transcript search and keyword summary; automatic display of relevant still frames next to the video; and a visual summary representing points with high learner activity. Participants in a user study commented that "it feels like watching with other students" and it was "more classroom-y" to watch videos with LectureScape, which shows how large-scale interaction data can support social and interactive video learning.

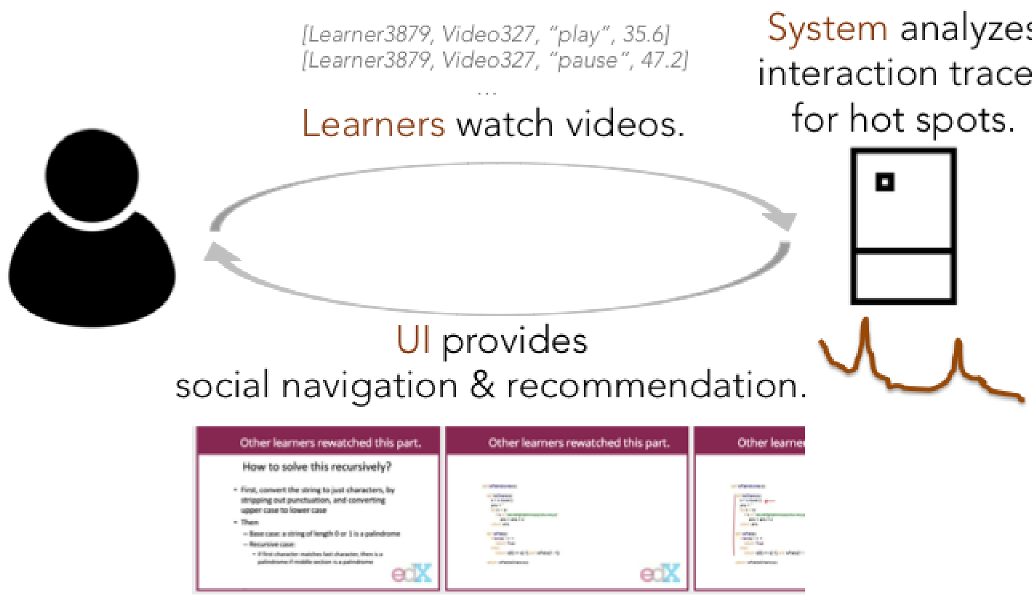
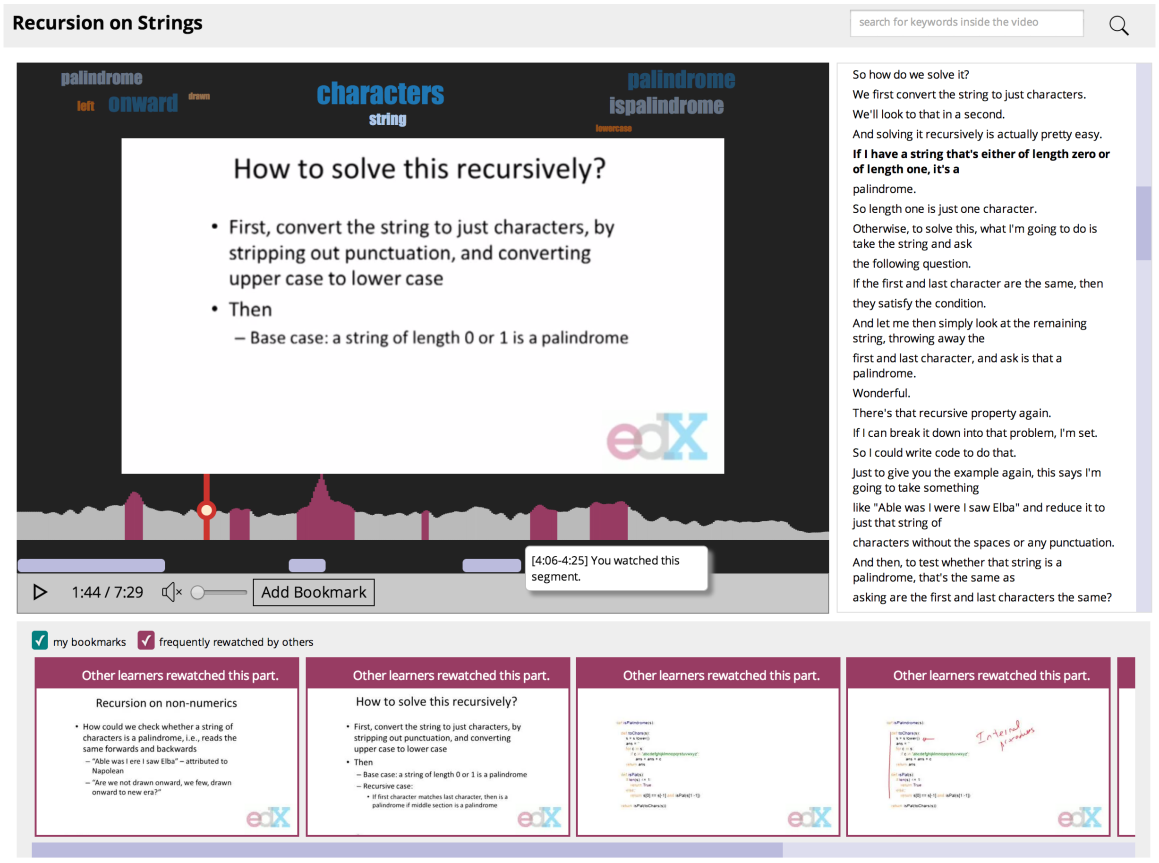

[Learner3879, Video327, "play", 35.6]
[Learner3879, Video327, "pause", 47.2]
...
Learners watch videos.
System analyzes interaction traces for hot spots.
UI provides social navigation & recommendation.
Other learners rewatched this part.
How to solve this recursively?
Other learners rewatched this part.
edX

Figure 1-4: This diagram describes how LectureScape implements passive learnersourcing. LectureScape is a video player that adapts to collective learner engagement. As learners watch a video, the system analyzes collective interaction traces to mine for specific positions that may indicate points of confusion or importance. Based on this information, the video player dynamically changes its behavior to recommend popular frames visited frequently by other learners.

1.3.2 Active Learnersourcing: Learner prompts contribute to new learning materials

How-to videos contain worked examples and step-by-step instructions for how to complete a task (e.g., math, cooking, programming, graphic design). My formative study demonstrated the navigational, self-efficacy, and performance benefits of having step-by-step information about the solution [89]. Education research has shown powerful learning gains in presenting the solution structure to learners. Solution structure refers to an organized representation of how to complete a procedural task, in terms of steps and groups of steps (subgoals). Steps and subgoals can be labeled and presented as an outline to provide learners with an overview of a solution. However, such information is not available for most existing how-to videos online, and requires substantial expert efforts to collect. This thesis introduces scalable methods for extracting steps and subgoals from existing videos that


Recursion on Strings
search for keywords inside the video
palindrome
left
onward
drawn
characters
string
palindrome
ispalindrome
lowercase
1:44 / 7:29
Add Bookmark
my bookmarks
frequently rewatched by others
Other learners rewatched this part.
Other learners rewatched this part.
Other learners rewatched this part.
Other learners rewatched this part.

How to solve this recursively?

- First, convert the string to just characters, by stripping out punctuation, and converting upper case to lower case
- Then
  - Base case: a string of length 0 or 1 is a palindrome

[4:06-4:25] You watched this segment.

So how do we solve it?
We first convert the string to just characters.
We'll look to that in a second.
And solving it recursively is actually pretty easy.
If I have a string that's either of length zero or of length one, it's a
palindrome.
So length one is just one character.
Otherwise, to solve this, what I'm going to do is take the string and ask
the following question.
If the first and last character are the same, then
they satisfy the condition.
And let me then simply look at the remaining string, throwing away the
first and last character, and ask is that a palindrome.
Wonderful.
There's that recursive property again.
If I can break it down into that problem, I'm set.
So I could write code to do that.
Just to give you the example again, this says I'm going to take something
like "Able was I were I saw Elba" and reduce it to just that string of
characters without the spaces or any punctuation.
And then, to test whether that string is a palindrome, that's the same as
asking are the first and last characters the same?

Recursion on non-numerics

- How could we check whether a string of characters is a palindrome, i.e., reads the same forwards and backwards
  - "Able was I ere I saw Elba" – attributed to Napolean
  - "Are we not drawn onward, we few, drawn onward to new era?"

How to solve this recursively?

- First, convert the string to just characters, by stripping out punctuation, and converting upper case to lower case
- Then
  - Base case: a string of length 0 or 1 is a palindrome
  - Recursive case:
    - If first character matches last character, then is a palindrome if middle section is a palindrome

Figure 1-5: LectureScape: a lecture video player powered by collective interaction data.

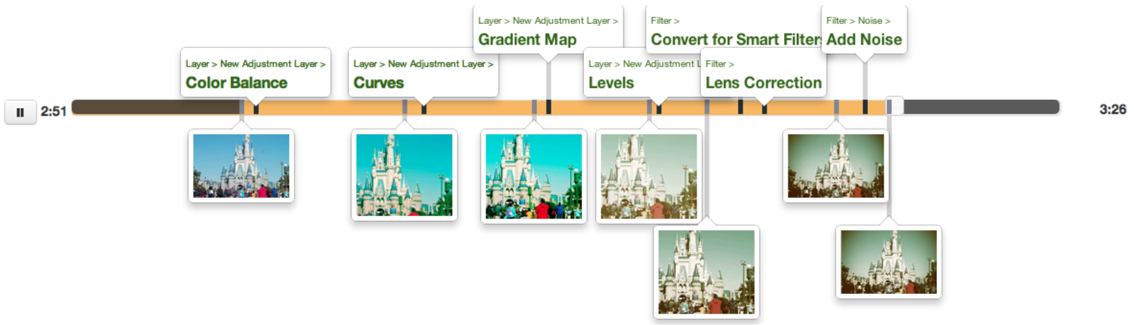


Figure 1-6: ToolScape: a step-aware how-to video player.

do not require experts, as well as an alternative video player where the solution structure is displayed alongside the video. These techniques actively prompt learners to contribute structured information in an in-video quiz format.

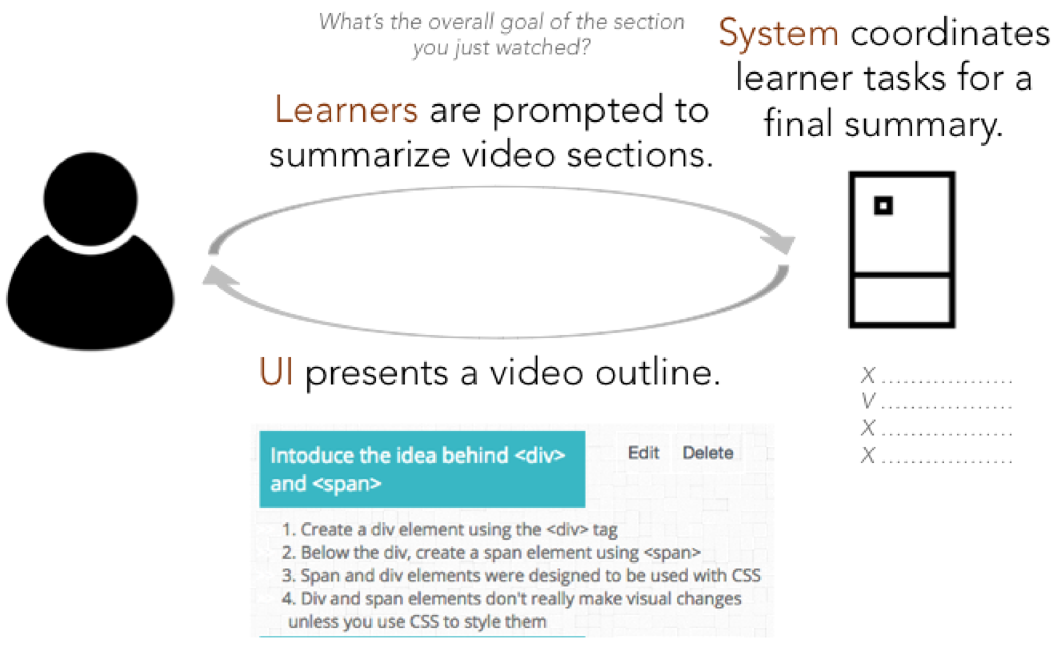

What's the overall goal of the section you just watched?
Learners are prompted to summarize video sections.
System coordinates learner tasks for a final summary.
UI presents a video outline.
Intoduce the idea behind <div> and <span>
Edit
Delete
1. Create a div element using the <div> tag
2. Below the div, create a span element using <span>
3. Span and div elements were designed to be used with CSS
4. Div and span elements don't really make visual changes unless you use CSS to style them
X
V
X
X

Figure 1-7: This diagram describes how Crowdy implements active learnersourcing. Crowdy is a video player that coordinates learners' collective efforts to generate subgoal labels, which are summarized, abstract descriptions for a group of low-level steps in a procedural task. As learners watch a video, the video player pauses and prompts learners to summarize video sections. The system coordinates learner tasks in multiple stages to reach a final set of summary labels for the clip. The video player dynamically displays the collectively generated video outline as future learners visit and watch the video.

ToolScape: Extracting step-by-step information from how-to videos

The findings from the formative study guided the design of ToolScape, a video player that displays step descriptions and intermediate result thumbnails in the video timeline (Figure 1-6) [89]. To enable non-experts to successfully extract step-by-step structure from existing how-to videos at scale, ToolScape includes a three-stage crowdsourcing workflow. It applies temporal clustering, text processing, and visual analysis algorithms to merge crowd output. The workflow successfully annotated 75 cooking, makeup, and Photoshop videos on YouTube of varying styles, with a quality comparable to trained annotators across all domains, with the average cost of about $1 for a minute of video.

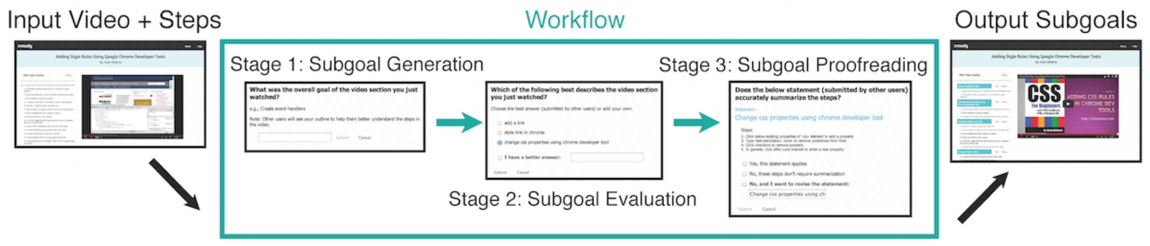

Input Video + Steps
Workflow
Output Subgoals
Stage 1: Subgoal Generation
Stage 2: Subgoal Evaluation
Stage 3: Subgoal Proofreading

Figure 1-8: Crowdy: a learnersourcing workflow for summarizing steps in a how-to video.

Crowdy: Learnersourcing section summaries from how-to videos

Taking a step further, what if learners, both an intrinsically motivated and uncompensated crowd, can generate summaries of individual steps at scale? This research question resulted in a learnersourcing workflow that periodically prompts learners who are watching the video to answer one of the pre-populated questions, such as "what was the overall goal of the video section you just watched" (Figure 1-8) [163]. The system dynamically determines which question to display depending on how much information has already been gathered for that section in the video, and the questions are designed to engage learners to reflect on the content. Learners' answers help generate, evaluate, and proofread subgoal labels, so that future learners can navigate the video with the solution summary. This led to a live deployment of Crowdy, a website with the learnersourcing workflow implemented on a set of introductory web programming videos. A study with crowd workers on Mechanical Turk showed that participants with Crowdy retained information about the solution better than those with a standard video player, and at the same level as those who saw expert-generated labels. The 25-day deployment attracted more than 1,200 learners who contributed hundreds of subgoal labels and votes. A majority of learner-generated subgoals were comparable in quality to expert-generated ones, and learners commented that the system helped them grasp the material.

When combined, the two methods (ToolScape for individual steps and Crowdy for subgoal labels) can fully extract a hierarchical solution structure from existing how-to videos.

1.4 Contributions

Two primary technical contributions of learnersourcing are 1) methods for collecting and processing large-scale data from learners, and 2) data-driven interaction techniques to augment existing video players. These techniques are then validated by a series of evaluation studies that measure pedagogical benefits, resulting data quality, and learners' qualitative experiences. This thesis combines these components to demonstrate that learnersourcing can improve the content and interfaces in a way neither experts, nor computers, nor existing crowdsourcing methods can achieve at scale.

The contributions are made possible by uniquely extending and combining the following families of techniques: **crowdsourcing** to aggregate small contributions into meaningful artifacts, **social computing** to motivate participation and build a sense of community among learners, **content-based video analysis techniques** such as computer vision and natural language processing to complement learner input, and **learning science** to inform the design of learnersourcing tasks that are pedagogically meaningful.

Thesis statement: In large-scale video learning environments, interfaces powered by learnersourcing can enhance content navigation, create a sense of learning with others, and ultimately improve learning.

1.5 Thesis Overview

Chapter 2 lays the groundwork for learnersourcing by covering major research challenges in the context of related work.

The main part of the thesis introduces four learnersourcing applications. These chapters span data analysis, interface design, technical solutions, usability and learning evaluation, and live deployment.

- Chapter 3 presents the first application of passive learnersourcing, which is a large-scale analysis of learners' in-video interaction in MOOCs. Specifically, it focuses on in-video dropout and interaction peak patterns.

- Chapter 4 presents the second application of passive learnersourcing, LectureScape, which is a lecture video player powered by learners' collective video interaction data.
- Chapter 5 presents the first application of active learnersourcing, ToolScape, which extracts step-by-step instructions from an existing how-to video. It features a three-stage workflow and a step-aware video player.
- Chapter 6 presents the second application of active learnersourcing, Crowdy, which learnersources section summaries of a how-to video. This chapter introduces a three-stage workflow and a learnersourcing-enhanced website for watching how-to videos. Combined with ToolScape, Crowdy can extract a hierarchical solution structure from existing how-to videos.

Chapter 7 summarizes design lessons from the four learnersourcing applications, and discusses major design dimensions and limitations to assist future researchers and practitioners in their learnersourcing design.

Finally, Chapter 8 reviews the contributions of the thesis proposes future research directions for learnersourcing.

Chapter 2

Related Work

There are four main areas of research that this thesis builds upon: **crowdsourcing** to solicit learner participation and coordinate learner tasks at scale; **interaction data processing** to handle collected interaction data from learners; **learning science** to make pedagogically meaningful enhancements to video learning; and **video applications** to better extract useful information from video content and design more interactive video interfaces. This chapter reviews literature in these four domains.

2.1 Crowdsourcing and Human Computation

A majority of existing crowdsourcing systems rely on paid crowd workers (e.g., workers on Mechanical Turk or oDesk) who are offered monetary reward upon completion of the task. When designing crowd-powered application, paid crowdsourcing makes it easy to use the power of the crowd on demand. When considering using learners as a crowd, however, system designers should consider a unique set of challenges, one of which is incentive design.

Learnersourcing is inspired by prior work on human computation [160], in which the system solicits human input for certain parts of the computation. Games with a Purpose [159] present computationally difficult problems (e.g., tagging objects from an image [160] or protein folding [`fold.it`]) as a game. As users play the game, the byproduct of the gameplay is collected and processed by the system. What makes this approach scalable and motivational is that users do not have to know how their input is used by the system, because they have the strong motivation to play the game anyway. In learnersourcing, learners' tasks need to be designed so that they are intrinsically useful and enjoyable.

Another way to motivate the crowd is to expose the underlying goal of the human computation system, especially if the goal is serving the public good. Examples include helping scientific discoveries (e.g., Zooniverse [www.zooniverse.org], FoldIt [fold.it], Lab in the Wild [www.labinthewild.org/]) or improving civic engagement (e.g., Ushahidi [www.ushahidi.com], Factful [87], and BudgetMap [91]). Zooniverse is a research platform where volunteers can assist professional researchers with various tasks that require human judgment or intelligence at scale. Tasks include classifying galaxies by analyzing photos, or identifying directions the wildebeest to track where they are going. Ushahidi allows the crowd to report incidents of violence and peace efforts, especially during a crisis. The collected reports collectively serve as an up-to-date dashboard that journalists and citizens alike can benefit from. In learnersourcing, the bigger cause presented to learners is that their contributions can not only help them but future learners learn better.

There also exist crowdsourcing applications that require domain knowledge and expertise beyond what the random crowd can handle. Recent research has begun using crowds within a community to tackle more domain-specific and complex problems. Communitysourcing [66] outsources tasks that community members are qualified and motivated to perform, such as computer science students at a university grading homework problems for free snacks offered from a vending machine. Cobi [90] asks authors of accepted papers at an academic conference to specify papers that are relevant to theirs. The authors' input is used by the system to detect preferences and constraints, which conference organizers consider when scheduling the conference.

Some existing applications can be categorized as learnersourcing, in which the crowd's motivation for participation is learning. Duolingo [www.duolingo.com/] asks language learners to translate text and uses the translated text to provide a commercial text translation service. Dontcheva et al. [40] augmented Photoshop to embed real-world image manipulation tasks for learners, thus satisfying the requesters' and the learners' needs at the same time. Learnersourcing applications introduced in this thesis differ from the two examples above in that the artifact the crowd creates directly benefits the crowd themselves, not external requesters. Mechanical MOOC [mechanicalmooc.org/] aims to create a MOOC experience without any instructor, by coordinating learner activities with a set of online communication tools. This thesis defines learnersourcing and discusses major design dimensions any learnersourcing application should consider, along with examples of learnersourcing in the context of video learning.

2.1.1 Crowdsourcing Workflows

Research on multi-stage crowd workflows inspired the design of our crowdsourcing and learnersourcing methods. Soylent [11] has shown that splitting tasks into the Find-Fix-Verify stages improves the quality and accuracy of crowd workers' results. Other multi-stage crowdsourcing workflows were designed for complex tasks, including nutrition information retrieval from food photos [129], real-time captioning of a speech [104], activity recognition from streaming videos [106], taxonomy generation [31], and search engine answer generation [12]. Systems have also been built to leverage the wisdom of the crowds to generate information on educational videos. For example, VidWiki [36] crowdsources incremental improvements to educational videos with annotations.

These applications demonstrated that crowdsourcing can yield results comparable to those of experts at lower cost. This thesis contributes to this line of work two crowd-powered workflows (Chapters 5 and 6) in the domain of video annotation and online education. These workflows are designed explicitly for a crowd of learners who are naturally watching the video with their own desire to learn. Designing human computation tasks for voluntary learners presents a unique set of challenges, which we address with our workflow.

2.1.2 Crowdsourced Data Annotation

Studies have shown that gathering data using crowdsourcing can be accurate and highly successful. Sorokin and Forsyth [154] used Amazon Mechanical Turk to generate quality image annotations at a cheap rate. Snow et al. [153] showed that aggregating individual annotators' input over a small number quickly can yield results comparable in quality to those produced by experts, even though annotations from individuals may produce noisy results. MacLean and Heer [109] demonstrated that in identifying medical terms in patient-authored text, crowdsourced annotations can train medical term classifiers that outperform systems by experts. This thesis introduces crowdsourced data annotation methods for extracting information from educational videos.

2.2 Interaction Data Processing

Interaction data refers to any kind of data the system can collect as users interact with the system. In video learning, examples include clickstream data generated from learners' interaction with the video player, such as clicking on the play button or scrubbing on the video timeline. Existing research has explored ways to use video interaction data to infer learners' engagement, either by using implicit user data (interaction log) or explicit user data (clicking the "important" button or voting).

Implicit user data has the benefit of requiring no additional action on user's part, because this data is automatically captured by the system while users naturally interact with videos. Shaw and Davis advocate using actual viewership data in modeling user interest [150]. Existing systems leverage scrubbing [169], zooming and panning [22], and playing and pausing [32] activities. SocialSkip [32] demonstrates that modeling users' video interactions can accurately capture user interest in information retrieval tasks. While this thesis adopts the idea of using video clickstream data from the literature, our analysis differs in that it uses large-scale interaction data from MOOCs, and that it focuses on in-video dropout and interaction peaks in the educational context. Passive learnersourcing primarily relies on implicit user data.

Explicit user data can be collected by asking users to make a specific action around their points of interest. Previous research used user rating data [130] or annotations [150]. CLAS [144] is a lecture video annotation tool where students click a button when they find a part of the video important. The system aggregates responses from all students in a class to visualize important points. Also, we extend prior work on providing social navigation for lecture videos [121] to support diverse learning tasks. Active learnersourcing primarily relies on explicit user data.

2.2.1 Modeling Collective User Data

There is a rich thread of research in using collective user interaction history data to analyze usage patterns and improve users' task performance. Interaction history data is automatically collected by applications during normal usage. Examples include Web browsers logging Web page visit history, search engines capturing query history, and video players storing video interaction clickstreams such as play and pause events. Read Wear [68] presented a visionary idea in this space to visualize users' read and edit history data in the scrollbar. Chronicle [58] captured and provided playback for rich, contextual user interaction history inside a graphical application. Dirty Desktops [71] applied magnetic forces to each interaction trace, which improved target selection for commonly used widgets. Patina [117] separated individual and collective history and added overlays on top of the GUI, to help people find commonly used menu items and discover new ways of completing desktop-related tasks. Causality [124] introduced an application-independent conceptual model for working with interaction history. This thesis introduces a technique for using video interaction history to support common navigation tasks in video-based learning.

2.2.2 Tools for temporal pattern analysis

Understanding temporal patterns in large-scale video interaction data requires powerful computational and visual tools.

Temporal pattern analysis of time-series data inspired the analytical methods in this thesis. Kleinberg [93] introduced a burst model for detecting meaningful structure in documents, and Jones and Diaz [78] applied this model among other temporal features to identify temporal patterns in search queries. Using search query and social media streams, researchers categorized search query patterns and trending events based on the shape of spikes [79, 98]. This thesis applies similar techniques to analyze video interaction patterns, which is enabled by large-scale data collected from educational videos.

Video analytics platforms can enable the visual sensemaking of large-scale data. General purpose video platforms such as Youtube provide advanced analytics [55, 166] for content authors. These services include dashboards showing viewership graphs over time for a video, and suggest focusing on rises and dips. Our approach considers more in-depth activity data such as play, pause, and skip events on the player, and content specific to educational videos, such as video kinds (lecture or tutorial), presentation styles (slide, head shot, etc.), and visual transitions between the presentation styles.

2.3 Learning Science

Constructivism and active learning theories emphasize the importance of the learner's active role in engaging with the learning material and the learning process. In active learning, students engage in reading, writing, discussion, and problem solving, rather than just listening to delivered lectures [16]. More specific activity examples include group discussions, short written exercises, learning by teaching, and reacting to a video. Previous research has shown the pedagogical benefits of active learning over traditional lectures [50, 142]. A meta-analysis of hundreds of research studies, dubbed ICAP [25], found that learning increases as engagement increases. ICAP provides a more precise definition to active learning by differentiating between the types of active learning, and ranks these activities as follows (from most to least learning impact): $\mathit{Interactive} > \mathit{Constructive} > \mathit{Active} > \mathit{Passive}$. Passive concerns receiving instruction without any overt action; Active involves students selecting and manipulating materials; Constructive goes beyond what is presented in the material and engages students to generate novel information; and Interactive involves two or more students collaboratively working together via dialog. For instance, simply watching a video is a Passive activity, while answering in-video quizzes and open-ended exercises can promote Active and Constructive learning. Examples in this thesis primarily fall under the Active and Constructive categories, although the Interactive mode is partly triggered as learners see and manipulate materials generated by other learners. A recent MOOC study also confirms ICAP by showing that students who engage in activities learn more than those who primarily watch or read expository content [94].

Another form of constructive learning is self-explanation, an activity that prompts learners to explain their answers and thinking [26]. It increases learning by encouraging students to make inferences and adjust mental models. A surprising research finding is that the learning benefits of self-explanations hold even without the presence of a teaching expert or any performance feedback [26]. Moreover, research has shown that these learning benefits extend to multimedia learning environments [167]. In terms of modality for explanation, students self-explained more frequently when asked to speak than when asked to type [64].

2.3.1 Video Learning Theories

In designing user interfaces for instructional videos, higher interactivity with the content has been shown to aid learning [46, 156]. Tversky et al. [156] state that controlling the playback of an animation such as stopping, starting and replaying can promote reinspection, which in turn can mitigate challenges in perception and comprehension, and further facilitate learning. Semantic indices and random access have been shown to be valuable in video navigation [1, 108, 171] and the lack of interactivity has been deemed a major problem with instructional videos [61]. Tutored Videotape Instruction [51] reported improved learning when pausing videos every few minutes for discussion. VITAL [141] allows students to clip and annotate on video segments, and further embed them in multimedia essays. While VITAL enables adding videos to an activity, active learnersourcing takes an opposite approach by adding activity to a video. In summary, this thesis introduces user interfaces for giving learners more interactivity in video navigation, and learnersourcing methods for acquiring metadata required to create such user interfaces at scale.

2.3.2 Learning Procedural Knowledge

How-to videos help learners master a sequence of steps to complete a procedural task. Chapters 5 and 6 introduce learnersourcing applications designed for how-to videos. There is a rich body of research focusing on mechanisms that improve learning procedural knowledge.

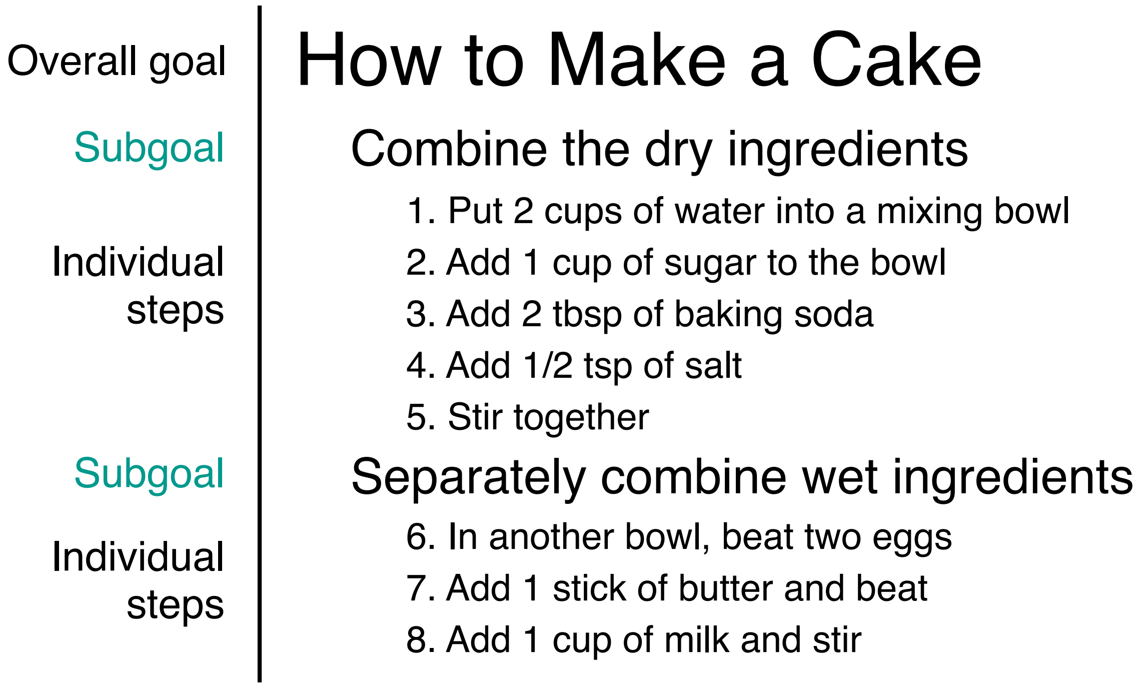

Overall goal
How to Make a Cake
Subgoal
Combine the dry ingredients
Individual steps
1. Put 2 cups of water into a mixing bowl
2. Add 1 cup of sugar to the bowl
3. Add 2 tbsp of baking soda
4. Add 1/2 tsp of salt
5. Stir together
Subgoal
Separately combine wet ingredients
Individual steps
6. In another bowl, beat two eggs
7. Add 1 stick of butter and beat
8. Add 1 cup of milk and stir

Figure 2-1: An example of the breakdown between goal, subgoals, and individual steps for a procedural task. Creating subgoals from individual steps requires self-explanation and summarization.

Eiriksdottir and Catrambone [44] explored the effects of presenting learners with different forms of instructional material for learning procedural knowledge. They discovered that including specific instructions helped learners complete the initial task but those learners did not retain the information. Conversely, learners presented with more holistic instructions had greater learning and transfer.

Previous research [113] has shown that people learn better from videos if they are simultaneously presented with the *subgoals*, which capture meaningful conceptual pieces of the procedure covered in the video (Figure 2-1). Presenting subgoals improves learning by reducing learners' cognitive load by abstracting away low-level details, and by encouraging learners to self-explain why a set of steps have been grouped [23].

Margulieux et al. showed that instructions including both specific steps and subgoals [113] resulted in improved learning and transfer, compared to those with the specific steps alone. Buykx and Petrie [21] included subgoals in recipe instructions and showed that displaying steps and subgoal information improves understanding in domains other than software applications. The TAPS method [24] proposes a systematic workflow for extracting subgoals for existing lecture videos with a domain expert and a knowledge extraction expert working together. This thesis suggests that knowledge extraction experts, namely learners who are novices in their domains, are a viable source for providing information to help other domain novices.

Regarding the effectiveness of learning with how-to videos, there have been mixed results on the effect of instruction media types on learning and performance. Palmiter et al. [131, 132] showed that while users with animated demonstrations completed tasks faster with fewer errors, they showed weaker performance in a retention task after a week. Harrison [62] found that users with animated tutorials learned skills quicker and performed better than with textual tutorials. There was no significant difference between still and animated graphics, however.

Tversky et al. [156] compiled research on animated instructions and concluded that there is no decisive evidence in favor of them. The authors suggest that the advantage is mostly from having more information or interactivity in the animated condition than the static condition. In a more recent meta-analysis of 26 studies comparing static graphics against animated instructions, animations showed a higher effect size [69]. In efforts to solicit when dynamic displays are more effective, the authors find that learners perform better when animated instructions are more representational, realistic, and related to procedural-motor knowledge.

2.4 Video Applications

2.4.1 Video Interfaces for Learning

Many existing online learning platforms support in-video quizzes for flipped classrooms and MOOCs. Khan Academy [khanacademy.org] links supplementary videos to practice problems, and EDpuzzle [edpuzzle.com] and eduCanon [educanon.com] allow inserting quizzes into any video. MOOC platforms such as Udacity [udacity.com], Coursera [coursera.org], and edX [edx.org] commonly pause the video every few minutes and ask short questions to engage learners and test learning.

While MOOC exercises are typically limited to multiple choice and short answer questions, recent systems have focused on incorporating more interactive modalities into video learning. RIMES [84] supports interactive multimedia exercises where learners can record their responses to instructors' exercises in drawing, audio, and video while watching a video. Mudslide [53] asks learners to spatially annotate points of confusion after watching a video, and automatically generates a heatmap of confusion for instructors to review. L.IVE [122] is a video learning platform that integrates video, comments, and assessments in a single view. Improving social learning support is another way to promote interactive learning, via discussion forums, chat rooms [34], peer feedback and grading [137], and live video discussions [99].

As units of learner activity grow in complexity, a major challenge is in making feedback and grading scale: open-ended short answers [8, 19] and essay writing [114] have been explored. For programming assignments, OverCode [54] clusters student solutions and provides a user interface for instructors to filter and explore them.

2.4.2 Video Content Analysis

Content-based video analysis [152] has long been an active research area. Previous research uses image analysis and computer vision to extract keyframes [52], shot boundaries [108], or visual saliency [70]. This thesis applies simple content-based analysis metrics such as pixel differences between video frames in the analysis and UI design. Further, we combine video interaction data and content-based analysis to gain a more complete understanding of learner engagement.

2.4.3 Video Navigation Techniques

To improve video navigation with interaction data, this thesis introduces novel techniques that 1) add richer interactions to the video timeline, 2) support enhanced in-video search, and 3) automatically summarize video content. We now review related work for each of the three techniques.

Video Timeline Enhancements

To improve video scrubbing, YouTube displays thumbnail previews for quickly skimming local frames, and Swift [116] overlays low-resolution thumbnails to avoid network latency delays. The content-aware timeline [139] extracts keyframes with content analysis and plays a video snippet around these points when the user scrubs the timeline. Elastic interfaces [115] use the rubber band analogy to control scrubbing speed and support precise navigation, as seen in the PV Slider [143] and Apple's iOS video interface. We extend this line of research by asking: "What if the scrubbing behavior adapts to learners' watching patterns, as collected from interaction history data?" To our knowledge, no video scrubbing technique has leveraged interaction history data.

Other timeline enhancement techniques for supporting video navigation rely on additional metadata and attributes from video. Chronicle [58] and Video Lens [119] add additional layers of information and facets to the video timeline to improve navigation, search, and browsing. Another thread of research focuses on how-to videos. Existing systems reveal step-by-step structure by adding rich signals to the video timeline, such as tool usage and intermediate results in graphical applications [29, 89].

Lecture videos have a different kind of structure from how-to videos, as they are often organized in conceptual units rather than a sequence of concrete steps. Therefore, capturing and presenting structural signals from lecture videos will need a different approach from how-to videos. For lecture videos, this thesis turns to interaction data that is automatically logged by the system as learners watch the video.

Video Search Techniques

Popular GUI applications such as Web browsers and text editors have incremental search features where the scrollbar and text visually highlight locations of search term occurrences. Also, video players on educational platforms such as edX show a synchronized transcript alongside the currently playing video. Learners can search for text in the transcript and then click to jump to the corresponding spot in the video. We improve these interfaces by augmenting search results with interaction data and visualizing them on the video timeline. Video Google [151] uses computer vision techniques to retrieve and index all the occurrences of various objects in a video. This thesis focuses on extracting higher level information from a video, such as a section summary, rather than fine-grained, object-level information.

Video Summarization

Existing video summarization techniques use video content analysis to extract keyframes [5, 52], shot boundaries [108], and visual saliency [70]. To provide an overview of the entire clip at a glance and support rapid navigation, recent research has used a grid layout to display pre-cached thumbnails [118], short snippets [74] in a single clip, or personal watching history for multiple clips [2], a conceptual hierarchy visualization [75], or a 3D space-time cube display [126].

For educational lecture videos, Panopticon [74], a system that lays out animated grid of video thumbnails for rapid overview, has been shown to shorten task completion time in seeking information inside videos [128]. For blackboard-style lecture videos, NoteVideo [123] reverse-engineers a rendered video to create a summary image and support spatial and temporal navigation.

Recent systems have attempted to create a hybrid of video and text presentation to support browsing and skimming of video. Video Digests [136] and VideoDoc [96] introduce new video formats that reveal a section structure with summaries and transcripts.

This thesis introduces alternative summarization techniques that use data from learners' interaction with the video. This method can be combined with content analysis to incorporate social signals in extracting highlights and navigational cues.

2.4.4 Video Annotation

This thesis focuses on providing a scalable and generalizable video annotation solution without relying on trained annotators, experts, or video authors. Video annotation tools capture moments of interest and add labels to them. Many existing tools are designed for dedicated annotators or experts in limited context. Domain-specific plug-ins [28, 56, 58] automatically capture task information, but require direct access to internal application context (e.g., Photoshop plug-ins accessing operation history). But plug-ins do not exist for most procedural tasks outside of software applications (e.g., makeup), which limits the applicability of this method.

Fully automated video annotation solutions rely primarily on computer vision techniques. Vision-based methods work best when a domain is well-specified and task properties are consistent (e.g., specific software applications [7, 138], graphical user interfaces [39, 168], or fixed camera positions [60]). Recent systems have used deep convolutional neural networks for higher accuracy annotation. A recent system combines text, speech, and vision to align textual step descriptions with video [110], which trains its model with a domain-specific classifier. For educational videos, EdVidParse [140] uses computer vision to annotate people and textual content, two of the common components in most videos. This thesis introduces annotation techniques involving human input that mitigate the limitations of automated methods.

Crowdsourcing video annotation has recently gained interest as a cost-effective method without relying on experts while keeping humans in the loop. Existing systems were designed mostly to collect training data for object recognition [170], motion tracking [155, 161], or behavior detection [134]. Rather than use crowdsourcing to support qualitative researchers, this thesis aims to support end users learning from videos. Adrenaline [10] uses crowdsourcing to find the best frame from a video in near real-time. While [10] and our work both aim to detect a time-specific event from a video stream, our work additionally labels the event and expands to capture surrounding context.

2.4.5 Interactive Tutorials

The HCI community has been actively exploring better formats for tutorials. Recent HCI research either takes a negative position on including video instructions [56, 82], or advocates for using them in context [28, 57, 100, 101, 138]. While the benefits of animated demonstrations are under debate, it is needless to say that a growing number of learners are using video tutorials to master new skills. Popular Photoshop tutorials on YouTube attract millions of viewers.

Tutorial Interfaces for Learning Software Applications

A rich body of previous research has looked into tutorial interfaces for learning software applications. Software applications make a compelling domain for adding learning support, because demonstrations can easily be captured with screen recording software, and application-specific context (e.g., list of commands in Photoshop) can be programmatically accessed. Chapters 5 and 6 contain examples addressing Photoshop and web programming tutorials, but the techniques introduced in these chapters are meant to generalize to other how-to domains.

Some of these interfaces focus on ways to visualize and discover the most suitable tutorials [95, 135], while other systems enhance tutorial content by building on the metadata available in software applications [28, 58, 89, 138]. Specific techniques include automatically generating step-level metadata by demonstration [28, 47, 56, 125], connecting to examples [47, 107, 138], enhancing the tutorial format with annotated information [20, 28, 29, 89, 101], and automatically capturing metadata by mining existing tutorials or video demonstrations [7, 48, 49, 100, 103].

The success of these systems in helping people learn to use software suggests that using existing information (such as the transcript to a tutorial video) is an effective way to improve learning. However, many of these systems rely on the underlying structure of software applications to generate content [58, 95, 135, 138] and cannot easily be generalized to domains other than software. This thesis demonstrates that learnersourcing workflows can provide annotations required to create these interfaces and further enable new ways to learn from tutorials.

Chapter 3

In-Video Dropouts and Interaction Peaks

This chapter presents the first example of passive learnersourcing, which is a large-scale analysis of learners' in-video interaction in Massive Open Online Courses (MOOCs). This chapter has adapted, updated, and rewritten content from a paper at Learning at Scale 2014 [86]. All uses of "we", "our", and "us" in this chapter refer to coauthors of the aforementioned paper.

With thousands of learners watching the same online lecture videos, analyzing video watching patterns provides a unique opportunity to understand how students learn with videos. This chapter reports a large-scale analysis of in-video dropout and peaks in viewership and student activity, using second-by-second user interaction data from 862 videos in four Massive Open Online Courses (MOOCs) on edX. We find higher dropout rates in longer videos, re-watching sessions (vs first-time), and tutorials (vs lectures). Peaks in re-watching sessions and play events indicate points of interest and confusion. Results show that tutorials (vs lectures) and re-watching sessions (vs first-time) lead to more frequent and sharper peaks. In attempting to reason why peaks occur by sampling 80 videos, we observe that 61% of the peaks accompany visual transitions in the video, e.g., a slide view to a classroom view. Based on this observation, we identify five student activity patterns that can explain peaks: starting from the beginning of a new material, returning to missed content, following a tutorial step, re-watching a brief segment, and repeating a non-visual explanation. Our analysis has design implications for video authoring, editing, and interface design, providing a richer understanding of video learning on MOOCs.

3.1 Motivation and Contributions

MOOCs often include hundreds of pre-recorded video clips. Recent research on the first edX course, 6.002x, has shown that learners spend a majority of their time watching videos [18, 149], but little research has been aimed at the click-level interactions within MOOC videos. With thousands of learners watching the same online lecture videos, video analytics can provide a unique opportunity in understanding how learners use video content and what affects their learning experience.

This chapter analyzes click-level interactions resulting from student activities within individual MOOC videos, namely playing, pausing, re-watching, and quitting. We analyze video player interaction logs from four MOOCs offered on the edX platform to identify temporal interaction patterns at the second-by-second level. Specific focus is given to 1) in-video dropout rates and 2) peaks associated with re-watching sessions and play events.

Video dropout, i.e., navigating away from a video before completion, is a measure of engagement. With more engaging videos students might stay until later in the video, resulting in lower dropout. Instructors using videos in their pedagogy need to know what aspects of their videos are the most engaging or most widely viewed. While existing analytics tools provide access to this data, they do not consider different video kinds (lecture or tutorial) and presentation styles (slides, head shot, etc.) specific to the educational context.

When a significant number of students interact with a common portion of a video, the resultant data can be binned to highlight *peaks* in the video timeline. **Peaks in viewership and student activity** can precisely indicate points of interest for instructors and students. These spikes, hereinafter referred to as interaction peaks, can indicate student confusion, introduction of important concepts, engaging demonstrations, or video production glitches. We manually inspected 80 videos from our set to understand why these peaks occur. One notable observation we made is that the peaks often coincide with visual transitions in a video, such as switching from a slide to a classroom view, or from handwritten notes to a software screencast. Combining the interaction data with visual content analysis, we identified five student activity types that can lead to a peak.

This chapter makes the following contributions:

- A first MOOC-scale in-video dropout rate analysis, finding higher dropout rates in longer videos, re-watching students (vs first-time watchers), and tutorials (vs lectures).
- A first MOOC-scale in-video interaction peak analysis, finding more frequent and sharper peaks in re-watching students (vs first-time watchers) and tutorials (vs lectures).
- Categorization of student activities responsible for a peak: starting from the beginning of a new material, returning to missed content, following a tutorial step, re-watching a brief segment, and repeating a non-visual explanation.
- Data-driven design implications for video authoring, editing, and interface design in the context of MOOCs that reflect the temporal interaction patterns of students.

3.2 Video Interaction Dataset

Our dataset consists of interaction logs from the edX video player over four courses offered in Fall 2012. Each log entry contains user name, time of access, video ID, event type, and internal video time, as documented in [43]. A play event is created when the user clicks the play button on the player or scrubs the playhead to a new position while the video is playing. A pause event is created when the user clicks the pause button or scrubs the playhead to a new position when the video is paused. Since a *mousemove* event is not separately logged, exact scrubbing time and position need to be calculated using the play or pause event at the end of a scrub.

Table 3.1 summarizes information on the four courses and their videos. We chose the courses offered at roughly the same time to minimize the effect of changes in the edX platform, logging method, and student population. They span different institutions, subject

Course	Subject	University	Students	Videos	Video Length	Processed Events
6.00x	Intro. CS & Programming	MIT	59,126	141	7:40	4,491,648
PH207x	Statistics for Public Health	Harvard	30,742	301	10:48	15,832,069
CS188.1x	Artificial Intelligence	Berkeley	22,690	149	4:45	14,174,203
3.091x	Solid State Chemistry	MIT	15,281	271	6:19	4,821,837
Total			**127,839**	**862**	**7:46**	**39,319,757**

Table 3.1: Overview of the four edX courses in our dataset offered in Fall 2012. *Students* refers to the number of students who watched at least one video, *Videos* is the number of all video clips posted, *Video Length* is the mean duration, and *Processed Events* is the number of total play and pause events captured by the video player.

fields (computer science, statistics, or chemistry), and recording styles. One of the authors manually labeled video types and presentation styles for all the videos in the video set. Video types represent a pedagogical purpose of a video, including introduction, tutorial, or lecture. Presentation styles represent the visual format of instruction: Powerpoint-style slide, code editor, head shot, classroom recording, and handwritten tablet annotations similar to those used in Khan Academy videos.

3.2.1 Data Processing Pipeline

Our data processing pipeline first reconstructs the watching history of each viewer and then aggregates the per-viewer history data to produce activity statistics for each second-long segment of the video. Specifically, the first step converts raw interaction log entries into watching segments. A watching segment keeps track of all continuous chunks of a clip watched by a user. It includes start and end time for every watched segment. The second step uses the segment information to create second-by-second counts of viewers, unique viewers, re-watching sessions, play events, and pause events. Re-watching sessions only consider a student watching a segment of a video twice or more. Play and pause events increment a bin count if the event is triggered within that bin. Finally, such information can be queried upon request for statistical analysis and further processing.

The data processing module was implemented using Insights, the open source learning analytics library [42], which supports streaming events over SOA (Service-Oriented Architecture) as well as handling requests for query and view. It also uses Python, MongoDB, and the d3 visualization library [17].

3.3 Analysis 1. In-Video Dropout

A dropout rate is defined by the percentage of students who start watching a video but leave before the video finished playing entirely. The dropout rate can reveal the factors that affect students to leave a video, helping video authors to consider them. Also, comparing this rate between videos can illustrate the relative difference in engagement. This analysis could provide valuable feedback to content creators whose courses are rapidly moving toward flipped environments where content consumption occurs online. To our knowledge, no previous work has studied the dropout rates within individual MOOC videos.

3.3.1 Method

For a video of length *n* seconds, let *viewcount(t)* denote the number of unique viewing sessions that include this second for each video. We compute the dropout rate of all videos in our set as: *1.0 - viewcount(n) / viewcount(0)*. Note that all edX videos automatically start playing once the page is open, which might affect the results.

3.3.2 Results

On average across all videos, about 55.2% of viewing sessions (std=14.7) were dropouts before the end. Out of the 55.2% that dropped out, 36.6% (std=11.1) occurred within the first 3% of the video length. This means that 18.6% of the dropouts occur during the rest of the length. It is notable that the dropout rate changes quite dramatically at the beginning of a video.

Why do so many students leave the video very early on? The student might have left the video shortly after it (auto-)started, or the auto-play feature in the edX video player inadvertently started a video. Misleading video titles or course navigation interfaces might be another reason. A tip for content owners on YouTube analytics [55] states that viewers leaving before 5-10 seconds probably means the video keyword or title might not accurately

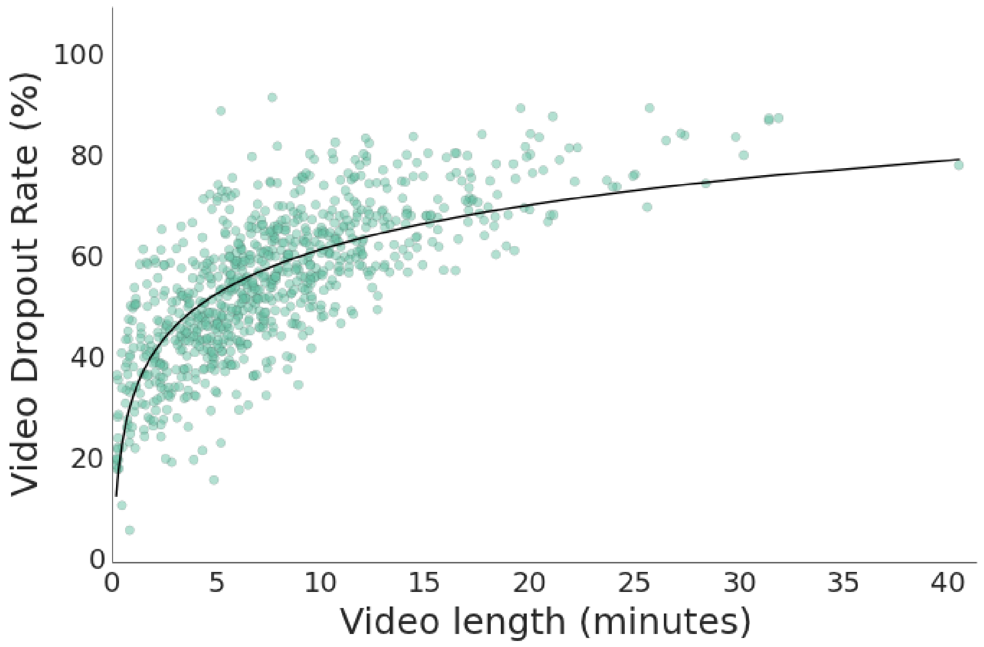


Figure 3-1: Longer videos exhibit higher dropout rates. Our linear regression model uses the log-transformed video length (x-axis) to predict the dropout rate (y-axis). The model fits the data well with r=0.55 with 95% CI = [0.50, 0.59].

represent the content. Additional analysis looking at the common page navigation paths of these early-dropping students might reveal issues with the video title or course navigation structure.

The dropout rate increases with video length (Figure 3-1). Linear regression shows that the logarithmic value of the video length significantly predicted the dropout rate (b = 0.13, t(848) = 32.22, $p < 0.01$). The overall model with the logarithmic value of the video length also predicted the dropout rate very well (adjusted $R^2 = 0.55$, F(1, 848) = 1038, $p < 0.001$). This suggests that for a five-minute video, the predicted dropout is 53% (35% in the first 3%), whereas for a 20-minute video the rate goes up to 71% (47% in the first 3%). With longer videos, students might feel bored due to a short attention span or experience more interruption.

A recent analysis of edX data [59] shows that learner engagement drops significantly if the video length is longer than 6 minutes. Their analysis differs from ours in that they

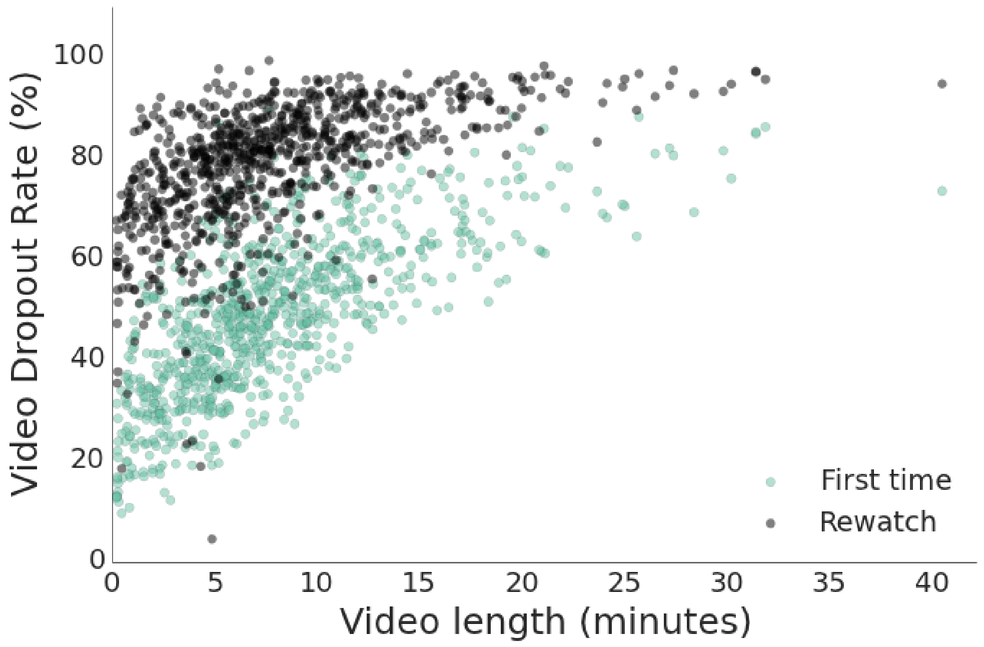


Figure 3-2: Re-watching students tend to drop out more, which might mean that re-watching students watch videos more selectively with a more specific need.

use viewing session length as engagement, as opposed to second-by-second dropout rates. Our analysis can provide additional evidence to the finding that shorter videos are more engaging because more students would drop out.

Another factor that might affect the dropout rate is whether the student watches the video for the first time. Students that are re-watching a video might have more specific information needs and selectively watch a video. Our analysis verifies this assumption as can be seen in Figure 3-2: the dropout rate of re-watchers (78.6%, std=54.0) was much higher than that of first-time watchers (48.6%, std=34.0). A Mann-Whitney's U test shows a significant effect ($Z = -30.7$, $p < 0.001$, $r = 0.74$).

Finally, we look at how video production types affect the dropout rate by comparing lecture videos and tutorial videos. Tutorial videos showed higher dropout rate (61.3%, std=38.3) than lecture videos (54.1%, std=36.3). A Mann-Whitney's U test shows a significant effect ($Z = -5.29$, $p < 0.001$, $r = 0.18$). One explanation is that lecture videos contain first-time introductions to concepts and sequential flow, whereas tutorial videos contain step-by-step instructions students can selectively review and follow along. The mean video length was not significantly different between the two video types ($p > 0.05$), limiting the effect of video length in the result.

3.4 Analysis 2. Interaction Peaks

In addition to staying in a video or leaving, students also actively play, pause, or skip the video to learn at their own pace. Uncovering meaningful patterns from these natural learning activities can provide an in-depth look at video learning on MOOCs. The temporal profiles of such patterns reveal time-specific interest, which might indicate student confusion, pacing issues in the video, useful information presented visually, or important concepts. Course instructors can refer to such information to attend to specific parts of a video. Comparing peak profiles between pedagogically different videos (lecture vs tutorial) can reveal the difference in students' consumption patterns, while comparison between watching contexts (first-time vs re-watching) might highlight different purposes in watching videos.

We investigate **temporal peaks** in the number of interaction events in particular, where a significantly large number of students show similar interaction patterns during a short time window. We use the following two peak definitions.

- A **re-watching session peak** is a sudden spike in the number of re-watching sessions during a period inside a video. We exclude first-time sessions because they tend to be more sequential. We instead focus on non-sequential, random access activities. Note that this measure is not per unique student. A student repeatedly watching a part of a video five times adds five to our measure.

- A **play event peak** is a sudden spike in the number of play events on the player. These events occur when a student clicks the play button or scrubs the playhead to a new position. We ignore autoplay events at the beginning of a video because they do not represent student-initiated activity.

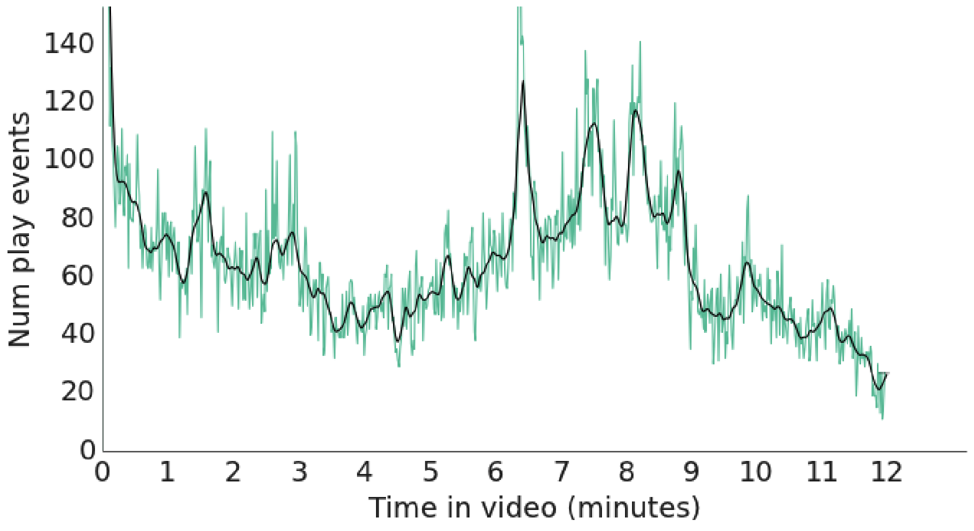


Figure 3-3: Even after aggregating data into bins of one second, the data is noisy (green curve). Kernel-based smoothing reduces noise in the data and helps salient patterns stand out (black curve).

3.4.1 Method

Raw watching session and interaction data are noisy (green curve in Figure 3-3). Identifying peaks in such noisy data both manually and automatically becomes difficult due to local maxima and false peaks. Following the bin-summarize-smooth framework [164], we first bin the data into one-second segments, which simplifies the computation and visualization. We then count all points in each bin to represent an aggregate number of events in a bin. To fight the noise and excessive variance in data and compensate for lost statistical power, we then apply smoothing to the binned and aggregated data (black curve in Figure 3-3). The smoothing technique we use is lowess (locally weighted scatterplot smoothing) [33], with the smoothing parameter of 0.02 after testing various values. A kernel smoother such as lowess is simple and efficient, works well with binned data [162], and is computationally tractable.

After smoothing, we apply a peak detection algorithm to both re-watching session counts and play event counts. The algorithm we use is a variant of the TwitInfo [111] algorithm. It uses a weighted moving average and variance to detect unusually large num-

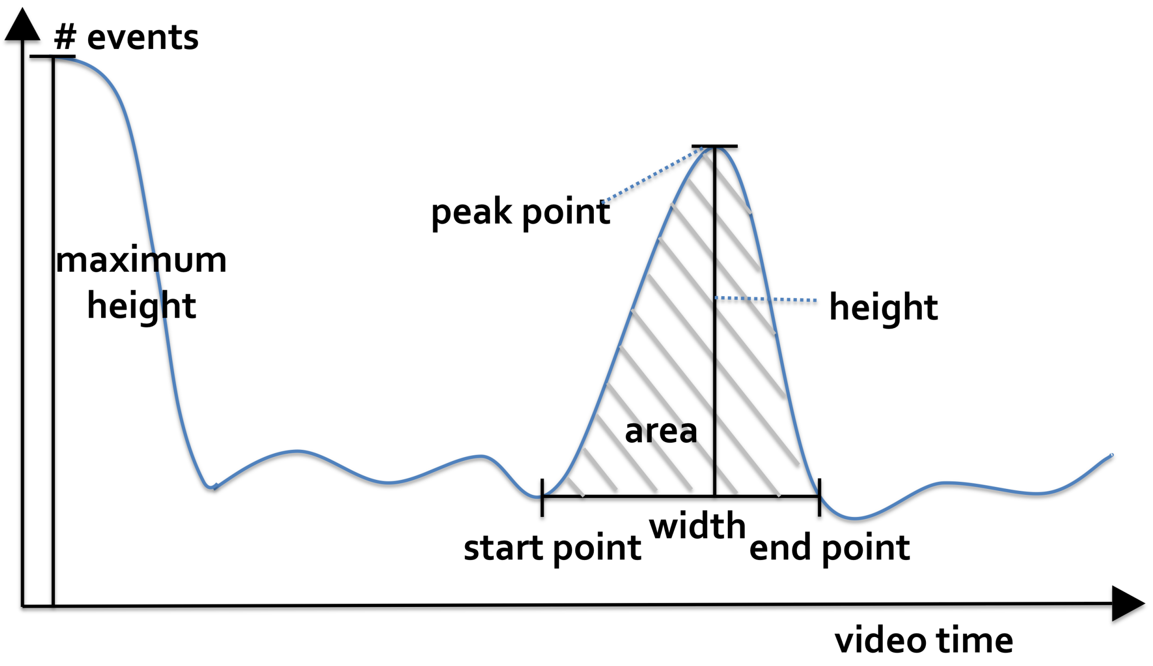


Figure 3-4: The location of a peak is determined by three time points (start, peak, and end). Width, height, and area determine the shape, sharpness, and intensity of the peak.

ber of events in time-series data, which applies well to the video context. We tested with different parameters in the algorithm to fit the time scale of our analysis, which is much shorter (the order of seconds and minutes) than what TwitInfo dealt with (hours and days).

One reason for using both re-watching and play counts is that they might capture different behaviors. We observe that video content includes both a time-specific event (e.g., a visual transition from a talking head to a slide) and a coherent segment that spans a longer period of time (e.g., a two-minute long explanation of a theorem). Play events capture a more precise timing of an event in a video, generally resulting in sharper, spiky peaks. They respond better to student activities at one-second granularity. Re-watching session counts tend to capture segments that occur over a longer period of time better, generally resulting in smoother, wider peaks.

When a re-watching session peak and a play event peak overlap, they point to a single event. When two peak windows overlap, we pick the re-watch peak because re-watch counts are always higher than play counts, possibly resulting in more informed peaks.

The features of a peak, such as width, height, and area, can indicate the strength of

Video Group	All	Lecture	Tutorial	First timers	Re-watchers
Peaks per Video	3.7	3.6	4.1	2.2	1.0
Normalized Height	7.7%	7.1%	10.2%	1.5%	3.1%
Normalized Width	2.7%	2.6%	3.1%	3.2%	3.7%
Normalized Area	4.1%	3.9%	4.8%	4.1%	4.7%

Table 3.2: Peak profile comparison reporting average values across all peaks detected for each video group. Tutorial videos resulted in more peaks than lecture videos. Likewise, re-watching sessions resulted in more peaks than first-time sessions. All differences between lecture and tutorial, and first time and re-watcing were statistically significant.

students' collective, time-specific interest. We compare these features between video types and student contexts. Previous work considered similar constructs in modeling temporal profiles of search queries [78]. A peak is characterized by descriptive properties as shown in Figure 3-4. It includes both start and end time markers, which determine the width or time duration of a peak. The peak point is the highest point between the [start, end] range, which determines the height. Finally, the area under a peak is the sum of event counts during the peak time window, which denotes the relative significance of a peak against the entire video. Multiple peaks of differing profiles might appear within a video clip. In reporting height, width, and area, we normalize the values by scaling between 0 and 1 to address high variability in event counts and durations across videos. For width, height, and area, we take a normalized range against the video duration, the maximum number of events, and the sum of all event counts, respectively.

3.4.2 Peak Profile Comparison

We now explore peak profiles for different video styles and watching behaviors. Overall, the mean number of peaks in a video was 3.7 (std=2.1). Of those, 2.2 (std=1.8) were re-watch peaks, and 2.3 (std=1.5) of them were play event peaks, which includes 0.8 duplicate peaks per video (i.e., play and re-watch peaks were overlapping). Considering that a mean video length was 7.8 minutes, a peak is detected roughly every two minutes in a video. Some videos exhibited as many as 11 peaks, while others did not show a notable peak. Table 3.2 summarizes the results in this section.

The mean width of a peak was 2.7% (std=3.5), and the median width was 9 seconds. This means that peaks in our analysis generally spanned less than 10 seconds including the rise and fall, which can point to highly time-specific events in a video. In the next section we attempt to explain what kind of events might be responsible for a peak.

The mean of normalized peak height was 7.7% (std=10.4) of the maximum height. This indicates that most peaks were quite small when compared against the maximum value of the measure. For play events, the maximum height was autoplay events at the beginning of the video, which gives a practical, comparative measure of the intensity of a peak. For example, if 10,000 students watched a lecture video and a peak had a height of 50%, this indicates that 5,000 more play button clicks were made within the peak range than in the time span just before and after the peak.

Finally, the mean of normalized peak area was 4.1% (std=4.5). This value maps to the activity dominance of a peak. A dominant single peak for a video might indicate that the peak was the single most important point of interest in the video. Conversely, a video with more peaks leaves relatively smaller area for individual peaks.

Lectures vs Tutorials

Tutorial videos generated stronger and more numerous peaks than lecture videos. The mean number of peaks in tutorial videos was 4.1 (std=1.9), compared to 3.6 (std=2.0) in lecture videos. A Mann-Whitney's U test shows a significant effect ($Z = -2.6$, $p < 0.01$, $r = 0.09$). Furthermore, peaks in tutorial videos were wider in width ($Z = -3.1$, $p < 0.001$, $r = 0.06$), taller in height ($Z = -7.5$, $p < 0.001$, $r = 0.13$), and larger in area ($Z = -5.5$, $p < 0.001$, $r = 0.10$) than those in lectures. Where does this difference come from?

Tutorial videos generally contain step-by-step instructions about solving a problem or using a tool. Many students follow along instructions from a tutorial at their own pace, and peaks normally occur at the step boundary. For example, a statistics course included a tutorial video on running a t-test using a statistics software package. In many cases, peaks occurred when the instructor issued commands in the tool or explained a key step in the solution, which might indicate that students re-watched these steps to make sure they follow the steps correctly. On the other hand, lecture videos are less segmented in structure with more continuous flows. Our observations show that peaks in lecture videos often relate to visual transitions in the video, such as from a slide to a talking head, or explanations of important concepts, such as introducing a theorem. While these points of interest in lecture videos attract many students to re-watch, the interaction peaks are not as sharp as in tutorial videos.

First-timers vs Re-watchers

Re-watching sessions generated stronger and more numerous peaks than first-time sessions. The mean number of peaks in re-watching sessions was 2.2 (std=1.7), whereas the mean was only 1.0 (std=1.3) in first-time sessions. A Mann-Whitney's U test shows a significant effect ($Z = -14.7$, $p < 0.001$, $r = 0.35$). Furthermore, re-watching session peaks were wider in width ($Z = -3.9$, $p < 0.001$, $r = 0.07$), taller in height ($Z = -23.8$, $p < 0.001$, $r = 0.45$), and larger in area ($Z = -2.9$, $p < 0.001$, $r = 0.05$) than first-time ones.

First-time watchers might watch videos more sequentially, because they want to master the material by watching through the lecture before diving deeper into specific parts. When re-watching, students tend to watch videos more selectively. It is notable that differences in peak height show a much higher effect size than differences in width and area. This suggests that students selectively pick parts to re-watch rather than watch through sequentially.

3.5 Analysis 3. Five causes for peaks

The peak profile analysis explains what peaks look like and how frequently they occur in different videos, but it does not reveal *why* they occur. We introduce a categorization of student activities surrounding a peak, by combining the peak profile analysis with visual content analysis. While our categorization is not conclusive, it provides an explanation of which semantic and contextual aspects of video might be responsible for a peak. This analysis suggests that no one reason can explain all peaks, and that video instructors should respond to each peak differently.

Our informal observations suggest that **visual transitions** in the video are often associated with a peak. A visual transition is a change between presentation styles shown in

Peak Category	**All**	**Lec.**	**Tut.**
Type 1. beginning of new material	25%	30%	12%
Type 2. returning to content	23%	25%	15%
Type 3. tutorial step	7%	0%	30%
Type 4. replaying a segment	6%	7%	1%
Type 5. non-visual explanation	39%	38%	42%
Number of videos	80	61	19
Peaks per video	3.6	3.6	3.5

Table 3.3: Five student activity types that lead to a peak are shown, along with their frequency distribution as manually labeled by the authors. We sampled 80 videos and labeled each peak to one of the activity types. Only Type 5 does not involve a visual transition.

a video. Presentation styles in our video set are slide, code, talking head, classroom view, studio view, Khan-style tablet, and demo videos. We coded each visual transition with presentation styles before and after the transition, using the following categorization scheme, which is slightly modified from Guo et al's scheme [59]:

- Slide – PowerPoint slide presentation with voice-over
- Code – video screencast of the instructor writing code in a text editor, IDE, or command-line prompt
- Talking head – close-up shots of an instructor's head filmed at an office desk
- Classroom view – video captured from a live classroom lecture
- Studio view – instructor recorded in a studio with no audience
- Khan-style – full-screen video of an instructor drawing freehand on a digital tablet, which is a style popularized by Khan Academy videos
- Demo video – playback of a video clip inside the main lecture video

Example transitions include changes from a slide to a talking head, a code editor to a demo video, a lecture podium view to a slide, etc. These transitions are often added at the production stage by video engineers, who mostly rely on their experiences to determine transition points. Our definition of visual transitions does not include incremental changes within a single style, e.g., an instructor typing in a new line of code in the code editor, adding an underline to highlight text, and walking a few steps in a classroom view.

3.5.1 Method

To explore the connection between visual transitions and interaction peaks, we apply a visual analysis technique to complement the log analysis. We use an image similarity metric that computes pixel differences between two adjacent frames to quantify the amount of visual changes in the video. Our pipeline first samples a video frame every second, computes the image similarity using a standard technique, Manhattan distance, in the RGB space, and finally stores the pixel distance value. We visualize this data to aid the following categorization process.

We sampled 80 videos out of 862 (9.3%) while keeping the balance between video lengths, lectures vs tutorials, and production styles. This set included 20 videos from each course.

The categorization process involved two phases. In the first phase, researchers watched the selected videos, especially paying attention to the detected peaks. The goal was to construct a set of categories for peaks, using the open card sorting method [146]. As the researchers watched videos, they grouped peaks into rough categories based on common properties, such as the existence of visual transitions before or after a peak window. They discovered five groups in this generative process and named each. Three data streams were visualized to help with the categorization process, namely play events (Figure 3-5 top), re-watching sessions (Figure 3-5 middle), and the pixel differences (Figure 3-5 bottom). In the second phase, a researcher labeled all peaks in the 80 videos to one of the categories generated in the first phase.

3.5.2 Results

Overall, 61% of the categorized peaks involved a visual transition before, and/or after the peak. The categories, their descriptions, and frequency are shown in Table 3.3. We now

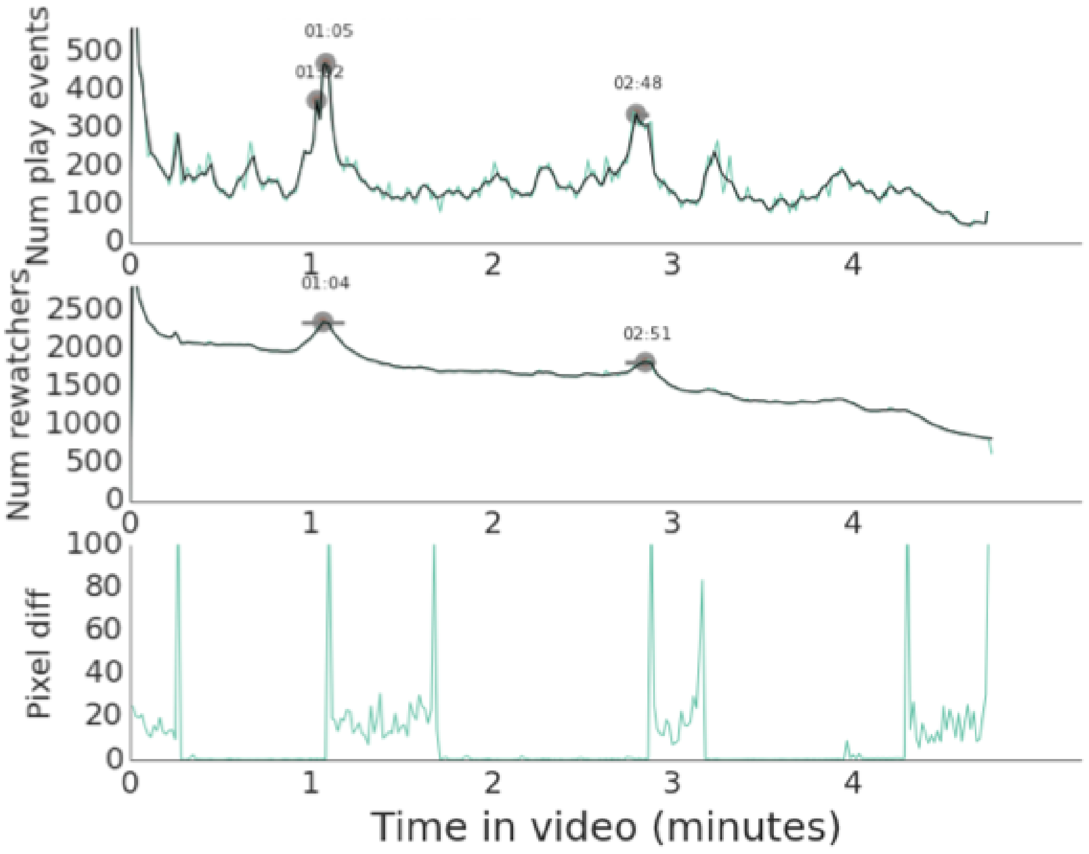


Figure 3-5: We visualize three streams of data to analyze interaction peaks in MOOC videos. The top graph shows play events, the middle graph shows re-watching sessions, and the bottom graph shows pixel differences over time. Detected peaks are marked with a gray point. In this example, the detected peaks coincide with a spike in pixel differences, which indicate a visual transition in video.

describe each student activity category in detail.

Type 1: starting from the beginning of a new material

In this category (25% of all peaks), students browse to the beginning of a new material, such as a new concept, example, or theorem. A peak caused by such activity includes a visual transition that precedes the peak. This indicates that students are interested in the content that comes after the visual transition, which is often where new units start. Students might want to review a confusing concept after mastering earlier ones, or re-visit a theorem proof sequence. These peaks might indicate good points to cut the longer video

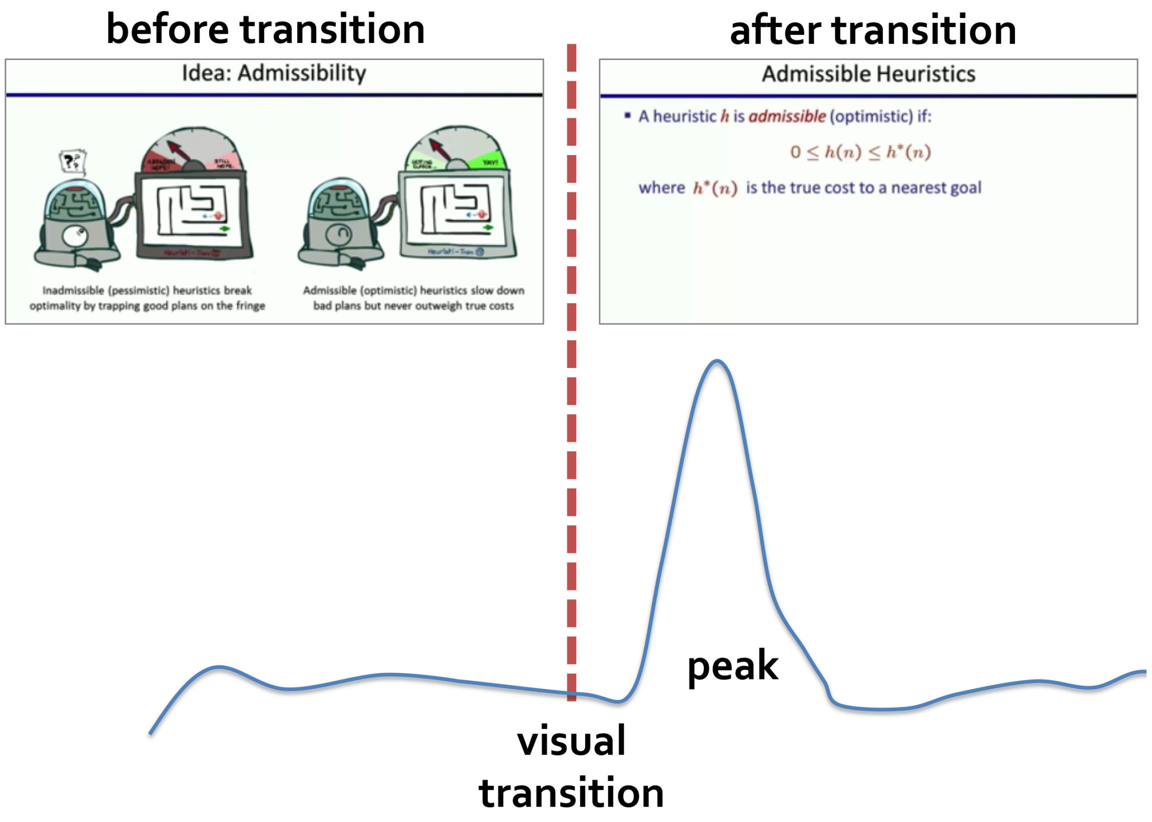
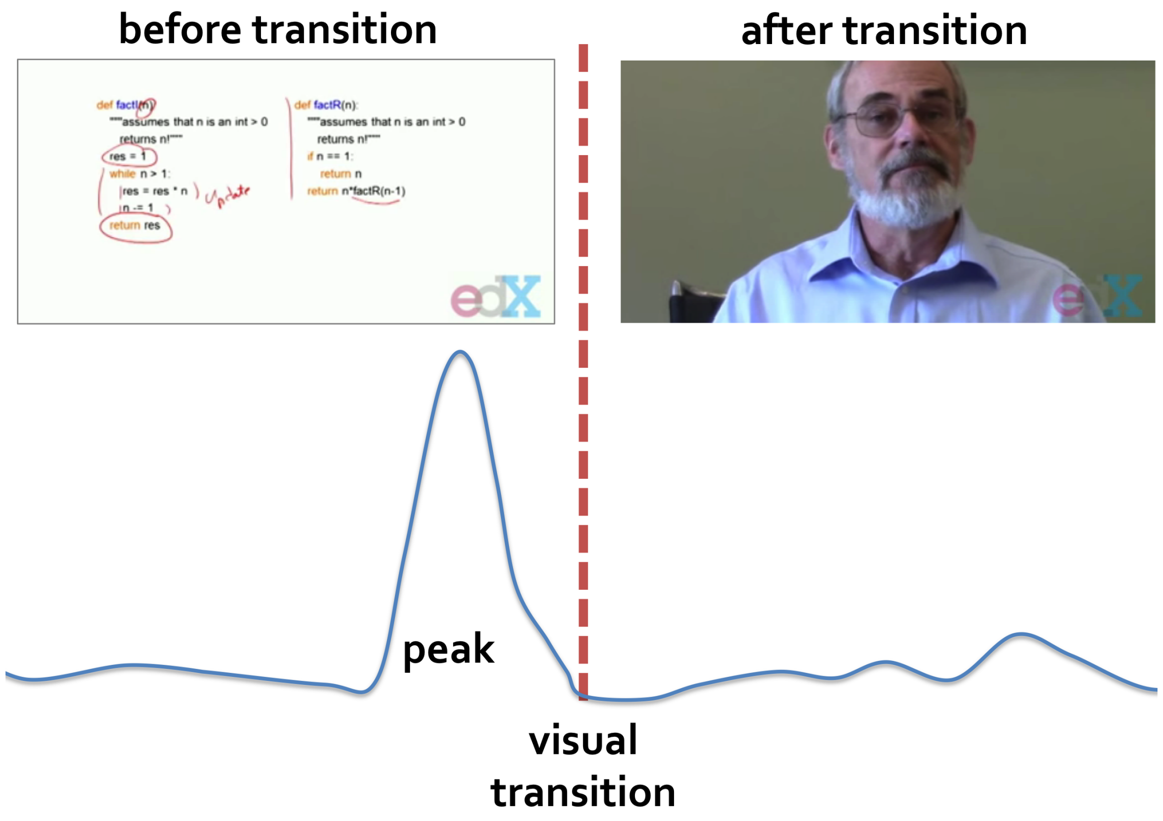

before transition
after transition
Idea: Admissibility
Inadmissible (pessimistic) heuristics break optimality by trapping good plans on the fringe
Admissible (optimistic) heuristics slow down bad plans but never outweigh true costs
Admissible Heuristics
A heuristic h is *admissible* (optimistic) if:
$0 \leq h(n) \leq h^*(n)$
where $h^*(n)$ is the true cost to a nearest goal
peak
visual transition

Figure 3-6: This peak represents the start of a new concept. The instructor started presenting a formal definition of a concept (admissibility) after changing the slide. The peak occurred when this concept explanation started.

into shorter segments, because they correspond to the beginning of a semantically different unit. Figure 3-6 shows an example from an AI course where a formal description of a concept (admissibility) started after presenting a motivating idea.

Type 2: returning to missed content

In this category (23% of all peaks), students return to visual content that disappears shortly after. A peak caused by such activity includes a visual transition that follows shortly after the peak. Often, the content that disappears is slides, code snippets, or board notes, but not talking heads or zoomed out views. An interpretation is that there is a pacing issue in the video. The visual transition was maybe too abrupt, not giving enough time for students to fully digest the content that disappeared. They need more time on the material,



```
def factI(n):
  """assumes that n is an int > 0
    returns n!"""
  res = 1
  while n > 1:
    res = res * n
    n -= 1
  return res
```

```
def factR(n):
  """assumes that n is an int > 0
    returns n!"""
  if n == 1:
    return n
  return n*factR(n-1)
```

Figure 3-7: This peak represents students returning to see the code snippet slide that disappeared after transitioning into the talking head. An abrupt transition might not give students enough time to comprehend what's presented.

but the video view suddenly changed and prevented access to the material. Also, note that what is shown during this peak type is often the final content that is complete, such as fully working code or a complete bullet point list. Many instructors make slides that advance progressively instead of showing everything at once to keep students' attention focused. When re-watching, students might want to skip to the final result without repeating all intermediate steps. Figure 3-7 shows an example where the code snippet suddenly disappeared and transitioned into the instructor talking.

Type 3: following a tutorial step

This category (7% of all peaks) is students following steps in the tutorial. Tutorials often contain step-by-step instructions students can follow, in the form of issuing a command or

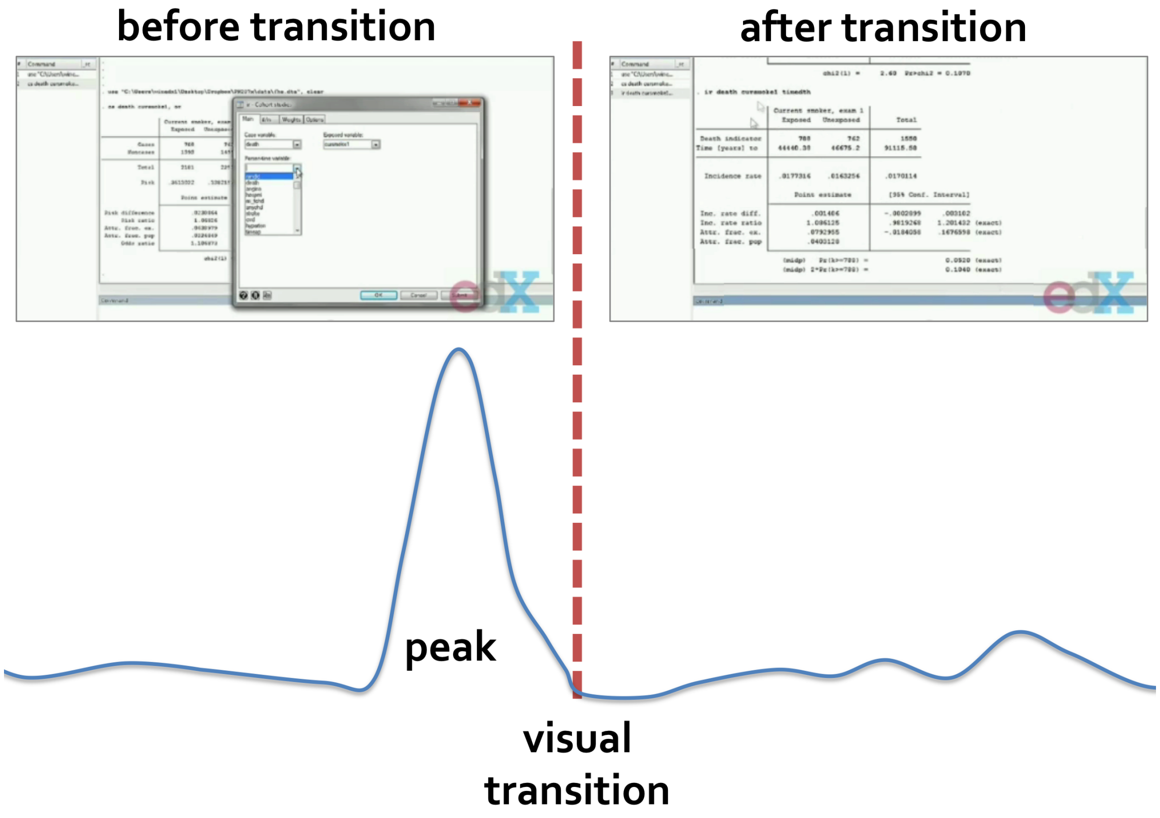


Figure 3-8: This peak represents students returning to a procedural step demonstrating how to run a command inside a statistics package. Students are more interested in following along the steps than the result afterward, probably because they can see the same result in their own application as well.

selecting a menu item from an application. Many students pause or re-watch right before an action takes place, possibly trying to replicate the step in their own tool. Since this was a recurring pattern in many of the tutorial videos, we assign a separate category. Figure 3-8 shows an example from a tutorial video where the instructor in the statistics course demonstrated how to run a command from a statistics package.

Type 4: re-watching a brief segment

In this category (6% of all peaks), visual transitions are located both before and after the peak. This indicates that students are interested in the content within the peak range. While much less common than the other types, this type gives more specific information about student behavior because reasons explaining both Type 1 and 2 can be applied here. Fig-

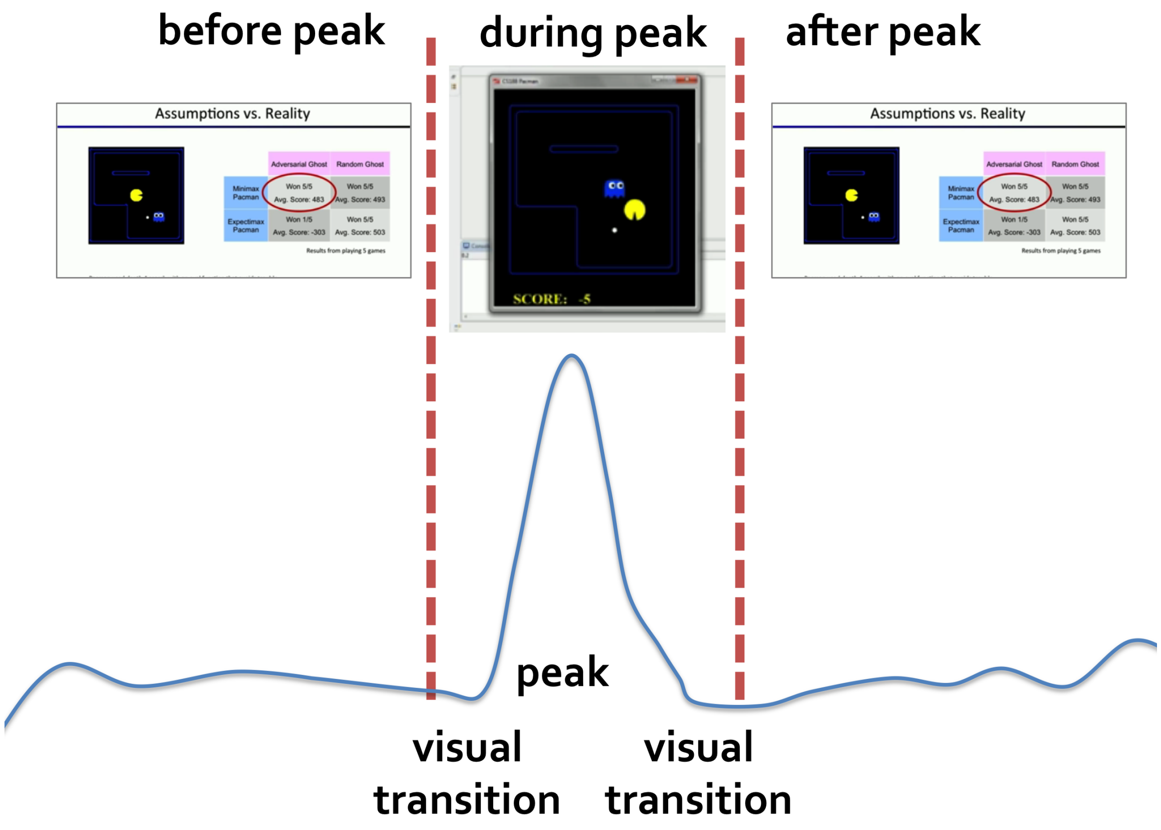
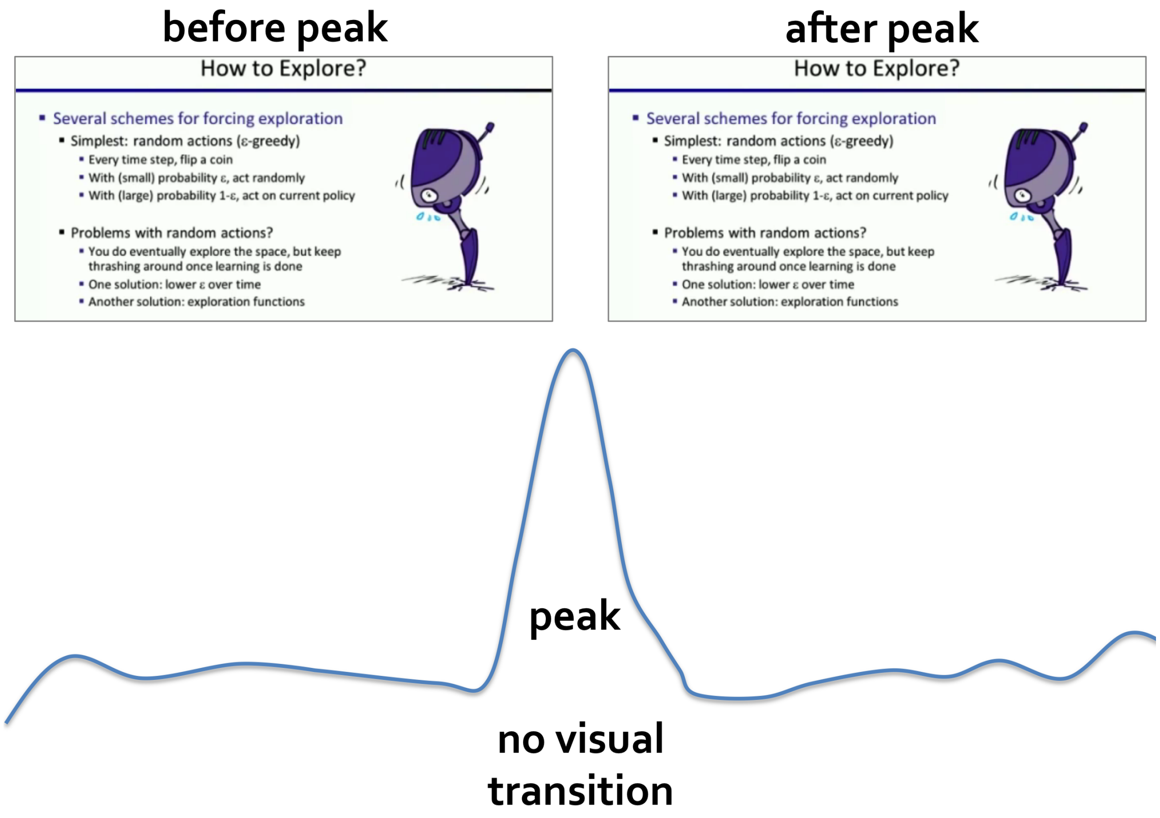


Figure 3-9: This peak represents a short range of interesting segment surrounded by visual transitions before and after. The instructor launched a game application that demonstrates the concept discussed. This engaging demo might have encouraged students to return to it.

ure 3-9 shows an example where the instructor briefly showed a demo application (during peak), and explained an underlying concept before and after the demo.

Type 5: repeating a non-visual explanation

In this category (39% of all peaks), students return to parts of a video that have no visual transitions nearby. What triggers a peak is non-visual activities in the video, such as a verbal instruction with semantic importance. We note that in many cases these peaks represent instructors introducing an important concept, re-emphasizing what has already been covered visually, or making a joke that results in a burst of laughter. Figure 3-10 shows an example where a peak occurred within a single slide. Here the instructor of the AI course



How to Explore?

- Several schemes for forcing exploration
  - Simplest: random actions (ε-greedy)
    - Every time step, flip a coin
    - With (small) probability ε, act randomly
    - With (large) probability 1-ε, act on current policy
  - Problems with random actions?
    - You do eventually explore the space, but keep thrashing around once learning is done
    - One solution: lower ε over time
    - Another solution: exploration functions

How to Explore?

- Several schemes for forcing exploration
  - Simplest: random actions (ε-greedy)
    - Every time step, flip a coin
    - With (small) probability ε, act randomly
    - With (large) probability 1-ε, act on current policy
  - Problems with random actions?
    - You do eventually explore the space, but keep thrashing around once learning is done
    - One solution: lower ε over time
    - Another solution: exploration functions

Figure 3-10: This peak represents important remarks from an instructor, without any visual transitions in the video. In this example the instructor was making an important point about random actions in reinforcement learning, the key topic of this AI lecture.

explained the concept of taking random actions to force exploration in reinforcement learning, which was the main topic of the video.

Are there differences between peak types?

We compared normalized width, height, and area between peak types to see if peak categories, defined by the semantics of the video, map to differences in the peak profile. We first compared peaks accompanying visual transitions (Type 1, 2, 3, 4) and peaks with non-visual explanation (Type 5). A Mann-Whitney's U test shows a significant effect of height ($Z = -3.0$, $p < 0.01$, $r = 0.18$) and area ($Z = -1.9$, $p < 0.05$, $r = 0.11$), but not of width. This shows that peaks were taller and larger in size when they had visual transitions nearby. One explanation might be that visual transitions, occurring at the exact same time for all students, lead students to act similarly around them. On the other hand, start and end times of a salient activity are less clear for non-visual explanations.

Next, we looked at differences between individual categories. A Kruskal Wallis test revealed a significant effect of category on normalized height ($\chi^2(4)=19.6$, $p < 0.001$). A post-hoc test using Mann-Whitney tests with Bonferroni correction showed the significant differences between Type 1 and Type 3 ($p < 0.01$, $r = 0.33$), and between Type 3 and Type 5 ($p < 0.001$, $r = 0.32$). This suggests that tutorial step peaks (Type 3) were significantly taller than new material peaks (Type 1) or non-visual explanation peaks (Type 5). There was no significant effect found for normalized width or area. One explanation might be that tutorial steps have a clear timestamp and span a shorter period of time. For example, time between a tutorial instructor typing a command and pressing enter can be very short. The student needs to pause the video within a very short time range to capture the timing with the full command entered. For new materials and non-visual explanations, a few seconds of difference is not crucial, which might lead to smoother peaks.

3.6 Design Implications for MOOC Video Interfaces

The micro-level analysis of students' video interaction introduced in this chapter can guide the design of better video learning experiences. Our analysis shows that students interact with MOOC videos differently, depending on the visual, pedagogical, and stylistic properties of the video. A primary finding from both the dropout and peak analyses is that students selectively pick parts of videos to watch. And the parts they choose tend to converge to form peaks. We argue that course instructors, video production engineers, platform designers, and even students can benefit from such information. We present a set of design implications from our results for different types of learners and videos addressed in this chapter.

[authoring] **Avoid abrupt visual transitions.** Type 2 peaks (returning to missed content) are likely to indicate too fast or abrupt transitions. These peaks often accompany informative slides, which can be made available outside the video as a screenshot or thumbnail for easier scanning and reviewing. Excessive visual transitions should be avoided because they might prevent students from referring to earlier content. A video interface can automatically generate a picture-in-picture or side-by-side view to retain access to earlier content. Alternatively, the instructor figure may be overlaid on top of the content all the time to avoid any transition at all.

[authoring] **Make shorter videos.** Long lecture videos lead to a higher dropout rate. When determining points to segment long videos, Type 1 peaks can be useful points because students watch from the beginning of that segment.

[interface] **Enable one-click access for steps in tutorial videos.** Important steps in a tutorial get clear peaks. These peaks can be used to automatically mark steps in a video, making it easy for students to non-sequentially access these points without having to rely on imprecise scrubbing. For example, ToolScape [89] is a tutorial video interface that adds an interactive timeline below a video to allow step-by-step navigation.

[interface] **Provide interactive links and screenshots for highlights.** Type 2 peaks (returning to missed content) suggest that missing content forces students to return. Providing static screenshots of the peak-creating informative frames might reduce the navigation overhead for students. Video interfaces might even consider multi-track streams, showing slide and instructor in separate channels that are available all the time. Type 5 peaks attract students with non-visual information, and our observation suggests that instructors make important points in these peaks. Interactive links to these points can be useful for students willing to find them later, which is especially difficult due to the lack of visual cues. Chapter 4 shows some of these ideas in a video player.

[interface] **Consider video summarization for selective watchers.** A common interaction pattern in our results is non-sequential and selective watching. Students re-watching videos tend to non-sequentially seek their points of interest. Peaks can be used to effectively summarize highlights from a video, which can be useful for students who re-watch or skim through the content while auditing. Chapter 4 shows some of these ideas in a video player.

3.6.1 MOOC video analytics platform

Techniques presented in this chapter can provide stakeholders in a MOOC with richer data about micro-level video interaction, which can help them make data-driven decisions about planning, recording, editing, and revising videos. To support exploration of in-video interaction data, we built a prototype MOOC video analytics platform. In addition to showing basic statistics per video, the enhanced video player synchronizes the video playhead with an overlay time bar on the visualization (Figure 3-11). This interface enables visually connecting video content to points with salient patterns in the graph.

We expect to support the sensemaking process for course instructors, video production engineers, and platform designers. Course instructors can use MOOC video analytics to respond to students' interest and confusion while a course is being offered. Further, they can also use data-driven metrics to revise videos for the next offering of the course. Video production engineers can better allocate their resources in the production effort. One concrete use case is to avoid excessive visual transitions that lead to Type 2 peaks. Platform designers can benefit from MOOC video analytics to enhance the video player interface. For example, they can attach interactive bookmarks for peaks to improve in-video navigation.

While the analysis for this chapter was done offline after the courses were complete, the analytics platform can also handle streaming events. This allows running our analytics framework for currently active courses, so that instructors can address student confusion inferred from the streaming video analytics during virtual office hours or in discussion forums.

3.7 Limitations

While our analysis methods identified video navigation patterns, understanding *why* we see these patterns is difficult. Because MOOCs do not have access to a broader learning context of a student, log entries cannot accurately represent learners' real intent (e.g., the same log may be recorded for the two following cases: start playing a video and walk out of the room, and start playing and pay attention). Also, video interactions might depend on other pedagogical methods in a MOOC such as problem sets, discussion forums, and exams.

Furthermore, presentation quality or storyline might also affect which parts of the video students come back to watch, but our analysis does not incorporate such data. Finally, our analysis does not consider different learner goals in MOOCs, such as completing, auditing, and disengaging [92]. Per-group analysis of our techniques might reduce noise and help us better reason about the dropout and peak results.

3.8 Conclusion

This chapter provides an in-depth look into how students interact with MOOC videos. We analyze data from four live courses on edX, focusing on in-video dropout rates, interaction peak profiles, and student activity categorization around peaks. We believe our data-driven analytic methods can help improve the video learning experience.

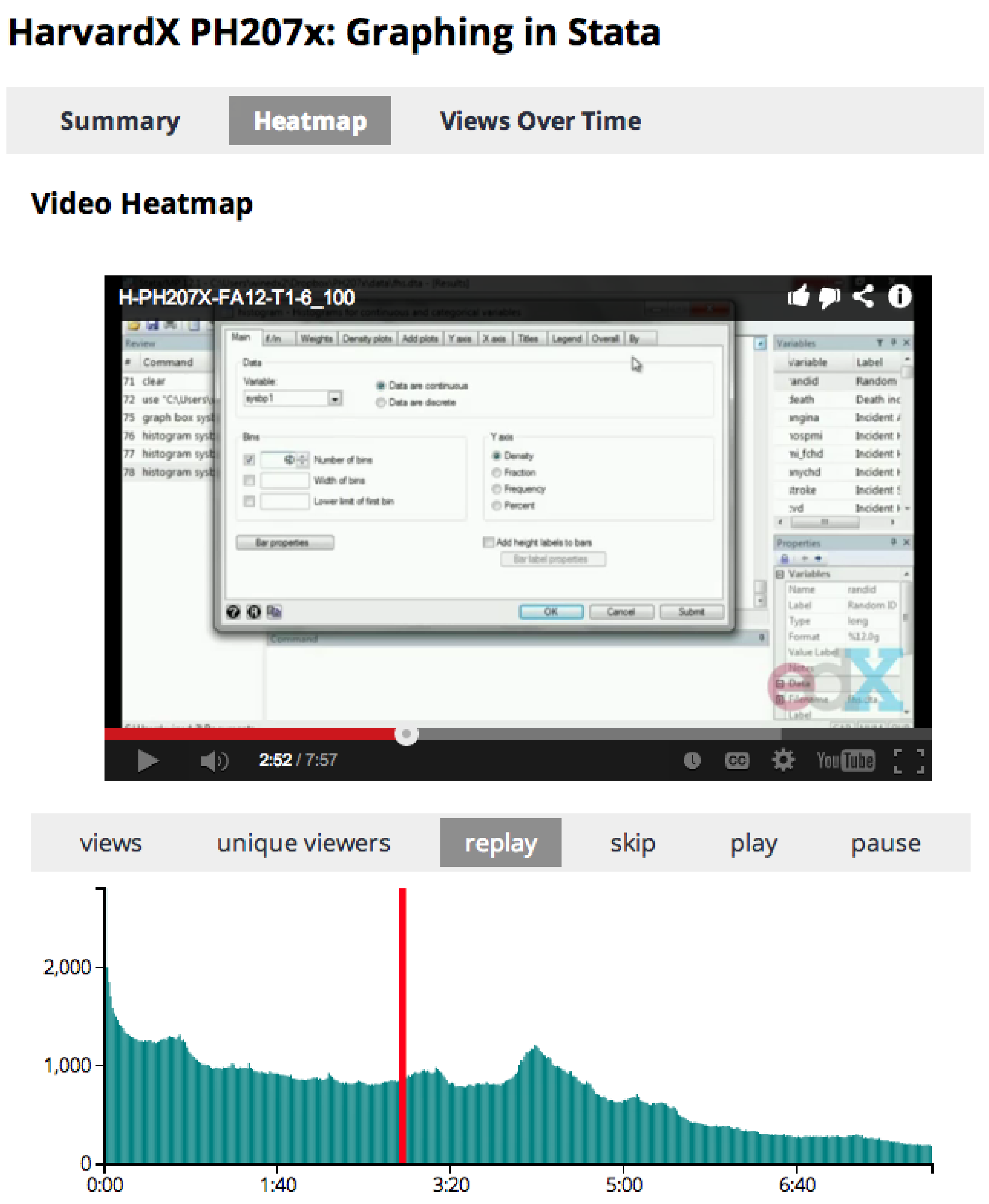


Figure 3-11: Our prototype video analytics dashboard supports synchronized video playback for various interaction measures.

Chapter 4

LectureScape: Data-Driven Navigation of Educational Videos

This chapter presents the second example of passive learnersourcing, LectureScape, which is a lecture video player powered by learners' collective video interaction data. This chapter has adapted, updated, and rewritten content from a year at UIST 2014 [85]. All uses of "we", "our", and "us" in this chapter refer to coauthors of the aforementioned paper.

With an unprecedented scale of learners watching educational videos on online platforms such as MOOCs and YouTube, there is an opportunity to incorporate data generated from their interactions into the design of novel video interaction techniques. Chapter 3 introduced how interaction data can be used to understand learners' collective video engagement and help instructors improve their videos. The interaction data analysis also led to a set of recommendations for better MOOC video design. This chapter takes a step further, and attempts to implement and test some of these recommendations in a user interface. The goal is to improve the video navigation and learning experience by exploring the design space of data-driven interaction techniques. We introduce a set of techniques that augment existing video interface widgets, including:

- a 2D video timeline with an embedded visualization of collective navigation traces
- dynamic and non-linear timeline scrubbing
- data-enhanced transcript search and keyword summary

- automatic display of relevant still frames next to the video
- a visual summary representing points with high learner activity

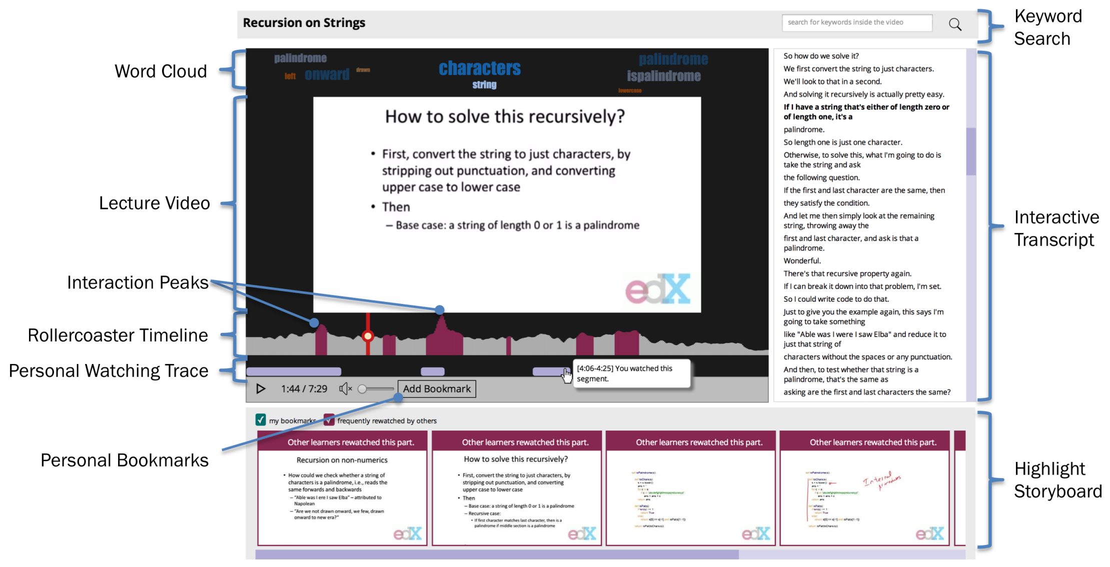
To evaluate the feasibility of the techniques, we ran a laboratory user study with simulated learning tasks. Participants rated watching lecture videos with interaction data to be efficient and useful in completing the tasks. However, no significant differences were found in task performance, suggesting that interaction data may not always align with moment-by-moment information needs during the tasks.

4.1 Motivation and Contributions

Millions of people watch free educational videos online on platforms such as Khan Academy, Coursera, edX, Udacity, MIT OpenCourseWare, and YouTube. For example, the "Education" channel on YouTube currently has over 10.5 million subscribers, and a typical MOOC has thousands of video-watching learners. In addition, learners also take paid video-centric courses on commercial platforms such as Lynda, Udemy, and numerous university e-learning initiatives.

The server logs of these platforms contain fine-grained, second-by-second data of learners' interactions with videos, which we refer to as *interaction traces*. This data is now being used for real-time analytics to optimize business metrics such as viewer engagement time. Researchers have also used this data to perform retrospective empirical analyses. For example, video analytics studies on MOOCs have compared the effects of video production methods on learner engagement [59] and identified common causes of peaks in learner activity within videos (Chapter 3).

Interaction data provides a unique opportunity to understand collective video watching patterns, which might indicate points of learner interest, confusion, or boredom in videos. However, to our knowledge, researchers have not yet attempted to feed these patterns back into the video navigation interface to support learners. While learners might have diverse goals in navigating through a video, existing video interfaces do not provide customized navigation support beyond scrubbing on a linear timeline slider with thumbnail previews



How to solve this recursively?

- First, convert the string to just characters, by stripping out punctuation, and converting upper case to lower case
- Then
  - Base case: a string of length 0 or 1 is a palindrome

[4:06-4:25] You watched this segment.

So how do we solve it?
We first convert the string to just characters.
We'll look to that in a second.
And solving it recursively is actually pretty easy.
If I have a string that's either of length zero or of length one, it's a
palindrome.
So length one is just one character.
Otherwise, to solve this, what I'm going to do is take the string and ask
the following question.
If the first and last character are the same, then
they satisfy the condition.
And let me then simply look at the remaining string, throwing away the
first and last character, and ask is that a palindrome.
Wonderful.
There's that recursive property again.
If I can break it down into that problem, I'm set.
So I could write code to do that.
Just to give you the example again, this says I'm going to take something
like "Able was I were I saw Elba" and reduce it to just that string of
characters without the spaces or any punctuation.
And then, to test whether that string is a palindrome, that's the same as
asking are the first and last characters the same?

Recursion on non-numerics

- How could we check whether a string of characters is a palindrome, i.e., reads the same forwards and backwards
  - "Able was I ere I saw Elba" – attributed to Napolean
  - "Are we not drawn onward, we few, drawn onward to new era?"

How to solve this recursively?

- First, convert the string to just characters, by stripping out punctuation, and converting upper case to lower case
- Then
  - Base case: a string of length 0 or 1 is a palindrome
  - Recursive case:
    - If first character matches last character, then is a palindrome if middle section is a palindrome

Figure 4-1: This chapter presents three sets of novel interaction techniques to improve navigation of educational videos. 1) Dynamic timelines (Rollercoaster Timeline, Interaction Peaks, and Personal Watching Trace), 2) Enhanced in-video search (Keyword Search and Interactive Transcript), 3) Highlights (Word Cloud, Personal Bookmarks, Highlight Storyboard). All techniques are powered by interaction data aggregated over all video watchers.

and synchronizing with a textual transcript. Adapting to collective video watching patterns can lead to richer social navigation support [37].

This chapter explores the design space of navigation techniques for educational videos that leverage interaction data. We introduce novel data-driven interaction techniques that process, visualize, and summarize interaction data generated by many learners watching the same video. For instance, in a typical MOOC, at least a few thousand learners watch each video. Based on prior findings about learner intent and typical formats of educational videos [59, 86], we have designed these techniques to support fluid and diverse video navigation patterns. Typical video watching scenarios include:

- Rewatch: "Although I understand the high-level motivation, I didn't quite get the formal definition of 'admissible heuristic' the first time I watched this lecture. So I want to rewatch the section explaining the formal definition."
- Textual search: "I want to jump to where the instructor first mentioned the phrase 'alpha-beta pruning.'"

- Visual search: "I remember seeing this code example in a diagram somewhere in the video. I want to find it again."
- Return: "Hey, that was annoying! I don't want to see the instructor's talking head. I'm not done looking at this PowerPoint slide yet. I want the slide back!"
- Skim: "This lecture seems somewhat trivial. I'll skim to see if there's something I probably shouldn't miss."

Specifically, we developed interaction techniques to augment a traditional Web video player with 1) a Rollercoaster timeline that expands the video timeline to a 2D space, visualizes collective interaction traces of all learners, and dynamically applies non-linear scrolling to emphasize interaction peaks, 2) enhanced in-video search that visualizes and ranks occurrences on the timeline and recommends salient keywords for each video section, and 3) a video summarization method that captures frames that are frequently viewed by other learners. These techniques combine learners' collective interaction traces with text and visual content analysis.

We package all of these techniques together in LectureScape, a prototype Web-based video interface shown in Figure 4-1. In a laboratory study with simulated search and skimming tasks, we observed that participants employ diverse video navigation patterns enabled by our techniques. Specifically, they noted that LectureScape helped them to quickly scan the video and efficiently narrow down to parts to direct their focus. They also found interaction data to be useful in identifying important or confusing pedagogical points within videos. However, no significant differences were found in task performance, suggesting that interaction data may not always align with moment-by-moment information needs participants had for the study tasks.

This chapter makes the following contributions:

- a conceptual design approach for interaction techniques that leverages information about other learners' behavior to improve the video learning experience,
- a set of novel video interaction techniques powered by real log data from learners in a MOOC platform, introducing 1) a 2D, non-linear timeline, 2) enhanced in-video search, and 3) a data-driven video summarization method,

- and an empirical evaluation of the techniques with learners, which enabled fluid and diverse video navigation.

4.2 Design Goals

This work focuses on supporting video navigation patterns common in online education, which differ from watching, say, a movie or TV show in a sequential, linear manner. Our designs are informed by quantitative and qualitative findings from analyses of educational videos, which suggest that learners often re-watch and find specific information from videos. A video clickstream analysis introduced in Chapter 3 identified *interaction peaks* inside a video, of which 70% of automatically-detected peaks coincided with visual and topical transitions. A challenge in using interaction data to support learning is that the meaning of an interaction peak can be ambiguous (e.g., interest, confusion, or importance). In this chapter, we do not assume a specific meaning behind interaction peaks, but do assume they are worth emphasizing regardless of the real cause. If a peak indicates importance, it would make sense to highlight it for future learners. If it indicates confusion, it may still make sense to emphasize so that learners would pay more attention.

To discover unsupported needs in lecture video navigation, we also conducted multiple rounds of feedback sessions with learners using our initial prototypes. The data analysis and interviews led to three high-level goals that informed our design of data-driven video interaction techniques.

Provide easy access to what other learners frequently watched. Our observations suggest that learners find it hard to identify and navigate to important parts of information-dense educational videos. To help a learner make more informed decisions about which part of the video to review, we leverage other learners' interaction traces, especially interaction peaks. We designed navigation techniques to emphasize these points of interest while the learner visually scans the video or physically scrubs the timeline.

Support both personal and collective video summaries. To prepare for homework assignments or exams, learners often take notes and watch videos multiple times to create a meaningful summary. Since there are often thousands of learners watching each video, we explore ways to present collective interaction traces as an alternative summary to complement each learner's personal summary. We extend prior work on social navigation in videos [121], history visualization, and revisitation mechanisms by supporting both manual bookmarking and automatic personal and collective watching traces.

Support diverse ways to search inside of a video. In our formative studies, learners described different ways they look for specific information inside a video. They would rely on both textual cues (e.g., topic and concept names) and visual cues (e.g., an image or a slide layout) to remember parts of the video. A more challenging case is when they cannot remember what the cue was for their particular information need. This observation inspired us to support both active search (e.g., when the learner has a clear search term), and ambient recommendations (e.g., when the learner does not know exactly what to search for). We designed techniques to enhance existing search mechanisms with interaction data, which provide social cues to serve both search scenarios.

4.3 Data-Driven Video Navigation Techniques

We introduce three interaction techniques to improve navigation of educational videos: an alternative timeline, search interface, and summarization method. Our main insight is to use the non-uniform distribution of learner activity within a video to better support common navigation patterns. Although the prototypes shown in this chapter use videos on edX, a MOOC (Massive Open Online Course) platform, the techniques can be implemented for other video platforms such as YouTube because they use only standard Web technologies.

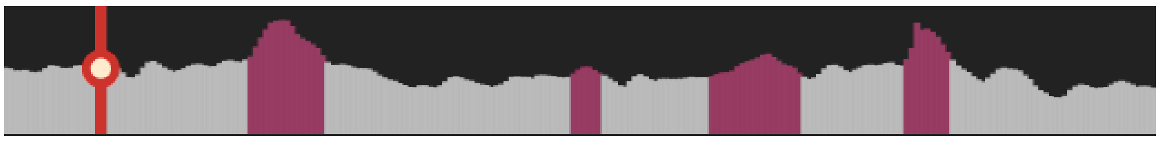


Figure 4-2: The 2D Rollercoaster timeline that appears below each video instead of a traditional 1D timeline. The height of the timeline at each point shows the amount of navigation activity by learners at that point. The magenta sections are automatically-detected interaction peaks.

4.3.1 The Rollercoaster Timeline: 2D, Non-Linear Timeline

To help learners identify and navigate to important parts of the video, we introduce the rollercoaster timeline. Unlike a traditional 1D timeline, the rollercoaster timeline is 2D with an embedded visualization of second-by-second learner interaction data (Figure 4-2). It visualizes the navigation frequency as a proxy of importance, as revealed by the behavior of other learners, and modifies the timeline scrubbing behavior to make precise navigation in important regions easier.

Navigation events are logged when the learner pauses and resumes the video, or navigates to a specific point. The Rollercoaster timeline uses navigation event counts as the vertical dimension. These counts are then applied lowess (locally weighted scatterplot smoothing) [33] for smoothing, with the smoothing parameter of 0.02. The smoothing process is identical to what has been described in Chapter 3. The Rollercoaster timeline can also visualize other kinds of interaction events, including the number of viewers, re-watchers, unique viewers, or play or pause button clicks.

2D timeline

If the learner wants to jump to a specific point in the video, he can click on any point in the 2D timeline, which will capture the x coordinate of the click and update the playhead. The embedded peak visualization shows the intensity and range of each peak, and the overall distribution of the peaks within a video. Since interaction peaks are highlighted in magenta and span a wider region than other points, the learner can visually review and navigate to the commonly revisited parts in the video. We use the Twitinfo [111] peak detection algorithm to detect peaks in the server log data.

Non-linear scrubbing with the phantom cursor

This timeline also enables dynamic, non-linear scrubbing, which takes advantage of interaction peaks. The basic idea is to apply friction while scrubbing around peaks, which leads to prolonged exposure so that learners can get a more comprehensive view of the frames near the peaks even when scrubbing quickly. Friction also makes it easier to precisely se-

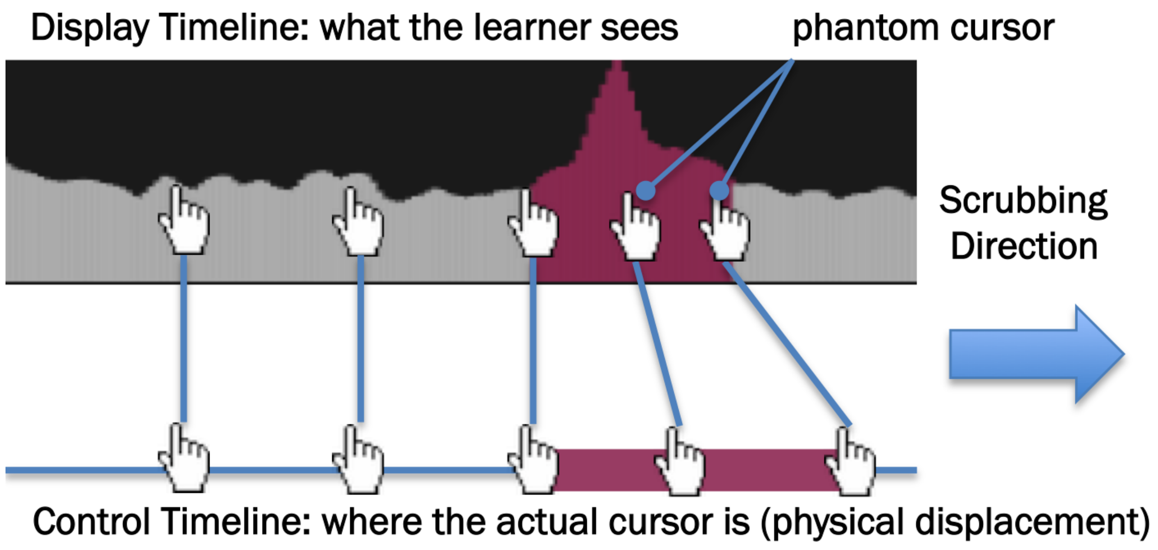

Display Timeline: what the learner sees
phantom cursor
Scrubbing Direction
Control Timeline: where the actual cursor is (physical displacement)

Figure 4-3: Non-linear scrubbing in the Rollercoaster timeline. To draw the learner's attention to content around interaction peaks, the phantom cursor decelerates scrubbing speed when the cursor enters a peak range.

lect specific frames within the range, since it lowers the frame update rate. It is an example of control-display ratio adaptation [15, 71, 148], dynamically changing the ratio between physical cursor movement and on-screen cursor movement.

Previous techniques have applied elastic, rubber band-like interactions to scrubbing [72, 115, 139, 143]. Our technique differs in that 1) it uses interaction data instead of content-driven keyframes, 2) elasticity is selectively applied to parts of the timeline, and 3) the playhead and the cursor are always synchronized, which reduced user confusion in our pilot studies.

When the mouse cursor enters a peak region while dragging, the dragging speed slows down relative to the dragging force, creating the sense of friction. The faster the dragging, the weaker the friction. We achieve this effect by temporarily hiding the real cursor, and replacing it with a *phantom cursor* that moves slower than the real cursor within peak ranges (Figure 4-3). The idea of enlarging the motor space around targets is inspired by Snap-and-Go [9].

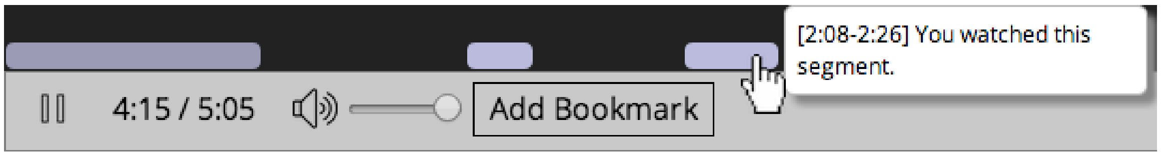


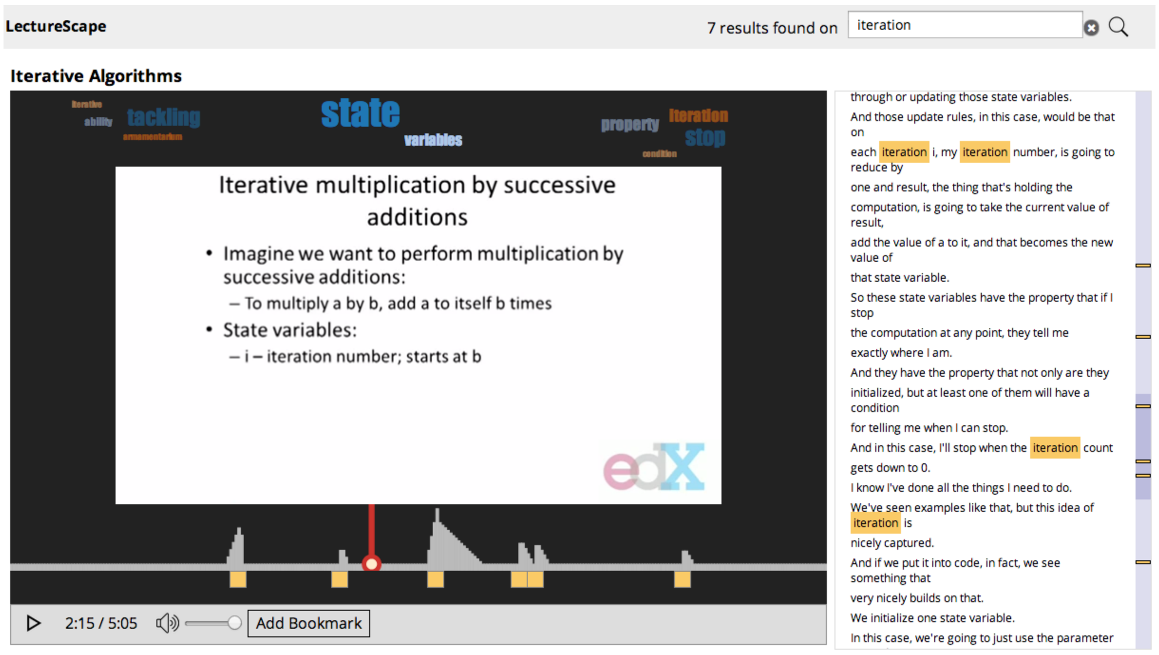
Figure 4-4: The real-time personal watching traces visualize segments of the video that the learner has watched.

Personal watching trace visualization

When we observed pilot study users navigating videos with our timeline, a common desire was to keep track of which parts of the video they personally watched, which might not align with the aggregate interaction peaks collected over all learners. Thus, we added another stream under the timeline to visualize each learner's personal watching traces. Previous research has separated personal and collective history traces to support GUI command selection [117], and added history indicators to a document scrollbar [3], which improved task performance in information finding. We extend these approaches to video navigation by using personal watching traces to support revisitation. Once the learner pauses the video or jumps to a new point, the current watching segment is visualized on a separate track below the timeline (Figure 4-4). Clicking on a generated segment replays the segment. More recent segments are displayed with higher opacity to further emphasize them over older ones. These traces can be stored on a per-user basis to help learners quickly find points of interest when they return to re-watch a video at a later date.

4.3.2 Keyword Search and Visualization

To better support searching for relevant information inside of a video, we use interaction data to power keyword search and transcript analysis. Instead of weighing all occurrences equally, our search technique rewards results in sections of the video where more learners watched. Since key concepts often appear dozens of times in a video, this feature helps the learner prioritize which parts of the video to review. Furthermore, to support novice learners who do not necessarily have the vocabulary to translate their information needs into a direct search query, we suggest major topics discussed in each section of the video in a word cloud. These topics serve as a keyword summary that can help learners recognize and remember the main topics discussed in each video.



Iterative multiplication by successive additions

- Imagine we want to perform multiplication by successive additions:
  - To multiply a by b, add a to itself b times
- State variables:
  - i – iteration number; starts at b

through or updating those state variables.
And those update rules, in this case, would be that on
each iteration i, my iteration number, is going to reduce by
one and result, the thing that's holding the computation, is going to take the current value of result,
add the value of a to it, and that becomes the new value of
that state variable.
So these state variables have the property that if I stop
the computation at any point, they tell me
exactly where I am.
And they have the property that not only are they
initialized, but at least one of them will have a condition
for telling me when I can stop.
And in this case, I'll stop when the iteration count
gets down to 0.
I know I've done all the things I need to do.
We've seen examples like that, but this idea of iteration is
nicely captured.
And if we put it into code, in fact, we see something that
very nicely builds on that.
We initialize one state variable.
In this case, we're going to just use the parameter

Figure 4-5: Our interaction data-driven keyword search brings in more context for the learner to decide where to focus on when searching for a keyword. For instance, the learner can visually check the distribution of when the lecturer said the keyword in the current video, which is useful in seeing where it was heavily discussed versus simply introduced.

Keyword search

If the learner has a keyword to search for, she can type it in the search field (top right in Figure 4-5), which searches over the full transcript of a video clip, and displays results both on the timeline and in the transcript. When the learner enters a search query, the timeline dynamically displays the search results instead of the default interaction visualization (see Figure 4-6). In this way, the timeline serves as a dynamic space for supporting different learner tasks by changing the peak points it emphasizes. Each result renders as a pyramid-shaped distribution, whose range is the duration of the sentence the word belongs to and whose peak is where the term is spoken. Figure 4-7 shows how hovering over the result displays a tooltip, and clicking on the result plays the video from the beginning of the sentence that includes the search term. This sentence-level playback provides the learner with more context surrounding the term.

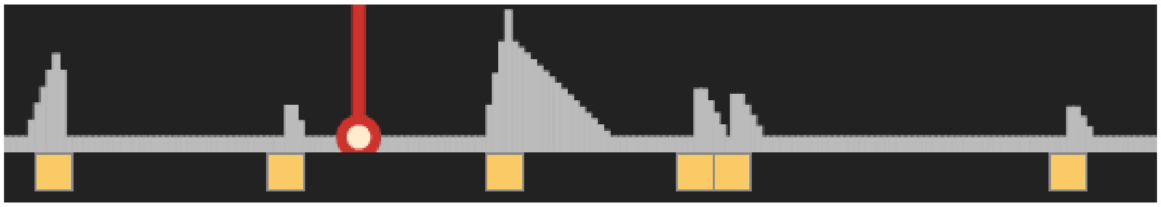
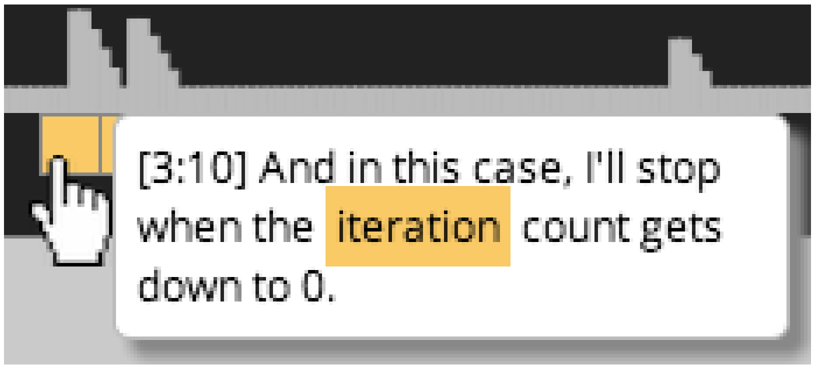


Figure 4-6: The search timeline appears below the video after each search and replaces the Rollercoaster timeline when a search term is entered. It visualizes the positions of search results, as well as the relative importance of each. Here the high peak in the middle indicates both that it contains the search term and that lots of learners watched that part.



[3:10] And in this case, I'll stop when the iteration count gets down to 0.

Figure 4-7: Hovering on a search result displays a tooltip with the transcript sentence that contains the search term. Clicking on the result plays the video starting at the beginning of the sentence to assist the learner with comprehending the surrounding context.

Because key terms are repeated many times even during a short video, it can be hard for the learner to locate the most important search result. For example, in a 5-minute edX video on iterative algorithms, “variable” is mentioned 13 times, and “state” 12 times. We use interaction data to rank search results, with the assumption that parts of the video more frequently watched by previous learners are likely to reflect the current learner’s interest. Our ranking algorithm analyzes learner activity around a search result and assigns a weight to the result, giving higher weights to sentences that were viewed by more learners. Each match within a sentence is then highlighted in the tooltip and the transcript. To support quick visual inspection of search results, we represent the computed score as the height on the timeline (gray peaks in Figure 4-6). If the term was mentioned multiple times in a sentence, we convolve the distributions for all occurrences and assign the maximum score to it.

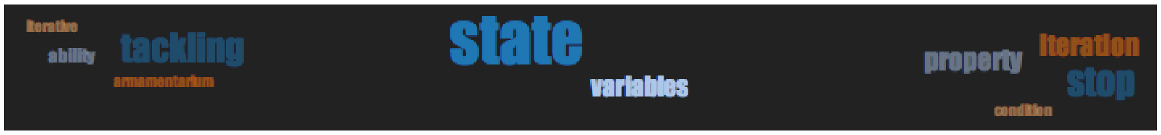


Figure 4-8: The word cloud displays automatically-extracted topics for the currently visible section of the video, providing a keyword-based summarization of the video. Clicking on any word triggers a search for it.

Word cloud: topic summarization and visualization

To address the low visual variation between frames in many videos and to help learners recognize and remember major topics in the clip, we use word clouds to dynamically display keywords in different segments of the video. We use TF-IDF (term frequency-inverse document frequency) scores for extracting keywords and weighing their importance. To compute the TF-IDF scores for the keywords in a transcript, we define a document as the transcription sentences between two consecutive interaction peaks, and the background corpus as the collection of all video transcripts in the entire course. This user activity-driven mechanism extracts self-contained segments from each video.

The visualization is positioned directly above the video as a panel consisting of three word clouds (see Figure 4-8), and gets updated at every interaction peak. The center cloud corresponds to the present segment being viewed, and two additional clouds represent the previous and upcoming segments, respectively. These displays are intended to give a sense of how lecture content in the current segment compares to that of surrounding segments. To bring focus to the current word cloud, we de-emphasize the previous and next word clouds by decreasing their opacity and word size relative to the current cloud. The text color is randomly assigned from a categorical color scheme provided by d3 [1], which is based on an ordinal scale. Clicking on any keyword in the cloud triggers a search using that term, visualizing the occurrences in the transcript as well as on the timeline.

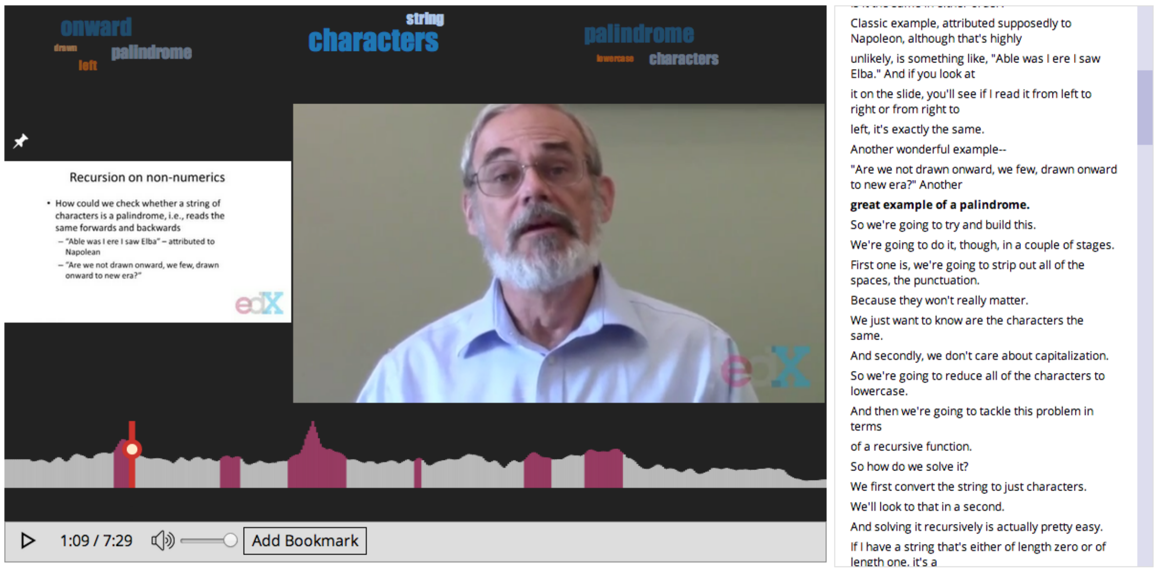
[1] https://github.com/mbostock/d3/wiki/Ordinal-Scales

4.3.3 Video Summarization with Highlights

To enable a quick overview of important points, we present a strip of visual highlights of selected video frames. Consistent with our design goal of providing access to both personal and collective interaction traces in the timeline, we support both collective and personal highlights. Collective highlights are captured by interaction peaks, while personal highlights are captured by the learner bookmarking a frame of interest.

Interaction peak highlights

We capture interaction peaks and provide one-click access to them to support common watching patterns such as jumping directly to these points. The storyboard-style display of the peak frames allows the learner to visually scan the video's progress (Figure 4-1). These highlights are visually-oriented, while the word cloud of Figure 4-8 is text-oriented.



Recursion on non-numerics

- How could we check whether a string of characters is a palindrome, i.e., reads the same forwards and backwards
  - "Able was I ere I saw Elba" – attributed to Napolean
  - "Are we not drawn onward, we few, drawn onward to new era?"

Classic example, attributed supposedly to Napoleon, although that's highly

unlikely, is something like, "Able was I ere I saw Elba." And if you look at

it on the slide, you'll see if I read it from left to right or from right to

left, it's exactly the same.

Another wonderful example--

"Are we not drawn onward, we few, drawn onward to new era?" Another

great example of a palindrome.

So we're going to try and build this.

We're going to do it, though, in a couple of stages.

First one is, we're going to strip out all of the spaces, the punctuation.

Because they won't really matter.

We just want to know are the characters the same.

And secondly, we don't care about capitalization.

So we're going to reduce all of the characters to lowercase.

And then we're going to tackle this problem in terms

of a recursive function.

So how do we solve it?

We first convert the string to just characters.

We'll look to that in a second.

And solving it recursively is actually pretty easy.

If I have a string that's either of length zero or of length one, it's a

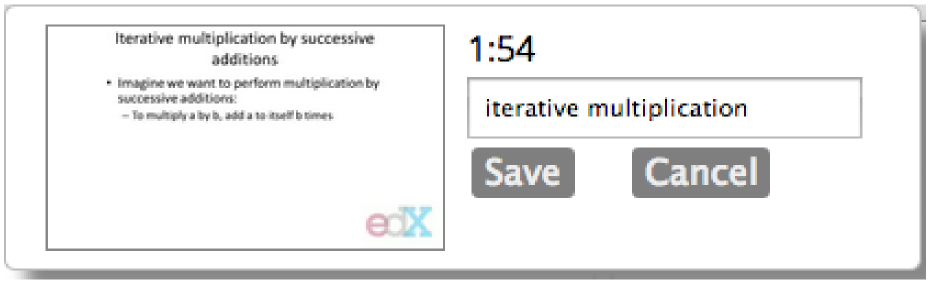
Figure 4-9: Our pinning algorithm analyzes interaction peaks and visual transitions in the video to display a smaller static frame (on the left) next to the video (on the right). Learners can manually pin any frame as well.

Pinning video frames

Most existing video players display only one frame at a time. This ubiquitous interface is sensible for the narrative structure of general-purpose videos such as TV shows, where a sequential flow is natural. However, educational videos are information-heavy, and active learning involves skimming and scrubbing [86]. For example, an instructor might verbally refer to the formula she described in the previous PowerPoint slide, but the formula might no longer be available on screen. A learner who wants to refer to that formula has to scrub the video timeline to go back to a frame with the relevant formula slide. To support watching patterns that are not easily supported by existing players, our video player pins a relevant frame next to the video stream for easy reference (Figure 4-9).

Our video player automatically determines a frame to pin. A relevant pinned frame should 1) not be identical to what is currently shown in the video, 2) include important content that is worth referencing, and 3) contain readable content such a textual slide or technical diagram, not merely a static frame of the instructor's face or students sitting in a classroom. Otherwise, juxtaposing a static frame next to the video might cause distraction and visual clutter. To meet these requirements, our pinning algorithm uses both interaction peaks and visual transitions. It automatically pins an interaction peak frame if there is a visual transition shortly after the peak. Checking for a visual transition ensures that the pinned frame is not visually identical to the frames right after the transition. To meet the requirement 3), we simply check whether the slide is visually static, with the assumption that the instructor's face or classroom capture signals jiggle in terms of pixel differences between frames. A more robust implementation can use a trained video parser such as EdVidParse [140]. jAlso, pinning an interaction peak frame ensures that the frame at least includes content viewed by many others.

The learner can also manually pin a frame by clicking the pin icon attached to each peak frame, which replaces the current pinned frame with the learner's. While the system attempts its best effort to show the most relevant frame at a given time, the learner also has the flexibility to control what gets displayed.


Iterative multiplication by successive additions

- Imagine we want to perform multiplication by successive additions:
  - To multiply a by b, add a to itself b times

1:54

iterative multiplication

Figure 4-10: The learner can add a labeled bookmark, which is added to the highlights stream below the video for visual preview and revisitation.

Personal bookmarks

While peak frames might be a reasonable summary of a video, individual learners might have summarization needs that are not captured by the collective traces. The learner can add personal bookmarks by clicking on the "Add Bookmark" button. The learner can see the captured frame at the point of click and add their own label for future reference (Figure 4-10). Once a bookmark is saved, it is added to the highlights stream, chronologically ordered along with other bookmarks and interaction peak highlights. This view allows the learner to choose between naturally-formed interaction peaks by other learners as well as self-generated bookmarks.

4.4 LectureScape: Web-Based Prototype

This section introduces LectureScape, a prototype lecture video player that combines all of the techniques in a unified interface (see Figure 4-1). The main goal of LectureScape is to give learners more control and flexibility in deciding how to navigate educational videos. LectureScape features the video player in the main view, along with the Rollercoaster timeline below the video and the word cloud above it. The interactive transcript is in the right sidebar, and the highlights are positioned at the bottom of the screen. The search box at the top enables keyword search. The widgets are all collapsible to reduce visual clutter and to hide unused features.

We implemented LectureScape using standard Web technologies: HTML5, CSS3, and JavaScript. The word cloud and Rollercoaster timeline are rendered with D3 [17]. Our customized HTML5 video player does not require additional re-encoding or streaming, and is independent of the encoding or streaming method used. It only requires interaction data for each video to activate the data-driven interaction techniques. Our data processing pipeline formats interaction traces and generates visual assets from an existing video.

First, three data streams are stored for each video: interaction data, TF-IDF results on the transcript, and visual transition data. Our peak detection algorithm runs on page load, which allows dynamic parameter tuning. Peak detection is run on both the interaction data and visual transition data, which returns interaction peaks and shot boundaries, respectively.

Second, the system generates thumbnail images for each second of a video. Because many of our interactions require displaying a thumbnail of a video frame on demand, low latency in image loading is crucial in supporting seamless interactions. It is especially important for educational videos whose visual changes are more subtle than in movies or TV shows, and whose on-screen information matters for learners to read and comprehend. Upon a page load, the system preloads all thumbnails for a video. When a learner drags the timeline, instead of loading the video each time a dragging event is triggered, our player pauses the video and displays an overlay screenshot. This results in much less latency and smoother dragging, similar to the benefits reported by Swift [116].

Consistent with previous research [86] and guidelines for online streaming video platforms [55], many viewers drop out early on in the video due to misfires and unwanted autoplays. This results in high volumes in interaction data at the beginning of the video, which might not be useful yet outscale the rest in quantity. To mitigate these effects, the Rollercoaster timeline excludes the interaction data within the first 3% of the video duration in setting the maximum y-axis value for the visualization.

4.5 Evaluation

To assess the feasibility of using interaction data to enhance video navigation, we conducted a user study comparing video players with and without our data-driven interaction techniques. We explored three research questions:

- **RQ1.** How do learners navigate lecture videos with LectureScape in typical kinds of learning tasks such as search and summarization?
- **RQ2.** How do learners interpret interaction data presented in LectureScape?
- **RQ3.** Are LectureScape's features useful and learnable?

4.5.1 Study Design

The study was a within-subjects design, where each learner used both LectureScape and a baseline interface that stripped off all interaction data-related features from LectureScape. The baseline interface still included the interactive transcript and preview thumbnails on hover, to emulate what is available in platforms such as edX or YouTube. To maintain uniformity in look and feel for our comparative study, the baseline interface had the same layout and visual design as LectureScape.

Learners performed three types of learning tasks for lecture videos: visual search, problem search, and summarization. These tasks represent realistic video watching scenarios from our observations, and match common evaluation tasks used in the literature on video navigation interfaces [38, 128].

- **Visual search** tasks involved finding a specific piece of visual content in a video. These tasks emulated situations when a learner remembers something visually and wants to find where it appeared in a video. For example, for a video about tuples in Python, a visual search task asked: *"Find a slide where the instructor displays on screen examples of the singleton operation."* Targets were slides that appeared briefly (less than 20 seconds) in the video. We mixed tasks that had targets around interaction peaks and non-peaks to investigate how LectureScape fares even when the target is not near a peak. For all visual search tasks, we provided learners with only the video timeline (linear timeline in baseline, 2D timeline in LectureScape) and removed all other features (e.g., transcript, word cloud) to restrict video navigation to timelines only. Learners were told to pause the video as soon as they navigated to the answer.
- **Problem search** tasks involved finding an answer to a given problem. These tasks

emulated a learner rewatching a relevant video to answer a discussion forum question or to solve a homework problem. For example, for a video about approximation methods, a problem search task asked: *"If the step size in an approximation method decreases, does the code run faster or slower?"* Learners were asked to find the part in the video that discussed the answer, and then state their answer.

- **Summarization** tasks required learners to write down the main points of a video while skimming through it. We gave learners only three minutes to summarize videos that were seven to eight minutes long, with the intent of motivating learners to be selective about what parts to watch.

All videos used in the study were from 6.00x, an introductory computer science course on edX. Interaction data was collected from server logs during the first offering of the course in fall 2012, which included 59,126 learners and 4,491,648 video clickstream events. The course has been recurring every semester since then. Each of the eight tasks in the study used different videos to minimize learning effects. We also chose videos of similar length, difficulty, and style within each task type to control for differences across videos.

4.5.2 Participants

We recruited 12 participants (5 male, mean age 25.6, stdev=11.0, max=49, min=18) who were currently enrolled in the introductory CS course (either on edX or on campus) to which the videos belong. The on-campus version of the course shares the same curriculum but is taught by different instructors. For recruitment, we posted a flyer on the course discussion forum on edX and the on-campus course website, both with consent from instructors. Furthermore, we picked videos for lessons given earlier in the semester, so that participants were likely to have already been exposed to that material before coming to our study, as is often the case in video re-watching scenarios. Four participants were enrolled in a current or previous edX offering of the course, while six were taking the on-campus version. Two had previously registered in the online offering but were currently taking the on-campus course. Participants received $30 for their time.

4.5.3 Procedure

A 75-minute study session started with 15-minute tutorials on both interfaces. Next, participants performed eight learning tasks: four visual search tasks, two problem search tasks, and two summarization tasks. Each task used a different video, so the total of eight videos were used in the study and by each participant. After each task, they answered questions about confidence in their answer and prior exposure to the video. After each task type, we interviewed them about their task strategy. For each task type, we counterbalanced the order of the interfaces and the assignment of videos to tasks. After completing all the tasks, participants completed a questionnaire on the usability of each interface and their experience and opinions about interaction data. All click-level interactions were logged by the server for analysis. The tasks, links to the videos used, and the survey are presented in Appendix A.

4.6 Results

4.6.1 RQ1. Navigation Patterns for Search and Summarization

In **visual search**, most participants in the baseline 1D timeline sequentially scanned the video using thumbnail previews or dragging. In contrast, participants using the 2D Rollercoaster timeline often jumped between interaction peaks to reach the general area of the answer. But the latter strategy did not help in many cases because interaction data represents collective interests in a video, not results for a search query.

For the two out of four tasks where the search targets were located near interaction peaks, it took participants in both conditions similar amounts of time (LectureScape: μ=85 seconds, σ=51, baseline: μ=80, σ=73). This difference was not statistically significant with the Mann-Whitney U (MWU) test (p>0.4, Z=-0.9). For the other two tasks where the search targets were outside of an interaction peak range, it took participants in the LectureScape condition longer to complete (μ=117, σ=44) than in the baseline condition (μ=90, σ=50), although the MWU test showed no statistical significance (p>0.1, Z=-1.5) The longer time might be due to the fact that many participants navigated by clicking on the peaks to see if the answer existed around peaks. Nonetheless, results show that LectureScape did not adversely affect task completion times even when the answer was not in peak ranges.

Because **problem search** tasks required some understanding of the context to provide an answer, participants often tackled the problem by narrowing down to a section of the video that mentioned relevant concepts and then watching the section until they found the answer. In the process of narrowing down to a video section, most participants in the baseline condition relied on searching for keywords (10 out of 12) and clicking on transcript text (11 / 12), while participants with LectureScape used search (6 / 12) and clicking on transcript text (6 / 12) less frequently, and additionally clicked on interaction peaks on the timeline (6 / 12) or highlights below the video (6 / 12). Over all problem search tasks, participants in the LectureScape condition completed them slightly faster (μ=96, σ=58) than participants in the baseline condition (μ=106, σ=58), although the difference was not significant (MWU test, p=0.8, Z=0.3).

In comparison to the other tasks, participants in **summarization** tasks did not rely on keyword search (1 / 12 in both conditions), because the task required them to quickly scan the entire video for the main points. Many participants with LectureScape scanned the peak visualization, word cloud, and highlights for an overview, and clicked on interaction peaks (9 / 12) or highlights (6 / 12) for a detailed review. In one video, all six participants with LectureScape visited an interaction peak almost at the end of the video (located at 6:35 in the 7:00 clip). This slide summarized main ideas of variable binding, which was the topic of the video. In contrast, in the baseline condition, only one learner navigated to this section of the video. Most participants spent majority of their time in the earlier part of the video in the baseline condition.

Despite inconclusive evidence on quantitative differences in task completion time, participants believed that they were able to complete the tasks faster and more efficiently with LectureScape than with the baseline interface. Answers to 7-point Likert scale questions on the overall task experience revealed significant differences between the two interfaces in participants' belief in speed (LectureScape: μ=5.8, Baseline: μ=4.8) and efficiency (LectureScape: μ=6.1, Baseline: μ=4.8). The MWU test shows that both differences were significant at $p<0.05$ for these questions.

4.6.2 RQ2. Perception of Interaction Data

Generally, participants' comments about watching videos augmented by others' interaction data were positive. Participants noted that *"It's not like cold-watching. It feels like watching with other students."*, and *"[interaction data] makes it seem more classroom-y, as in you can compare yourself to what how other students are learning and what they need to repeat."*

In response to 7-point Likert scale questions about the experience of seeing interaction data, participants indicated that they found such data to be "easy to understand" (μ=5.9), "useful" (5.3), "enjoyable" (5.2), that interaction peaks affected their navigation (5), and that interaction peaks matched their personal points of interest in the video (4.4).

In an open-ended question, we asked participants why they thought interaction peaks occurred. Common reasons provided were that these parts were "confusing" (mentioned by 8 / 12), "important" (6 / 12), and "complex" (4 / 12). Identifying the cause of a peak might be useful because they can enable more customized navigation support. While most participants mentioned that highlighting confusing and important parts would be useful, some noted that personal context may not match the collective patterns. One said, *"If it were a topic where I was not very confident in my own knowledge, I would find it very helpful to emphasize where others have re-watched the video. If however it was a topic I was comfortable with and was watching just to review, I would find it frustrating to have the physical scrolling be slowed down due to others' behavior while watching the video."*

4.6.3 RQ3. Perceived Usability of LectureScape

Many participants preferred having more options in navigating lecture videos. As one learner noted, *"I like all the extra features! I was sad when they got taken away [for the baseline condition]."* Also, when asked if the interface had all the functions and capabilities they expected, participants rated LectureScape (6.4) significantly higher than the baseline interface (4.3) ($p<0.001$, $Z=-3.2$ with the MWU test).

However, some expressed that LectureScape was visually complex, and that they would have liked to hide some widgets not in use at the moment. They found it more difficult to use than the baseline (ease of use: 4.7 vs. 6.3, $p<0.001$, $Z=2.7$ with the MWU test). This perception leaves room for improvement in the learnability of the system. A participant commented: *"[LectureScape] is fairly complex and has a lot of different elements so I think it may take a bit of time for users to fully adjust to using the interface to its full potential."* These comments are consistent with our design decision to support collapsible widgets to reduce visual clutter, although the version of LectureScape used in the study had all features activated for the purpose of usability testing.

Due to the limitations of a single-session lab study, few participants actively used personal bookmarks or personal history traces. A longitudinal deployment might be required to evaluate the usefulness of these features.

4.6.4 Navigation Pattern Analysis

Now we provide a detailed analysis of how participants navigated videos during the study. In all tasks, most participants' strategy was to start with an overview, and then focus on some parts in detail. Participants alternated between the overview and focus stages until they found what they were looking for, or covered all major points in the video for summarization.

While the high-level strategy was similar in the two video interfaces, participants' navigation patterns within each of the two stages differed noticeably. With LectureScape, participants used more diverse options for overview and focused navigation, making more directed jumps to important points in the video. With the baseline interface, most participants sequentially scanned the video for overview and clicked on the timeline or transcript for focusing. Another common pattern in the baseline condition was to make short and conservative jumps on the timeline from the beginning of the video, in order not to miss anything important while moving quickly.

In the **overview** stage, most participants tried to scan the video to grasp the general flow and select a few points to review further. One learner described her strategy with

LectureScape in this stage: *"having this idea of 'here's where other people have gone back and rewatched, being able to visually skim through very quickly and see titles, main bullet points, and following along with the transcript a little bit as well was definitely helpful."* Although visual scanning did not result in click log entries, our interviews with participants confirm that it was a common pattern. They pointed out three main features in LectureScape that supported overview:

- the 2D timeline with an overall learner activity visualization: *"I could use the 2D overlay to scroll through... I think I took a quick scan through and saw the general overview of what was on the slides."*
- highlight summaries with a strip of screenshots: *"They would get me close to the information I needed. They also made it easier to quickly summarize."*
- the word cloud with main keywords for sections in the video: *"I just looked at the top keywords, then I watched the video to see how [the instructor] uses those keywords."*

After scanning for an overview, participants chose a point in the video to watch further. All of the methods described above provide a single-click mechanism to directly jump to an "important" part of the video. Participants reviewed data-driven suggestions from LectureScape to make informed decisions. The log analysis reveals that participants using LectureScape made direct jumps such as clicking on a specific point in the timeline or a highlight 8.4 times on average per task, in contrast to 5.6 times in the baseline condition.

In the **focus** stage, participants watched a segment in the video and reviewed if content is relevant to the task at hand. In this stage they relied on methods for precise navigation: scrubbing the timeline a few pixels for a second-by-second review, and re-watching video snippets multiple times until they fully comprehend the content. With LectureScape, participants had options to use the slowed-down scrubbing around peaks in the rollercoaster timeline, automatic pinning of the previous slide, and sentence-level playback in search. To navigate back to the previously examined point, participants frequently used timestamped anchors attached to search results, interaction peaks, and highlights.

In summary, with LectureScape, participants used more navigation options in both the overview and focus stages. A learner commented that *"[LectureScape] gives you more* *options. It personalizes the strategy I can use in the task."* They had more control in which part of the video to watch, which might have led them to believe that they completed the tasks faster and more efficiently.

4.7 Discussion

Availability of interaction data: Discussion throughout this chapter assumes the availability of large-scale interaction data. With modern Web technologies, clickstream logging can be easily added with APIs for video event handling.

There remain unanswered questions around interaction data, such as "Will using the data-driven techniques bias the data so that it reinforces premature peak signals and ignores other potentially important ones?", and "How many data points are required until salient peaks and patterns emerge?" Our future work will address these questions through a live deployment on a MOOC platform such as edX. We will also explore other types of interaction data such as active bookmarking and content streams such as voice to enrich video-based learning.

Adaptive video UI: The data-driven techniques introduced in this chapter open opportunities for more adaptive and personalized video learning experiences. In this chapter, we demonstrated how collective viewership data can change the video interface dynamically, influencing the physical scrubbing behavior, search ranking algorithm, and side-by-side frame display. We envision future video UIs that adapt to collective usage. Also, incorporating interaction data can lead to personalized video learning. Because interaction data is likely to represent an average learner, comparing personal history traces against collective traces may help model the current user more accurately and improve personalization.

Beyond MOOC-style lecture videos: While this chapter used MOOC-style lecture videos for demonstration, we believe our techniques can generalize to other types of educational videos such as programming tutorial screencasts, how-to demonstration videos, and health education videos. We expect to apply the techniques introduced in this chapter to these videos.

4.8 Conclusion

This chapter introduces a novel concept of designing video interaction techniques by leveraging large-scale interaction data. We present three sets of data-driven techniques to demonstrate the capability of the concept: 2D, non-linear timeline, enhanced in-video search, and a visual summarization method. In a lab study, participants found interaction data to draw attention to points of importance and confusion, and navigated lecture videos with more control and flexibility.

Ultimately, the design techniques we have presented provide enriched alternatives to conventional video navigation. We envision engaging a community of learners in creating a social, interactive, and collaborative video learning environment powered by rich community data.

Chapter 5

ToolScape: Crowdsourcing Step Labels for How-to Videos

This chapter presents the first example of active learnersourcing, ToolScape, which extracts step-by-step instructions from an existing how-to video. This chapter has adapted, updated, and rewritten content from a paper at CHI 2014 [89]. which combines and extends the ideas proposed in two Extended Abstracts at CHI 2013 [83, 127]. All uses of “we”, “our”, and “us” in this chapter refer to coauthors of the aforementioned papers.

Millions of learners today use how-to videos to master new skills in a variety of domains. But browsing such videos is often tedious and inefficient because video player interfaces are not optimized for the unique step-by-step structure of such videos. This research aims to improve the learning experience of existing how-to videos with *step-by-step annotations*.

We first performed a formative study to verify that annotations are actually useful to learners. We created ToolScape, an interactive video player that displays step descriptions and intermediate result thumbnails in the video timeline. Learners in our study performed better and gained more self-efficacy using ToolScape versus a traditional video player.

To add the needed step annotations to existing how-to videos at scale, we introduce a novel crowdsourcing workflow. It extracts step-by-step structure from an existing video, including step times, descriptions, and before and after images. We introduce the Find-Verify-Expand design pattern for temporal and visual annotation, which applies clustering, text processing, and visual analysis algorithms to merge crowd output. The workflow does not rely on domain-specific customization, works on top of existing videos, and recruits untrained crowd workers. We evaluated the workflow with Mechanical Turk, using 75 cooking, makeup, and Photoshop videos on YouTube. Results show that our workflow can extract steps with a quality comparable to that of trained annotators across all three domains with 77% precision and 81% recall.

5.1 Motivation and Contributions

How-to videos on the web have enabled millions of learners to acquire new skills in procedural tasks such as folding origami, cooking, applying makeup, and using computer software. These videos have a unique step-by-step structure, which encourages learners to sequentially process and perform steps in the procedure [157]. While most text- and image-based tutorials (e.g., webpages) are naturally segmented into distinct steps, how-to video tutorials often contain a single continuous stream of demonstration. Because comprehensive and accurate step-by-step information about the procedure is often missing, accessing specific parts within a video becomes frustrating for learners. Prior research shows that higher interactivity with the instructional content aids learning [46, 156], and that the completeness and detail of step-by-step instructions are integral to task performance [44].

To better understand the role of step-by-step information in how-to videos, we ran a formative study where learners performed graphical design tasks with how-to videos. For this study, we designed ToolScape, an interactive how-to video player that adds step descriptions and intermediate result thumbnails to the video timeline. Learners using ToolScape showed a higher gain in self-efficacy and rated the quality of their own work higher, as compared to those using an ordinary video player. Moreover, external judges gave higher ratings to the designs produced by learners using ToolScape.

Providing such navigation support for how-to videos requires extracting step-by-step information from them. One solution is to ask instructors to include this information at tutorial generation time, but this adds overhead for instructors and does not solve the problem for existing videos. Another approach uses automatic methods such as computer vision.

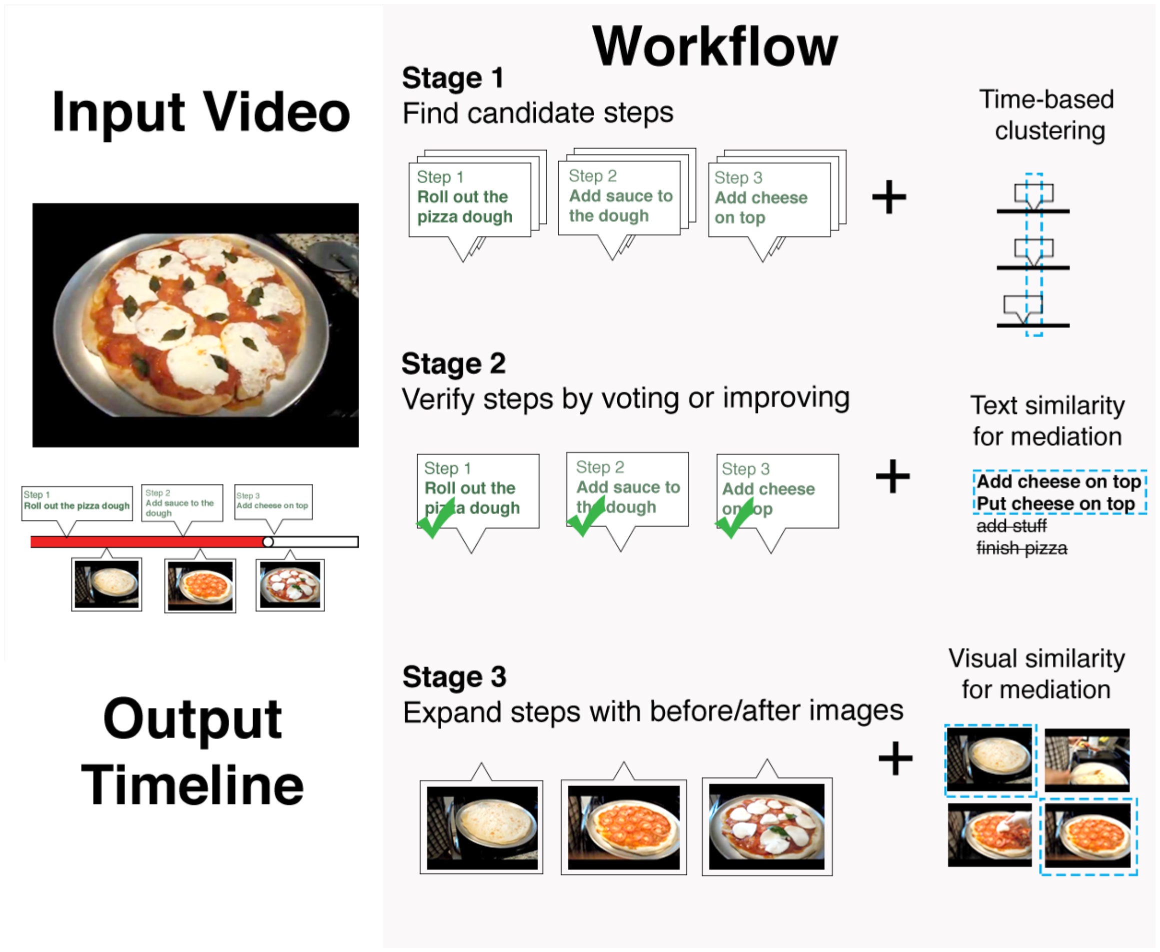

Input Video
Step 1 Roll out the pizza dough
Step 2 Add sauce to the dough
Step 3 Add cheese on top
Output Timeline
Workflow
Stage 1
Find candidate steps
Step 1 Roll out the pizza dough
Step 2 Add sauce to the dough
Step 3 Add cheese on top
Time-based clustering
Stage 2
Verify steps by voting or improving
Text similarity for mediation
Add cheese on top
Put cheese on top
~~add stuff~~
~~finish pizza~~
Stage 3
Expand steps with before/after images
Visual similarity for mediation

Figure 5-1: Our crowdsourcing workflow extracts step-by-step information from a how-to video with their descriptions and before/after images. It features the Find-Verify-Expand design pattern, time-based clustering, and text/visual analysis techniques. Extracted step information can be used to help learners navigate how-to videos with higher interactivity.

Previous research [7, 138] has shown success in limited domains with extensive domain-specific customization. When working with "videos in the wild", however, vision-based algorithms often suffer from low-resolution frames and a lack of training data. A scalable solution applicable beyond limited task domains and presentation formats is not yet available.

To address the issues of high cost or limited scalability with existing methods, we introduce a crowdsourcing workflow for annotating how-to videos, which includes the Find-Verify-Expand design pattern shown in Figure 6-1. It collects step-by-step information from a how-to video in three stages: (1) find candidate steps with timestamps and text descriptions, (2) verify time and description for all steps, and (3) expand a verified step with before and after images. The workflow does not rely on domain-specific knowledge, works on top of existing videos, and recruits untrained, non-expert crowd workers. For quality control, the workflow uses time-based clustering, text processing, and visual analysis to merge results and deal with noisy and diverse output from crowd workers.

To validate the workflow with existing how-to videos, we asked crowd workers on Mechanical Turk to annotate 75 YouTube how-to videos spanning three domains: cooking, makeup, and graphics editing software. Results show that the crowd workflow can extract steps with 77% precision and 81% recall relative to trained annotators. Successfully extracted steps were on average 2.7 seconds away from ground truth steps, and external evaluators found 60% of before and after images to be accurately representing steps.

The contributions of this chapter are as follows:

- A how-to video player interface and experimental results showing that increased interactivity in a video player improves learners' task performance and self-efficacy.
- A domain-independent crowd video annotation method and the Find-Verify-Expand design pattern for extracting step-by-step task information from existing how-to videos.
- A novel combination of time-based clustering, text processing, and visual analysis algorithms for merging crowd output consisting of time points, text labels, and images.
- Experimental results that validate the workflow, which fully extracted steps from 75 readily available videos on the web across three distinct domains with a quality comparable to that of trained annotators.

5.2 Efficacy and Challenges of Video Annotation

To motivate the design of our crowdsourcing workflow for how-to video annotations, we first performed a formative study to (1) verify that annotations are actually useful to learners, and (2) reveal the challenges of manually annotating videos and show the need for a more scalable technique.

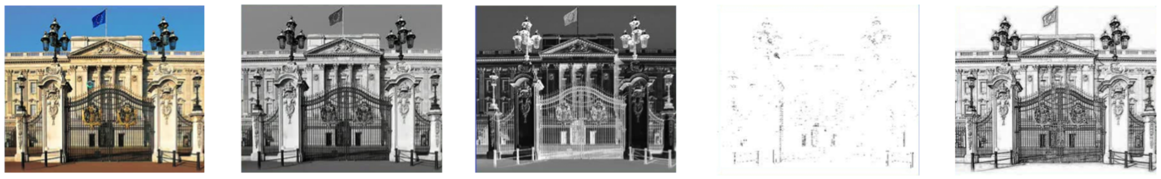


Figure 5-2: Progress in many how-to videos is visually trackable, as shown in screenshots from this Photoshop how-to video. Adding step annotations to videos enables learners to quickly scan through the procedure.

5.2.1 Annotations on How-To Videos

How-to videos often have a well-defined step-by-step structure [58]. A *step* refers to a low-level action in performing a procedural task. Literature on procedural tasks suggests that step-by-step instructions encourage learners to sequentially process and perform steps in the workflow [157] and improve task performance [44]. Annotations can make such structure more explicit. In this chapter, we define *annotation* as the process of adding step-by-step information to a how-to video. In determining which information to annotate, we note two properties of procedural tasks. First, for many domains, task states are visually distinct in nature, so progress can be visually tracked by browsing through a video (Figure 5-2). Examples include food in cooking videos, a model's face in makeup videos, and an image being edited in Photoshop videos. Second, how-to videos contain a sequence of discrete steps that each advance the state of the task (Figure 5-3). Our annotation method uses these two properties to accurately capture a sequence of steps, extracting timestamps, textual descriptions, and before and after images for each step.

We manually created a corpus of annotations for 75 how-to videos in three procedural task domains: cooking, applying makeup, and using Photoshop. We used this corpus to create our interface in the formative study, and ground truth data for evaluating our crowdsourcing workflow. We collected videos from YouTube's top search results for "[domain] [task name]" (e.g., "cooking samosa", "Photoshop motion blur").

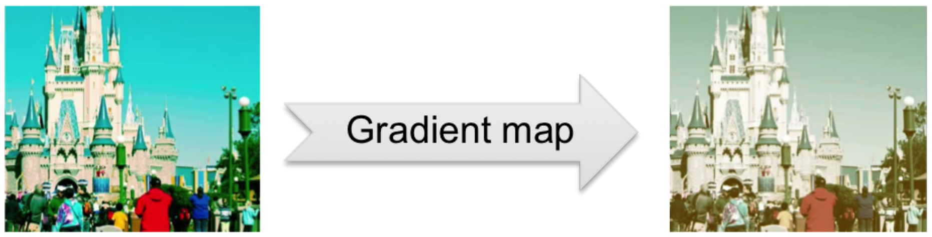


Figure 5-3: How-to videos often contain a series of task steps with visually distinct before and after states. Here the author applied the "Gradient map" tool in Photoshop to desaturate the image colors.

5.2.2 ToolScape: Step-Aware Video Player

To display step annotations, we created a prototype video player named ToolScape. ToolScape augments an ordinary web-based video player with a rich timeline containing links to each annotated step and its respective before and after thumbnail images (Figure 5-4). ToolScape is a Javascript library that manages a timestamped list of steps and before/after images, which can connect to any embedded video player with a "play from this time point" Javascript API call.

In the timeline, the top and bottom streams represent annotated steps and thumbnail images from the video, respectively (Figure 5-4(a), (c)). Clicking on a step or image moves the video player's slider to 5 seconds before the moment it occurred. The 5-second buffer, determined from pilot testing, helps learners catch up with the context preceding the indicated moment. Finally, ToolScape supports annotations of "dead times" at the beginning and end of videos (Figure 5-4(b)), which often contain introductory or concluding remarks. Pilot user observations showed that learners often skip to the main part of the tutorial. In our manually annotated video corpus, on average, 13.7% of time at the beginning and 9.9% at the end were "dead times" with no task progress.

5.2.3 Formative Study Design

To assess the effects of step annotations, we ran a formative study on novice Photoshop learners watching how-to videos on image manipulation tasks. We compared the experiences of learners using ToolScape and a baseline video player without the interactive timeline. We hypothesized that interacting with step annotations provided by ToolScape

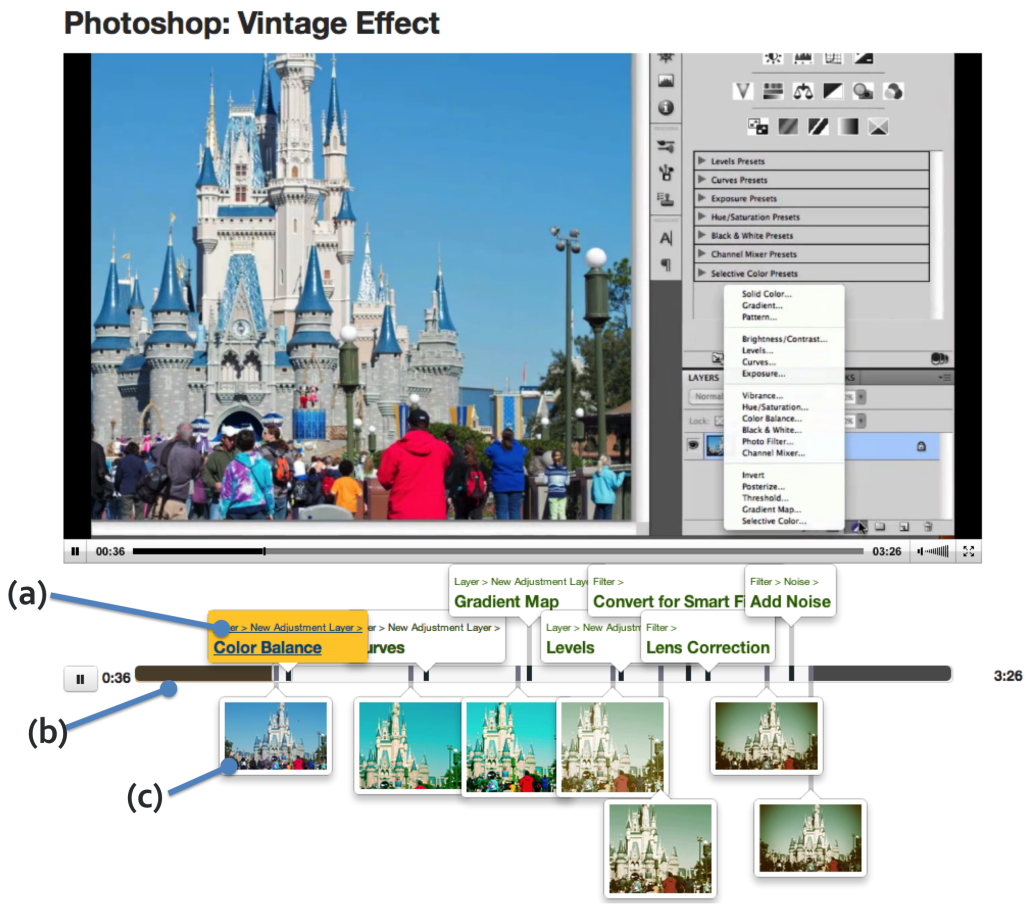


Figure 5-4: ToolScape augments a web-based video player with an interactive timeline. Annotations are shown above the timeline (a), screenshots of intermediate states are shown below the timeline (c), and the gray regions at both ends (b) show "dead times" with no meaningful progress (e.g., waiting for Photoshop to launch).

improves both task performance and learner satisfaction. Specifically:

H1 Learners complete design tasks with a higher self-efficacy gain when watching how-to videos with ToolScape.

H2 Learners' self-rating of the quality of their work is higher when watching with ToolScape.

H3 Learners' designs when watching with ToolScape are rated higher by external judges.

H4 Learners show higher satisfaction with ToolScape.

H5 Learners perceive design tasks to be easier when watching with ToolScape.

In addition to external ratings (H3), our measures of success include self-efficacy (H1) and self-rating (H2). In the context of how-to videos, these measures are more significant than just user preference. Educational psychology research shows that self-efficacy, or confidence in application of skills, is an effective predictor of motivation and learning [6, 172]. Positive self-rating has also been shown to accurately predict learning gains [147]. Finally, we chose not to count errors made in repeating tutorial steps as in [28], because our goal was to help users explore and learn new skills in open-ended design tasks.

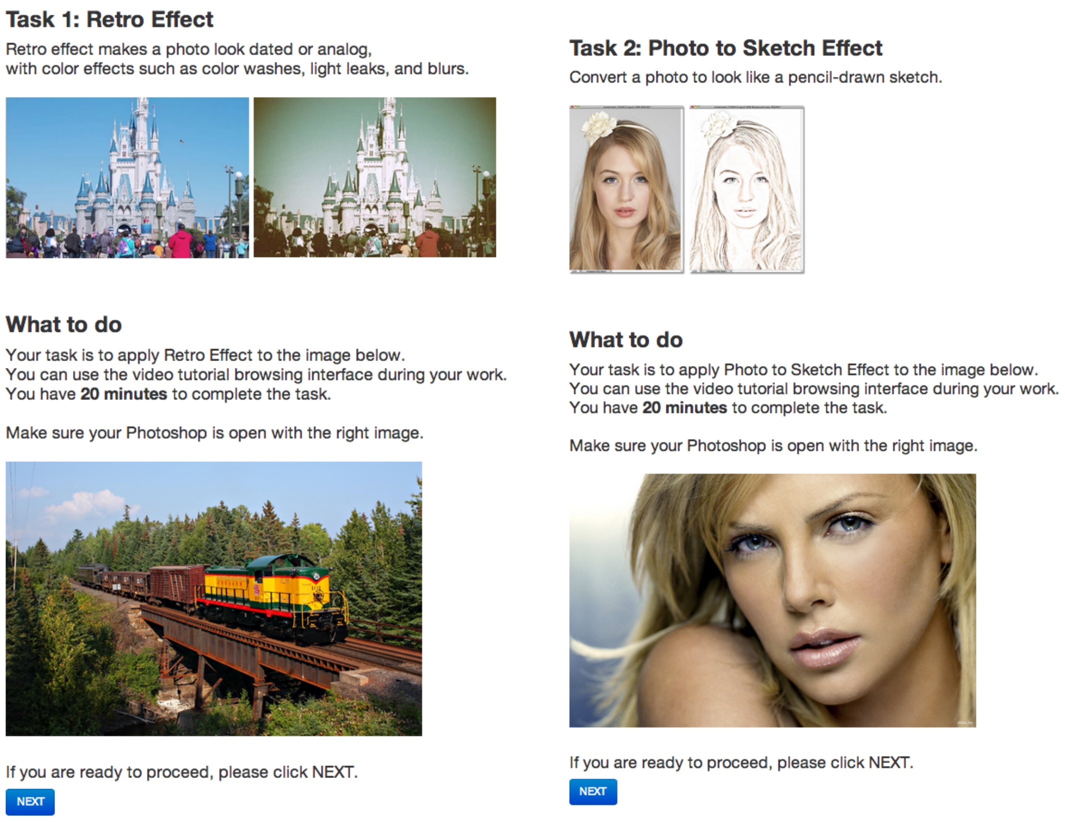
Participants: We recruited twelve participants through university mailing lists and online community postings. Their mean age was 25.2 ($\sigma = 3.2$), with 8 males and 4 females. Most rated themselves as novice Photoshop users, but all had at least some experience with Photoshop. They received $30 for up to two hours of participation, on either a Mac or PC.

Tasks and Procedures: Our study had 2 x 2 conditions: two tasks each using ToolScape and baseline video players. We used a within-subject design with interface, task, and order counterbalanced. Each participant performed two image manipulation tasks in Photoshop: applying retro effect and transforming a photo to look like a sketch. In both interface conditions, we provided participants with the same set of how-to videos; the interface was the only difference. In addition, we disallowed searching for other web tutorials to ensure that any effect found in the study comes from the interaction method, not the content.

After a tutorial task covering all features of the video player interface, we asked participants self-efficacy questions adapted from Dow et al. [41], whose study also measured participants' self-efficacy changes in a design task. The questions asked: On a scale of 1 (not confident at all) to 7 (very confident), how confident are you with...

- solving graphic design problems?
- understanding graphic design problems?
- applying design skills in practice?
- incorporating skills from video tutorials in your design?

Next, participants attempted two 20-minute image manipulation tasks in Photoshop, with instructions shown in Figure 5-5. Participants could freely browse and watch the 10 how-to videos we provided (with annotations in the ToolScape condition). After each task,


Task 1: Retro Effect

Retro effect makes a photo look dated or analog,
with color effects such as color washes, light leaks, and blurs.

What to do

Your task is to apply Retro Effect to the image below.
You can use the video tutorial browsing interface during your work.
You have **20 minutes** to complete the task.

Make sure your Photoshop is open with the right image.

If you are ready to proceed, please click NEXT.

Task 2: Photo to Sketch Effect

Convert a photo to look like a pencil-drawn sketch.

What to do

Your task is to apply Photo to Sketch Effect to the image below.
You can use the video tutorial browsing interface during your work.
You have **20 minutes** to complete the task.

Make sure your Photoshop is open with the right image.

If you are ready to proceed, please click NEXT.

Figure 5-5: These task instructions are shown before the participant starts working on their image manipulation task. It includes a description of the effect to be implemented and a before-and-after example image pair.

we asked questions on task difficulty, self-rating, and interface satisfaction. The questions in the survey are presented in Appendix B.

We also asked the self-efficacy questions again to observe any difference, followed by a 15-minute open-ended interview.

Finally, we asked four external judges to evaluate the quality of all transformed images by ranking them, blind to user and condition. They ranked the images from best to worst, based on how well each participant accomplished the given task.

5.2.4 Formative Study Results

H1 (higher self-efficacy for ToolScape) is supported by our study. For the four self-efficacy questions, we take the mean of the 7-point Likert scale ratings as the self-efficacy score. The participants' mean initial score was 3.8; with the baseline video player, the score after

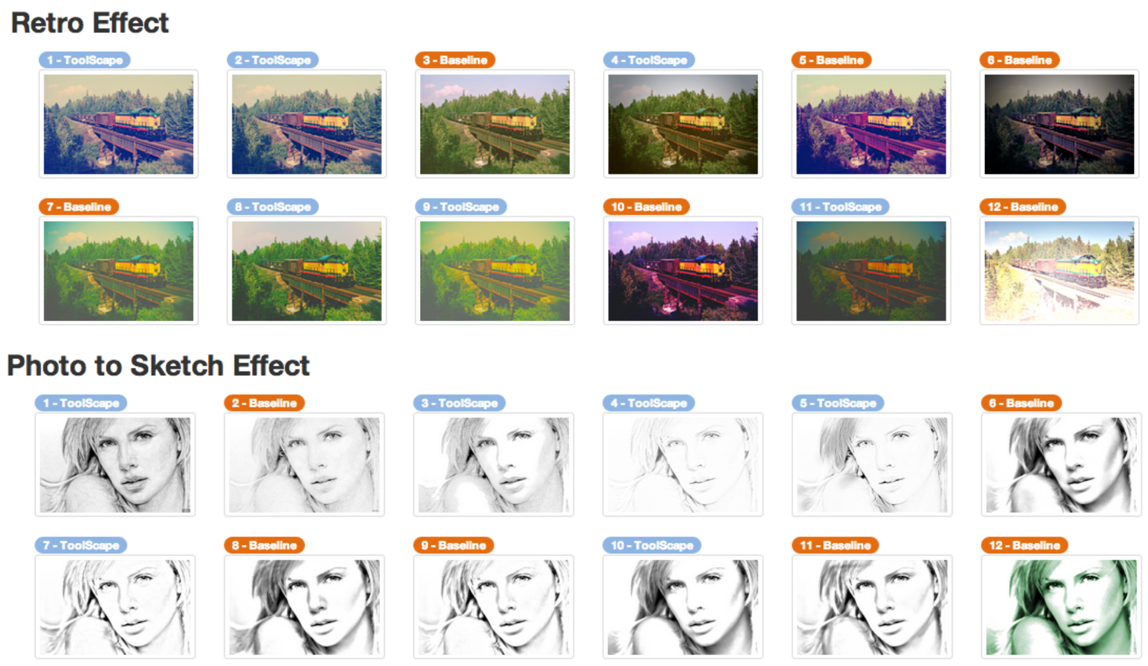


Figure 5-6: Rank-ordered designs created by the participants in the formative study. For each image, the top label displays the rank of the image and the condition it was created in (either ToolScape or Baseline). The mean rankings were lower in ToolScape (5.7) than in Baseline (7.3), which maps to higher quality designs.

the task was 3.9 (+0.1) whereas with ToolScape the score was 5.2 (+1.4), which meant that learners felt more confident in their graphical design skills after completing tasks with ToolScape. (For H1, H2, and H4, differences between interfaces were significant at p<0.05 using a Mann-Whitney U test.)

H2 (higher self-rating for ToolScape) is supported. Participants rated their own work quality higher when using ToolScape (mean rating of 5.3) versus baseline (mean of 3.5).

H3 (higher external judge rating for ToolScape) is supported. The overall ranking was computed by taking the mean of the four judges' ranks. The mean rankings (lower is better) for output images in the ToolScape and Baseline conditions were 5.7 and 7.3, respectively. A Wilcoxon Signed-rank test indicates a significant effect of interface (W=317, Z=-2.79, p<0.01, r=0.29). Furthermore, nine of the twelve participants produced higher-rated images with ToolScape. The ranking method yielded high inter-rater reliability (Krippendorff's alpha=0.753) for ordinal data. Figure 5-6 shows all the designs created by participants.

H4 (higher satisfaction with ToolScape) is supported. Mean ratings for ToolScape and Baseline were 6.1 and 4.5, respectively.

H5 (easier task difficulty perception for ToolScape) is not supported: The mean ratings for ToolScape and Baseline were 4.0 and 3.7, respectively. Combined with H2 and H3, this might indicate that participants did not find the tasks easier yet still produced better designs with greater confidence.

In conclusion, ToolScape had a significant effect on learners' belief in their graphical design skills and output quality. They also produced better designs as rated by external judges. Note that participants were watching the same video content in both conditions. Thus, the video annotation browsing interface affected design outcomes. Participants especially enjoyed being able to freely navigate between steps within a video by clicking on annotations.

5.2.5 Lessons for Video Browsing Interfaces

The features of ToolScape that provided higher interactivity and non-sequential access were highly rated and frequently used. In participants' responses to the 7-point Likert scale questions on the usability of interface features, the time-marked image thumbnails (6.4) and step links (6.3) were among the highest rated, as well as the graying out of "dead times" with no workflow progress (6.5). Participants noted, "It was also easier to go back to parts I missed.", "I know what to expect to get to the final result.", and "It is great for skipping straight to relevant portions of the tutorial."

All participants frequently used the ability to click on timeline links to navigate directly to specific images and steps. They clicked the interactive timeline links 8.9 times on average ($\sigma = 6.7$) in a single task. We also analyzed the tracking log, which records an event when the user clicks on an interactive link or a pause button, or drags the playhead to another position. The learners watched videos less linearly with ToolScape: The ToolScape condition recorded 150 such events, versus only 96 in the Baseline condition. In ToolScape, 107 out of 150 events were interactive link clicks and 43 were pause button clicks or direct scrubbing on the player. These findings indicate that interactive links largely replaced the need for pause or scrubbing, and encouraged the stepwise navigation of the procedure.

5.2.6 Lessons for How-To Video Annotation

The study results suggest that annotated step information makes how-to videos much more effective for learners. However, the bottleneck is in obtaining the annotations. Here are some lessons from our experience annotating videos by hand:

- Extracting step information from how-to videos involves detecting timing, generating a natural language description of a step, and capturing before and after states.
- It often requires multi-pass watching, which adds to task complexity. Before knowing what each step is, the annotator cannot extract before and after thumbnail images. This experience supports a design choice to split the work into multiple stages so that in each stage, the annotator's attention is focused on a single, simple task.
- Hand annotation is time-consuming. Roughly three times the original video length was required by trained annotators to annotate each how-to video.
- Timing detection is difficult. Sometimes there is an interval between when a step is spoken and demonstrated. Also, if the goal is to find a starting time of a step, the annotator has to watch, verify, and scroll back to mark as a valid step.

These lessons informed the design of our crowdsourced how-to video annotation method, which we now present.

5.3 Crowdsourcing Workflow: Find-Verify-Expand

Using lessons from our formative study, we designed a three-stage crowdsourcing workflow for annotating how-to videos with procedural steps, timings, textual descriptions, and before and after thumbnail images. This workflow works with any how-to video regardless of its domain, instructional style, and presentation. It also collects annotations with untrained crowd workers (e.g., workers on Mechanical Turk).

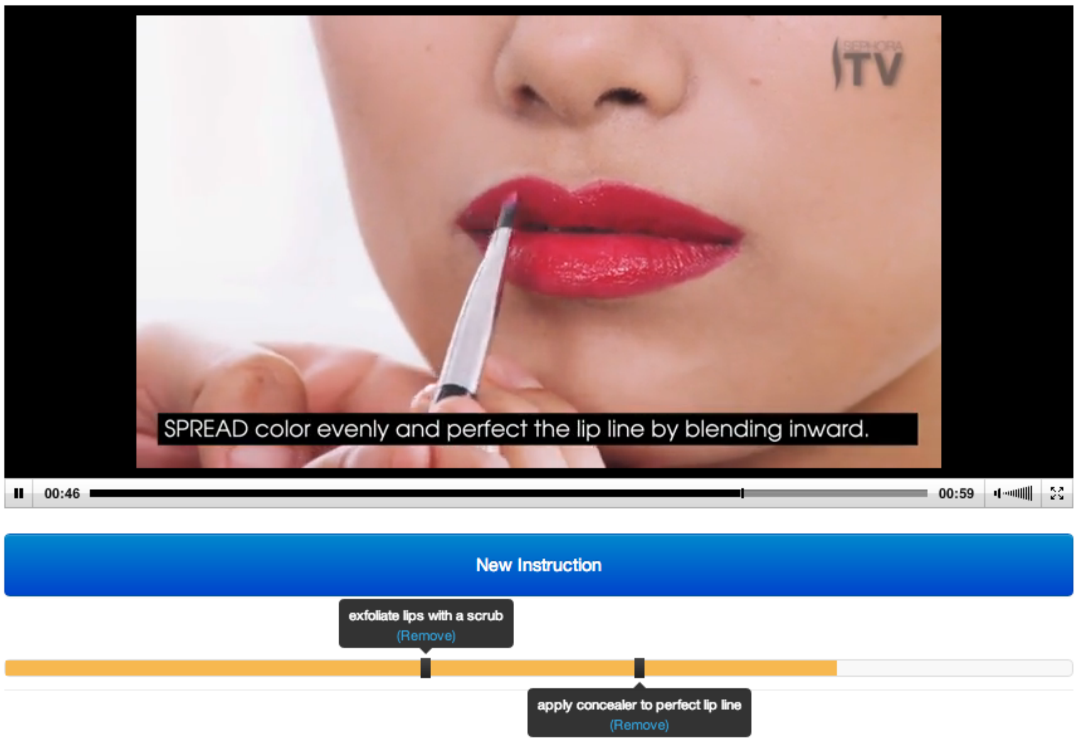

SPREAD color evenly and perfect the lip line by blending inward.
00:46
00:59
New Instruction
exfoliate lips with a scrub
(Remove)
apply concealer to perfect lip line
(Remove)

Figure 5-7: In the Find stage, the crowd worker adds new steps to the timeline by clicking on the "New Instruction" button.

Inspired by crowd design patterns that segment a bigger task into smaller micro-tasks [11], our workflow decomposes the annotation task into three stages and each video into shorter segments. This design addresses the task complexity and multi-pass overhead problems of manual annotation.

We developed a generalizable crowd workflow pattern called **Find-Verify-Expand** (Figure 6-1) for detecting temporal and visual state changes in videos, such as steps in a how-to video, highlights from a sports game, or suspicious incidents from a surveillance video. The unique *Expand* stage captures surrounding context and causal relationships (e.g., before/after images for a step in a how-to video) by expanding on the detected event (e.g., a step in a how-to video). To better handle crowd output coming from timing detection and image selection, we apply clustering algorithms and text and visual analysis techniques to intelligently merge results from workers.

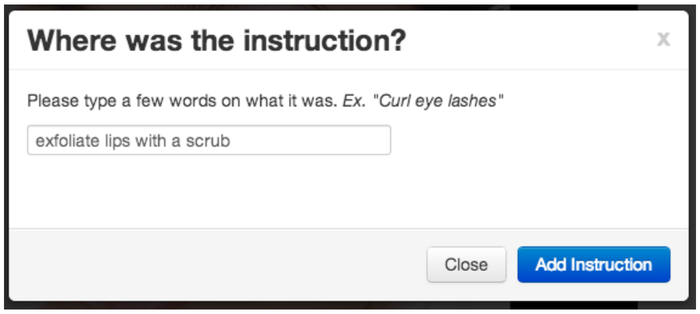

Where was the instruction?
x
Please type a few words on what it was. *Ex. "Curl eye lashes"*
exfoliate lips with a scrub
Close
Add Instruction

Figure 5-8: Upon clicking on the "New Instruction" button, a popup window asks the worker to describe what the step is about in free-form text.

5.3.1 Stage 1: Find Candidate Steps

This crowd task collects timestamps and text descriptions for possible steps from a video segment. While watching the video, the worker adds a step by clicking on the "New Instruction" button every time the instructor demonstrates a step (Figure 6-3). Each time the worker clicks on the button, the task prompts the worker to describe the step in free-form text (Figure 5-8). The same segment is assigned to three workers, whose results get merged to create candidate steps.

Pre-processing: A video is segmented into one-minute chunks. We learned from pilot runs that longer video segments lead to lower annotation accuracy toward the end and slower responses on Mechanical Turk. However, a drawback in using segmented video is the possibility of missing steps near segment borders. We address this issue by including a five-second overlap between segments, and attaching the final segment to the prior one if it is shorter than 30 seconds.

Task Design: For quality control, the task first ensures that the user has audio by giving a test that asks the worker to type in a word spoken from an audio file. Our pilot runs showed that labeling accuracy drops significantly when the worker does not listen to audio. Secondly, we disable the Submit button until the video playhead reaches the end to ensure that the worker watches the entire segment. Finally, when the worker clicks on the "New Instruction" button, the video pauses and a dialog box pops up to ask what the step was. Our initial version simply added a tick on the timeline and continued playing without pausing or asking for a label. But this resulted in workers clicking too many times (as many as 100 for

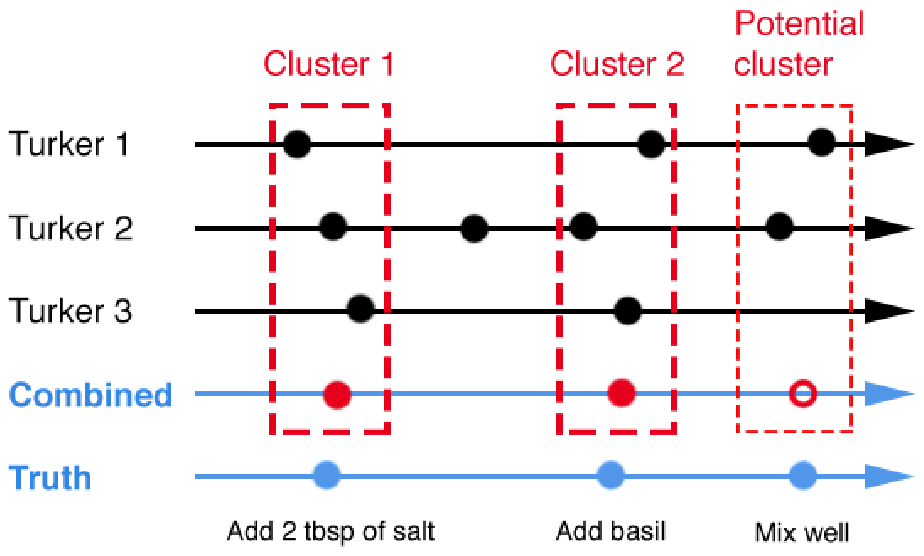


Figure 5-9: Our clustering algorithm groups adjacent time points into a candidate step. It further adds a potential cluster as a candidate, which might turn out to be a proper step once checked in the Verify stage. This inclusive strategy mitigates the effect of clustering errors.

a 60-second chunk) without thinking. The prompt adds self-verification to the task, which encourages the worker to process the workflow by each step. The prompt also includes an example label to show the format and level of detail they are expected to provide (Figure 5-8).

Post-Processing: The workflow intelligently merges results from multiple workers to generate step candidates. To cluster nearby time points given by different workers into a single step, we use the DBSCAN clustering algorithm [45] with a timestamp difference as the distance metric. The clustering idea is shown in Clusters 1 and 2 in Figure 5-9. The algorithm takes ϵ as a parameter, which is defined by the maximum distance between two points that can be in a cluster relative to the distance between farthest points. We train ϵ once initially on a small set of pilot worker data and ground truth labels. Our tests show that the values between 0.05 and 0.1 yield high accuracy, regardless of domain or video. We configured the algorithm to require at least two labels in every cluster, similar to majority voting among the three workers who watched the segment. We considered other clustering algorithms such as K-Means, but many require the number of clusters as an input parameter. In video annotation, the number of steps is neither known a priori nor consistent across videos.

Depending on videos and parameters, the DBSCAN algorithm might over-generate (false positive) or under-generate (false negative) clusters. We bias the algorithm to over-generate candidate steps ('potential cluster' in Figure 5-9) and aim for high recall over high precision, because the first stage is the only time the workflow generates new clusters. We improve the initial clusters in three ways, with the goal of higher recall than precision. First, we take into account the textual labels to complement timing information. The clustering initially relies on workers' time input, but using only time might result in incorrect clusters because steps are distributed unevenly time-wise. Sometimes there are steps every few seconds, and other times there might be no step for a minute. We run a string similarity algorithm between text labels in border points in clusters, to rearrange them to the closer cluster. Second, we break down clusters that are too large by disallowing multiple labels from one worker to be in a cluster. Finally, if there are multiple unclustered points within ϵ between clusters, we group them into a candidate cluster. For each cluster, we take a mean timestamp as the representative time to advance to the Verify stage.

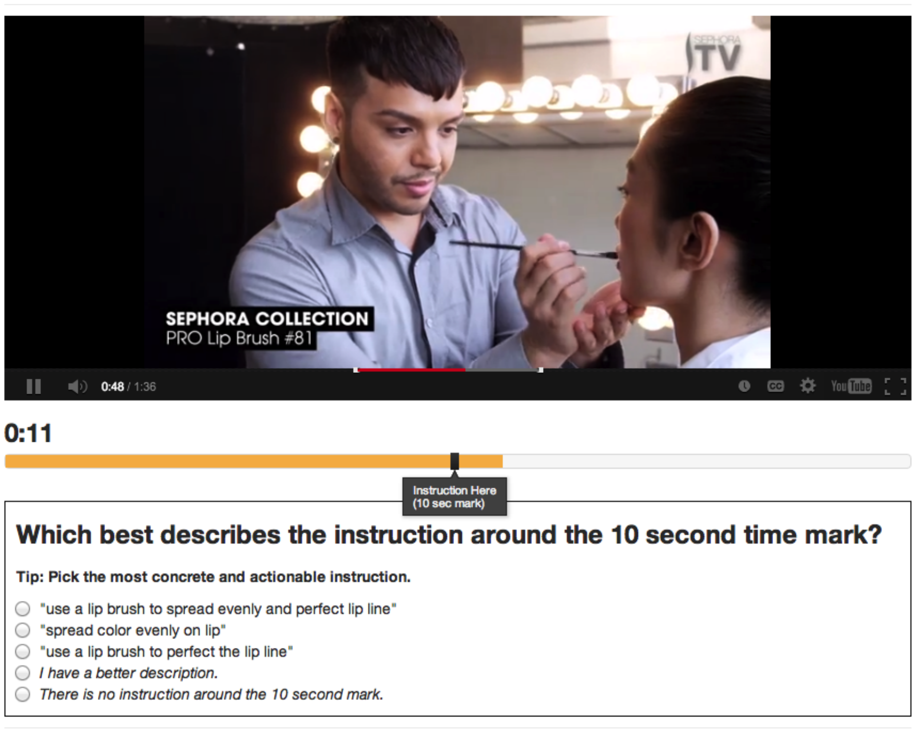
5.3.2 Stage 2: Verify Steps

Here the worker's verification task is to watch a 20-second clip that includes a candidate step and textual descriptions generated from the prior stage, and vote on the best description for the step (Figure 6-4). The workflow assigns three workers to each candidate step, whose votes are later merged.

Pre-processing: For each of the candidate steps from Stage 1, the workflow segments videos into 20-second clips around each step (10 seconds before and after).

Task Design: To prevent workers from selecting the first result without reviewing all options, we randomize the order of options presented each time. We also lowercase all labels to prevent capitalized descriptions from affecting the decision. Also, the Submit button becomes clickable only after the worker finishes watching the 20-second clip.

In addition to candidate text descriptions, two additional options are presented to work-



Which best describes the instruction around the 10 second time mark?

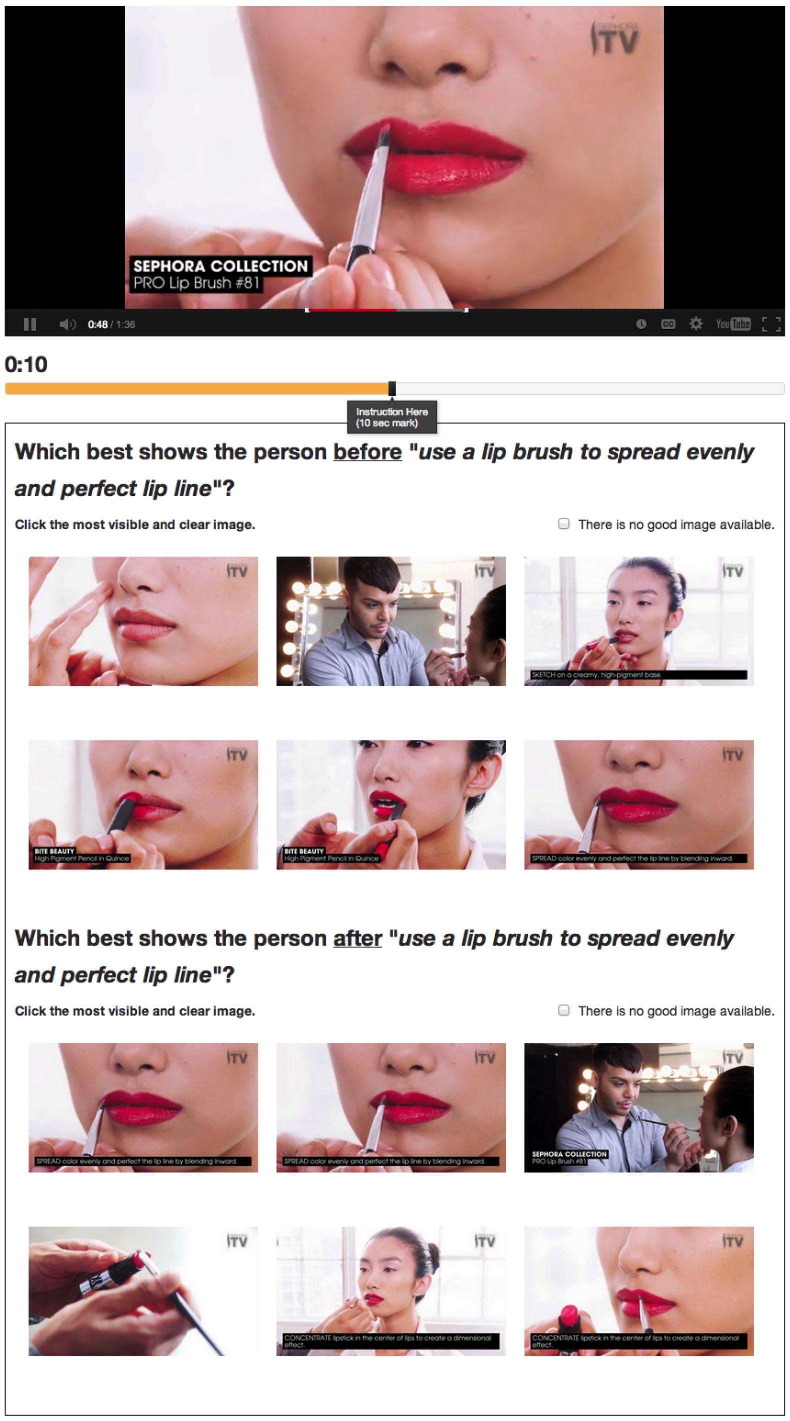
Tip: Pick the most concrete and actionable instruction.

- ○ "use a lip brush to spread evenly and perfect lip line"
- ○ "spread color evenly on lip"
- ○ "use a lip brush to perfect the lip line"
- ○ *I have a better description.*
- ○ *There is no instruction around the 10 second mark.*

Figure 5-10: The Verify stage asks the worker to choose the best description of a candidate step. The options come from workers in the Find stage. Additional default options allow the worker to either suggest a better description or mark the step as invalid.

ers: "I have a better description", which improves the step label, and "There is no instruction", which filters out false positives from Stage 1.

Post-Processing: Two possible outputs of this stage are 1) finalizing the timestamp and description for a valid step, or 2) removing a false step. The workflow uses majority voting to make the final decision: If two or more workers agreed on a description, it becomes the final choice. If workers are split between three different options, it checks if some of the selected text descriptions are similar enough to be combined. We first remove stop words for more accurate comparisons, and then apply the Jaro-Winkler string matching algorithm [76]. If the similarity score is above a threshold we configured with initial data, we combine the two descriptions with a longer one. If not, it simply picks the longest one from the three. The decision to pick longer description for tie-breaking comes from a pilot observation that longer descriptions tend to be more concrete and actionable (e.g., "grate three cups of cheese" over "grate cheese").



Which best shows the person before "*use a lip brush to spread evenly and perfect lip line*"?

Click the most visible and clear image.

☐ There is no good image available.

Which best shows the person after "*use a lip brush to spread evenly and perfect lip line*"?

Click the most visible and clear image.

☐ There is no good image available.

Figure 5-11: The Expand stage asks the worker to choose the best before and after images for a step. The worker visually reviews the thumbnail options and clicks on images to decide.

5.3.3 Stage 3: Expand with Before and After Images for Steps

This final stage collects the before and after images of a step, which visually summarize its effect. This stage captures surrounding context and causal relationships by expanding on what is already identified in Find and Verify. The worker's task here is to watch a 20-second video clip of a step, and select a thumbnail that best shows the work in progress (e.g., food, face, or Photoshop image) before and after the step (Figure 6-5). The workflow assigns three workers to each step.

Pre-processing: This stage uses a 20-second video clip of a step verified in Stage 2, and uses its final text label to describe the step. It creates thumbnails at two-second intervals to present as options, 10 seconds before and after the step.

Task Design: Our initial design asked workers to click when they see good before and after images, but this resulted in low accuracy due to variable response time and the lack of visual verification. We then simplified the task to a multiple choice question. Selecting from static thumbnail images makes the task easier than picking a video frame.

Post-Processing: Similar to the Verify stage, we apply majority voting to determine the final before and after images. For merging and tie breaking, we use Manhattan distance, an image similarity metric that computes pixel differences between two images.

5.4 Evaluation

We deployed our annotation workflow on Mechanical Turk and evaluated on:

- **Generalizability**: Does the workflow successfully generate labels for different types of how-to tasks in different domains with diverse video production styles?

- **Accuracy**: Do collected annotations include all steps in the original video, and avoid capturing too many (false positive) or too few (false negative)? How do textual descriptions generated by crowd workers compare to those generated by trained annotators?

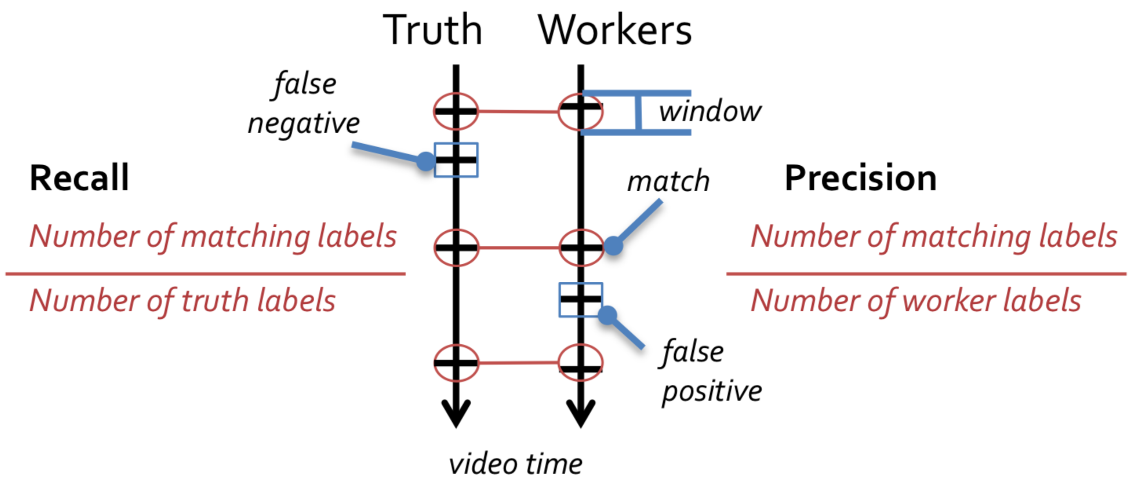


Figure 5-12: Our evaluation uses the Hungarian method to match extracted and ground truth steps with closest possible timestamp within a 10-second window size. Then we compute precision and recall, which indicate if our workflow over- or under-extracted steps from ground truth.

5.4.1 Methodology

We used our workflow and Mechanical Turk to fully extract step information from the 75 how-to videos in our annotation corpus, with 25 videos each in cooking, makeup, and graphics editing software (Photoshop). We did not filter out videos based on use of subtitles, transitions, or audio, to see if our annotation workflow is agnostic to presentation styles. Out of 75 videos in our set, 7 did not have audio, and 27 contained text overlays. For each domain, we picked five tasks to cover diverse types of tasks: Cooking – pizza margherita, mac and cheese, guacamole, samosa, and bulgogi; Makeup – bronze look, reducing redness, smokey eyes, bright lips, and summer glow; Photoshop: motion blur, background removal, photo to sketch, retro effect, and lomo effect. The mean video length was 272 seconds, summing to over 5 hours of videos.

5.4.2 Results

Our evaluation focuses on comparing the quality of step information produced by our crowdsourcing workflow against ground truth annotations from our corpus.

Domain	From Turkers	After Stage 1	After Stage 2	Ground Truth
Cooking	64.0	21.0	19.5	17.9
Makeup	53.3	16.4	15.3	14.8
Photoshop	43.5	12.5	12.2	12.4
All	53.6	16.7	**15.7**	**15.0**

Table 5.1: The mean number of steps generated by the workflow in each stage. At the end the workflow extracted 15.7 steps per video, which is roughly equivalent to 15.0 from ground truth. Stage 1 clusters the original Turker time points, and Stage 2 merges or removes some.

	Stage 1. Time only		Stage 2. Time + Text	
Domain	Precision	Recall	Precision	Recall
Cooking	0.76	0.89	0.77	0.84
Makeup	0.72	0.8	0.74	0.77
Photoshop	0.81	0.82	0.79	0.79
All	0.76	0.84	0.77	0.81

Table 5.2: When considering time information only, recall tends to be higher. When considering both time and text descriptions, incorrect text labels lower both precision and recall, but removing unnecessary steps in Stage 2 recovers precision.

Domain	Mean Distance (sec)	Stdev
Cooking	2.20	2.03
Makeup	3.37	2.39
Photoshop	2.52	2.24
All	2.66	2.26

Table 5.3: The mean time distance between extracted steps and ground truth steps was only 2.7 seconds on average, for all the steps that had a matching ground truth step. This means that the time-based clustering (Figure 5-9) accurately detected step timings.

The Turk crowd and trained annotators (two co-authors with educational video research experience) generated similar numbers of steps (Table 5.1). In Stage 1, 361 one-minute video segments were assigned to Turkers, who generated 3.7 candidate steps per segment, or 53.6 per video. Clustering reduced that to 16.7 steps per video. Stage 2 further removed over-generated steps, resulting in 15.7 per video, which is nearly equivalent to the ground truth of 15 steps per video.

Precision indicates how accurate extracted steps are compared to ground truth, while *recall* shows how comprehensively the workflow extracted ground truth steps (Figure 5-12). We present precision and recall results considering only the timing of steps (Stage 1), and both the timing and the textual description accuracy (Stage 2). For matching crowd-extracted steps to ground truth steps, we use the Hungarian method [97] whose cost matrix is filled with a time distance between steps.

Evaluating Stage 1: Find

We consider only precision and recall of times in the Stage 1 evaluation because final textual descriptions are not yet determined. Detecting the exact timing of a step is not straightforward, because most steps take place over a time period, and verbal and physical steps are commonly given with a time gap.

To more accurately account for the timing issue, we set a highest threshold in time difference that accepts a Turker-marked point as correct. We set the threshold to 10 seconds, which indicates that a step annotation more than 10 seconds off is discarded. This threshold was based on heuristics from step intervals in our corpus: We hand-annotated, on average, one step every 17.3 seconds in our video corpus (mean video length / number of steps in ground truth = 272/15.7), so a maximum 10-second difference seems reasonable.

The mean distance between ground truth steps and extracted steps (ones within 10 seconds of the ground truth) was only 2.7 seconds (Table 5.3). This suggests that for matched steps, the time-based clustering successfully detected the timing information around this distance. When considering only time accuracy, our workflow shows 0.76 precision and 0.84 recall (Table 5.2).

Evaluating Stage 2: Verify

Here we combine the accuracy of both timing and text descriptions. Precision for this stage captures what fraction of steps identified by the workflow are both placed correctly on the time line and whose description reasonably matches the ground truth. The analysis shows 0.77 precision and 0.81 recall over all the videos (Table 5.2).

For text accuracy measurement, we use the string similarity algorithm to see if a suggested description is similar enough to a description from ground truth. We apply the same threshold as what we configured in the workflow for tie breaking in the Verify stage. The precision and recall both go down when the text similarity condition is added, but precision recovers from the post-processing of steps in this stage. Two enhancements contribute to this recovery: removing steps that workers indicated as “no instruction” from the task, and merging nearby steps that have identical descriptions.

In 76% of the steps, two or more Turkers agreed on a single description. For the rest, the tie breaking process determined the final description. For 13% of the steps, Turkers provided their own description.

Evaluating Stage 3: Expand

This evaluation should judge if crowd-selected before and after images correctly capture the effect of a step. Because this judgment is subjective, and there can be multiple correct before and after images for a step, we recruited six external human evaluators to visually verify the images. We assigned two evaluators to each domain based on their expertise and familiarity with the domain, and gave a one-hour training session on how to verify before and after images. For each workflow-generated step, we presented an extracted text description along with a before and after image pair. Their task was to make binary decisions (yes / no) on whether each image correctly represents the before or after state of the step.

We used Cohen’s Kappa to measure inter-rater agreement. The values were 0.57, 0.46, 0.38, in cooking, makeup, and Photoshop, respectively, which show a moderate level of agreement [4]. Results show that on average, both raters marked 60% of before and after images as correct. At least one rater marked 81.3% as correct.

Cost, time, and tasks

We created a total of 8,355 HITs on Mechanical Turk for annotating 75 videos. With three workers on each task and a reward of \$0.07, \$0.03, and \$0.05 for Find, Verify, and Expand, respectively, the average cost of a single video was \$4.85, or \$1.07 for each minute of a how-to video. The Expand stage was more costly (\$2.35) than the first two; thus, time points and text descriptions can be acquired at \$2.50 per video. The average task submission time was 183, 80, and 113 seconds for Find, Verify, and Expand, respectively.

Results summary

In summary, our workflow successfully extracted step information from 75 existing videos on the web, generalizing to three distinct domains. The extracted steps on average showed 77% precision and 81% recall against ground truth, and were 2.7 seconds away from ground truth. Human evaluators found 60% of before and after images to be accurate.

5.5 Discussion and Limitations

We now discuss qualitative findings from the experiment, which might have practical implications for future researchers designing crowd workflows.

Detecting precise timing of a step. We observed that Turkers add new steps with higher latency than trained annotators, resulting in Turker-labeled time points being slightly later than those by annotators for the same step. The trained annotators often rewinded a few seconds to mark the exact timing of a step after seeing the step, whereas most Turkers completed their tasks in a single pass. While this might be a limitation of the workflow, our results show that a reasonable window size mitigates such differences. For improving timing accuracy, time-shifting techniques [105] can be explored.

Handling domain and video differences. Extraction accuracy in our workflow was consistent across the three domains with different task properties. This finding validates our domain-agnostic approach based on the general properties of procedural tasks. Photoshop videos were often screencasts, whereas cooking and makeup videos were physical demonstrations. Cooking videos contained higher number and density of steps than makeup or Photoshop videos, while Photoshop and makeup videos often had longer steps that required fine-grained adjustments and tweaking. Also, some videos were studio-produced with multiple cameras and high-quality post-processing, while others were made at home with a webcam. Our workflow performed robustly despite the various differences in task properties and video presentation styles.

Extracting steps at different conceptual levels. Video instructors present steps at different conceptual levels, and this makes it difficult to keep consistent the level of detail in Turkers' step detection. In a makeup video, an instructor said "Now apply the bronzer to your face evenly", and shortly after applied the bronzer to her forehead, cheekbones, and jawline. While trained annotators captured this process as one step, our workflow produced four, including both the high-level instruction and the three detailed steps. Turkers generally captured steps at any level, but our current approach only constructs a linear list of steps, which sometimes led to redundancy. Previous research suggests that many procedural tasks contain a hierarchical solution structure [23], and we plan to extend this work to hierarchical annotation.

5.6 Conclusion

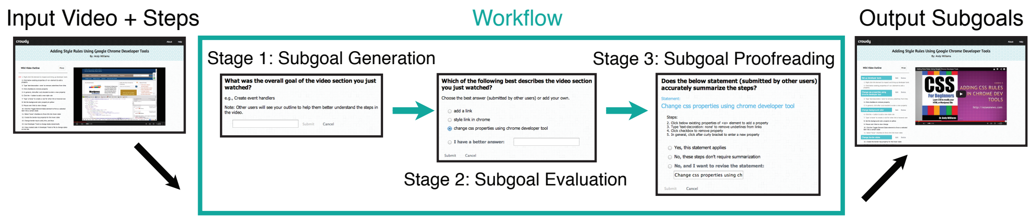
This chapter presents a scalable crowdsourcing workflow for annotating how-to videos. The Find-Verify-Expand pattern efficiently decomposes the complex annotation activity into micro-tasks. Step information extracted from the workflow can enable new ways to watch and learn from how-to videos. We also present ToolScape, an annotation-enabled video player supporting step-by-step interactivity, which is a potential client of this workflow. Our lab study shows the value of accessing and interacting with step-by-step information for how-to videos. Participants watching videos with ToolScape gained higher self-efficacy, rated their own work higher, and produced higher-rated designs.

Chapter 6

Crowdy: Learnersourcing Subgoal Labels for How-to Videos

This chapter presents the second example of active learnersourcing, Crowdy, which learnersources section summaries of a how-to video. This chapter has adapted, updated, and rewritten content from a paper at CSCW 2015 [163], which extends the idea proposed in a Works in Progress at CHI 2013 [88]. All uses of "we", "our", and "us" in this chapter refer to coauthors of the aforementioned papers.

Websites like YouTube host millions of how-to videos, but their interfaces are not optimized for learning. Previous research suggests that people learn more from how-to videos when the videos are accompanied by outlines showing individual steps and labels for groups of steps (subgoals) [23, 113]. We envision an alternative video player where the steps and subgoals are displayed alongside the video. To generate this information for existing videos, we use learnersourcing. To demonstrate this method, we deployed a live website with a workflow for constructing subgoal labels implemented on a set of how-to videos. A study with Turkers showed higher learning outcomes with Crowdy when compared against the baseline video interface. The Crowdy group also performed as well as the group that were shown expert-generated labels. For the four videos with the highest participation, we found that a majority of learner-generated subgoals were comparable in quality to expert-generated ones. Learners commented that the system helped them grasp the material, suggesting that our workflow did not detract from the learning experience.


Input Video + Steps
Workflow
Output Subgoals
Stage 1: Subgoal Generation
Stage 2: Subgoal Evaluation
Stage 3: Subgoal Proofreading
Submit
Cancel

What was the overall goal of the video section you just watched?

e.g., Create event handlers

Note: Other users will see your outline to help them better understand the steps in the video.

Which of the following best describes the video section you just watched?

Choose the best answer (submitted by other users) or add your own.

- add a link
- style link in chrome
- [✓ change css properties using chrome developer tool]
- I have a better answer:

Does the below statement (submitted by other users) accurately summarize the steps?

Statement:
Change css properties using chrome developer tool

Steps:
2. Click below existing properties of <a> element to add a property
3. Type 'text-decoration: none' to remove underlines from links
4. Click checkbox to remove property
5. In general, click after curly bracket to enter a new property

- Yes, this statement applies
- No, these steps don't require summarization
- No, and I want to revise the statement: Change css properties using ch

Figure 6-1: We present a three-stage workflow to generate labels for groups of steps (subgoal labels) for how-to videos. This workflow is designed to engage people actively trying to learn from the video to contribute the information.

6.1 Motivation and Contributions

Previous research [113] has shown that people learn better from videos if they are simultaneously presented with the *subgoals*, which capture meaningful conceptual pieces of the procedure covered in the video (Figure 2-1). Presenting subgoals improves learning by reducing learners' cognitive load by abstracting away low-level details, and by encouraging learners to self-explain why a set of steps have been grouped [23]. Ideally, to help people learn better from how-to videos, a video interface could display a list of subgoals as learners watch a video. Traditionally, labeling subgoals requires domain experts and knowledge extraction experts [24], but this process will not scale to the millions of existing how-to videos on the web. While makers of how-to videos may eventually be convinced to add subgoal labels for each video they create, we seek a scalable mechanism for improving the learning experience for viewers of existing videos.

In this chapter, we turn to learners for help in generating subgoal labels at scale for how-to videos on the web (Figure 6-1). We rely on active learnersourcing, a method for human computation for gathering information from people trying to actively learn from a video. Our solution is a three-stage workflow with microtasks for learners to complete as they watch the videos. We hypothesize that learners will learn the material better with this workflow while generating expert-quality subgoals at the same time.

In our learnersourcing workflow for constructing subgoal labels, as learners watch a video, the video stops periodically and the system asks one of three kinds of questions. The choice of question depends on how much information has already been gathered for that section in the video, and the questions are designed to engage learners to reflect on the content.

- **Generate**: In the first stage of the workflow, learners are asked to summarize the preceding short video section, which generates candidate subgoal labels for the section.
- **Evaluate**: After enough candidate subgoals have been generated for a video section, learners are routed to the second stage where they answer an automatically generated multiple choice question. The question asks them to pick the best subgoal from candidates contributed by previous learners. Learners' answers are then used by the system to evaluate the learner-generated subgoals.
- **Proofread**: In the third stage, learners make improvements to the most popular subgoal by considering its scope and language.

Once learners stop making changes to a subgoal, we can consider it final and future learners can be presented a final list of subgoals when watching the video.

The contributions of this chapter are as follows:

- An implementation of a three-stage learnersourcing workflow to generate high-quality subgoal labels from how-to videos in a scalable way.
- Results from a study demonstrating the pedagogical benefits of participating in the learnersourcing workflow.
- Results from an in-the-wild study demonstrating the workflow's effectiveness in generating high-quality subgoals. 14 out of 17 subgoals had at least one out of four evaluators rate the learner-generated subgoal label as equal to or better than the expert-generated label.
- Results from interviews with learners who participated in the in-the-wild study suggesting that the system helped them pay attention to videos and grasp the material better.

6.2 Learnersourcing Subgoal Labels

In order to generate subgoal labels for how-to videos, we use a learnersourcing approach. Our design goals are:

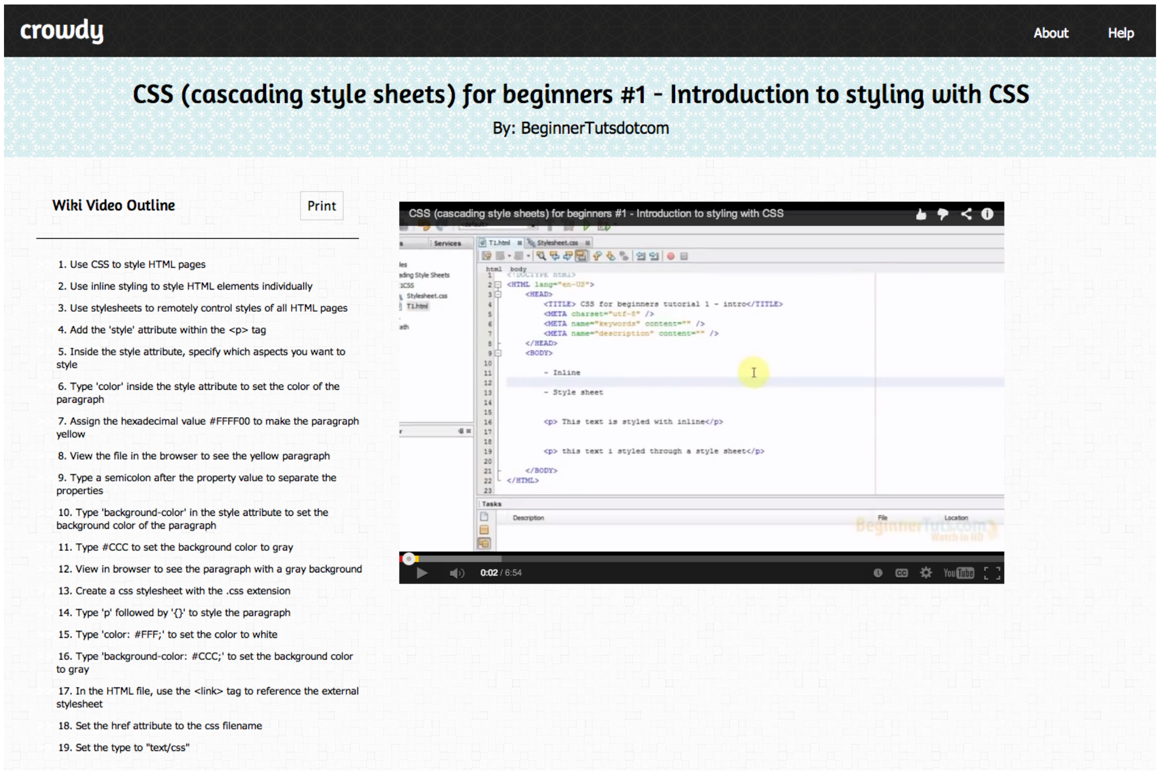
- To create a method for generating subgoal labels that does not require experts,
- To enable learners to generate expert-quality subgoals collaboratively,
- To design human computation tasks that improve learning and are engaging.

Due to the variety in video domain and presentation style on the web, an automatic approach to generating subgoals does not seem feasible. For example, some videos include text, some are silent and only show a person completing certain actions, and others rely on video and sound for demonstration. Automatically adapting to the various presentation and instructional styles is difficult, and we are looking for a solution that can generalize to any how-to video on the web. Additionally, subgoals are often at a higher conceptual level than what the video explicitly mentions, so automatically generating subgoals would require interpreting and understanding the material.

We built a website, Crowdy, to implement our workflow for learnersourcing subgoal labels. It is an alternative site to watch videos that were originally hosted on YouTube. The front page features all videos for the current deployment. When a user selects a video, Crowdy shows a separate page with an embedded video player and interactive outline (Figure 6-2). Our learnersourcing workflow asks specific questions as learners watch the videos, and in turn generates high-quality subgoals for them. Crowdy was created using standard web technologies (HTML, CSS, jQuery), with a Django backend and the YouTube API (`https://developers.google.com/youtube/`) for the video player.

6.3 Workflow Design

In order to enable learners to generate subgoal labels from how-to videos, we designed a three-stage workflow that engages learners to create and refine each others' subgoal labels. Initially, learners are presented with a video player and an interactive outline panel seeded



CSS (cascading style sheets) for beginners #1 - Introduction to styling with CSS

By: BeginnerTutsdotcom

Wiki Video Outline

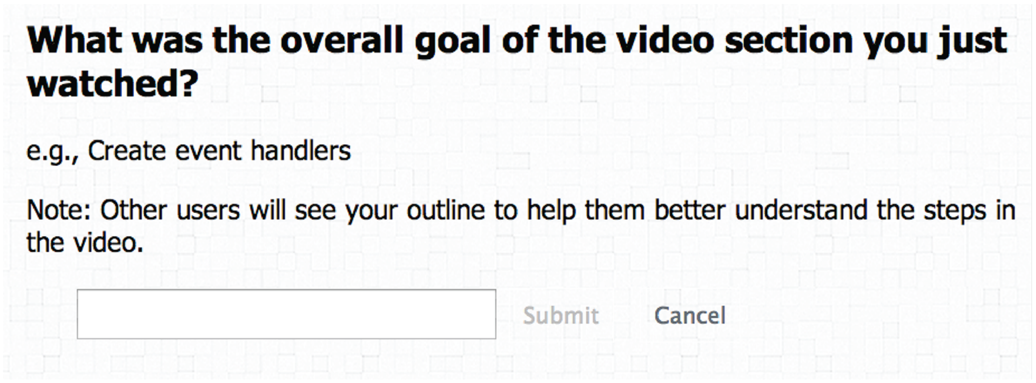
1. Use CSS to style HTML pages
2. Use inline styling to style HTML elements individually
3. Use stylesheets to remotely control styles of all HTML pages
4. Add the 'style' attribute within the <p> tag
5. Inside the style attribute, specify which aspects you want to style
6. Type 'color' inside the style attribute to set the color of the paragraph
7. Assign the hexadecimal value #FFFF00 to make the paragraph yellow
8. View the file in the browser to see the yellow paragraph
9. Type a semicolon after the property value to separate the properties
10. Type 'background-color' in the style attribute to set the background color of the paragraph
11. Type #CCC to set the background color to gray
12. View in browser to see the paragraph with a gray background
13. Create a css stylesheet with the .css extension
14. Type 'p' followed by '{}' to style the paragraph
15. Type 'color: #FFF;' to set the color to white
16. Type 'background-color: #CCC;' to set the background color to gray
17. In the HTML file, use the <link> tag to reference the external stylesheet
18. Set the href attribute to the css filename
19. Set the type to "text/css"

```
<!DOCTYPE html>
<HTML lang="en-US">
    <HEAD>
        <TITLE> CSS for beginners tutorial 1 - intro</TITLE>
        <META charset="utf-8" />
        <META name="keywords" content="" />
        <META name="description" content="" />
    </HEAD>
    <BODY>

        - Inline

        - Style sheet


        <p> This text is styled with inline</p>


        <p> this text i styled through a style sheet</p>

    </BODY>
</HTML>
```

Figure 6-2: For each video on Crowdy, we show the video player as well as a side panel with the individual steps. After learners answer a question, the subgoal they added or picked appears in the outline. Learners can click on steps or subgoals to jump to various parts in the video. The question block (Figure 6-3, 6-4, 6-5) temporarily replaces the video player when it appears. The "Print" button enables exporting the outline in a text format for learners to review later even without the video.

with the low level steps that the video goes through (Figure 6-2). By aggregating the micro-contributions from learners, Crowdy adds subgoal labels to this outline. The steps are presented as a vertical timeline, such that clicking on a step moves the playhead to that time in the video. This research assumes that the low-level steps have already been generated for a video, and focuses only on generating subgoal labels. Although we manually added the steps for this research, individual steps could be obtained by crowdsourcing [89] or automatically in part, possibly by processing a transcript or notes from the video creators.

As learners watch a video, they are stopped periodically and asked a question about the preceding video section. Although we experimented with having a Wiki-like interface where learners were asked to contribute subgoal labels but were not forced to, we found that this direct question approach resulted in more frequent and higher quality participation.


What was the overall goal of the video section you just watched?

e.g., Create event handlers

Note: Other users will see your outline to help them better understand the steps in the video.

Figure 6-3: In Stage 1, learners are asked to generate a new subgoal label after watching a video segment.

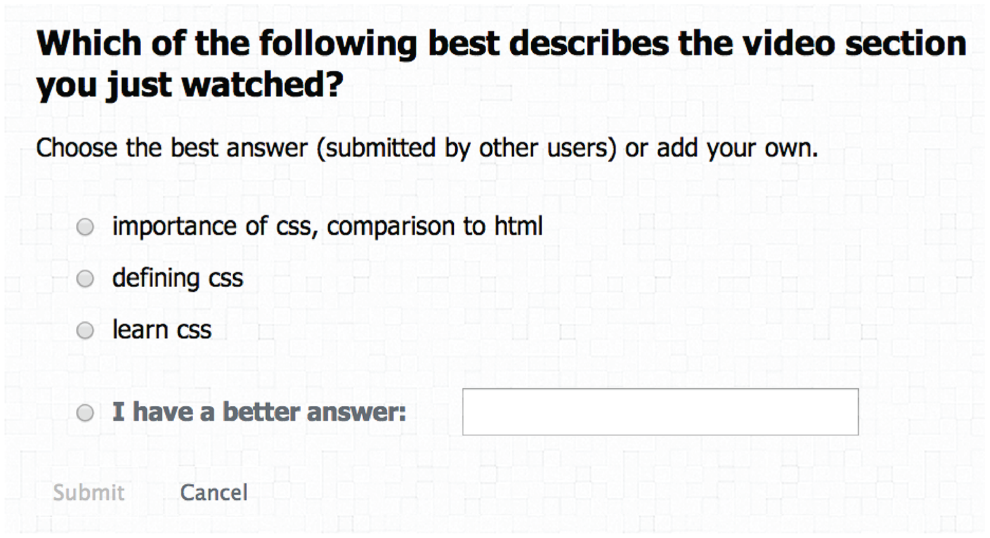
We chose a multi-stage workflow because we wanted to make each task small enough so that learners could complete it without getting too distracted from the video. To ensure a high quality of final subgoal labels, we use three mechanisms used in prior crowd-powered systems: we solicit multiple candidate solutions, we use voting to identify the best candidates, and we allow for iterative refinement.

We automatically route each subgoal to the necessary stage, so a learner might experience different stage prompts even while watching a single video. By removing the need for human intervention and judgment to move to the next stage, we aim to standardize and improve the efficiency of the workflow.

6.3.1 Stage 1: Generate

The goal of the first stage (Figure 6-3) is to have learners generate candidate subgoal labels from scratch. After a certain interval, learners are asked to summarize what they have just watched. While the learner is answering the question, the individual steps that were covered in the most recent video segment are bolded in the outline panel. Learners are given the option to cancel out of the question and are not forced to submit an answer.

In order to minimize spam answers, we made sure learners could easily cancel out of the prompt. We also designed the question such that it does not require prior familiarity with the concept of subgoals, but such that the answer to the question can serve as a subgoal label. From our preliminary user tests, we found that learners preferred jargon-free


Which of the following best describes the video section you just watched?

Choose the best answer (submitted by other users) or add your own.

- importance of css, comparison to html
- defining css
- learn css
- I have a better answer:

Figure 6-4: In Stage 2, learners vote on a subgoal they find most accurate.

questions (i.e., using ‘goal’ instead of ‘subgoal’).

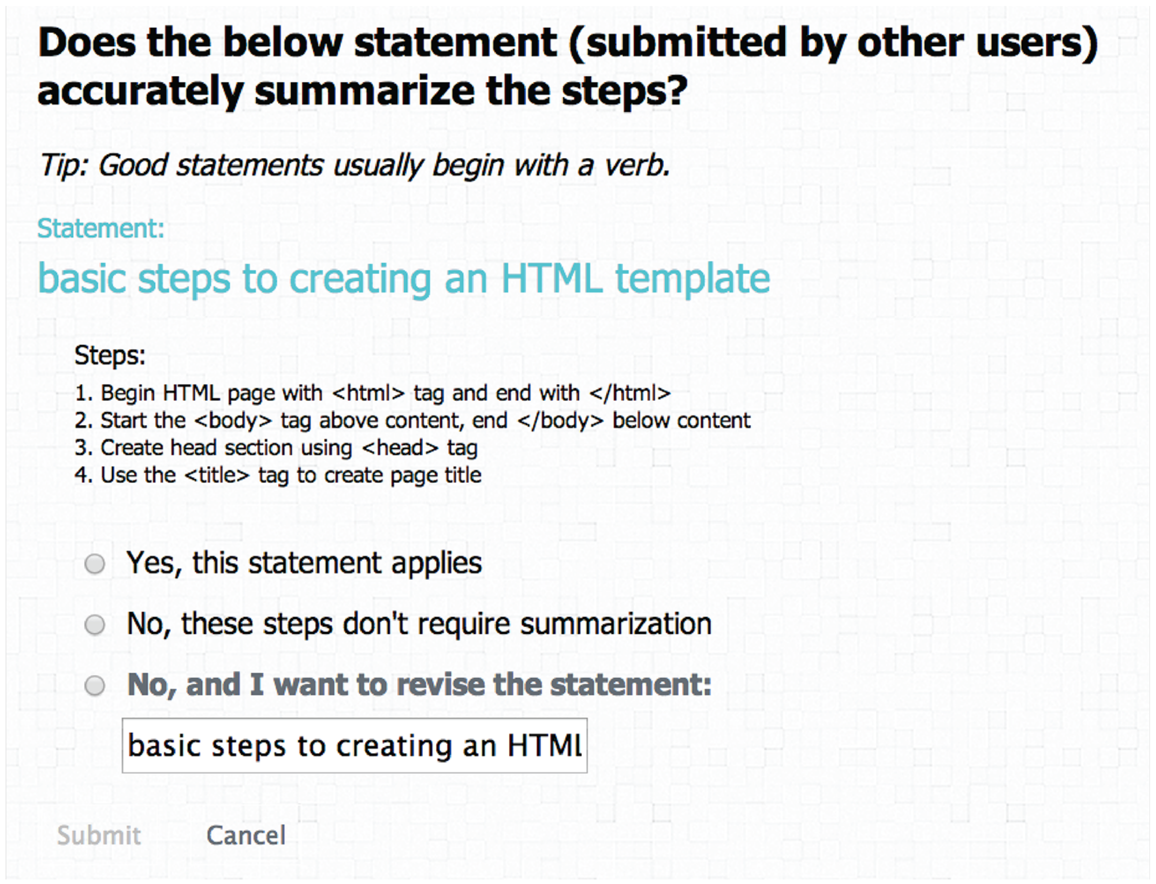
When there have been three candidate subgoal labels generated from the first stage, the next set of learners are asked the stage 2 question.

6.3.2 Stage 2: Evaluate

The subgoal labels that learners generate in stage 1 are not always of the highest quality. Learners may interpret the question in different ways or have different levels of familiarity with the video’s topic, which cause their subgoal labels to be different from each other. In stage 2 (Figure 6-4), learners are presented with the previously generated subgoals and asked to choose which they feel best captures the preceding section.

The goal of the second stage is to choose the most relevant answer for the video section, as well as weed out potential spam answers. The input to the question in this stage are the subgoal labels previously generated by learners. The system presents to learners in this stage a multiple choice question with previous learners’ labels as options. The number of subgoal labels presented is limited to maximum 4 to control for the complexity of the question. The system randomly chooses a subset of all generated subgoals each time.

The learners who are presented with the stage 2 question have the option to choose one of the subgoals, add their own, or abstain. We do not give learners any indication as to



Does the below statement (submitted by other users) accurately summarize the steps?

Tip: Good statements usually begin with a verb.

Statement:

basic steps to creating an HTML template

Steps:

1. Begin HTML page with <html> tag and end with </html>
2. Start the <body> tag above content, end </body> below content
3. Create head section using <head> tag
4. Use the <title> tag to create page title

- ○ Yes, this statement applies
- ○ No, these steps don't require summarization
- ○ **No, and I want to revise the statement:**

basic steps to creating an HTMl

Figure 6-5: In Stage 3, learners proofread the most popular subgoal for language and scope.

which subgoal is most popular during this stage. The subgoal that the learner chooses gets an upvote, and the ones that were not chosen receive a downvote. We use the upvote and downvote measures to determine when to move on to the third stage. This helps interpret user actions by weeding out answers that are consistently not chosen and considering newly generated subgoals. When the number of upvotes or the difference in upvotes between the top two subgoals reaches a certain threshold, we accept that subgoal as the winner of stage 2 and route the question to stage 3 for future learners.

6.3.3 Stage 3: Proofread

Learners in stage 2 evaluate the previously generated subgoal labels, but the most popular subgoal label from stage 2 is not necessarily a high-quality label by our standards. We define high-quality subgoal labels as having the correct scope (the subgoal label accurately reflects the underlying steps) and being concrete and descriptive. For example, for a section in a CSS video about changing the color of a div, a label like “CSS video” would be too broad, whereas a label such as “Use CSS to change div color” would be more appropriate.

The goal of this stage (Figure 6-5) is for learners to evaluate the most popular subgoal label from stage 2 for quality and eventually agree on a final label for that subgoal. The input to this stage is the most popular subgoal from stage 2, determined by the relative numbers of upvotes and downvotes. Learners are presented with this subgoal label and the steps that it covers, and have the option to refine the label or keep it as is. Ideally, when N learners accept the label without making a change we can assume that the label is final and future learners would be shown this subgoal without being asked a question. The default value for N is 3.

We expect alterations at this stage to be minimal because the necessary changes are often subtle and require a certain understanding of the material. By specifically asking learners to iterate on the subgoal label, we envisioned that some would accept the challenge and improve the quality of the subgoal. If a revision is entered, the system waits until N more people accept the revised subgoal before declaring it final.

6.3.4 Pedagogical Benefits in Learner Prompts

Because we are asking learners to summarize the information presented in the video, we believe our tasks may offer a pedagogical benefit to learners. At the same time, a major concern in designing Crowdy was to design the workflow such that participation in the process of subgoal label creation would not detract from the learning experience. Our core design decision to use short prompts interspersed throughout the video as the way to collect input from learners was informed by prior research in education and learning technologies, which we review here.

The subgoal learning model suggests that the process of grouping a set of solution steps involves self-explanation [24]. Previous research on self-explanation [14, 65, 158] suggests that when learners are asked to explain examples to themselves, they learn the material better and have greater awareness of their own understanding. Self-explanation encourages learners to engage in deep cognitive processing, which results in more robust learning. Our learner prompts also aim to encourage active learning [142], as opposed to passively watching a video, which is shown to engage learners in the learning process and improve retention of the material.

Research on designing learner prompts shows that learners self-correct misconceptions by answering the prompts [165]. Existing online learning platforms such as Coursera add frequent micro-quizzes inside videos to help people learn better from video lectures, and research on retrieval practice might provide support for in-video quizzes [80, 81]. In multimedia learning, different learner prompts are shown to provide different pedagogical benefits [120]. The learning gain was higher when learners picked the best explanation than when they were asked to type in a possible explanation [77], possibly because the higher cognitive load in generating an explanation might break the flow in multimedia learning.

These findings suggest that answering learner prompts while watching a video may actually offer a pedagogical benefit to learners.

6.4 Pilot Study

To iterate on the workflow design and get a general sense of the quality of learner-generated subgoals, we conducted a pilot study in a user interface design class (6.813/6.831) at MIT in Spring 2014. The class covers the theory behind designing usable interfaces, and students are responsible for creating and testing a web interface. Therefore, we felt it was a suitable deployment for a series of introductory web programming how-to videos. While students are expected to complete programming assignments and a project involving web interfaces, these topics are not explicitly covered during class time.

There were approximately 280 students in the class, and most were computer science majors. By the end of the class we had brought together 21 videos on HTML, CSS, jQuery/JavaScript, and Meteor (https://www.meteor.com/), a JavaScript platform for building web apps) into one place. These videos had an average duration of 6:18 minutes (stdev=2:49). The videos were all existing videos on YouTube. For each video we generated the individual steps and learners went through the three-stage workflow to generate subgoals. There was an average of 76 students participating in each video (stdev=39), 20 subgoals generated per video (stdev=16), and 193 actions (stdev=395), which are logged when learners answer a question or manipulate subgoals in any way. We deployed the videos periodically based on the topics for the upcoming problem sets/projects. The video links were embedded into assignment handouts as a resource. Although students had to be responsible for the material in the videos in order to succeed in the class, watching the videos was optional. We used a group of college students to iterate on our design, but in the future we will consider how using different groups of learners may affect the success of our workflow, especially if there is no constraint on who can participate.

We did not conduct a rigorous evaluation of the results from this deployment, and instead used the periodic deployments as an opportunity to iterate on our design. We found that participation was consistent throughout the semester, although participation varied greatly per video, likely based on video quality and difficulty of the material. In the beginning of the deployment we received a lot of spam answers, which we attribute to unclear instructions on the website that did not emphasize the option to cancel out of a question. Additionally, it was not obvious that a user's answers would be seen by others. The instructions in our final workflow state that the answers will be seen by other learners, and we added a 'Cancel' button to reduce spam.

6.4.1 Design Dimensions Refined through Pilot Study

Through our pilot deployment, we refined important design dimensions for the workflow such as the correct interval to ask questions, whether it is important to show steps, and whether it is important that learners are asked a question at every interval. During the deployment, we analyzed the submitted subgoals and talked with users to determine the pain points of our system. For example, we added an interactive tutorial to Crowdy because first-time users commented that they were caught off-guard and confused by the questions. We then iterated on our design and continued our analysis to arrive at our final system.

Question Interval

For the pilot iteration we asked questions at predetermined, fixed intervals because we wanted to focus on the overall design of the questions. We started using a 30-second interval based on our analysis of short (<5 minutes) videos. We arrived at this interval by manually annotating the subgoals for the short videos we used and averaging the time at which the subgoals occurred. However, the web programming videos we used in our pilot deployment were often longer than five minutes. We ended up with a one minute interval for our live deployment, based on users' feedback that frequent stops can be annoying to them. In the future, we would like to make the question intervals adapt to the domain, context, or video length.

Steps vs. No Steps

We also tested whether the workflow would be successful if the individual steps were not shown, in case step generation becomes a future bottleneck. We found no significant difference between subgoals generated when steps were present and not present. However, users liked the extra information provided and the ability to navigate to different points in the video by clicking on the steps. The presence of steps allowed users who may have lost attention during the video to generate a high-quality subgoal by examining the steps that were covered instead of submitting spam. Therefore we decided to show the steps for the live deployment.

Random vs. Not Random

We also tested whether learners would generate high quality subgoals if they were not asked a question at every interval boundary. Half of the users were asked a question randomly at each interval, and we found that the random interval led to subgoals that varied in scope. One learner who had the random condition and was only asked two questions, both at the very end of the video. Both of the subgoals she generated were summaries of the entire video, instead of being related to the preceding short section. Asking questions at regular intervals tended to lead to greater consistency in subgoals.

6.5 Evaluation Overview

The central premise of learnersourcing is that it helps learners learn the material better *while* their collective contributions generate meaningful artifacts for future learners. Our evaluation focuses on verifying this premise in two separate research questions: 1) Does participating in learnersourcing yield better learning outcomes? 2) Are the learnersourced subgoals in good quality?

6.6 Study 1: Pedagogical Benefits of Learnersourcing

The goal of this part of evaluation was to quantitatively determine the learning benefit of prompts specifically designed to generate subgoals. We created three interface conditions that simulated realistic video learning scenarios. While participants watched a how-to video, the video interface either 1) simply played the video without displaying any additional information or prompting learners (baseline condition, Figure 6-6), 2) displayed expert-generated subgoals but didn't prompt learners (expert condition, Figure 6-7), or 3) prompted them to generate a subgoal every minute and displayed their responses in the sidebar (Crowdy condition, Figure 6-8)

The baseline condition represents a majority of video interfaces today, which plays the video content without any additional support. The expert condition is meant to create a highly customized environment where subgoals are labeled by experts and displayed to learners. While it may be desirable to have more content labeled by experts, limited expert availability and time prevent a vast majority of educational videos from getting labeled. For the study, two researchers with expertise in statistics discussed together to generate subgoals for the material. Crowdy is an alternative way to get a video labeled, with learners' collective efforts without relying on experts. More importantly, the process of making these efforts may aid learning. We hypothesize that Crowdy will be more effective in learning than the baseline condition, and as effective as the expert condition.

The video used in the study was an intro-level statistics video explaining how to compute a population standard deviation [1]. The video was from Khan Academy.

[1] https://www.khanacademy.org/math/probability/descriptive-statistics/

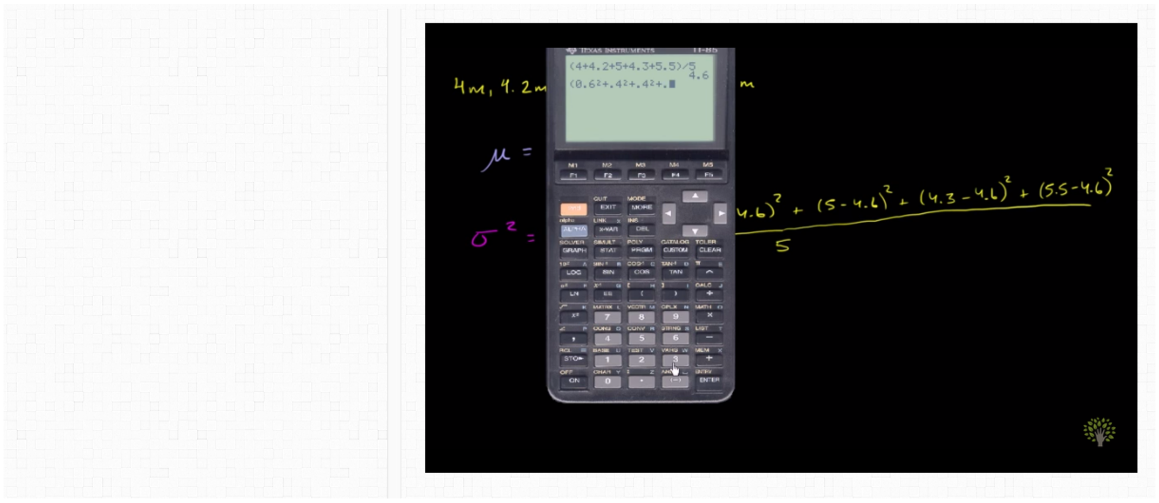
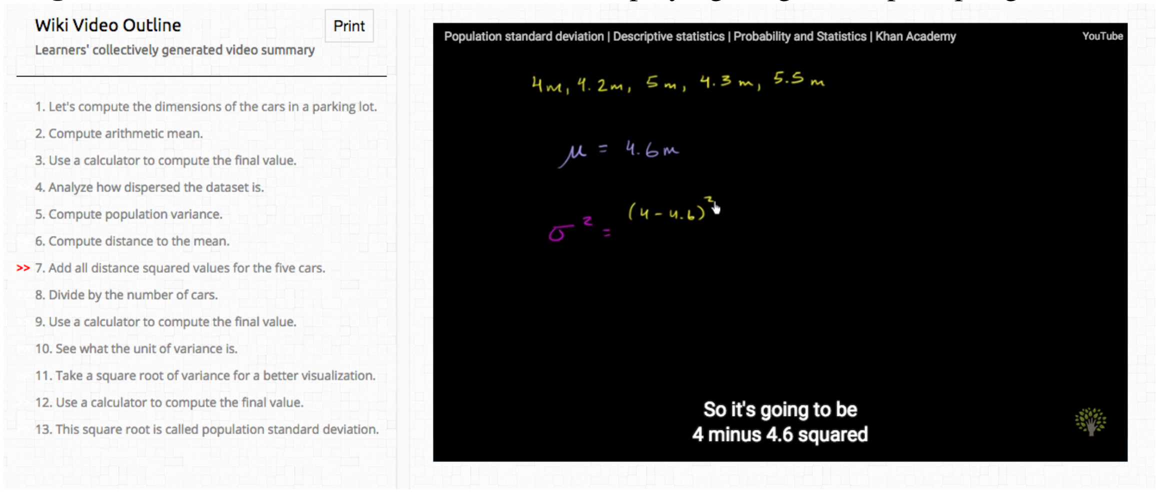


Figure 6-6: The baseline condition without displaying subgoals or prompting learners.



Wiki Video Outline

Learners' collectively generated video summary

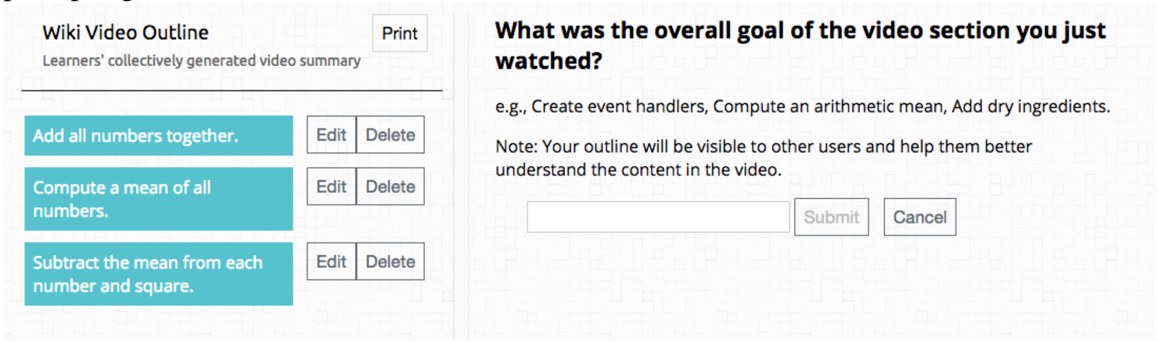
Print

1. Let's compute the dimensions of the cars in a parking lot.
2. Compute arithmetic mean.
3. Use a calculator to compute the final value.
4. Analyze how dispersed the dataset is.
5. Compute population variance.
6. Compute distance to the mean.
7. >> Add all distance squared values for the five cars.
8. Divide by the number of cars.
9. Use a calculator to compute the final value.
10. See what the unit of variance is.
11. Take a square root of variance for a better visualization.
12. Use a calculator to compute the final value.
13. This square root is called population standard deviation.

Population standard deviation | Descriptive statistics | Probability and Statistics | Khan Academy

So it's going to be
4 minus 4.6 squared

Figure 6-7: The expert condition with expert-generated subgoals displayed but without prompting.



Wiki Video Outline

Learners' collectively generated video summary

Print

Add all numbers together. Edit Delete

Compute a mean of all numbers. Edit Delete

Subtract the mean from each number and square. Edit Delete

What was the overall goal of the video section you just watched?

e.g., Create event handlers, Compute an arithmetic mean, Add dry ingredients.

Note: Your outline will be visible to other users and help them better understand the content in the video.

Submit Cancel

Figure 6-8: The Crowdy condition with learner prompts and display of learners' own responses.

Measures of learning used in the study were two-fold. First, a pretest and a posttest were given to participants immediately before and after watching the video. Total of six questions were selected from “A Database of Sample Statistics Quiz Questions for DSc310” by Brian Schott [2], which is based on “Microfiche Copy of the Data Base for the Statistical Test Item Collection System (STICS)” by NSF [3]. The same questions were asked twice, and a learning gain was measured by the difference in the number of correct answers between the pretest and the posttest. The questions used are presented in Appendix C. Second, a retention test was given after three to five days after the initial video watching. It asked if learners were able to remember the solution information after a time interval. The test asked learners to “Explain the process of calculating a population standard deviation, step by step, as detailed as possible.” For assessment, each response was evaluated against the ground truth answer generated by researchers, which consists of six subgoals that should be included in the answer, namely:

1. Compute the arithmetic mean.
2. For each data point, find the distance from the mean.
3. Square the distance values.
4. Add all the squared distances.
5. Divide this number by the number of data points.
6. Square root the answer.

Each response was assessed by the number of attempted subgoals and the number of correct subgoals, extending the subgoal learning evaluation method used in [112, 113]. A researcher decomposed each response into a set of subgoals, or units of information described in a solution. The total number of subgoals in a response is attempted subgoals, and ones that semantically match with any of the six in the ground truth are counted toward correct subgoals. We use these two measures as proxies for learning in the retention test.

variance_std_deviation/v/population-standard-deviation

[2]http://www2.gsu.edu/~dscbms/ibs/ibsd.html

[3]http://www2.gsu.edu/~dscbms/ibs/tests/readme.txt

Additionally, we aimed to see if Crowdy's prompting incurs high task load, which may detract learners from the main learning activity. After watching the video, participants completed the NASA TLX test [63] for measuring task load, in a 7-point scale.

In summary, our hypotheses are as follows:

H1 Retention scores are higher in Crowdy and expert conditions than the baseline.

H2 Immediate test scores are higher in Crowdy and expert conditions than the baseline.

6.6.1 Participants

We initially recruited 300 participants through Amazon's Mechanical Turk. While 300 finished the first round study, due to attrition in the retention test phase, 229 of them completed the entire study. All data reported hereinafter are based on the 229 participants. Participants' mean age was 32.4 (stdev = 9.3; min = 18; max = 67), with 140 male. They received $2 for their participation in the first round, and were invited to participate in the retention test 3 days later via email. Upon completion of the retention test they received another $1 reward. Most participants reported a low level of knowledge in statistics (5-point Likert scale, 1: novice, 5: expert, M = 2.1, stdev = 1.0) and low familiarity with the topics covered in the video (5-point Likert scale, 1: not at all familiar, 5: very much familiar, M = 2.0, stdev = 1.1). For all self-rated prior knowledge questions, there was no significant difference between the interface conditions.

While crowd workers on Mechanical Turk are not voluntary learners on Crowdy, we believe Turkers are a viable population to evaluate learnersourcing with. In a previous study where participants were given critical-thinking problems, Turkers' performance was statistically equivalent to that of workers at a university in the United States [133]. A reason we chose not to run a lab study is because we thought Turkers might better represent self-directed, online learners than lab study participants who are likely to be highly educated and motivated. Another benefit is ease of recruiting a large number of participants, which may better simulate visitors on a public website like Crowdy. To look at how real learners on Crowdy use the site for learning and subgoal labeling, Study 2 reports results from a live deployment.

6.6.2 Procedure

This study used a between-subject design, where each participant was randomly assigned to one of the three interface conditions. For each condition, 100 Turkers were recruited in the initial session.

Initial Session: Participants first completed a pre-questionnaire that asked demographic information and their prior knowledge in statistics. Then the pretest questions were asked, along with a self-rated confidence level for each answer, in a 5-point Likert scale. Upon submitting the pre-questionnaire, participants watched the video using the interface they were assigned. The video player control was disabled to prevent Turkers from skipping the playback. Once the video reached the end, the post-questionnaire was presented, which included posttest questions that were identical to the pretest questions. Additionally, participants completed the NASA TLX questions, which measured task load. Six 7-point Likert scale questions were asked, addressing mental demand, physical demand, temporal demand, performance, effort, and frustration. Task completion time was also measured. When Turkers submitted the post-questionnaire, a confirmation code was displayed, which they had to enter in a form field to get credit.

Retention Test: All Turkers who completed the initial session were invited to participate in the retention study. The invitation email was sent to these Turkers three days after the initial session, and they could complete the retention test for three days, which means that they had between three to five days of time interval between video watching and retention test.

6.6.3 Results

H1: Retention Test

Attempted Subgoals: Participants in the Crowdy (M = 4.77, stdev = 1.67) and expert (M = 4.75, stdev = 1.76) groups submitted more subgoals than the baseline group (M = 4.09, stdev = 1.95), as shown in Figure 6-9. Bartlett's test did not show a violation of homogeneity of variances ($\chi^2(2)= 1.95$, $p = 0.38$). One-way ANOVA reveals that interfaces had a significant effect on the number of attempted subgoals (F(2, 226)=3.6, $p< 0.05$, partial

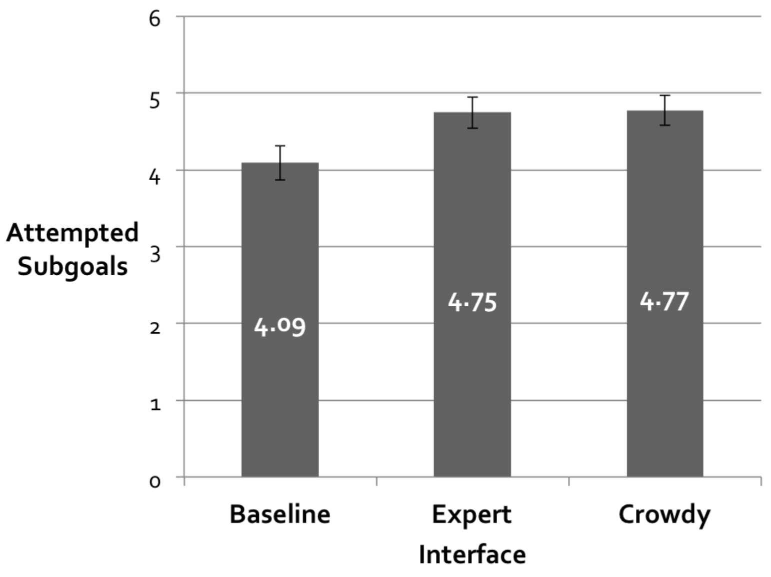


Figure 6-9: The number of attempted subgoals in the retention test. Crowdy = Expert > Baseline. Error bar: standard error.

η^2=0.03). Because the study design included the baseline as a reference condition, our analysis focuses on planned contrasts against the baseline using a Dunnett's test. Dunnett's post-hoc test shows significant differences in both Crowdy-baseline ($p < 0.05$, Cohen's d = 0.38) and expert-baseline ($p < 0.05$, Cohen's d = 0.35) pairwise comparisons.

Correct Subgoals: Participants in the Crowdy (M = 4.39, stdev = 1.85) and expert (M = 4.52, stdev = 1.83) groups submitted more correct subgoals than the baseline group (M = 3.63, stdev = 2.08), as shown in Figure 6-10. Bartlett's test did not show a violation of homogeneity of variances ($\chi^2(2)$= 1.55, p = 0.46). One-way ANOVA reveals that interfaces had a significant effect on the number of correct subgoals (F(2, 226)=4.8, $p < 0.01$, partial η^2=0.04). Dunnett's post-hoc test shows significant differences in both Crowdy-baseline ($p < 0.05$, Cohen's d = 0.38) and expert-baseline ($p < 0.01$, Cohen's d = 0.45) pairwise comparisons. There was no significant difference between the Crowdy and expert groups.

Dropout Rate: Results may be skewed if participants in a particular condition completed the retention test significantly more than those in other conditions. For this reason, we analyzed the dropout rate for each condition. Across all conditions, 76% of the people returned for the retention test (229 out of 300), with 79 in baseline, 75 in expert, and 75

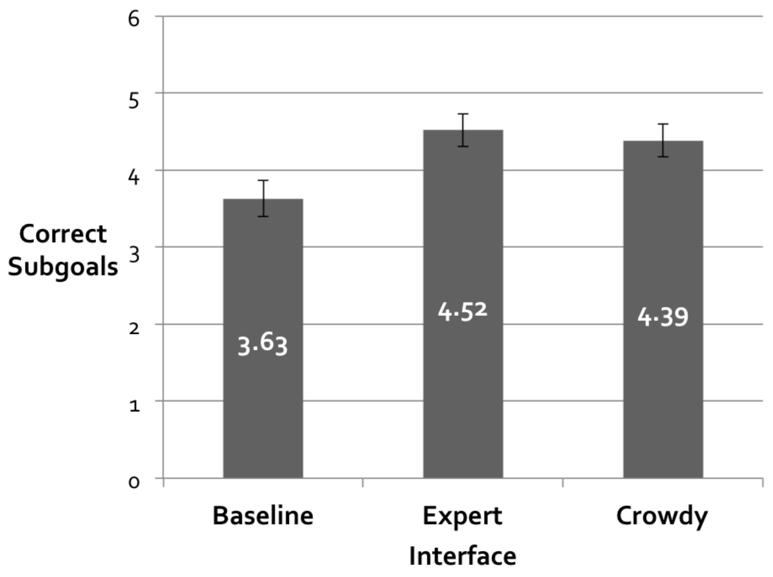


Figure 6-10: The number of correct subgoals in the retention test. Crowdy = Expert > Baseline. Error bar: standard error.

in Crowdy groups. A Chi-square test, in which whether a participant returned for the retention test was coded as a binary variable, revealed that there is no statistically significant difference in the dropout rate between conditions ($\chi^2(2, N = 300) = 0.59$, $p = 0.74$, $\phi = 0.04$).

These results support the hypothesis that with Crowdy, participants were able to reconstruct the solution procedure better than the baseline group in the retention test. This implies that Crowdy provides pedagogical benefits to learners with its prompting mechanism over the baseline video interface. Also, the Crowdy group performed as well as the expert group, which suggests that even in the absence of expert-generated subgoals, learner prompts provide equally effective learning support.

H2: Pretest and Posttest

A learning gain was measured as the pretest score subtracted from the posttest score, which was assessed shortly before and after watching the video in the initial session. Participants in all the groups showed positive learning gains, as shown in Figure 6-11. Paired t tests revealed statistical significance between the pretest and posttest scores (Baseline: t(78)

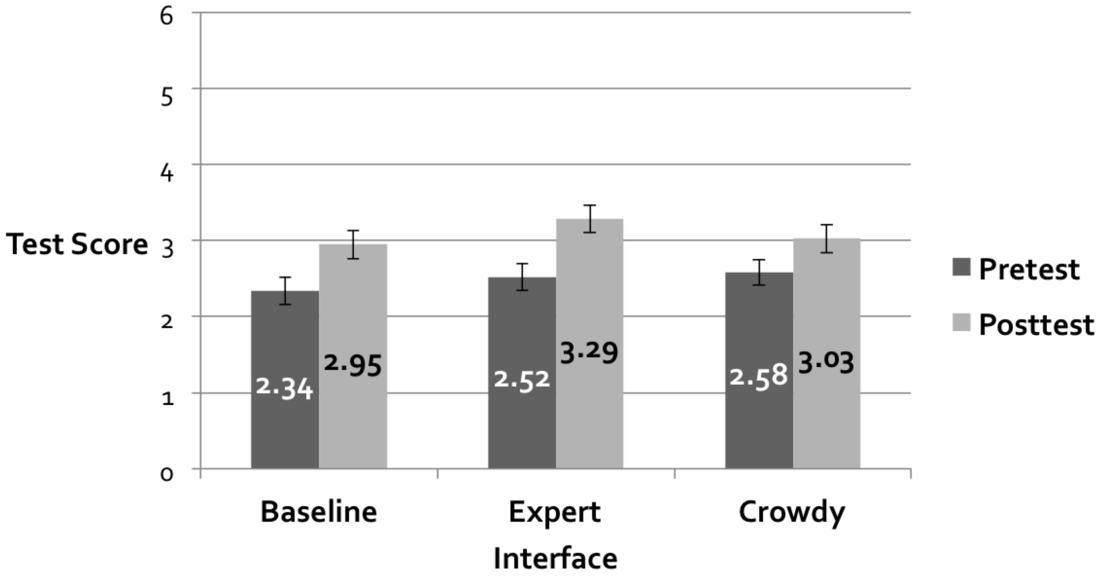


Figure 6-11: The pretest and posttest scores. No significant difference between the interfaces. Error bar: standard error.

= 3.8, $p < 0.005$, Cohen's d = 0.43; Expert: t(74) = 5.9, $p < 0.005$, Cohen's d = 0.69; Crowdy: t(74) = 3.1, $p < 0.005$, Cohen's d = 0.37).

While participants performed better after watching the video in all the groups, there was no significant difference between the groups in terms of learning gains (one-way ANOVA p = 0.31).

The Crowdy group did not outperform the baseline group, although the Crowdy group and the expert group performed equally well. These results do not support H2.

Task Load

The task load score was computed by taking a mean value of responses to the six NASA TLX questions (7-point Likert scale, 1: low workload, 7: high workload). One-way ANOVA revealed that there was no statistically significant difference between the groups (p = 0.20; Baseline: 3.80, stdev=1.06; Expert: 3.77, stdev=0.95; Crowdy: 4.04, stdev=0.94).

Results suggest that answering learnersourcing prompts did not add a significant workload to participants.

Task Completion Time

Task completion includes time taken to complete the pre-questionnaire, video playback, any learner prompts, and the post-questionnaire in the initial session. Participants took around half an hour to complete a session across groups (M=1762 seconds, stdev = 685). One-way ANOVA revealed that there was no statistically significant difference between the groups (p = 0.06), although the baseline group (M = 1616 seconds, stdev = 615) generally took less time than the expert (M = 1818, stdev = 732) and the Crowdy (M = 1861, stdev = 690) groups.

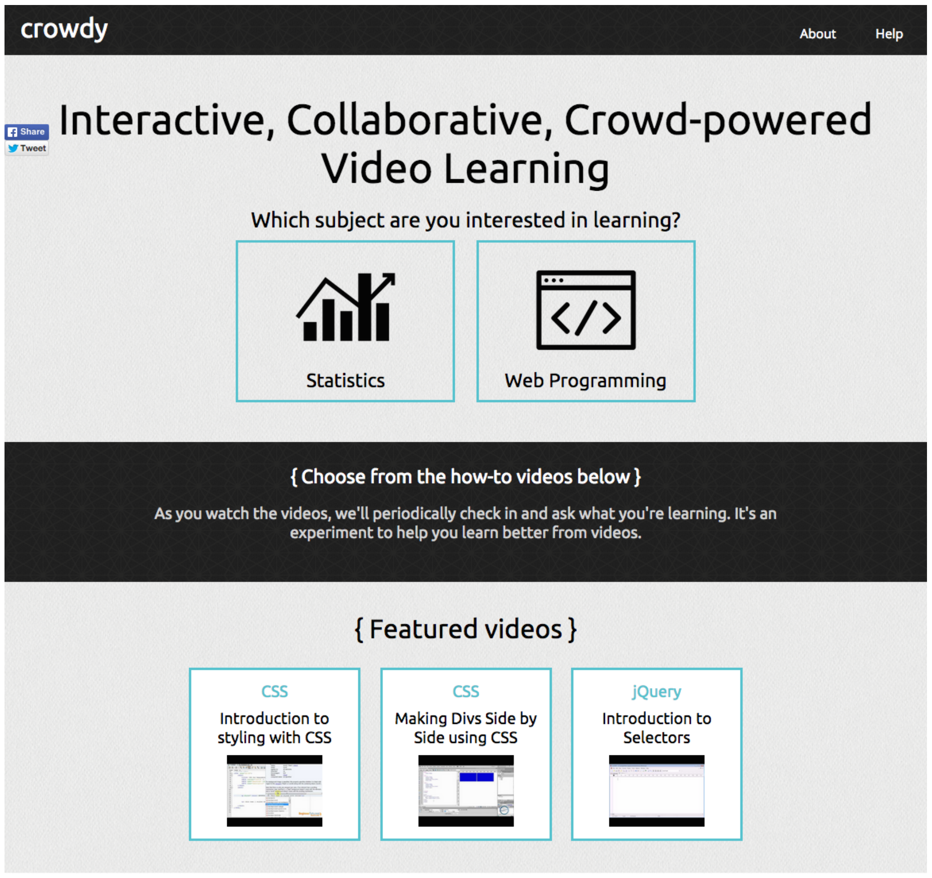
Considering that the video used in the study was 8 minutes long, participants in the expert and Crowdy groups took roughly half the video length of additional time (4 minutes) to complete a session. Because the Crowdy group was prompted 8 times during video watching, each subgoal generation activity took approximately 30 seconds with a rough estimate.

6.7 Study 2: Quality of Learnersourced Subgoals

The goal of this part of evaluation was to test our claim that learners can collaboratively generate subgoals of similar caliber to those generated by domain experts. We chose to deploy the Crowdy system live to test this claim with real users on the web. After improving the workflow based on the pilot study, we publicly launched a website that embeds videos and implements our three-stage workflow, as seen in Figure 6-12.

6.7.1 Study Structure

Our study targeted anyone interested in learning introductory web programming. Crowdy was presented as a way to learn web programming through collaborative video watching. In order for the site to stand alone, we included 15 videos (average duration=6:15, stdev=2:08) covering the basics of HTML, CSS, and JavaScript/jQuery. We refined the original selection of videos from the pilot study to remove less popular ones and added some to increase coverage. A summary of the videos included is presented in Table 6.1. We manually an-



Interactive, Collaborative, Crowd-powered Video Learning

Which subject are you interested in learning?

{ Choose from the how-to videos below }

As you watch the videos, we'll periodically check in and ask what you're learning. It's an experiment to help you learn better from videos.

{ Featured videos }

CSS
Introduction to styling with CSS

CSS
Making Divs Side by Side using CSS

jQuery
Introduction to Selectors

Figure 6-12: The Crowdy website features how-to videos with the learnersourcing workflow embedded.

notated each of the videos to generate a list of individual steps to be shown alongside the video. All of the videos started with zero subgoals, and all contributions were from people visiting the site voluntarily.

6.7.2 Deployment Strategy and Participation

We used various strategies to promote the website. We emailed to lists at universities, posted on social media (Twitter, Facebook, Reddit, Hacker News), reached out to local hacker schools, and bought keywords on Google Ads. Furthermore, we added our link to YouTube comments on the original videos we used to attract learners watching the videos on YouTube to visit Crowdy. In our analysis, we included data from May 5 to May 29, 2014 (25 days).

Video Title	Domain	YouTube Views	Length
Learn HTML- Basic Structure and How to Link Pages	HTML	8,745	4:48
Div and Span	HTML	12,934	6:09
HTML Website Tables & Layouts Tutorial	HTML	210,777	4:19
How to create forms in HTML5	HTML	9,983	9:23
Introduction to styling with CSS	CSS	3,150	6:55
Adding Style Rules Using Google Chrome Developer Tools	CSS	205	4:47
CSS Absolute and Relative Positioning Tutorial	CSS	34,113	6:56
Making Divs Side by Side using CSS	CSS	17,644	4:24
Z-Index CSS Tutorial	CSS	8,158	7:33
Introduction to Selectors	jQuery/JavaScript	40,825	2:06
ID selector	jQuery/JavaScript	36,795	5:53
addClass	jQuery/JavaScript	13,871	4:46
Click Event Handler	jQuery/JavaScript	11,234	6:56
Working on the Drag and Drop Program	jQuery/JavaScript	41,615	9:28
Finishing the Drag and Drop Program	jQuery/JavaScript	38,145	9:16

Table 6.1: How-to videos used in our live deployment. We evaluated the quality of learnersourced subgoal labels for the four bolded videos because they were the most popular videos on Crowdy.

During the 25-day deployment, 1,268 users visited 2,838 pages, with an average session duration of 1.3 minutes. The top 4 traffic sources were direct access (41.31%, including mailing list clicks and typing in the URL), Twitter (23.81%), Google organic search (12.39%, which are clicks on search results, not advertisements), and Facebook (8.84%). All traffic data was collected with Google Analytics.

6.7.3 Evaluation Method

We chose the four most popular videos on Crowdy (boldface in Table 6.1) HTML Intro, Intro to CSS, Making divs side by side, and Intro to selectors) to evaluate the quality of learnersourced subgoal labels. These videos resulted in 17 final subgoal labels.

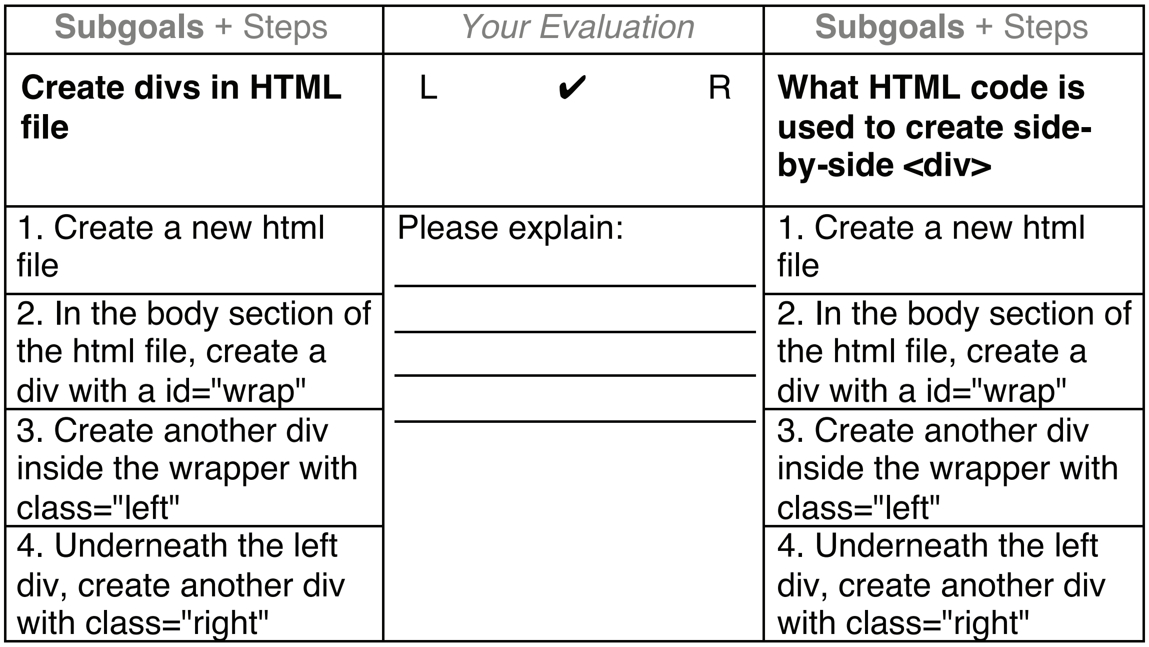
To judge the quality of the learner-generated subgoals, we compared them to expert-quality subgoals. We recruited two faculty and staff members in computer science at a university, who have extensive experience in web programming, to be our domain experts. After explaining what subgoals are and providing examples, the domain experts watched the four videos that we are evaluating and generated subgoals using our system. Because we only allowed learners to generate subgoals at predetermined intervals, we asked the domain

Expert Subgoal	Evaluation	Learner Subgoal
Learn HTML- Basic Structure and How to Link Pages		
Create top-level structure of an HTML file	Expert better	The video stopped right as it was starting to talk about the head section
View HTML page in a Web browser	Expert better	Save and open a HTML document in the browser
Add a hyperlink	Learner better	Create link to another page
Create target page of the hyperlink	Match	Create another page to link to and from
View and navigate between both pages	Expert better	Create HTML pages and links
Introduction to styling with CSS		
Understand what CSS is	No majority	Introducing CSS, comparison to HTML
Two kinds of CSS styling	Match	Inline styling, and difference between this and stylesheets
Set text color of a paragraph	Match	Using the style and color attributes of inline styling
Set background color of a paragraph	Expert better	Problems with inline styling, solutions
Create an external stylesheet	Match	Create a stylesheet to set paragraph colors
Linking stylesheet to HTML file	Match	Linking CSS documents to HTML
Making Divs Side by Side using CSS		
Create divs in HTML file	Match	What HTML code is used to create side-by-side <div>
Create CSS for wrapper and left div	Match	Define div appearance with style blocks
Create CSS for right div	Expert better	Style the left and right div, color, height, width
Position divs side by side	Match	Put the two divs side by side and adjust spacing between them
Introduction to Selectors		
Create HTML and JavaScript files	Learner better	Selector introduction
See example of a jQuery ID selector	No majority	Using selectors and jQuery to create event handlers for page objects

Table 6.2: Expert labels compared to Learner subgoals for the four videos selected for evaluation. For each pair, we identify whether a majority of evaluators felt the subgoals matched or that one was better than the other. In some cases there was no clear majority between evaluators, which we have also noted.

experts to generate labels using the same intervals for direct comparison. In the future, we plan to evaluate subgoals at the entire video level without the set interval. Table 6.2 compares all the subgoal labels from both the experts and learners. For learner-generated subgoals, we took the subgoals that were in stage 3 or most popular in stage 2 for the four videos we selected. At the end of our deployment, five of the subgoals were in stage 3, but most had a clear majority.

To measure how the learner-generated subgoals compare to the expert-generated subgoals, we created a worksheet (Figure 6-13) to have another group of domain experts rate whether the subgoals matched or which one was better, and why. We randomized which side of the worksheet each of the subgoals was on, so evaluators did not know which were generated by learners and which were by experts. They were instructed to rate based on how well the subgoals covered the given steps, and whether they were concrete and descriptive. We recruited four domain experts who were graduate students in computer science with substantial web programming experience to rate 17 subgoals in the four videos.



Subgoals + Steps	*Your Evaluation*	**Subgoals** + Steps
Create divs in HTML file	L ✔ R	**What HTML code is used to create side-by-side <div>**
1. Create a new html file	Please explain: ____ ____ ____ ____	1. Create a new html file
2. In the body section of the html file, create a div with a id="wrap"		2. In the body section of the html file, create a div with a id="wrap"
3. Create another div inside the wrapper with class="left"		3. Create another div inside the wrapper with class="left"
4. Underneath the left div, create another div with class="right"		4. Underneath the left div, create another div with class="right"

Figure 6-13: An excerpt from the worksheet given to domain experts to evaluate learner-generated subgoals. The subgoals were randomly placed on the left and right so experts had no notion of which was which. Evaluators were asked to choose whether the subgoals matched, or which was better, and provide an explanation.

We used a scoring scheme to better compare the subgoal labels. Each evaluator's rating is assigned either -1 (expert label is better), 0 (expert and learner label match), or 1 (learner label is better), which match the three options the evaluators were given for each subgoal. The score of -4 means all raters thought the expert label was better, while 4 means all raters thought the learner label was better.

6.7.4 Results

During the 25 day deployment, learners generated 109 subgoals in stage 1, added 14 new subgoals in stage 2, and edited 3 subgoals in stage 3. There were 109 upvotes in stage 2, and 13 upvotes in stage 3.

Table 6.2 shows all 17 labels generated by experts and learners. For 14 out of 17 labels, the learner label was voted as matching or better than the expert label from at least one of the four raters. When we used majority voting to determine a winner in the comparative

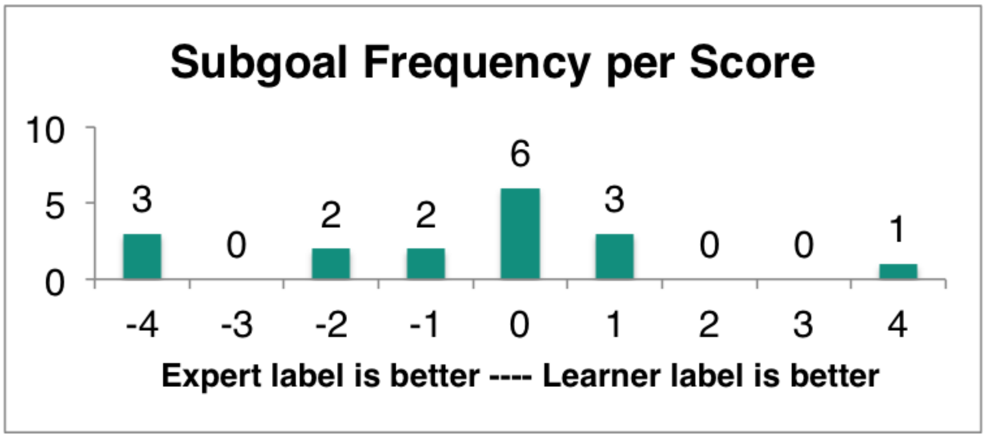


Figure 6-14: The distribution of subgoal label scores. The scoring scheme assigned -1 if an expert label is better, 1 if a learner label is better, and 0 if they match. While a majority of the subgoals were within the -2 and 2 range, three subgoals had unanimous votes for the expert label, and one subgoal had unanimous votes for the learner label.

evaluation, eight labels were considered matching between the learner label and the expert label, five were voted in favor of the expert label, two were voted in favor of the learner label, one was a tie between matching and in favor of the expert label, and one was a tie between favoring the expert and learner labels. Figure 6-14 shows the distribution of scores using the scoring scheme. Due to the highly subjective nature of the matching task, inter-rater reliability (Fleiss' $\kappa = 0.19$) showed a slight level of agreement [102].

Participation

Based on our logging, we found that 119 users (unique sessions) participated in a single video, 17 participated in two, and 14 participated in three or more. Only 26% of users participated in multiple videos.

We now discuss interesting cases in detail: labels for which the expert label was unanimously voted as better, labels for which the learner label got worse in stage 3, and labels that were voted to be matching.

Example of subgoals where expert's is better

Scoping of a subgoal label matters. Most evaluators judged the subgoals on whether they were accurate representations of the given steps. A common complaint was over subgoals that were too specific, or ones that did not seem to capture the steps at all. Evaluators tended to prefer subgoals that were informative instead of those that captured only generic statements. For example, evaluators preferred the expert subgoal "View and navigate between both pages" over the learner subgoal "Create HTML pages and links" for a subgoal in the introduction to HTML video because the learner subgoal was too broad.

Expert subgoal: View HTML page in a web browser

Learner subgoal: Save and open a HTML document in the browser

Steps:

1. Save file with .html extension
2. Open file in web browser to view

Evaluators felt the expert subgoal was more holistic, whereas the learner subgoal simply enumerated the steps that were covered. This subgoal was added in stage 1, and was chosen over other answers such as "Save and open HTML", "HTML format", "describe page tags", and "Implementation of an HTML page". Out of these choices, "Save and open a HTML document in the browser" is the most accurate and the most specific. However, as a subgoal it is not ideal because it exactly replicates the steps. This subgoal correctly summarizes the steps, as asked in stages 1 and 2, but could have been altered to be less specific in stage 3.

This suggests a lack of instruction on our part in stage 3 where learners are asked to make sure the subgoals cover the provided steps, but are not instructed to make subgoals neither too broad nor too specific. This sort of instruction could be added to stage 3, or we could implement multiple versions of stage 3, where we ask learners about different aspects of the subgoal.

Example where final subgoal got worse in stage 3

Expert subgoal: Create top-level structure of an HTML file

Learner subgoal (most popular subgoal after stage 2): Basic structure of HTML template

Learner subgoal (subgoal altered in stage 3): The video stopped right as it was starting to talk about the head section

In this example, a learner who saw a stage 3 question for this subgoal changed the subgoal to something of much lower quality. The final subgoal in this case does not summarize the video section. Although learners only rarely made a drastic replacement to a subgoal in stage 3, we need to consider mediation methods to prevent this from happening in the future. One possibility is to accumulate votes throughout the stages and only allow minor changes to subgoals that already have a lot of votes.

Example of subgoals that match

Expert subgoal: Set text color of a paragraph

Learner subgoal: Using the style and color attributes of inline styling

Steps:

1. Type 'color' inside the style attribute to set the color of the paragraph
2. Assign the hexadecimal value #FFFF00 to make the paragraph yellow
3. View the file in the browser to see the yellow paragraph
4. Type a semicolon after the property value to separate the properties

These subgoals are slightly different, yet the majority of evaluators felt they accurately covered the steps. This suggests that there can be variability in subgoals. Learners and experts (or people in general) can have different interpretations on what the 'correct' subgoal is. Further study will need to see if learner-generated subgoals, even if imperfect, still lead to improved learning gains as observed in literature.

6.7.5 Interviews with Users

To better understand learners' experiences, we conducted semi-structured interviews with three learners who have used Crowdy. Learners identified themselves to us via email after using our system to ask a question or express their interest in the site. They had various programming experiences, but all were web programming novices. We conducted 30-minute video interviews where we asked them about their experience on the site and tried to learn about their thought process while they used Crowdy.

Learners in general felt that Crowdy's prompts while watching a video helped them pay attention, and one specifically noted that answering the questions helped him remember the concepts. This suggests that the learnersourcing prompts may have had a positive effect on learning. Learners also liked being able to reference the video's steps as they watched. According to one learner, *"having the steps there and knowing that I was going to have to fill out goals made me pay attention to the video slightly differently."* In contrast, he felt that if he had watched the video with no steps or questions, he would have mindlessly followed along without learning anything. Another learner commented, *"When I first watched the first video I didn't really know what to expect, but in the second and third videos I was more... attentive to watching, to trying to understand what exactly am I watching instead of blindly watching."* Learners also recognized that the quality of the underlying video affected how well they learned the material, even if the prompts helped them engage with the material.

Learners had different critical approaches to the multiple choice questions in stage 2. One learner never contributed additional subgoals in stage 2 because he felt *"the ones that were there were pretty descriptive already."* Another added his own answers instead of choosing an existing subgoal about 50 percent of the time. He said, *"I would choose one if it was close to what I would have written... I think I was pretty critical, actually."* Learners liked seeing labels added by other learners. One learner said the choices *"...made me feel as though I was on the same wavelength still."*

The set interval for the learnersourcing prompts was hit or miss. For some videos it seemed okay, but for others it felt too short and others too long. Some learners felt the questions were asked at suboptimal times, especially for the shorter videos. One mentioned *"on a really short video, it was a little harder to see the goals,"* and another said, *"The introduction videos didn't cover very much content, so they could use a longer span, while something that covers more information would need to stop more frequently."*

6.7.6 Interview with Video Creator

We also interviewed a creator of one of the videos to get the author's perspective on the value of having subgoal labels added to the video. We contacted some of the authors through YouTube messaging, and one author volunteered to share his thoughts. We showed him the learner-generated subgoals from his video and asked questions over email. For his video, he felt that most of the subgoals were correct except for the last one, which was too broad in his opinion.

He was happy that the outline and the questions might make learners stay and watch the entire video, or be able to easily find the sections that they want to watch. He commented, *"Having pop up questions means the viewer has to be paying attention."* This means that our system can be beneficial to both video learners and video creators because it has the opportunity to enhance what the author has created with the help of learners watching the video.

6.8 Discussion

We now discuss the strengths and weaknesses of the Crowdy system and its learning approach, with respect to learning benefits, learner motivation, and generalizability.

6.8.1 Crowdy's Video Learning Model

Our two-part evaluation demonstrates that participants learn the material better when they engage in the subgoal generation activity. The contributed subgoals by learners prove to be high quality, which in turn can help future learners learn better.

Crowdy presents a video learning model that combines in-video quizzes and note-taking. In-video quizzes are often presented to assess learners' understanding of the material. Crowdy uses the in-video quiz format to not just give learners a chance to self-explain, but also to use the answers as meaningful artifacts for future learners. On the other hand, note-taking is often an open-ended activity learners engage in whenever and however they want to. Crowdy adds structure to note-taking by presenting learner prompts in specific formats and in pre-determined time intervals. The prompted, structured note-taking enables the system to process, assess, and combine learners' notes, to create meaningful artifacts for future learners.

6.8.2 Pedagogical Benefits of Learnersourcing

Why does Crowdy lead to higher learning gains in retention? The Crowdy group performed better in the retention test than the baseline group. Our results are consistent with the subgoal learning literature, which reports that when subgoal labels were shown to participants, they performed better in problem solving in a retention test than those shown no labels [23]. With the self-explanation prompts, participants may be more likely to learn the subgoals better, because the prompts encourage them to reflect on the material and summarize into higher-level descriptions. This process may promote subgoal learning, which reduces extraneous cognitive load and helps with mental model creation in the learning process [23]. Also, Crowdy's prompts are an active and constructive mode learning, which is shown to yield better learning outcomes than the passive mode of learning [27].

The Crowdy and expert groups performed equally well in the retention test. Most studies in the literature have been using manually labeled subgoals by researchers and experts [23, 112, 113]. Our results contribute a new finding that learners can provide expert quality labels while also benefiting from actively learning the subgoals themselves. This finding suggests that experts' video labeling efforts may be replaced with learners' collective efforts. This has implications for a large number of educational videos online, especially those not produced by companies or schools with high budget. With Crowdy's video learning intervention, the long tail of online videos can benefit from an enhanced learning experience.

Why does Crowdy fail to yield higher learning gains in pretest-posttest? Although we initially hypothesized that Crowdy will also lead to higher learning gains in immediate testing, our results show that Crowdy was as effective as the baseline condition. Some possible explanations are as follows: First, Crowdy's summarization prompts may not have been as useful in solving the conceptual questions used in the pretest and posttest (the questions are presented in Appendix C). This is in contrast to reconstructing the solution procedure asked in the retention test, which more directly matches with the subgoal generation task given while watching the video. The mismatch between the learning activity and the test may also explain why the expert condition did not result in better learning. Second, we used a video from Khan Academy, which is highly engaging and easy to understand. Additional prompts may have less room for improving the learning experience, as the source material already does a good job. Third, we used a relatively short video (eight minutes long), which might have been short enough to keep participants' attention throughout. Finally, Turkers with monetary reward may have been incentivized to pay attention to the material.

Small effect sizes? While the reported effect sizes are in the small to medium range [35], Crowdy's video learning approach is effective for two additional reasons. First of all, Crowdy's intervention was simple and short. It only involved answering simple prompts while watching a video, with no additional workload and small added time. Also, with Crowdy, learners not only learn the material better but also generate subgoal labels that contribute to creating a video outline.

6.8.3 Improving the Workflow for Better Subgoal Labels

Why does Crowdy need all three stages?

Stage 1 is necessary for the system to obtain initial subgoals from scratch. The quality of these initial subgoals largely determines the overall quality of the final subgoal label because future steps mostly focus on refining existing steps, not generating completely new ones. In Figure 6-15, raw subgoals in stage 1 vary in quality. In our deployment, there were power users who submitted a set of high quality subgoals in stage 1, which usually ended up being the most popular all the way through. How can we better guide learners to come up with quality subgoals? The current learner prompt provides an example, but more contextual, domain-specific examples and counterexamples that highlight the desired properties in a subgoal label might help.

Stage 2 is necessary to remove spam and to pick the best label from candidates, but

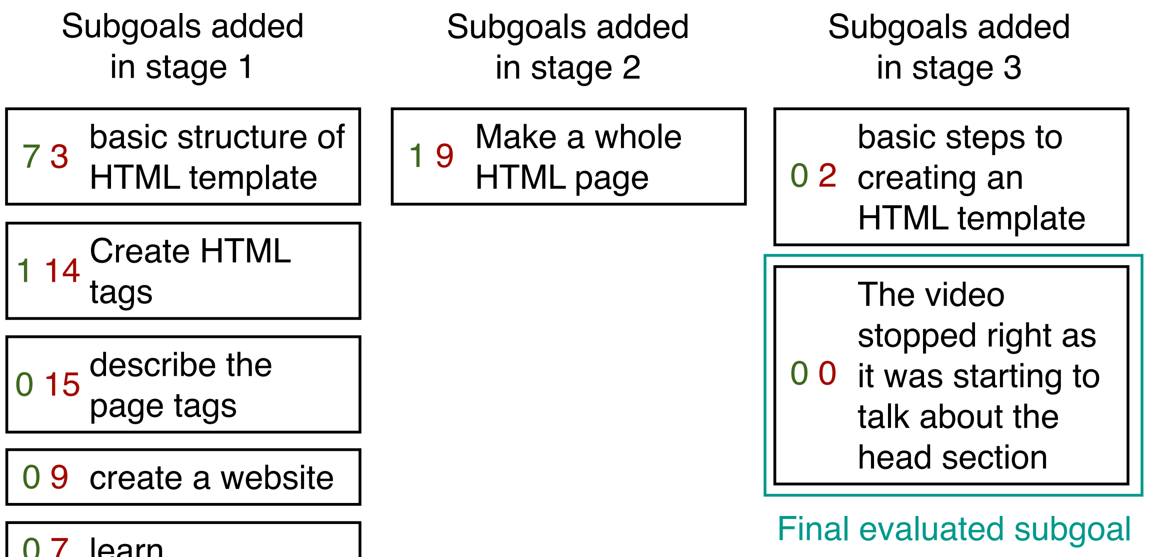

Subgoals added in stage 1
Subgoals added in stage 2
Subgoals added in stage 3
7 3 basic structure of HTML template
1 14 Create HTML tags
0 15 describe the page tags
0 9 create a website
0 7 learn
1 9 Make a whole HTML page
0 2 basic steps to creating an HTML template
0 0 The video stopped right as it was starting to talk about the head section
Final evaluated subgoal

Figure 6-15: The subgoals that were added in each of the stages for the introduction to HTML video. The green numbers refer to upvotes, and the red numbers refer to downvotes. In stage 3, we expected small iterations, but found that one learner drastically changed the tone of the original subgoal.

results suggest that more guidance on which one to pick would improve the voting process. Our current design assumes that one label is going to be significantly better than others, but there is the possibility that they will be close. In such cases, the learner prompt can suggest more specific guidelines by encouraging them to consider scope and language and presenting contrasting good and bad examples. Another potential problem we noticed is that learners are reluctant to submit their own labels in this stage and resort to the best one from the candidates, even when they find the existing ones unsatisfactory. We plan to study the voting process in more depth, and design ways to lower the cost of label revision in stage 2.

Stage 3 is necessary to adjust the scope and language for finalization, but it still triggers learner prompts at a pre-defined subgoal label boundary, preventing learners from looking for video-wide consistency and scoping improvements. Also, in stage 3 we expect learners to only make incremental changes, but a learner who totally missed the point could change a previously 'good' subgoal. This is what happened in Figure 6-15. We are considering mediation methods to prevent later stage input from completely overwriting the earlier

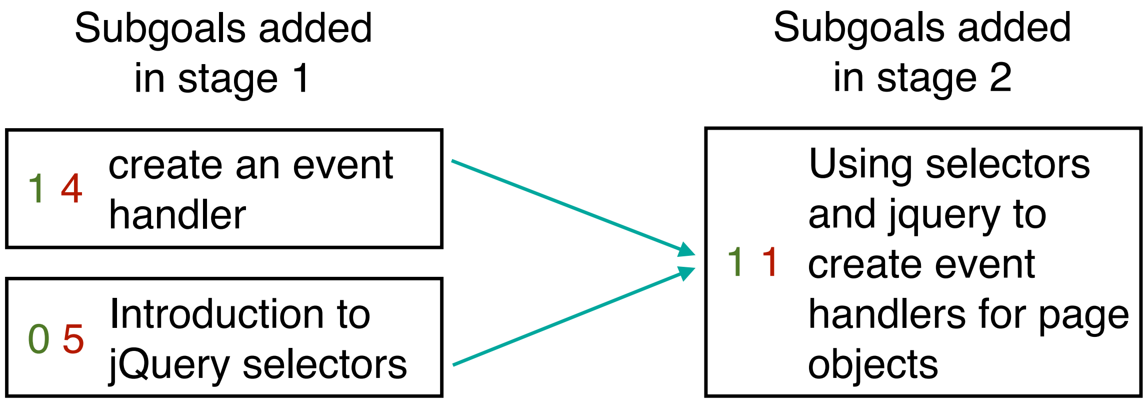

Subgoals added in stage 1
Subgoals added in stage 2
1 4 create an event handler
0 5 Introduction to jQuery selectors
1 1 Using selectors and jquery to create event handlers for page objects

Figure 6-16: The subgoals that were added in each of the stages for one portion of the introduction to jQuery selectors video. The green numbers refer to upvotes, and the red numbers refer to downvotes. The subgoal added in stage 2 is a combination of the subgoals added in stage 1, which suggests that the learner who created this subgoal may have used the stage 1 subgoals as a baseline.

stage input and are looking to automatic methods for evaluating subgoals before they are considered final.

In many cases, we found an improvement in subgoals added as learners progressed through the stages. For example, in the “Introduction to Selectors” video, a stage 2 user combined the subgoals “create an event handler” and “Introduction to jQuery selectors” to create the final subgoal “Using selectors and jquery to create event handlers for page objects” (Figure 6-16). This suggests that even if incomplete, the subgoals in stage 1 are a good baseline for the final selected subgoal.

In general, we found that our learnersourcing workflow succeeded because it allowed many learners to contribute at a micro level, and learners were generally successful at identifying the ‘best’ subgoal in each set. For learnersourcing systems in general, we believe it is important to consider the amount of information given to learners at each task so they are equipped to contribute without being distracted from learning. The type of prompt is also important so learners are contributing as much information as possible and still benefiting from the original educational material.

In the future, we would like to make the question interval more context-specific. Some ideas for informed segmentation are to analyze the audio track for speech breaks, run topic modeling on the transcript to detect topic transitions, or leverage signals from previous learners' in-video interaction [85, 86]. Allowing learners to move subgoals to the correct spot and making sure questions aren't asked in the middle of words could be a preliminary step. Another approach would be to add additional steps to the learnersourcing workflow to allow learners to contribute the subgoal boundaries.

Only 26% of users participated in multiple videos, so we want to look for ways to encourage learners to stay on the site, perhaps by recommending videos or visualizing the learners' collective progress.

6.8.4 Limitations

More domains: Our evaluation covered two procedural task domains: web programming and statistics videos. The videos we chose were how-to videos that had hands-on, concrete, demonstrated tasks through which instructors walked the learners. Although there is nothing in the workflow that is specific to the two domains we used, we don't have any quantitative data about learner-generated subgoals from other domains to show that the workflow is domain-independent. We plan to include videos from other how-to domains such as cooking or home improvement. Expanding the approach to more conceptual and less structured videos, such as lecture videos, is another avenue for improvement.

Longer term, larger deployment: Our 25-day deployment with over 1,000 users was helpful for us to get the initial idea of how our workflow works, but we believe a longer and larger deployment will be helpful. While many people visited the website out of curiosity, not many learners were seriously willing to learn web programming. As a result, not many videos received enough learner input to advance to stage 3. We plan to host more videos and continue the live deployment, which will provide us with more data for a deeper analysis on how learners in the wild use our workflow.

6.9 Conclusion

This chapter introduces Crowdy, an active learnersourcing application for generating subgoal labels from a how-to video. It presents a multi-stage workflow where learners collaboratively contribute useful information about the video while answering reflective summary questions. Our study shows that Crowdy provides learning benefits in a retention test, over a baseline video watching interface. Our deployment study shows that a majority of learner-generated subgoals were comparable in quality to expert-generated subgoals, and that learners reported they were paying more attention to the material and felt they understood the material better after going through the workflow.

Chapter 7

Discussion

This chapter compares learnersourcing against paid crowdsourcing, lays out the design space of passive and active learnersourcing, and presents design lessons for future researchers and practitioners. I will also discuss some of the limitations of learnersourcing and the approaches introduced in the thesis.

7.1 Learnersourcing vs Crowdsourcing

What makes learnersourcing different from crowdsourcing? Knowing the differences helps illustrate what learnersourcing is and what aspects designers should consider when designing learnersourcing applications. This section compares learnersourcing against generic crowdsourcing, specifically paid, microtask crowdsourcing commonly found on platforms such as Mechanical Turk.

7.1.1 Purpose

From a requester's point of view, while paid crowdsourcing mainly aims to collect desired data, learnersourcing attempts to help learners learn better while collecting desired data. This difference may constrain the range of tasks amenable to learnersourcing, because both goals should be met in a single learnersourcing task.

Design Dimension	**Learnersourcing**	**Crowdsourcing**
Purpose	learning + data collection	data collection
Task property	pedagogically meaningful and engaging	simple and standalone
Task context	task given while learning	task given because the worker decided to work for hire
Design challenge	incentive design	quality control
Goal visibility	overall contribution visible	microscopic, focused design
Cost	learners' time and effort	money

Table 7.1: Comparison between learnersourcing and paid crowdsourcing, along major design dimensions.

7.1.2 Task Property and Context

A fundamental difference in learnersourcing tasks and crowdsourcing tasks is that learnersourcing tasks are embedded in an existing learning material or interface, whereas crowdsourcing tasks are often presented as standalone.

This means that learnersourcing tasks need to be contextual: they should closely align with the underlying material (e.g., a learnersourcing prompt should ask about the video the learner is currently watching). This characteristic in learnersourcing can be both a challenge and an opportunity. It can be challenging to design a pedagogically meaningful and engaging task within the scope of the original material. The quality of the original material may affect participation and data quality in learnersourcing. Also, learners may not be willing to participate if the task seems tangential or unrelated to the learning material. On the other hand, it can be an opportunity as learnersourcing tasks can leverage the original material to their favor. For example, Crowdy presents subgoal generation prompts as if they were in-video quizzes.

In crowdsourcing, requesters often create a self-contained environment in which workers complete tasks. By designing simple, standalone tasks, requesters can have more control in task design. Workers only see the task because they explicitly decided to work for hire.

7.1.3 Design Challenge

The main design challenge in learnersourcing is incentive design, which is about motivating learners to actively participate in the tasks while generating useful human computation artifacts. In active learnersourcing, tasks are often displayed after interrupting the learner, or by embedding additional complexity to the learning interface. These properties make it challenging to design a successful learnersourcing application, because a learnersourcing prompt should meet multiple criteria at the same time: be engaging, not detract learners from the main learning experience, collect useful information from the learner, and help with learning.

Two useful design tips in learnersourcing are: 1) to appeal to learners' self-motivation by clearly communicating that engaging in the prompt can help them learn better, and 2) to appeal to learners' altruism by clearly communicating that their contribution can help other learners.

In paid crowdsourcing, the foremost importance is in collecting clean and consistent data. There are many ways workers might submit low quality work: they might optimize for earning the most money with minimal amounts of work [11] or exhibit spamming behaviors. Quality control mechanisms (e.g., gold standard or majority voting) need to be in place to ensure that workers provide quality input. While important, incentive design in paid crowdsourcing can be controlled with the amount of reward to certain extent.

In learnersourcing, on the other hand, quality control may not be as critical of an issue. Learners are incentivized to watch these videos based on their own desire to learn, not because they will receive payment. Learners may provide higher quality answers than a paid crowd because they are motivated by their interest in the instructional material.

Because tasks are presented only after the learner is engaged in the learning material, there is likely a higher barrier to entry for spammers. But there is still a danger of learners submitting minimal work to return to the main material as soon as possible. Seamlessly integrating the learnersourcing task into the learning experience is a way to address this issue.

7.1.4 Goal Visibility

In paid crowdsourcing, workers often work on a microtask without necessarily knowing how their work is going to be used. Similarly, many human computation applications such as reCAPTCHA [1] and ESP Game [160] do not expose their underlying goal of data collection. Such design helps workers focus on the task at hand without raising complexity of work.

In learnersourcing design, however, it is recommended that the underlying goal is clearly conveyed to learners, so that knowing the goal can further incentivize them to participant in the task. Also, another notable difference is that in learnersourcing, learners themselves are the target population that the human computation outcome benefits. For example, the learner who contributed to generating subgoal labels for a video can benefit from navigating the video with improved subgoal labels when they revisit the video. This design also makes it possible to progressively benefit learners as more data becomes available: as more learners contribute, the quality of the learnersourced artifact will be increased. In this sense, learnersourcing is similar to peer production communities like Wikipedia.

Cost

The cost structure differs significantly between learnersourcing and crowdsourcing. Learnersourcing relies on learners' time and effort, without paying for their work. In learnersourcing design, the cost structure may be harder to quantify than in paid crowdsourcing, which makes it difficult to experiment with various parameters. In paid crowdsourcing, the primary cost is money.

[1]http://www.google.com/recaptcha/

7.2 Design Guidelines for Learnersourcing Applications

7.2.1 Passive Learnersourcing

Passive learnersourcing involves capturing meaningful learning behaviors from interaction data. This is a challenging task because it requires carefully constructing a data processing pipeline, which explains high-level learning behaviors and powers data-driven interaction techniques.

A design process for passive learnersourcing used in this thesis includes the following stages:

1. Consider an ideal situation for learning, and identify what's currently missing (e.g., social recommendation, points of confusion within a clip).
2. Look at the interaction data and decide which signals can generate the needed artifact in Stage 1 (e.g., play button clicks, in-video dropout behavior).
3. Come up with a technical solution to extract the signals in Stage 2 from the actual data (e.g., aggregate play button click times, peak detection, curve smoothing).
4. Build this signal into the online learning user interface. Iterate on the design (e.g., visualize interaction peaks, recommend frequently revisited frames).

During this process, there are many design dimensions a learnersourcing designer has to consider.

- **Interaction data**: Which interaction behavior can you leverage from the learner? (e.g., clicking on the play button, dropping out from a video)
- **Content data**: What kind of information can you leverage from the content? (e.g., visual analysis of a video frame, transcript, instructor audio)
- **Usage scenario**: What kind of instructor's or learner's task do you want to support with this data? (e.g., instructor updating their video, learner reviewing for an example, learner skimming through a video)

- **Personal vs Collective**: What if collective behavior patterns do not match with personal patterns? (e.g., In LectureScape, an important design decision was to display both personal and collective traces because there may be differences between the two. A participant in the lab study encountered a mismatch, and he was able to recognize that the location was not a confusing point for him but he could see why other learners might have been confused.)
- **Feedback**: How should the application explain the meaning of a collective behavior pattern to the user? (e.g., "this is where other learners were frequently confused about")

7.2.2 Active Learnersourcing

Active learnersourcing involves designing an explicit learning activity, with the two goals of collecting useful information and designing pedagogically meaningful activity. This is a challenging task because meeting both goals at the same time requires an understanding of crowdsourcing and instructional design.

A design process for active learnersourcing used in this thesis includes the following stages:

1. Consider an ideal situation for learning, and identify what's currently missing (e.g., video outline, subgoal labels, step-level indices).
2. Identify what an individual learner can contribute to generating the needed artifact in Stage 1 (e.g., a step label, a subgoal label, an upvote).
3. Design an activity so that it is pedagogically beneficial to the learner while generating the expected individual outcome in Stage 2 (e.g., "summarize the section of the video you just watched," "which of the following best summarizes the section of the video you just watched?").
4. Come up with a technical solution to meaningfully aggregate the collective learner contributions in Stage 3 (e.g., multiple stage workflow, string comparison, voting process).

5. Build this activity into the online learning user interface. Iterate on the design (e.g., add a sidebar with subgoals, add an interactive timeline).

During this process, there are many design dimensions a learnersourcing designer has to consider.

- **Motivation**: How should the application incentivize learners in the task? Even if it helps with their learning, learners still might hesitate to participate. How should the application convey the value of the activity?
- **Quality Control**: How should the application prevent the gaming or spamming behavior? In comparison to generic crowdsourcing, spamming or minimal work becomes less of an issue because learners are a self-selected crowd actively trying to learn. A bigger issue is when learners submit incorrect answers or attempt to game the system, especially if individual contributions are visible to other learners or grading is related to the participation in learnersourcing tasks.
- **Data Aggregation**: How should the application process and aggregate learners' contributions? This is especially challenging in learnersourcing as opposed to generic crowdsourcing, because learnersourcing tasks tend to be more subjective and complex than simple crowdsourcing tasks.
- **Interruption**: What is the right amount of interruption that a learner prompt should make? How complex should the task be?
- **Presentation**: How can a learner's contribution be visible to the learner and to the entire learner community? How should the community progress be presented?

7.3 Possible Applications Using Learnersourced Artifacts

7.3.1 Learner Engagement

Chapters 3 and 4 demonstrate how interaction data can be used to infer and improve learner engagement. While the method used in this thesis primarily focused on interaction peaks from data in science and engineering MOOCs, future work can extend this work to analyze more courses, data streams, and interaction patterns. We hope to analyze humanities and professional courses, and compare results against the current data from science and engineering courses. Another potential data stream is text from transcripts, textbooks, and lecture slides. Text analysis can complement vision-based techniques. In contrast to peaks, dips in viewership and interaction counts might be an informative pattern to investigate. Dips might represent boredom and loss of interest.

7.3.2 Video Annotation

Chapters 5 and 6 present methods for extracting steps and subgoals form existing how-to videos. What novel applications can we design once we have the extracted information? First, better video search can be made possible with finer-grained video indices and labels. For example, ingredient search for cooking or tool name search for Photoshop can show all videos and time points that cover a specific tutorial element. Furthermore, video players can present alternative examples to a current step. If a learner is watching how to apply the eyeliner, the interface detect steps and subgoals from other videos that involve the eyeliner, and snow snippets from other videos that include demonstrations of the eyeliner. This allows the learner to hop between different use cases and context for the step of interest, which can potentially improve learning outcomes.

We believe the Find-Verify-Expand workflow presented in Chapter 5 can generalize to annotating broader types of metadata beyond steps from how-to videos. For example, from a soccer video this pattern can extract goal moments with Find and Verify, and then use Expand to include a crucial pass that led to the goal, or a ceremony afterward. Generally, the pattern can extract metadata that is human-detectable but hard to completely automate. It is a scalable method for extracting time-sensitive metadata and annotating streaming data, which can be applied to video, audio, and time-series data.

7.4 Generalizing Learnersourcing

The core idea of learnersourcing is that with an inherently motivated crowd, we can design applications in which small contributions from members of a crowd can create artifacts that directly benefit the crowd. It is possible to expand the definition of learnersourcing to a broader community context. This section attempts to explore how the idea of learnersourcing might generalize to domains beyond education.

I have explored the domains of collaborative planning and civic engagement, in which a community of users perform inherently meaningful tasks while collaborating in the sensemaking and content generation process.

7.4.1 Community-driven conference scheduling

The onus of large-scale event planning often falls on a few organizers, and it is difficult to reflect the diverse needs and desires of community members. The Cobi project engages an entire academic community in planning a large conference. Cobi elicits community members' preferences and constraints, and provides a scheduling tool that empowers organizers to take informed actions toward improving the schedule [90]. Community members' self-motivated interactions and inputs guide the conflict resolution and schedule improvements. Because each group within a community has different information needs, motivations, and interests in the schedule, we designed custom applications for different groups: Frenzy [30] for program committee members to group and label papers sharing a common theme; authorsourcing [90] for paper authors to indicate papers relevant to theirs; and Confer [13] for attendees to bookmark papers of interest. Cobi has scheduled CHI and CSCW, two of the largest conferences in HCI, since 2013. It has successfully resolved conflicts and incorporated preferences in the schedule, with input from hundreds of committee members, and thousands of authors and attendees.

7.4.2 Budgetary discussion support for taxpayers

A lot of community practice and civic issues matter to members, but the decision process and complicated structure are often not accessible to the members. The conceptual idea behind learnersourcing presents a technique for engaging community members in the process, while members engage in natural or voluntary or intrinsically motivated activities. The extensiveness and complexity of a government budget hinder taxpayers from understanding budgetary information and participating in deliberation. In an interdisciplinary team, I helped build interactive and collaborative web platforms in which users can contribute to the overall pulse of the public's understanding and sentiment about budgetary issues. Factful [87] is an annotative news reading application that enhances the article with fact-checking support and contextual budgetary information. Users' fact-checking requests and results are accumulated to help future users engage in fact-oriented discussions. BudgetMap [91] allows users to navigate the government budget with social issues of their interest. Users can make links between government programs and social issues by tagging. In our 5-day live deployment, more than 1,600 users visited the site and made 697 links between social issues and budget programs.

Both examples presented above follow the learnersourcing model in that they create a virtuous feedback loop in which users engage in microtasks to contribute to generating community artifacts. In Cobi, paper authors provide useful information for scheduling, while they learn about relevant papers in the community. In BudgetMap, taxpayers provide useful tagging information for budget items, while they learn about the budget with improved awareness. In addition to the domains discussed above, the conceptual idea in learnersourcing can generalize to various social domains such as open government, nutrition, healthcare, and accessibility, just to name a few. In these domains, we can design novel community-driven workflows and interfaces to support planning, discussion, decision making, and creative processes.

7.5 Limitations of Learnersourcing

This section offers reflections on the limitations of learnersourcing.

7.5.1 Complexity of Tasks and Outputs

Active learnersourcing tasks inherently interrupt learners to provide input to the system. When carefully designed, these tasks can lead to an improved learning experience, but there is a danger of distracting learners from their main task of watching the video. This constraint puts an upper bound in the amount of interruption that can be made in active learnersourcing tasks. A tradeoff involved in designing an active learnersourcing task is between the amount of interruption in prompting and learner motivation. There is also a related tradeoff between the amount of useful information collected from each learner and learner motivation. As the task gets complicated and takes longer, the learner is less likely to pick up the task. A general rule of thumb is as follows: the more aligned the task is with the main learning material, the more likely that the learner will be engaged in the task.

7.5.2 Data Blindness

Passive learnersourcing relies on massive amounts of learning interaction data, such as clickstream data from the video player. Because video interaction logs only reflect click behaviors, they are not a perfect proxy for learners' true engagement and learning. Even if significant trends are identified, drawing causality beyond correlation may be extremely difficult.

This is a tradeoff that needs to be considered: while data in passive learnersourcing can be relatively easily collected with no interruption, its explanatory power is weaker than data from learners' explicit input or observations made in person. A potential solution is to combine passive and active learnersourcing in a single application, in a way that the two models complement each other.

7.5.3 Privacy

Any interaction design methodology that feeds users' data back into the design should be careful about privacy. In deploying the data-driven video techniques, it would be essential to inform users of the exact data that is being collected, and present what benefits they can expect by opting-in to contribute their data. This is perhaps not a severe problem in

LectureScape, because it only collects anonymized, aggregated usage data. But as future learnersourcing application designers wish to provide more personalized support, privacy concerns should be carefully addressed in the design stage.

7.5.4 Is This a Temporary Solution?

As artificial intelligence advances, it may be possible to generate many of the results presented in the thesis automatically. For instance, what if there is a powerful computer vision and natural language processing technique for accurately summarizing a video? Does that make learnersourcing obsolete? Is learnersourcing a temporary solution until automated solutions become feasible?

I would argue that learnersourcing is still valuable even with the presence of powerful automated solutions. The value of learnersourcing comes from the participation aspect: engaging in learnersourcing directly rewards the participant with improved learning, sense of community, and better navigation experience. For this reason, in Crowdy, we still prompt learners with the learnersourcing questions even after the video is fully labeled with subgoals. Indeed, automated solutions can be used in combination with learnersourcing in several ways. First, learnersourcing can provide high quality training data for machine learning methods. Second, automated solutions can be used to grade or provide feedback to learners as they complete their learnersourcing tasks. Finally, automated solutions can be used to generate adaptive and personalized learnersourcing prompts.

Chapter 8

Conclusion

This dissertation introduced learnersourcing: a novel crowdsourcing model in which users' lightweight contributions improve content and interfaces for future learners. To conclude, this chapter will summarize the main contributions of the thesis and propose important directions for future research.

8.1 Summary of Contributions

Broadly, this thesis makes contributions in crowdsourcing, learning at scale, and video interaction.

- **Crowdsourcing**: novel workflows for extracting semantic and complex information from videos; and decomposition patterns that are designed to provide pedagogical benefits to crowd workers engaged in the task.

- **Learning at scale**: a virtuous feedback loop in which learners' improved learning experience in turn generates useful artifacts for future learners; MOOC-scale data analyses of click-level video interaction; and novel systems for improving the status quo in video learning at scale, with improved interactivity and sense of community.

- **Video interaction**: video interaction techniques powered by user-generated data; and video interfaces that dynamically evolve over time as more users interact with the content.

8.2 Future Directions

Learnersourcing can serve as an enabling technology for the next-generation learning platforms and learning at scale research. Further, this conceptual idea can be applied to broader societal context to lower the barrier to individuals' participation and impact within a community, by engaging them in activities meaningful to both themselves and the community.

8.2.1 Innovative Learnersourcing Applications

I envision more advanced learnersourcing applications that enable more constructive and interactive video learning experience at scale. While the active learnersourcing examples in this thesis used how-to videos covering procedural tasks, a similar workflow can be applied to lecture videos on MOOCs that are much less structured than how-to videos. Collaborative summarization on lecture videos might be a useful exercise for learners that deeply engage them in the learning process. Further, learnersourcing workflows can be designed for synchronous, direct interaction between learners, which can result in more complex and creative outcomes. For example, learners who are on the same video page can be given an opportunity to watch the video together and work on creating an outline together.

Another promising future direction is to push the boundaries of learnersourcing by broadening the application scope and domain. For instance, learners can be asked to submit their own version of explanation as a video clip. This gives learners a chance to self-explain, while the collected explanations can enable multiple learning paths and personalization for future learners. A technical challenge will be in interpreting and labeling multimedia inputs from learners. Another avenue for future research is supporting diverse problem domains: a programming environment that learnersources code snippets and documentation; a writing tool that learnersources alternate expressions and phrases for inspiration and learning; and a graphical design tool that learnersources visual assets other learners can extend.

8.2.2 Learning Platforms of the Future

While existing online learning platforms have provided access to more learners, the learning experience is still limited to passively watching videos and answering canned questions. I envision educational technologies that truly scale: a course that is created, organized, and taught entirely by learners. Learners will actively engage in 1) creating various artifacts including quizzes, explanations, and examples, 2) providing feedback and iterating on the artifacts, and 3) labeling the artifacts with metadata for better search and organization. With a learnersourced course that builds itself, learners are at the center of both the instructional design and learning experience.

8.2.3 Science of Learning at Scale

Despite the rapid growth, we do not yet have good answers to whether, what, how, and why people learn in massive learning platforms. The field needs to develop theories, instructional guidelines, and design principles for learning at scale settings. This research direction calls for formal and data-driven frameworks for conducting controlled experiments and adaptively modifying the learning interface. The frameworks can help researchers experiment with various learnersourcing models in vivo, with different video formats, interaction patterns, learner prompts, and adaptation techniques. These efforts will spur the design of new instructional formats and technologies, which in turn can advance the emerging science of learning at scale.

8.2.4 Sociotechnical Application Design

Learnersourcing is a novel application of crowdsourcing to lower the barrier to participation for individuals within a community. I believe this framework has potential for broader societal impact. Novel coordination and incentive mechanisms can be designed for broader domains including civic engagement, healthcare, and accessibility. We need to investigate generalizable design principles and computational methods applicable across multiple domains. Furthermore, the idea of learnersourcing can be extended to capture community interactions beyond clickstream and answers to given prompts. How can we capture discussion, collaboration, decision making, and creative processes by individuals and groups at scale? Learnersourcing presents a promising vision for involving members of a community in collectively creating artifacts and value beneficial to the community.

Appendix A

Materials for the LectureScape Study

This chapter presents the study tasks and questionnaires used in a laboratory study described in Chapter 4.

The tasks and their instruction given to participants are presented below. Part 1 is **Visual search** tasks, which asked participants to find a specific piece of visual content in a video. Part 2 is **Problem search** tasks, which asked participants to find an answer to a given problem. Part 3 is **Summarization** tasks, which asked participants to write down the main points of a video while skimming through it. For each task type, we counterbalanced the order of the interfaces and the assignment of videos to tasks.

A.1 Study Task Part 1.

Instruction: You might remember something from a video you already watched, and want to go back to that part for review. In this task, your goal is to find a point in the video that contains the given information.

A.1.1 Task A.

This video is about branching programs. From this video, find a slide where the instructor displays on screen what "elif" in Python stands for. Pause the video where the answer is visible.

Video link: https://www.youtube.com/watch?v=FMGal3lXcjw

A.1.2 Task B.

This video is about tuples. From this video, find a slide where the instructor displays on screen examples of the singleton operation. Pause the video where the answer is visible.

Video link: https://www.youtube.com/watch?v=d-SBFpxf8Bk

A.1.3 Task C.

This video is about floating point accuracy. From this video, find a slide where the instructor displays on screen the equation abs(x-y) ¡ 0.0001. Pause the video where the answer is visible.

Video link: https://www.youtube.com/watch?v=jq7Sujh5uDA

A.1.4 Task D.

This video is about understanding root finding. From this video, find a slide where the instructor displays on screen the abstraction property of functions. Pause the video where the answer is visible.

Video link: https://www.youtube.com/watch?v=mylsICZfBpo

A.2 Study Task Part 2.

Instruction: Let's assume that a fellow learner posted a question on the discussion forum. You want to post an answer to the question by watching a video on the topic. Your goal is to find a point in the video that contains an answer to the problem. You do not need to provide related context or background information for this task. A final answer will suffice.

A.2.1 Task A.

This video is about approximation methods. Approximation methods generate a number of different guesses that differ by a small amount (step). They help reach an answer that is close enough to the exact answer.

You want to answer the following question: “If the step size in an approximation method decreases, does the code run faster or slower?” Please find a part in the video that discusses the relationship between the step size and the amount of code execution required to find a solution.

Note that you need to both 1) find the part in the video that contains the answer, and 2) provide your answer.

Video link: `https://www.youtube.com/watch?v=pGd3WqZK4Cg`

A.2.2 Task B.

This video is about bisection search. Bisection search works by cutting the size of the range of values to search in half. It reduces the computation time by making smart guesses.

You want to answer the following question: “Would bisection search work well if the value we’re estimating grows as the input value grows?” Please find a part in the video that discusses the relationship between the function ordering and the feasibility of using bisection search.

Note that you need to both 1) find the part in the video that contains the answer, and 2) provide your answer.

Video link: `https://www.youtube.com/watch?v=Zoy7t4LbAPY`

A.3 Study Task Part 3.

Instruction: For this video, you’re only given three minutes to skim through it. While you skim, please summarize the main points of the video for other learners to review later. The researcher will give you 1-minute and 30-second warnings. Please write down your summary points below. Note that your summary should be in phrases, not just keywords.

- (O) “iterative algorithms capture states that update on each loop.”
- (X) “iterative algorithms”

A.3.1 Task 7. (your answer below)

Video link: `https://www.youtube.com/watch?v=qic9_yRWj5U`

A.3.2 Task 8. (your answer below)

Video link: `https://www.youtube.com/watch?v=lTnTlmM33dA`

A.4 Post-Task Questionnaire

The following questionnaire was given to participants after each task.

- What is your participant ID?
- What is the ID of the task that you just completed?
- How confident are you in your answer? (1: Not really, 7: Very much)
- Have you watched this video before?
  - Yes, multiple times.
  - Yes, once.
  - Yes, but I skimmed.
  - No.
  - I can’t remember.
- If you answered ”Yes” to the previous question, how much did you remember from the video before doing this task? (1: Nothing, 7: Everything)

A.4.1 Workload Assessment

The following assessment is used to measure your personal opinion on how much workload was required of you during the task you just completed.

Mental Demand

How mentally demanding was the task? (1: Very low, 7: Very high)

Physical Demand

How physically demanding was the task? (1: Very low, 7: Very high)

Temporal Demand

How hurried or rushed was the pace of the task? (1: Very low, 7: Very high)

Performance

How successful were you in accomplishing what you were asked to do? (1: Perfect, 7: Failure)

Effort

How hard did you have to work to accomplish your level of performance? (1: Very low, 7: Very high)

Frustration

How insecure, discouraged, irritated, stressed, and annoyed were you? (1: Very low, 7: Very high)

A.5 Post-Study Questionnaire

The following questionnaires were given to participants after all main tasks.

A.5.1 Questionnaire for the baseline interface

During the study, the baseline interface was simply referred to as Interface #1. All questions are asked in a 7-point Likert scale, with 1 indicating *strongly disagree* and 7 indicating *strongly agree*.

Instruction: Refer to the print out of an interface screenshot that says ”Interface #1”.

This questionnaire gives you an opportunity to tell us your reactions to the interface. Your responses will help us understand what aspects of the interface you are particularly concerned about and the aspects that satisfy you. To as great a degree as possible, think about all the tasks that you have done with the interface while you answer these questions. Please read each statement and indicate how strongly you agree or disagree with the statement.

1. Overall, I am satisfied with how easy it is to use this system.
2. It was simple to use this system.
3. I can effectively complete my work using this system.
4. I am able to complete my work quickly using this system.
5. I am able to efficiently complete my work using this system.
6. I feel comfortable using this system.
7. It was easy to learn to use this system.
8. I believe I became productive quickly using this system.
9. The interface of this system is pleasant.
10. I like using the interface of this system.
11. This system has all the functions and capabilities I expect it to have.
12. Overall, I am satisfied with this system.

For watching online lecture videos generally, not just for the study tasks today, how useful do you think the various features are in this interface?

Please refer to the printed interface screenshot to see what each feature is.

1. Timeline display
2. Timeline clicking and dragging

3. Thumbnail preview
4. Play / Pause button
5. Time display
6. Volume control
7. Transcript text
8. Interactive transcript (automatic scrolling and highlighting current line)
9. Keyword search

Please leave any comments about this interface. (strengths and weaknesses)

A.5.2 Questionnaire for LectureScape

During the study, the LectureScape interface was simply referred to as Interface #2. All questions are asked in a 7-point Likert scale, with 1 indicating *strongly disagree* and 7 indicating *strongly agree*.

Instruction: Refer to the print out of an interface screenshot that says ”Interface #2”.

This questionnaire gives you an opportunity to tell us your reactions to the interface. Your responses will help us understand what aspects of the interface you are particularly concerned about and the aspects that satisfy you. To as great a degree as possible, think about all the tasks that you have done with the interface while you answer these questions. Please read each statement and indicate how strongly you agree or disagree with the statement.

1. Overall, I am satisfied with how easy it is to use this system.
2. It was simple to use this system.
3. I can effectively complete my work using this system.
4. I am able to complete my work quickly using this system.
5. I am able to efficiently complete my work using this system.
6. I feel comfortable using this system.
7. It was easy to learn to use this system.
8. I believe I became productive quickly using this system.
9. The interface of this system is pleasant.

10. I like using the interface of this system.
11. This system has all the functions and capabilities I expect it to have.
12. Overall, I am satisfied with this system.

Other learners' watching history

Please answer the following questions about seeing how other learners watched the same video.

1. ”Interaction peaks” affected my video navigation during the tasks.
2. ”Interaction peaks” matched with my personal points of interest in the video.
3. Other learners' watching history was easy to understand.
4. Other learners' watching history was informative.
5. It was enjoyable to see other learners' watching history in a video.
6. Seeing other learners' watching history in a video was useful.
7. Why do you think ”interaction peaks” occur?

For watching online lecture videos generally, not just for the study tasks today, how useful do you think the various features are in this interface?

Please refer to the printed interface screenshot to see what each feature is.

1. Timeline display
2. Timeline clicking and dragging
3. Highlighting interaction peaks
4. Thumbnail preview
5. Personal watching history visualization
6. Play / Pause button
7. Time display
8. Volume control
9. Adding bookmarks
10. Transcript text
11. Interactive transcript (automatic scrolling and highlighting current line)
12. Keyword search

13. Search result highlights below the timeline
14. Search result ranking
15. Word cloud above the player
16. Screenshot highlights at the bottom
17. Pinning a screenshot for a side-by-side view

Please leave any comments about this interface. (strengths and weaknesses)

Appendix B

Materials for the ToolScape Study

This chapter presents the study tasks and questionnaires used in a laboratory study described in Chapter 5.

B.1 Post-Task Questionnaire

The following questionnaire was given to participants at the end of each design task. The questions used a 7-point Likert scale, with 1 indicating *strongly disagree* and 7 indicating *strongly agree*.

Instruction: Please answer the following questions about your experience in this design task.

B.1.1 Overall Experience

- The task was difficult to perform.
- The quality of my final image is high.

B.1.2 Video Browsing Interface

- It was easy to use.
- It was easy to understand.

- It was enjoyable.
- It was informative.

Appendix C

Materials for the Crowdy Study

The following questions were used as a pretest and posttest material, which were asked before and after participants watched the video.

Q1 You have found the following ages (in years) of all 6 lions at your local zoo: 11, 5, 12, 5, 7, 2 What is the standard deviation of the ages of the lions at your zoo? Please use the calculator. You may round your answers to the nearest tenth.

Q2 Listed below are two sample data sets, A and B. Which data set has the larger standard deviation? (Hint: you can answer this question by inspecting the two data sets without calculating the actual standard deviation.)

Data Set A: 1, 2, 3, 4, 5, 6, 7, 8, 9

Data Set B: 8, 9, 9, 9, 10, 11, 11, 12

1. A
2. B
3. They have the same standard deviation.
4. Cannot be answered with the given information.

Q3 What is the relationship between the variance of a set of scores and the standard deviation of those scores?

1. standard deviation = (variance)2
2. variance = (standard deviation)2
3. variance = mean/standard deviation
4. standard deviation = mean/variance

Q4 If the variance of a distribution is 9, the standard deviation is:

1. 3
2. 6
3. 9
4. 81
5. impossible to determine without knowing n.

Q5 The standard deviation of a group of scores is 10. If 5 were subtracted from each score, the standard deviation of the new scores would be

1. 2
2. 10/25
3. 5
4. 10

Q6 The population standard deviation of the following five numbers 3,3,3,3,3 is:

1. 0
2. 3
3. 9

4. 11.25

5. 45

Bibliography

[1] Gregory D. Abowd. Classroom 2000: An experiment with the instrumentation of a living educational environment. *IBM systems journal*, 38(4):508–530, 1999. 45

[2] Abir Al-Hajri, Gregor Miller, Matthew Fong, and Sidney S. Fels. Visualization of personal history for video navigation. In *CHI '14*, 2014. 50

[3] Jason Alexander, Andy Cockburn, Stephen Fitchett, Carl Gutwin, and Saul Greenberg. Revisiting read wear: Analysis, design, and evaluation of a footprints scrollbar. In *CHI '09*, pages 1665–1674, 2009. 87

[4] Duglas G Altman. *Practical statistics for medical research*, volume 12. CRC Press, 1991. 127

[5] F. Arman, R. Depommier, A. Hsu, and M.-Y. Chiu. Content-based browsing of video sequences. In *MULTIMEDIA '94*, pages 97–103, 1994. 50

[6] A. Bandura. Self-efficacy: toward a unifying theory of behavioral change. *Psychological review*, 84(2):191, 1977. 112

[7] Nikola Banovic, Tovi Grossman, Justin Matejka, and George Fitzmaurice. Waken: Reverse engineering usage information and interface structure from software videos. In *UIST '12*, 2012. 51, 52, 107

[8] Sumit Basu, Chuck Jacobs, and Lucy Vanderwende. Powergrading: a Clustering Approach to Amplify Human Effort for Short Answer Grading. *TACL*, 1:391–402, 2013. 48

[9] Patrick Baudisch, Edward Cutrell, Ken Hinckley, and Adam Eversole. Snap-and-go: Helping users align objects without the modality of traditional snapping. In *CHI '05*, pages 301–310, 2005. 86

[10] Michael S. Bernstein, Joel Brandt, Robert C. Miller, and David R. Karger. Crowds in two seconds: Enabling realtime crowd-powered interfaces. In *UIST '11*. ACM, 2011. 51

[11] Michael S. Bernstein, Greg Little, Robert C. Miller, Björn Hartmann, Mark S. Ackerman, David R. Karger, David Crowell, and Katrina Panovich. Soylent: a word processor with a crowd inside. In *UIST '10*, pages 313–322, 2010. 41, 117, 169

[12] Michael S. Bernstein, Jaime Teevan, Susan Dumais, Daniel Liebling, and Eric Horvitz. Direct answers for search queries in the long tail. In *CHI '12'*, pages 237–246, 2012. 41

[13] Anant Bhardwaj, Juho Kim, Steven Dow, David Karger, Sam Madden, Rob Miller, and Haoqi Zhang. Attendee-sourcing: Exploring the design space of community-informed conference scheduling. In *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014. 175

[14] Katerine Bielaczyc, Peter L Pirolli, and Ann L Brown. Training in self-explanation and self-regulation strategies: Investigating the effects of knowledge acquisition activities on problem solving. *Cognition and instruction*, 13(2):221–252, 1995. 139

[15] Renaud Blanch, Yves Guiard, and Michel Beaudouin-Lafon. Semantic pointing: Improving target acquisition with control-display ratio adaptation. In *CHI '04*, pages 519–526, 2004. 86

[16] Charles C. Bonwell and James A. Eison. *Active Learning: Creating Excitement in the Classroom. 1991 ASHE-ERIC Higher Education Reports.* ERIC, 1991. 44

[17] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *Visualization and Computer Graphics, IEEE Transactions on*, 17(12):2301–2309, 2011. 57, 93

[18] Lori Breslow, David E. Pritchard, Jennifer DeBoer, Glenda S. Stump, Andrew D. Ho, and Daniel T. Seaton. Studying learning in the worldwide classroom: Research into edX's first MOOC. *Research and Practice in Assessment*, 8, Summer 2013. 27, 54

[19] Michael Brooks, Sumit Basu, Charles Jacobs, and Lucy Vanderwende. Divide and correct: using clusters to grade short answers at scale. In *Learning at Scale '14*, pages 89–98. ACM, 2014. 48

[20] Andrea Bunt, Patrick Dubois, Ben Lafreniere, Michael A. Terry, and David T. Cormack. Taggedcomments: Promoting and integrating user comments in online application tutorials. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 4037–4046, New York, NY, USA, 2014. ACM. 52

[21] Lucy Buykx and Helen Petrie. Recipe sub-goals and graphs: An evaluation by cooks. In *Proceedings of the ACM Multimedia 2012 Workshop on Multimedia for Cooking and Eating Activities*, CEA '12, pages 59–64, New York, NY, USA, 2012. ACM. 46

[22] Axel Carlier, Vincent Charvillat, Wei Tsang Ooi, Romulus Grigoras, and Geraldine Morin. Crowdsourced automatic zoom and scroll for video retargeting. In *Multimedia '10*, pages 201–210, 2010. 42

[23] Richard Catrambone. The subgoal learning model: Creating better examples so that students can solve novel problems. *Journal of Experimental Psychology: General*, 127(4):355, 1998. 46, 129, 131, 132, 161

[24] Richard Catrambone. Task analysis by problem solving (taps): uncovering expert knowledge to develop highquality instructional materials and training. In *2011 Learning and Technology Symposium*, 2011. 47, 132, 139

[25] Michelene TH Chi. Active-constructive-interactive: A conceptual framework for differentiating learning activities. *Topics in Cognitive Science*, 1(1):73–105, 2009. 44

[26] Michelene TH Chi, Miriam Bassok, Matthew W. Lewis, Peter Reimann, and Robert Glaser. Self-explanations: How students study and use examples in learning to solve problems. *Cognitive science*, 13(2):145–182, 1989. 45

[27] Michelene TH Chi and R Wylie. Icap: A hypothesis of differentiated learning effectiveness for four modes of engagement activities. *Educational Psychologist*, to appear. 161

[28] Pei-Yu Chi, Sally Ahn, Amanda Ren, Mira Dontcheva, Wilmot Li, and Björn Hartmann. 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, pages 93–102, New York, NY, USA, 2012. ACM. 50, 51, 52, 112

[29] Pei-Yu (Peggy) Chi, Joyce Liu, Jason Linder, Mira Dontcheva, Wilmot Li, and Björn Hartmann. Democut: generating concise instructional videos for physical demonstrations. In *UIST '13*. ACM, 2013. 49, 52

[30] Lydia B. Chilton, Juho Kim, Paul André, Felicia Cordeiro, James A. Landay, Daniel S. Weld, Steven P. Dow, Robert C. Miller, and Haoqi Zhang. Frenzy: Collaborative data organization for creating conference sessions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 1255–1264, New York, NY, USA, 2014. ACM. 175

[31] Lydia B. Chilton, Greg Little, Darren Edge, Daniel S. Weld, and James A. Landay. Cascade: crowdsourcing taxonomy creation. In *CHI '13*, pages 1999–2008, 2013. 41

[32] Konstantinos Chorianopoulos. Collective intelligence within web video. *Human-centric Computing and Information Sciences*, 3(1):10, 2013. 42

[33] William S Cleveland. Lowess: A program for smoothing scatterplots by robust locally weighted regression. *The American Statistician*, 35(1):54–54, 1981. 61, 85

[34] Derrick Coetzee, Armando Fox, Marti A. Hearst, and Bjoern Hartmann. Chatrooms in MOOCs: All Talk and No Action. In *Learning at Scale '14*, L@S '14, pages 127–136, New York, NY, USA, 2014. ACM. 48

[35] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37–46, 1960. 162

[36] Andrew Cross, Mydhili Bayyapunedi, Dilip Ravindran, Edward Cutrell, and William Thies. Vidwiki: Enabling the crowd to improve the legibility of online educational videos. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '14, pages 1167–1175, New York, NY, USA, 2014. ACM. 41

[37] A. Dieberger, P. Dourish, K. Höök, P. Resnick, and A. Wexelblat. Social navigation: Techniques for building more usable systems. *Interactions*, 7(6):36–45, November 2000. 81

[38] Wei Ding and Gary Marchionini. A study on video browsing strategies. Technical report, College Park, MD, USA, 1997. 95

[39] Morgan Dixon and James Fogarty. Prefab: implementing advanced behaviors using pixel-based reverse engineering of interface structure. In *CHI '10*, pages 1525–1534, 2010. 51

[40] Mira Dontcheva, Robert R Morris, Joel R Brandt, and Elizabeth M Gerber. Combining crowdsourcing and learning to improve engagement and performance. In *the 32nd annual ACM conference*, pages 3379–3388, New York, New York, USA, 2014. ACM Press. 40

[41] Steven P. Dow, Alana Glassco, Jonathan Kass, Melissa Schwarz, Daniel L. Schwartz, and Scott R. Klemmer. Parallel prototyping leads to better design results, more divergence, and increased self-efficacy. *ACM Trans. Comput.-Hum. Interact.*, 17(4):18:1–18:24, December 2010. 112

[42] edX. edX Insights. `https://github.com/edx/insights`. 56

[43] edX. Tracking Logs – edX 0.1 documentation. `http://data.edx.org/en/latest/internal_data_formats/tracking_logs.html`. 55

[44] Elsa Eiriksdottir and Richard Catrambone. Procedural instructions, principles, and examples how to structure instructions for procedural tasks to enhance performance, learning, and transfer. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 53(6):749–770, 2011. 46, 106, 109

[45] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, 1996. 119

[46] Erika L Ferguson and Mary Hegarty. Learning with real machines or diagrams: application of knowledge to real-world problems. *Cognition and Instruction*, 13(1):129–160, 1995. 26, 45, 106

[47] Jennifer Fernquist, Tovi Grossman, and George Fitzmaurice. Sketch-sketch revolution: an engaging tutorial system for guided sketching and application learning. In *UIST '11*, pages 373–382, 2011. 52

[48] Adam Fourney, Ben Lafreniere, Richard Mann, and Michael Terry. "then click ok!": extracting references to interface elements in online documentation. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 35–38, New York, NY, USA, 2012. ACM. 52

[49] Adam Fourney and Michael Terry. Mining online software tutorials: Challenges and open problems. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '14, pages 653–664, New York, NY, USA, 2014. ACM. 52

[50] Scott Freeman, Sarah L. Eddy, Miles McDonough, Michelle K. Smith, Nnadozie Okoroafor, Hannah Jordt, and Mary Pat Wenderoth. Active learning increases student performance in science, engineering, and mathematics. *PNAS*, 111(23), June 2014. 8410-8415, PMID: 24821756. 44

[51] James F. Gibbons. Tutored Videotape Instruction. 1977. 45

[52] A. Girgensohn and J. Boreczky. Time-constrained keyframe selection technique. In *Multimedia Computing and Systems*, volume 1, pages 756–761 vol.1, 1999. 48, 50

[53] Elena L. Glassman, Juho Kim, Andrés Monroy-Hernández, and Meredith Ringel Morris. Mudslide: A spatially anchored census of student confusion for online lecture videos. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1555–1564, New York, NY, USA, 2015. ACM. 48

[54] Elena L. Glassman, Jeremy Scott, Rishabh Singh, Philip J. Guo, and Robert C. Miller. Overcode: Visualizing variation in student solutions to programming problems at scale. *ACM Trans. Comput.-Hum. Interact.*, 22(2):7:1–7:35, March 2015. 48

[55] Google. YouTube Analytics. `http://www.youtube.com/yt/playbook/yt-analytics.html#details`. 44, 57, 94

[56] Floraine Grabler, Maneesh Agrawala, Wilmot Li, Mira Dontcheva, and Takeo Igarashi. Generating photo manipulation tutorials by demonstration. In *SIGGRAPH '09*, pages 1–9, 2009. 50, 51, 52

[57] Tovi Grossman and George Fitzmaurice. Toolclips: an investigation of contextual video assistance for functionality understanding. In *CHI '10*, 2010. 51

[58] Tovi Grossman, Justin Matejka, and George Fitzmaurice. Chronicle: Capture, exploration, and playback of document workflow histories. In *Proceedings of the 23rd Annual ACM Symposium on User Interface Software and Technology*, UIST '10, pages 143–152, New York, NY, USA, 2010. ACM. 43, 49, 50, 52, 109

[59] Philip J. Guo, Juho Kim, and Rob Rubin. How video production affects student engagement: An empirical study of mooc videos. In *Proceedings of the First ACM Conference on Learning @ Scale*, L@S '14, pages 41–50, New York, NY, USA, 2014. ACM. 28, 58, 66, 80, 81

[60] Ankit Gupta, Dieter Fox, Brian Curless, and Michael Cohen. Duplotrack: A reatime system for authoring and guiding duplo model assembly. In *UIST '12*, 2012. 51

[61] Rassule Hadidi and Chung-Hsien Sung. Students' acceptance of web-based course offerings: an empirical assessment. *AMCIS 1998*, 1998. 26, 45

[62] Susan M Harrison. A comparison of still, animated, or nonillustrated on-line help with written or spoken instructions in a graphical user interface. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 82–89. ACM Press/Addison-Wesley Publishing Co., 1995. 47

[63] Sandra G Hart and Lowell E Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. *Advances in psychology*, 52:139–183, 1988. 146

[64] Robert GM Hausmann and MicheleneT H. Chi. Can a computer interface support self-explaining. *Cognitive Technology*, 7(1):4–14, 2002. 45

[65] Robert GM Hausmann, Timothy J Nokes, Kurt VanLehn, and Sophia Gershman. The design of self-explanation prompts: The fit hypothesis. In *Proceedings of the 31st Annual Conference of the Cognitive Science Society*, pages 2626–2631, 2009. 139

[66] Kurtis Heimerl, Brian Gawalt, Kuang Chen, Tapan Parikh, and Björn Hartmann. Communitysourcing: Engaging local crowds to perform expert work via physical kiosks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1539–1548, New York, NY, USA, 2012. ACM. 40

[67] James Hiebert and Patricia Lefevre. Conceptual and procedural knowledge in mathematics: An introductory analysis. 1986. 27, 28

[68] William C. Hill, James D. Hollan, Dave Wroblewski, and Tim McCandless. Edit wear and read wear. In *CHI '92*, pages 3–9, 1992. 43

[69] Tim N Höffler and Detlev Leutner. Instructional animation versus static pictures: A meta-analysis. *Learning and instruction*, 17(6):722–738, 2007. 47

[70] Xiaodi Hou and Liqing Zhang. Saliency detection: A spectral residual approach. In *CVPR '07*, pages 1–8, 2007. 48, 50

[71] Amy Hurst, Jennifer Mankoff, Anind K. Dey, and Scott E. Hudson. Dirty desktops: Using a patina of magnetic mouse dust to make common interactor targets easier to select. In *UIST '07*, pages 183–186, 2007. 43, 86

[72] Wolfgang Hürst, Georg Götz, and Philipp Jarvers. Advanced user interfaces for dynamic video browsing. In *MULTIMEDIA '04*, pages 742–743, 2004. 86

[73] Kaltura Inc. The state of video in education 2015: A kaltura report. *ACM Trans. Comput.-Hum. Interact.*, 2015. 27

[74] Dan Jackson, James Nicholson, Gerrit Stoeckigt, Rebecca Wrobel, Anja Thieme, and Patrick Olivier. Panopticon: A parallel video overview system. In *UIST '13*, pages 123–130, 2013. 50

[75] Michel Jansen, Willemijn Heeren, and Betsy van Dijk. Videotrees: Improving video surrogate presentation using hierarchy. In *CBMI 2008*, pages 560–567. IEEE, June 2008. 50

[76] Matthew A. Jaro. Advances in record-linkage methodology as applied to matching the 1985 census of tampa, florida. *Journal of the American Statistical Association*, 84(406):414–420, 1989. 121

[77] Cheryl I. Johnson and Richard E. Mayer. Applying the self-explanation principle to multimedia learning in a computer-based game-like environment. *Comput. Hum. Behav.*, 26(6):1246–1252, November 2010. 140

[78] Rosie Jones and Fernando Diaz. Temporal profiles of queries. *ACM Transactions on Information Systems (TOIS)*, 25(3):14, 2007. 43, 63

[79] Sanjay Kairam, Meredith Morris, Jaime Teevan, Dan Liebling, and Susan Dumais. Towards supporting search over trending events with social media. In *ICWSM '13*, 2013. 43

[80] Jeffrey D Karpicke and Janell R Blunt. Retrieval practice produces more learning than elaborative studying with concept mapping. *Science*, 331(6018):772–775, 2011. 140

[81] Jeffrey D Karpicke and Henry L Roediger. The critical importance of retrieval for learning. *science*, 319(5865):966–968, 2008. 140

[82] Caitlin Kelleher and Randy Pausch. Stencils-based tutorials: design and evaluation. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 541–550. ACM, 2005. 51

[83] Juho Kim. Toolscape: Enhancing the learning experience of how-to videos. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 2707–2712, New York, NY, USA, 2013. ACM. 105

[84] Juho Kim, Elena L. Glassman, Andrés Monroy-Hernández, and Meredith Ringel Morris. Rimes: Embedding interactive multimedia exercises in lecture videos. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1535–1544, New York, NY, USA, 2015. ACM. 48

[85] Juho Kim, Philip J. Guo, Carrie J. Cai, Shang-Wen (Daniel) Li, Krzysztof Z. Gajos, and Robert C. Miller. Data-driven interaction techniques for improving navigation of educational videos. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 563–572, New York, NY, USA, 2014. ACM. 32, 79, 165

[86] Juho Kim, Philip J. Guo, Daniel T. Seaton, Piotr Mitros, Krzysztof Z. Gajos, and Robert C. Miller. Understanding in-video dropouts and interaction peaks inonline lecture videos. In *Proceedings of the First ACM Conference on Learning @ Scale*, L@S '14, pages 31–40, New York, NY, USA, 2014. ACM. 32, 53, 81, 92, 94, 165

[87] Juho Kim, Eun-Young Ko, Jonghyuk Jung, Chang Won Lee, Nam Wook Kim, and Jihee Kim. Factful: Engaging taxpayers in the public discussion of a government budget. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2843–2852, New York, NY, USA, 2015. ACM. 40, 176

[88] Juho Kim, Robert C. Miller, and Krzysztof Z. Gajos. Learnersourcing subgoal labeling to support learning from how-to videos. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 685–690, New York, NY, USA, 2013. ACM. 131

[89] Juho Kim, Phu Tran Nguyen, Sarah Weir, Philip J. Guo, Robert C. Miller, and Krzysztof Z. Gajos. Crowdsourcing step-by-step information extraction to enhance existing how-to videos. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '14, pages 4017–4026, New York, NY, USA, 2014. ACM. 33, 35, 49, 52, 75, 105, 135

[90] Juho Kim, Haoqi Zhang, Paul André, Lydia B. Chilton, Wendy Mackay, Michel Beaudouin-Lafon, Robert C. Miller, and Steven P. Dow. Cobi: A community-informed conference scheduling tool. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology*, UIST '13, pages 173–182, New York, NY, USA, 2013. ACM. 40, 175

[91] Nam Wook Kim, Chang Won Lee, Jonghyuk Jung, Eun-Young Ko, Juho Kim, and Jihee Kim. Budgetmap: Issue-driven navigation for a government budget. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '15, pages 1097–1102, New York, NY, USA, 2015. ACM. 40, 176

[92] René F. Kizilcec, Chris Piech, and Emily Schneider. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *LAK '13*, pages 170–179, 2013. 77

[93] Jon Kleinberg. Bursty and hierarchical structure in streams. *Data Mining and Knowledge Discovery*, 7(4):373–397, 2003. 43

[94] Kenneth R. Koedinger, Jihee Kim, Julianna Zhuxin Jia, Elizabeth A. McLaughlin, and Norman L. Bier. Learning is not a spectator sport: Doing is better than watching for learning from a mooc. In *Proceedings of the Second (2015) ACM Conference on Learning @ Scale*, L@S '15, pages 111–120, New York, NY, USA, 2015. ACM. 26, 45

[95] Nicholas Kong, Tovi Grossman, Björn Hartmann, Maneesh Agrawala, and George Fitzmaurice. Delta: A tool for representing and comparing workflows. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '12, pages 1027–1036, New York, NY, USA, 2012. ACM. 52

[96] Rebecca P. Krosnick. Videodoc: Combining videos and lecture notes for a better learning experience. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2015. 50

[97] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 126

[98] Anagha Kulkarni, Jaime Teevan, Krysta M Svore, and Susan T Dumais. Understanding temporal query dynamics. In *WSDM '11*, pages 167–176, 2011. 43

[99] Chinmay Kulkarni, Julia Cambre, Yasmine Kotturi, Michael S. Bernstein, and Scott R. Klemmer. Talkabout: Making distance matter with small groups in massive classes. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work &#38; Social Computing*, CSCW '15, pages 1116–1128, New York, NY, USA, 2015. ACM. 48

[100] Ben Lafreniere, Tovi Grossman, Justin Matejka, and George Fitzmaurice. Investigating the feasibility of extracting tool demonstrations from in-situ video content. In *Proceedings of the 32Nd Annual ACM Conference on Human Factors in Computing Systems*, CHI '14, pages 4007–4016, New York, NY, USA, 2014. ACM. 51, 52

[101] Benjamin Lafreniere, Tovi Grossman, and George Fitzmaurice. Community enhanced tutorials: improving tutorials with multiple demonstrations. In *CHI '13*, pages 1779–1788, 2013. 51, 52

[102] J Richard Landis, Gary G Koch, et al. The measurement of observer agreement for categorical data. *biometrics*, 33(1):159–174, 1977. 156

[103] Gierad Laput, Eytan Adar, Mira Dontcheva, and Wilmot Li. Tutorial-based interfaces for cloud-enabled applications. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 113–122, New York, NY, USA, 2012. ACM. 52

[104] Walter Lasecki, Christopher Miller, Adam Sadilek, Andrew Abumoussa, Donato Borrello, Raja Kushalnagar, and Jeffrey Bigham. Real-time captioning by groups of non-experts. In *UIST '12*, pages 23–34, 2012. 41

[105] Walter S. Lasecki, Christopher D. Miller, and Jeffrey P. Bigham. Warping time for more effective real-time crowdsourcing. In *CHI '13*, pages 2033–2036, 2013. 128

[106] Walter S. Lasecki, Young Chol Song, Henry Kautz, and Jeffrey P. Bigham. Real-time crowd labeling for deployable activity recognition. In *CSCW '13*, pages 1203–1212, 2013. 41

[107] Yong Jae Lee, C. Lawrence Zitnick, and Michael F. Cohen. Shadowdraw: real-time user guidance for freehand drawing. *ACM Trans. Graph.*, 30(4):27:1–27:10, July 2011. 52

[108] Francis C. Li, Anoop Gupta, Elizabeth Sanocki, Li-wei He, and Yong Rui. Browsing digital video. In *CHI '00*, pages 169–176, 2000. 45, 48, 50

[109] Diana Lynn MacLean and Jeffrey Heer. Identifying medical terms in patient-authored text: a crowdsourcing-based approach. *Journal of the American Medical Informatics Association*, 20(6):1120–1127, 2013. 42

[110] Jonathan Malmaud, Jonathan Huang, Vivek Rathod, Nick Johnston, Andrew Rabinovich, and Kevin Murphy. What's cookin'? interpreting cooking videos using text, speech and vision. *arXiv preprint arXiv:1503.01558*, 2015. 51

[111] Adam Marcus, Michael S Bernstein, Osama Badar, David R Karger, Samuel Madden, and Robert C Miller. Twitinfo: aggregating and visualizing microblogs for event exploration. In *CHI '11*, pages 227–236, 2011. 61, 85

[112] Lauren E Margulieux, Richard Catrambone, and Mark Guzdial. Subgoal labeled worked examples improve k-12 teacher performance in computer programming training. In *Proceedings of the 35th Annual Conference of the Cognitive Science Society*, Austin, TX, USA, 2013. 145, 161

[113] Lauren E. Margulieux, Mark Guzdial, and Richard Catrambone. Subgoal-labeled instructional material improves performance and transfer in learning to develop mobile applications. In *Proceedings of the Ninth Annual International Conference on International Computing Education Research*, ICER '12, pages 71–78, New York, NY, USA, 2012. ACM. 46, 131, 132, 145, 161

[114] John Markoff. New Test for Computers: Grading Essays at College Level. *The New York Times*, April 2013. 48

[115] Toshiyuki Masui, Kouichi Kashiwagi, and George R. Borden, IV. Elastic graphical interfaces to precise data manipulation. In *CHI '95*, pages 143–144, 1995. 49, 86

[116] Justin Matejka, Tovi Grossman, and George Fitzmaurice. Swift: Reducing the effects of latency in online video scrubbing. In *CHI '12*, pages 637–646, 2012. 49, 94

[117] Justin Matejka, Tovi Grossman, and George Fitzmaurice. Patina: Dynamic heatmaps for visualizing application usage. In *CHI '13*, pages 3227–3236, 2013. 43, 87

[118] Justin Matejka, Tovi Grossman, and George Fitzmaurice. Swifter: Improved online video scrubbing. In *CHI '13*, pages 1159–1168, 2013. 50

[119] Justin Matejka, Tovi Grossman, and George Fitzmaurice. Video lens: Rapid playback and exploration of large video collections and associated metadata. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 541–550, New York, NY, USA, 2014. ACM. 49

[120] Richard E Mayer and Roxana Moreno. Nine ways to reduce cognitive load in multimedia learning. *Educational psychologist*, 38(1):43–52, 2003. 140

[121] Robert Mertens, Rosta Farzan, and Peter Brusilovsky. Social navigation in web lectures. In *HYPERTEXT '06*, pages 41–44, 2006. 43, 84

[122] Toni-Jan Keith Palma Monserrat, Yawen Li, Shengdong Zhao, and Xiang Cao. L.IVE: An Integrated Interactive Video-based Learning Environment. In *CHI'14*, CHI '14, pages 3399–3402, New York, NY, USA, 2014. ACM. 48

[123] Toni-Jan Keith Palma Monserrat, Shengdong Zhao, Kevin McGee, and Anshul Vikram Pandey. Notevideo: Facilitating navigation of blackboard-style lecture videos. In *CHI '13*, pages 1139–1148, 2013. 50

[124] Mathieu Nancel and Andy Cockburn. Causality: A conceptual model of interaction history. In *CHI '14*, 2014. 43

[125] Cuong Nguyen and Feng Liu. Making software tutorial video responsive. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1565–1568, New York, NY, USA, 2015. ACM. 52

[126] Cuong Nguyen, Yuzhen Niu, and Feng Liu. Video summagator: An interface for video summarization and navigation. In *CHI '12*, pages 647–650, 2012. 50

[127] Phu Nguyen, Juho Kim, and Robert C. Miller. Generating annotations for how-to videos using crowdsourcing. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '13, pages 835–840, New York, NY, USA, 2013. ACM. 105

[128] James Nicholson, Mark Huber, Daniel Jackson, and Patrick Olivier. Panopticon as an elearning support search tool. In *CHI '14*, 2014. 50, 95

[129] Jon Noronha, Eric Hysen, Haoqi Zhang, and Krzysztof Z. Gajos. Platemate: crowdsourcing nutritional analysis from food photographs. In *UIST '11*, pages 1–12, 2011. 41

[130] Dan R Olsen and Brandon Moon. Video summarization based on user interaction. In *EuroITV '11*, pages 115–122, 2011. 42

[131] Susan Palmiter and Jay Elkerton. An evaluation of animated demonstrations of learning computer-based tasks. In *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology*, pages 257–263. ACM, 1991. 47

[132] Susan Palmiter, Jay Elkerton, and Patricia Baggett. Animated demonstrations vs written instructions for learning procedural tasks: a preliminary investigation. *International Journal of Man-Machine Studies*, 34(5):687–701, 1991. 47

[133] Gabriele Paolacci, Jesse Chandler, and Panagiotis G Ipeirotis. Running experiments on amazon mechanical turk. *Judgment and Decision making*, 5(5):411–419, 2010. 146

[134] Sunghyun Park, Gelareh Mohammadi, Ron Artstein, and Louis-Philippe Morency. Crowdsourcing micro-level multimedia annotations: The challenges of evaluation and interface. In *CrowdMM Workshop*, 2012. 51

[135] Amy Pavel, Floraine Berthouzoz, Björn Hartmann, and Maneesh Agrawala. Browsing and analyzing the command-level structure of large collections of image manipulation tutorials. Technical Report UCB/EECS-2013-167, EECS Department, University of California, Berkeley, Oct 2013. 52

[136] Amy Pavel, Colorado Reed, Björn Hartmann, and Maneesh Agrawala. Video digests: A browsable, skimmable format for informational lecture videos. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 573–582, New York, NY, USA, 2014. ACM. 50

[137] Chris Piech, Jonathan Huang, Zhenghao Chen, Chuong Do, Andrew Ng, and Daphne Koller. Tuned models of peer assessment in MOOCs. *arXiv preprint arXiv:1307.2579*, 2013. 48

[138] Suporn Pongnumkul, Mira Dontcheva, Wilmot Li, Jue Wang, Lubomir Bourdev, Shai Avidan, and Michael F. Cohen. Pause-and-play: Automatically linking screencast video tutorials with applications. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 135–144, New York, NY, USA, 2011. ACM. 51, 52, 107

[139] Suporn Pongnumkul, Jue Wang, Gonzalo Ramos, and Michael Cohen. Content-aware dynamic timeline for video browsing. In *UIST '10*, pages 139–142, 2010. 49, 86

[140] Michele Pratusevich. Edvidparse: Detecting people and content in educational videos. Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2015. 51, 92

[141] Michael Preston, Gordon Campbell, Herbert Ginsburg, Peter Sommer, Frank Moretti, Michael Preston, Gordon Campbell, Herbert Ginsburg, Peter Sommer, and Frank Moretti. Developing New Tools for Video Analysis and Communication to

Promote Critical Thinking. In *EdMedia'05*, volume 2005, pages 4357–4364, June 2005. 45

[142] Michael Prince. Does Active Learning Work? A Review of the Research. *J. of Engineering Education*, 93(3), July 2004. 223-231. 44, 140

[143] Gonzalo Ramos and Ravin Balakrishnan. Fluid interaction techniques for the control and annotation of digital video. In *UIST '03*, pages 105–114, 2003. 49, 86

[144] E.F. Risko, T. Foulsham, S. Dawson, and A. Kingstone. The collaborative lecture annotation system (clas): A new tool for distributed learning. *Learning Technologies, IEEE Transactions on*, 6(1):4–13, 2013. 42

[145] Bethany Rittle-Johnson and Martha Wagner Alibali. Conceptual and procedural knowledge of mathematics: Does one lead to the other? *Journal of educational psychology*, 91(1):175, 1999. 27, 28

[146] Gordon Rugg and Peter McGeorge. The sorting techniques: a tutorial paper on card sorts, picture sorts and item sorts. *Expert Systems*, 14(2):80–93, 1997. 67

[147] D.H. Schunk. Goal setting and self-efficacy during self-regulated learning. *Educational psychologist*, 25(1):71–86, 1990. 112

[148] I. Scott MacKenzie and Stan Riddersma. Effects of output display and control–display gain on human performance in interactive systems. *Behaviour & Information Technology*, 13(5):328–337, 1994. 86

[149] Daniel T. Seaton, Yoav Bergner, Isaac Chuang, Piotr Mitros, and David E. Pritchard. Who does what in a massive open online course? *Communications of the ACM*, 2014. 27, 54

[150] Ryan Shaw and Marc Davis. Toward emergent representations for video. In *Multimedia '05*, pages 431–434, 2005. 42

[151] Josef Sivic and Andrew Zisserman. Video google: A text retrieval approach to object matching in videos. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 1470–1477. IEEE, 2003. 50

[152] S.W. Smoliar and HongJiang Zhang. Content based video indexing and retrieval. *MultiMedia, IEEE*, 1(2):62–72, 1994. 48

[153] Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast—but is it good?: Evaluating non-expert annotations for natural language tasks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 254–263, Stroudsburg, PA, USA, 2008. Association for Computational Linguistics. 42

[154] Alexander Sorokin and David Forsyth. Utility data annotation with amazon mechanical turk. *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, 0:1–8, 2008. 41

[155] Ian Spiro, Graham Taylor, George Williams, and Christoph Bregler. Hands by hand: Crowd-sourced motion tracking for gesture annotation. In *CVPR Workshop, 2010*, pages 17–24. IEEE, 2010. 51

[156] Barbara Tversky, Julie Bauer Morrison, and Mireille Betrancourt. Animation: can it facilitate? *International journal of human-computer studies*, 57(4):247–262, 2002. 26, 45, 47, 106

[157] Hans van der Meij, Peter Blijleven, and Leanne Jansen. What makes up a procedure? *Content & Complexity*, 2003. 106, 109

[158] Kurt VanLehn, Randolph M Jones, and Michelene TH Chi. A model of the self-explanation effect. *The journal of the learning sciences*, 2(1):1–59, 1992. 139

[159] Luis Von Ahn. Games with a purpose. *Computer*, 39(6):92–94, 2006. 39

[160] Luis von Ahn and Laura Dabbish. Labeling images with a computer game. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 319–326, New York, NY, USA, 2004. ACM. 39, 170

[161] Carl Vondrick, Deva Ramanan, and Donald Patterson. Efficiently scaling up video annotation with crowdsourced marketplaces. In *ECCV 2010*, pages 610–623. Springer, 2010. 51

[162] MP Wand. Fast computation of multivariate kernel estimators. *Journal of Computational and Graphical Statistics*, 3(4):433–445, 1994. 61

[163] Sarah Weir, Juho Kim, Krzysztof Z. Gajos, and Robert C. Miller. Learnersourcing subgoal labels for how-to videos. In *Proceedings of the 18th ACM Conference on Computer Supported Cooperative Work &#38; Social Computing*, CSCW '15, pages 405–416, New York, NY, USA, 2015. ACM. 36, 131

[164] Hadley Wickham. Bin-summarise-smooth: a framework for visualising large data. Technical report, had.co.nz, 2013. 61

[165] Joseph Jay Williams, Geza Kovacs, Caren Walker, Samuel Maldonado, and Tania Lombrozo. Learning online via prompts to explain. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '14, pages 2269–2274, New York, NY, USA, 2014. ACM. 140

[166] Wistia. Wistia Product Features. `http://wistia.com/product#analyze`. 44

[167] Ruth Wylie and Michelene TH Chi. 17 The Self-Explanation Principle in Multimedia Learning. *The Cambridge Handbook of Multimedia Learning*, page 413, 2014. 45

[168] Tom Yeh, Tsung-Hsiang Chang, and Robert C. Miller. Sikuli: using gui screenshots for search and automation. In *UIST '09*, pages 183–192, 2009. 51

[169] Jude Yew, David A. Shamma, and Elizabeth F. Churchill. Knowing funny: genre perception and categorization in social video sharing. In *CHI '11*, pages 297–306, 2011. 42

[170] Jenny Yuen, Bryan Russell, Ce Liu, and Antonio Torralba. Labelme video: Building a video database with human annotations. In *ICCV 2009*, pages 1451–1458. IEEE, 2009. 51

[171] Dongsong Zhang, Lina Zhou, Robert O. Briggs, and Jay F. Nunamaker Jr. Instructional video in e-learning: Assessing the impact of interactive video on learning effectiveness. *Information & Management*, 43(1):15 – 27, 2006. 45

[172] Barry J. Zimmerman, Albert Bandura, and Manuel Martinez-Pons. Self-motivation for academic attainment: The role of self-efficacy beliefs and personal goal setting. *American Educational Research Journal*, 29(3):663–676, 1992. 112