Leveraging Community-Generated Videos and Command Logs to Classify and Recommend Software Workflows

Xu Wang[1,2], Benjamin Lafreniere[1], Tovi Grossman[1]

[1]Autodesk Research
{tovi.grossman, ben.lafreniere}@autodesk.com

[2]Carnegie Mellon University
xuwang@cs.cmu.edu

ABSTRACT

Users of complex software applications often rely on inefficient or suboptimal workflows because they are not aware that better methods exist. In this paper, we develop and validate a hierarchical approach combining topic modeling and frequent pattern mining to classify the workflows offered by an application, based on a corpus of community-generated videos and command logs. We then propose and evaluate a design space of four different workflow recommender algorithms, which can be used to recommend new workflows and their associated videos to software users. An expert validation of the task classification approach found that 82% of the time, experts agreed with the classifications. We also evaluate our workflow recommender algorithms, demonstrating their potential and suggesting avenues for future work.

Author Keywords

Software learning; application logs; topic modeling; workflow recommendation; community-generated videos.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation: Misc.

INTRODUCTION

Modern complex software applications often include hundreds or thousands of commands, which further form a much larger number of workflows a user can use. The large variety of commands and workflows raises two issues. First, pre-designed tutorials cannot exhaustively cover all the different workflows. Second, users may get stuck in inefficient or suboptimal ways of completing tasks if they are not aware that better workflows exist.

Prior work on software learning addressed this *awareness* problem [12] at a command-level granularity by recommending individual commands [24, 28], or videos based on command usage [29]. However, command-based recommendations may not consider the user's higher-level workflow needs, or help the user to understand how to use already-known commands in new ways.

CHI 2018, April 21–26, 2018, Montreal, QC, Canada
© 2018 Association for Computing Machinery.
ACM ISBN 978-1-4503-5620-6/18/04…$15.00
https://doi.org/10.1145/3173574.3173859

Knowing the tasks that a user is working on, and the workflows they are using, is the first step towards providing better-personalized software learning support. For example, upon recognizing a user's workflow, the application could recommend alternative or more efficient workflows, display sample tutorial videos to help with their task, or provide links to relevant community-created content. By investigating software learning recommendation systems at a *workflow* level, this work complements the existing body of software learning research that focuses on individual commands.

With the above as motivation, this paper contributes a hierarchical approach to mining user workflows at both task and command-set levels. In the first layer of our hierarchical approach, we use Bi-term Topic Modeling (BTM) [39] to infer 18 high-level user *tasks* (*e.g.,* "Rendering", "Beginner Sketching", and "Advanced Surface Modeling") from command logs associated with 11,713 videos of people using the software. A study found an 82% expert agreement with the algorithm's classification of videos into task categories. In the second layer, we apply a frequent itemset mining and ranking approach [9] to acquire frequent *patterns* of commands under each task. For example, a pattern under the *task "*Beginner Sketching*"* may look like *{Center Rectangle, Create Sketch, Sketch Dimension, Edit Sketch Dimension}*.

Based on this hierarchical understanding of user workflows from command logs, we propose and evaluate a design space of four algorithms which recommend community-generated videos to the user, demonstrating relevant workflows. We evaluate the performance of the four algorithms along the dimensions of relevance and novelty. Users had high ratings on the relevance of the videos, with *pattern-based* recommendations being more relevant and familiar to the user than *task-based* recommendations.

Our work contributes a new method to infer user workflows and shows how this method can be utilized to recommend learning videos for a 3D design application. We conclude by discussing how our approach to workflow identification can generalize to other applications and can inform future designs of support systems for software learning.

RELATED WORK

Our work directly builds upon prior work on tools to support software learning, especially recommender systems and community-enhanced software learning systems. We also draw upon methods used in the literature to mine user data. In this section, we review related work in each of these areas.

Relevant Applications for Software Learning

There have been many efforts in the HCI community to design tools to support the learning and use of software, ranging from reflective visualization tools [6, 23, 26], to tools that provide more active support, such as in-software recommendations and feedback [24, 28, 29]. A range of different channels for providing support have been investigated, from in-application contextual help [13], to community support [22], to auto-generation of demonstration videos [18].

Tools to Raise User Awareness

Visualization tools are designed to raise users' awareness of their usage patterns and performance, with the goal of encouraging users to adopt more efficient methods. Such visualization tools have been shown to be successful in raising awareness [6, 23, 26], but can be limited in that they reflect existing behavior, rather than providing users with insights into new ways of working. Other work has adopted more proactive approaches, *e.g.,* CommunityCommands [28] uses collaborative filtering to recommend new commands, and Ambient Help [29] continuously recommends video resources based on the user's recently used commands.

The above systems operate at the *command level*. In this work, we also take an proactive recommendation approach, but we develop tools to understand user tasks and recommend personalized resources at a *workflow level*. Our approach is informed by the collaborative filtering algorithms developed in CommunityCommands [28], and explores a design space for workflow-based recommender systems.

Community Enhanced Learning

Prior work on software learning and "learner-sourcing" [15] has demonstrated the benefits of community-created learning resources, and the potential of repurposing community-created content for software learning purposes.

CADament [22] allows players to acquire new skills by observing the workflows of their opponents in a multiplayer game. CoScripter [20] enables end-users to create and share scripts to automate web-based processes. FollowUs [18] enables community-enhanced tutorials, which improve as more users work with them. Techniques have also been explored to extract command demonstrations from workflow videos [19], and to elicit workflow metadata for how-to videos [16].

Motivated by the work above, our approach leverages community-generated workflow videos to model common workflows in an application, and repurposes these videos as a means for presenting recommended workflows to the user.

User Data Mining

Dev and Liu [9] provides a good summary of prior work on user behavior modeling [3, 30, 31], event sequence [33] and clickstream data modeling [37]. In their work [9], they use a frequent pattern mining approach to identify user tasks in a photo editing application from command logs. They also developed a ranking algorithm to select more coherent patterns. Our work directly builds upon this approach, extending it to be more applicable for software with more diverse usage domains, and utilizing it to provide workflow recommendations.

Outside of the software learning literature, topic modeling is a common generative model to extract topics from a corpus [11]. Prior work has successfully applied topic modeling to mine user behaviors from sequence data. Huynh et al. [14] used this approach to identify routine behaviors from sensor data. In this case, a *topic* is a behavior (*e.g.,* having lunch), and the *words* are activities associated with this topic (*e.g.,* walking freely, picking up cafeteria food, queuing in the line, etc.) Wen and Rose [1] used a similar approach to identify click patterns in data from Massive Open Online Courses.

In our work, we develop a hierarchical approach combining topic modeling and frequent pattern mining approaches to mine software workflows at two levels: a task level, and a finer-grained command-pattern level.

DATASET

The software application we target with our approach is a 3D modeling application designed for consumer, commercial and educational use. The application has over 1,000 commands, separated into a set of high-level workspaces (Model, Sketch, Assemble, etc.) Collectively, this rich feature set enables a wide variety of workflows, including modeling, mesh editing, simulation, and animation. Even for a single task, users can take many different approaches. For example, for basic modeling, users can start from primitive shapes (*e.g.,* a box or cylinder), or can sketch in 2D then transform the sketch to 3D using extrude or other operations.

We collected detailed natural usage logs for 20,000 users of the software from June 25 to August 25, 2017, including 255,643 user sessions and 20 million command invocations. In addition to these usage logs, we collected video data from an online community repository where users upload videos of their usage of the software. These videos have an associated meta-data file with time-stamped command usage data. We collected data for 11,713 videos, with 470,811 commands invoked across these videos. We preprocessed the command logs of the videos to be in the same format as the natural logs from the product, and also collected video attribute data including video length and view count.

UNDERSTANDING USER TASKS FROM VIDEOS

The first step to recommend personalized resources to users is to understand what the users are doing in the software. To this end, we developed a hierarchical approach to mine user workflows at both task and command-set levels. In the first layer of our hierarchical approach, we used topic modeling and inferred 18 meaningful user *tasks* (topics). In the second layer, we mined frequent command *patterns* for videos of each task respectively, resulting in 233 patterns in total. The rationale for this two-level approach is that simply mining frequent command patterns from the entire corpus of command logs can lead to an over-representation of command patterns for frequently-performed tasks, with those for less-frequent tasks drowned out by the volume of log data for more-frequent tasks. Adding an initial stage of topic modeling allows less frequent but distinct activities to be captured as well.

In this section, we describe the limitations of the state-of-the-art approach [9] to mining frequent tasks on our dataset, and then introduce the first layer of our hierarchical approach – using topic modeling to mine user tasks.

Frequent Pattern Mining Approach

The current practice of identifying frequent user tasks involves applying frequent pattern mining techniques, such as frequent itemset mining and sequential pattern mining [1, 2, 8, 27]. However, such existing techniques do not account for the unique characteristics of software log data. For example, in software logs it is common for users to perform a task by executing a set of operations contiguously with no, or few, outliers. Users may also execute a required operation multiple times within the duration of a task. Recent work by Dev and Liu [9] developed an outlier-based ranking algorithm to rank frequent patterns mined from user log data, which addressed the above challenges specific to software logs. We adopted their approach as a starting point.

Initially, we applied Dev and Liu's approach to our dataset without modifications, but found that it mainly identified patterns related to the software's most frequently-used workspace (Sketching). This suggests that for complex software applications with diverse usage domains, a frequent item-set mining approach may not be sufficient to identify patterns representative of the full range of workflows in the software.

Topic Modeling Approach

A topic model is a type of statistical model for discovering abstract "topics" that occur in a collection of documents. Although it was originally developed as a text-mining technique, topic modeling has also been used to mine human behaviors [14, 1]. We consider the way a software user composes a session to be similar to the generative process of a document in topic modeling. Users first decide which task to work on, and then choose commands for that task. Additionally, for complex software applications, it is often the case that users will use slightly different command sets to accomplish similar tasks. The relationship between commands may not be captured by frequent pattern mining approaches, since they measure the co-occurrence of a set of commands. We see an opportunity for topic models to capture these relationships.

We define the problem of inferring user tasks from in-situ command logs as an unsupervised machine learning task, due to the lack of ground truth data. In this work, we collected a large dataset of community-generated videos showing various uses of the software, with corresponding command logs. This enables us to first infer user tasks through unsupervised machine learning and then validate the results using the additional context provided by the videos.

We used the command logs from the video dataset as training data for the topic model, treating each video as a document, and each command as a word. After labeling and validating the inferred topics, we applied the topic model to the community logs to classify user tasks for all users.

Data Preprocessing

We extracted the command logs from the video dataset, and filtered out 34 "stop word" commands identified by domain experts, such as "Constrained Orbit", "Free Orbit", "Pan", and "Cancel". We only included videos with at least 2 unique commands. This resulted in 11,713 videos with 952 unique commands, and 470,811 total command invocations.

LDA vs. BTM

We initially applied a common topic modeling algorithm, Latent Dirichlet Allocation (LDA) [7] (using Gensim [10]) to infer topics from the command logs of the video dataset. To select the number of topics, K, we tested values from 5 to 100 in increments of 5. A researcher and a domain expert analyzed the output for each value of K to identify the most sensible results. New topics ("Animation") emerged at K=20, as compared to K=15, and for $K \geq 25$, we started to see overlapping topics, especially related to sketching. We did not further refine the final number of topics by testing values of K between 20 and 25, due to limitations in time with the domain expert. Based on the above, we finalized at K=20.

A limitation of the LDA algorithm is the sparsity issue. Some videos contain a small number of commands, resulting in a sparse document-word (video-command) matrix. To address this, we adopted the Bi-term Topic Modeling (BTM) approach [39], which is designed to infer topics from short texts. BTM explicitly models word co-occurrence patterns to enhance the topic learning, and uses the aggregated patterns in the whole corpus when learning topics, to solve the problem of sparse word co-occurrence patterns. We applied BTM on the command logs of the video dataset and used a similar approach to tune the number of topics parameter, which also generated optimal performance for K = 20 topics.

The researcher and a domain expert did a qualitative comparison of the 20 topics generated by LDA and BTM respectively. We concluded that BTM generated more coherent topics, with fewer overlapping topics, and additional topics not identified by LDA. Thus, we used BTM to classify user tasks. We refer to the topics generated by BTM as "*tasks*".

Output of BTM

The output of BTM includes (1) a topic-word (task-command) distribution matrix, and (2) a document-topic (video-task) distribution matrix. Using the video-task matrix, we assigned each of the 11,713 videos as belonging to the "task" (*i.e.,* topic) with the highest weight for that video. We also define the following two similarity terms:

Task-Task similarity—each task is represented in a task-command vector by the topic-word (task-command) matrix. We define the task-task similarity as the cosine similarity between the two task-command vectors. We used a similar definition of task-task similarity as used in Labeled LDA [34].

Id	Expert 1	Expert 2	Final Task Name	Videos
1	Intermediate Sketching	Offsetting sketch geometry and trimming lines back that are overlapping	Intermediate sketching (offsetting sketch geometry and trimming lines)	239
2	Advanced Surfacing	Surfacing and Sculpting	Advanced surfacing (surfacing and sculpting)	296
3	Beginners Design	Basic Part Modeling	Basic part modeling	673
4	Advanced sketching	Editing Splines in a sketch	Editing splines in a sketch	293
5	Design (sketch and features)	Extruding text	Creating features from sketches	1251
6	Surfacing	Fixing Surfaces/Patching up surfaces	Fixing or patching surfaces	249
7	Rendering	Add appearances, Rendering a design	Rendering (adding appearances, rendering a design)	166
8	Simulation	Simulation	Simulation	116
9	More aimless clicking	Editing a design	Editing a design	506
10	Drawing Creation	Creating a drawing of a design	Creating a drawing of a design	131
11	Expert Sketching	Creating construction planes and then creating sketches on those planes	Intermediate sketching (creating construction planes)	998
12	Intermediate CAM	Creating CAM tool paths	Intermediate CAM (creating CAM tool paths)	835
13	Someone is aimlessly clicking around	Assembling components	Copy and pasting components, and assembling them	528
14	Industrial Design from image	Inserting a canvas then using a TSpline body to match the canvas	Industrial design based on a reference image	104
15	Animations	Creating an animation	Animation	53
16	Sketching	Constraining a sketch	Beginner sketching (constraining/dimensioning a sketch, fully defining a sketch)	458
17	Sculpt	Editing a TSpline body	Sculpting (editing a T-spline body)	793
18	*3rd Party Add In*	*Not sure*	*Dropped this topic*	*445*
19	Parametric Design	Dimensioning a sketch	Beginner sketching (constraining/dimensioning a sketch, fully defining a sketch)	1139
20	Assembly	Assembling components	Assembly	583

Table 1. Task names, as labeled by experts, with the number of videos that are categorized for each task.

Video-Video similarity—each video is represented in a video-task vector by the document-topic (video-task) matrix. We define the similarity between two videos as the cosine similarity between the two video-task vectors.

Cosine similarity can compare documents in terms of their subject matter [35], rather than length or other attributes. This makes it appropriate for calculating similarity of videos and tasks, which may vary in length but concern the same high-level content. While cosine similarity comes from a different modeling approach than probabilistic modeling, we selected it over probability-based metrics (*e.g.,* Kullback-Leibler Divergence) because it is well-known, easy to implement, and has been found to be as effective as probability-based metrics in applications similar to our own (*e.g.*, filtering redundant documents [40], and computing similarity between user-topic vectors generated by LDA [32]).

Expert Labeling

While the topic modeling algorithm provides an association of commands to topics, it does not provide a semantically meaningful label for these "tasks". To generate such labels, we recruited two domain experts from the company that developed the software. For each "task", we showed the top 10 weighted commands for the task, and the top three videos ranked by weight for that task. We included three multiple-choice questions and one open-ended question in the survey to understand whether experts found the "tasks" to be meaningful, and to label each with a name.

Expert Labeling Survey Results

In the open-ended question, we asked the experts to give a name to each task (*i.e.,* what they think the user is working on when seeing these commands used together). The experts' responses are shown in Table 1. Following the survey, a researcher met each expert to discuss their understanding and finalize a name for each task (also shown in Table 1). We dropped one task that was indicated as not meaningful by the experts, and combined two tasks on beginner sketching, resulting in 18 distinct tasks. The number of videos that are categorized into each task is shown in column "Videos".

To further assess the effectiveness of the algorithm, we asked experts to indicate whether the commands composing each task are frequently used together. In a multiple-choice question, we asked each expert to rate "How frequent/likely do you think these commands would be used together?" The experts respectively rated 15/20 and 17/20 tasks as meaningful (Figure 1). This provides initial validation of our approach of task recognition using topic modeling.

The experts also rated the helpfulness of the videos in deciding the name of the task. Experts found the videos to be helpful or neutral for 16/20 and 16/20 of the tasks respectively. The two experts also showed high agreement on this question, with a Pearson coefficient of 0.48 (p=0.03) between their ratings. This is a promising result, indicating that the community corpus of videos can be used to demonstrate new workflows to users as part of a recommender system.

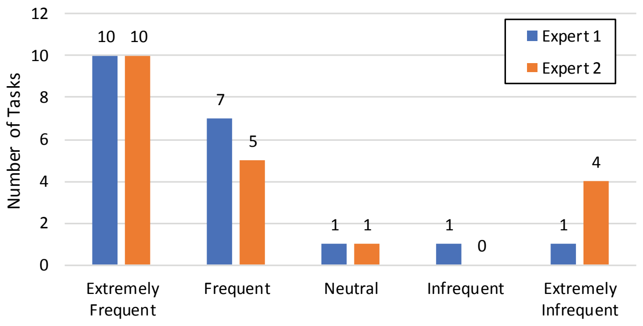


Figure 1. Expert ratings for Q1: "How frequently/likely do you think these commands would be used together?"

STUDY 1: TASK CATEGORIZATION VALIDATION

The aim of our first study was to evaluate the topic modeling approach for categorizing user tasks. Based on the labels provided by the domain experts, we asked a new set of ten users to watch a selection of videos, and identify the task demonstrated in the video, and the similarity of pairs of videos. The goal was to evaluate whether users' responses would match the topic modeling algorithm's categorization.

Video Study Design

Based on several rounds of piloting, we developed the following two question types answered by users in this study.

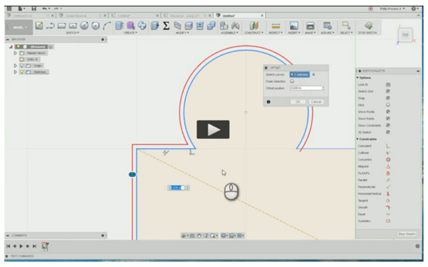
Labeling Questions

The first question type asked participants to view a single video, and choose a task category for that video from a list of four possible options (Figure 2). The list of choices for each video was constructed so that it contained the label provided by the topic modeling algorithm, as well as three other tasks. The three additional tasks were selected by ranking the remaining 17 tasks produced by the topic modeling approach by similarity to the task of the target video (using the task-task similarity metric as defined earlier). This ranked list of tasks was divided into three roughly equal-sized tiers, and one task was selected from each tier. This results in each question containing a mix of tasks that our algorithm believes are close to the video, and that are far from it.

For example, the BTM algorithm categorizes the video shown in Figure 2 as belonging to "Intermediate sketching (offsetting sketch geometry and trimming lines)". The three additional choices, ranked by their similarity to this task, are "Beginner sketching" (0.8), "Editing splines in a sketch" (0.25), and "Simulation" (0.12). Using this approach, we include choices that are similar to the task chosen by the algorithm, and ones that are different, without overwhelming the participant with all 18 possible choices. This enables us to investigate whether the distribution of answers over choices is consistent with the similarity between tasks.

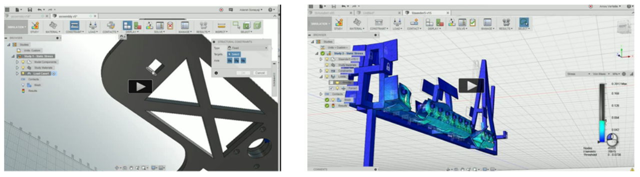
Similarity Questions

The second question type asked the participant to watch a pair of videos, and evaluate whether they believed the tasks being performed in the two videos were similar or not, on a 5-point Likert scale (Figure 3). To construct the similarity questions, we randomly selected a video from the dataset as the target video, and ranked all the other videos based on their similarity to the target video (using the video-video similarity metric as defined earlier).



Which of the following categories best describes the task being performed in the video?
- ○ Simulation
- ○ Intermediate sketching (offsetting sketch geometry and trimming lines)
- ○ Editing splines in a sketch
- ○ Beginner sketching (constraining/dimensioning sketch, fully defining a sketch)

Figure 2. Example labeling question – Select the category that best describes the task being performed in the video.



To what extent do you agree with the following statement:

The tasks being performed in the two videos are similar.
- ○ Strongly Agree
- ○ Agree
- ○ Neither Agree Nor Disagree
- ○ Disagree
- ○ Strongly Disagree

Figure 3. Similarity question – Rate the similarity of the tasks performed in the two videos.

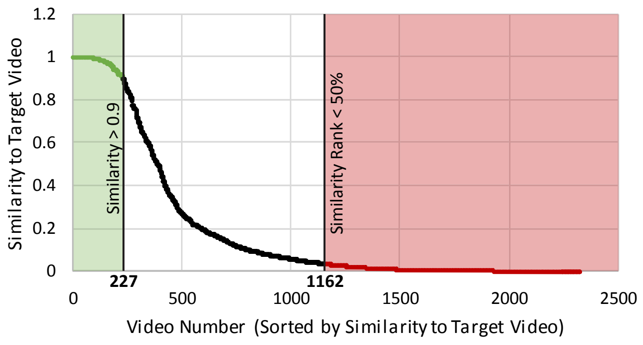


Figure 4. Similarity between a video and all other videos. We selected similar videos from the green shaded area, and dissimilar videos from the red shaded area.

The pair of videos shown to the participant was either similar or dissimilar. To form a *similar pair*, we identified videos with a similarity score higher than 0.9 to the target video (the green shaded area in Figure 4), and randomly selected one such video. To form a *dissimilar pair*, we randomly selected a video from the bottom half of all videos, ranked by similarity to the target video (the red shaded area in Figure 4). Our approach for selecting similar/dissimilar items was developed in an ad-hoc manner, based on experimentation with our dataset. In particular, we found that a simpler criterion (*e.g.,* selecting the top 10% of videos, ranked by similarity [28]) did not account for tasks with few videos. For example, the Animation task included only 53 videos, so selecting 10% of all videos ranked by similarity to a video on Animation would select many videos outside this topic. Using a threshold on similarity score avoided this problem.

Participants

We recruited 10 users (8 male, 2 female) that self-identified as intermediate or expert-level users of the software. Participants were given a $25 gift card for participating.

Study Design

Each participant answered 11 labeling questions, and 18 similarity questions (including 9 video pairs that our algorithm indicated were similar, and 9 that our algorithm indicated were dissimilar), covering a total of 47 videos across 29 questions. In order to cover a larger variety of videos, we designed the study so that 2 participants worked on questions with the same set of 47 videos, with 5 sets of 47 videos in total. We also made sure videos of each task were equally represented in the questions. In total, we covered 235 videos in the study, which ranged in duration from 10-60 seconds. To counterbalance, in each set, we reversed the order of similarity and labeling questions for the two participants. Questions within each category were presented in a random order. To make sure the participants watched the videos and treated the questions seriously, we asked participants to provide a short text justification of their response to each question.

Quantitative Results and Analysis

For similarity questions, if the participant answers “Strongly Agree” or “Agree” for a similar pair of videos, or “Strongly Disagree” or “Disagree” for a dissimilar pair of videos, we count the answer as consistent with the algorithm, otherwise as inconsistent. Participants’ overall agreement on similarity questions with the algorithm was 71%. We also computed the users’ average rating for similar pairs (1.4) and dissimilar pairs (3.4), where the rating is computed on a 1-5 scale, with 5 meaning “similar” and 1 meaning “dissimilar”. For the labeling questions, participants agreed with the algorithm’s classification 82% of the time. The performance of each participant is shown in Table 2.

Video Set	Similarity Agreement	Rating for dissimilar pairs	Rating for similar pairs	Labelling Agreement
1	83%	1.3	4.2	82%
	78%	1	3.4	82%
2	61%	1.3	2.4	91%
	72%	1.4	3.1	73%
3	78%	1.1	4	100%
	94%	1.2	4.3	91%
4	61%	1.7	2.8	64%
	67%	1	2.6	73%
5	61%	2	3.6	91%
	56%	2	2.8	73%
TTL	**71%**	**1.4**	**3.3**	**82%**

Table 2. Summary of results for Study 1, grouped by video set.

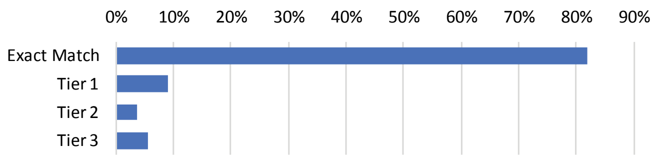


Figure 5. Summary of agreement between the algorithm and participants’ responses to the labeling questions.

From the results, we found that participants had a high level of agreement with the algorithm. The main source of inconsistency was the lack of agreement when our algorithm considered two videos to be similar – participants were more conservative about judging two videos to be similar.

In particular, participants had high agreement with the algorithm on labeling questions. As shown in Figure 5, 82% of responses matched the algorithm’s classification. Moreover, approximately half of the responses that did not match the algorithm were in Tier 1, which is the closest to the classification of the algorithm without being an exact match. Overall, 91% of responses were either exact matches or in Tier 1.

Qualitative Analysis of User Feedback

To better understand the circumstances under which participants agreed or disagreed with the algorithm’s classification, we examined the justifications they provided for their ratings. In general, we found that participants were able to give detailed descriptions of the tasks being performed in the videos. Some examples of justifications are provided below:

“They are both linked with motion, both using joints to drive the parts or restrict movement.”

“Both are short videos that go into the render environment and start in-canvas rendering.”

“Fairly advanced sketching manipulations, all 2D. 2nd video, manipulating 3D assembly with joints and alignments, no actual changes to the geometry just their orientation.”

“The first video is using a sketch on a plane to use as a cutting tool to split a body - the second uses the time line and preferences to modify an existing model.”

Participants’ justifications indicated a range of different standards for judging similarity. While we asked participants to judge based on whether the two workflows were working toward a similar goal, even if their individual approaches were different, or the end results were different (*e.g.*, one succeeded while the other failed), we observed that many participants judged similarity based on other standards, such as the expertise level of the approaches shown in the video, or the specific operations used (*e.g.*, “Both navigate the design space and add features. The former looks unprofessional, but latter looks very skilled.”, and “Both videos create a sketch, but other operations are different”.) Participants also mentioned that their answer could go either way, depending on how similarity was defined (*e.g.,* “It depends on how vague you want to go with the similarities – you could say that they are using some type of constraint by aligning faces or changing dimension etc. – but it is vague”).

Further, some participants were particularly strict when judging similarity. In the examples below, participants’ justifications for their ratings suggest an agreement that the tasks had a similar goal, but their ratings do not reflect this:

"1 is creating a drawing view. 2 is editing a drawing view's scale. The general end goal is to end up with a drawing." (Rated as "Neutral")

"Both attempt to simulate how parts work in the physical world" (Rated as "Extremely different")

"1 is creating a body from a TSpline. 2 is creating variations of TSpline bodies and ends up with bodies." (Rated as "Neutral")

Overall, our study results suggest that the BTM algorithm does infer meaningful user tasks that are consistent with the understanding of experienced users of the software. This is encouraging evidence for the value of topic modeling approaches for software log data. Our findings also indicate the value of gathering a corpus of video and associated log data for a software application, as it can be used to validate approaches for modeling user tasks from log data.

HIERARCHICAL TASK IDENTIFICATION

Study 1 validates the first layer of our hierarchical approach, with which we can infer the high-level tasks (topics) the user is working on. However, even when two videos are of the same high-level task, they may contain very different command sets. The second layer of our approach allows us to acquire finer-grained command sets under each task.

We began with the output of BTM, which assigned each video to a task (*i.e.,* the task with the highest weight for that video in the video-task matrix). For the set of videos under each task, we applied the FP-Growth algorithm (using SPMF library [36]) on the command logs to identify frequent patterns. We set different thresholds for each task based on how many videos there were – our rule of thumb was that the number of frequent patterns acquired for each task should be within the range of 5-10% of the total number of videos for that task. For the patterns acquired under each task, we applied the ranking algorithm developed by Dev and Liu [9] and set the minimal length for a pattern to be 3 and the cutout cohesion score to be 2. By choosing cohesion score of 2, we allowed 1 outlier for a pattern with 3 commands. For example, the pattern {*Construct Sketch, Draw Line, Add Geometry Constraint to Sketch*} contained three commands and had a cohesion score of 2, because for the 206 times that this pattern appeared, at least half of the times there was another command that appeared in the sequence other than the three commands in the pattern *(i.e.,* an outlier). Examining the video command logs, there were cases where the three commands appeared together with no outliers, and other cases such as {*Draw Line, Construct Sketch, Trim Sketch, Add Geometry Constraint*}, and {*Draw Line, Construct Sketch, Add Tangent Handle, Add Tangent Handle, Add Geometry Constraint*} where this was not the case. Setting the cutout cohesion score to 3 would result in a loss of such length-3 patterns that appeared frequently with 1 outlier in between the commands. We refer the reader to Dev and Liu [9] for additional context surrounding our choice of cohesion score and allowing outliers. Using this approach, we got 233 frequent patterns in total for the 18 tasks. The final distribution of command patterns by task is shown in Table 3.

Task Name	Count
Intermediate sketching (offsetting sketch geometry and trimming lines)	9
Advanced surfacing (surfacing and sculpting)	1
Basic part modeling	0
Editing splines in a sketch	19
Creating features from sketches	49
Fixing or patching surfaces	4
Rendering (adding appearances, rendering a design)	6
Simulation	34
Editing a design	9
Creating a drawing of a design	12
Intermediate sketching (creating construction planes)	13
Intermediate CAM (creating CAM toolpaths)	10
Copy and pasting components, and assembling them	4
Industrial design based on a reference image	2
Animation	1
Sculpting (editing a T-spline body)	2
Beginner sketching (constraining/dimensioning a sketch, fully defining a sketch)	57
Assembly	1

Table 3. Distribution of patterns by task.

Comparing this hierarchical approach with simply applying the above command-set identification to all data, we found greater diversity in the tasks identified. Specifically, without first applying BTM, all 53 frequent patterns acquired were for sketch-related tasks.

Examining the output of our approach indicated that it provided reasonable results. For example, for the Beginner Sketching task, we found patterns such as *{Center Rectangle, Create Sketch, Sketch Dimension, Edit Sketch Dimension}* showing the user created a sketch, drew a rectangle, and edited its dimensions. For the Intermediate Sketching task, we found patterns such as *{Line, Extrude, Stop Sketch, Trim}* showing the user is drawing and trimming lines in a sketch, and extruding from the sketch. For the Assembly task, we found patterns such as *{Activate Environment, Joint, Drag Joint Origin}* which shows the user activated the workspace, dragged the joint origin and then made a joint.

RECOMMENDER SYSTEM

Using the hierarchical approach, we trained a topic model using BTM on the command logs of 11,713 videos, to infer 18 topics representing high-level tasks in the software (first layer), and then acquired frequent patterns for each of these tasks (second layer), resulting in 233 patterns in total. These topics and command patterns allow us to infer a task distribution for each user, and to characterize users and videos based on which of the patterns are exhibited in their log data. Specifically, we used this model of tasks and command patterns to design and implement four collaborative-filtering algorithms to recommend workflows and associated videos.

To form the community for collaborative filtering, we sampled 20,000 users and collected their log data for the period of June 25 to August 25 of 2017. In all, this included 255,643 user sessions and about 20 million command invocations. We applied the model trained using BTM in the first layer of the hierarchical approach on the community data to infer task usage for each user, resulting in a user-task distribution matrix. We then searched for the appearance of each of the 233 patterns acquired from the second layer of the hierarchical approach in the command logs of each user session. This results in a user-pattern frequency matrix.

The recommender system works in three steps. First, for a given user, the system gets his/her task and command pattern usage from the above matrices. Second, it selects either a task (first layer) or a command pattern (second layer) to recommend using collaborative filtering. Finally, it selects a candidate video to recommend based on the chosen task or pattern.

Recommender System Design Space

Based on this hierarchical understanding of user workflows both at a task level (inferred from the topic modeling approach), and at a pattern level (inferred from the frequent pattern mining approach), we propose and evaluate a design space for workflow-based video recommender systems. The first dimension of our design space is *granularity*, basing our recommendation on either a *topic level* or a *pattern level*. Videos that are recommended at a topic level target a general task, *e.g.,* sketching, whereas videos that are recommended at a pattern level target a specific pattern of commands, *e.g.,* drawing a line, applying a constraint, and editing dimensions. The second dimension we evaluated was *topic relevance*. Recommendations were either *most-familiar topic* (MFT), meaning the recommended videos match the user's most commonly-used topic, or *less-familiar topics* (LFT), meaning the recommended videos are outside of the user's most commonly-used topic. The intention is for the MFT recommendations to be more familiar and relevant, and for LFT recommendations to be more novel. Exploring this dimension allows user to explore the tradeoff between relevance and novelty [21]. Based on these two dimensions, we proposed a design space of four workflow based video recommendation algorithms (Table 4).

	Most-Familiar Topic	Less-Familiar Topics
Topic Level	*1. Topic-MFT*	*2. Topic-LFT*
Pattern Level	*3. Pattern-MFT*	*4. Pattern-LFT*

Table 4. A design space for workflow recommender systems.

Recommendation Algorithms

All four algorithms make recommendations in two stages. First, a topic is selected (topic-level algorithms), or a set of five patterns[1] is selected (pattern-level algorithms). Second, based on the selected topic or patterns, videos are chosen that either belong to the topic, or contain the selected patterns. The following sections describe the specific algorithms.

Algorithm 1: Topic-MFT

Step 1: *Compute the task distribution for the target user.*

Step 2: *Choose the task the user has most frequently used.* For example, in Figure 6 we visualize the task distribution for a target user. In this case, we would select Task 5.

[1] We select five patterns to increase the diversity of videos recommended for pattern-level algorithms – we felt that recommending multiple videos of one pattern only would result in videos that were too similar to one another.

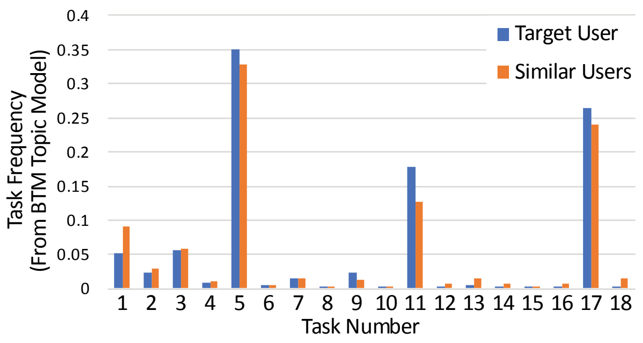


Figure 6. Example task distribution, user vs. similar users.

Step 3: *Select five videos for the chosen task.* We select all videos in the dataset that belong to this task, and compute the similarity between the target user's task usage with all videos selected. Videos with a similarity greater than 0.9 are ranked by their view counts, and the top five videos are selected. The threshold 0.9 followed a similar rationale as introduced in Study 1. The task similarity step guarantees that the videos will be close to the user's typical workflows, and the view count ranking ensures that we select higher-quality videos.

Algorithm 2: Topic-LFT

Step 1: *Compute the task distribution for the target user and all 20,000 other users in the community.*

Step 2: *Find similar users.* We select users that have a task similarity score larger than 0.9 with the target user. Task similarity here is defined as the cosine similarity between the user-task vectors. Figure 6 shows the comparison of task distributions between a target user and similar users.

Step 3: *Compare target user to similar users*. We compute the task weight difference between the target user and similar users, and select the task that has the largest delta between similar users and the target user. For the example in Figure 6, Task 1 would be recommended. We then use a similar algorithm as in Approach 1, Step 3 to select five videos.

Algorithms 3 & 4: Pattern-MFT, Pattern-LFT

Step 1: *Compute the pattern frequency distribution for the target user, and all 20,000 other users in the community.*

Step 2: *Find similar users based on pattern frequency.* Similarity here is defined as the cosine similarity between user-pattern frequency vectors. Since the pattern frequency similarity is much lower than task similarity, it was difficult to determine a threshold for selecting similar users. We chose N=200 to select the top 200 users based on the ranking of pattern frequency similarity with the target user.

Step 4: *Compute expected pattern frequency for the target user.* To calculate the expected frequency for each pattern, we follow the method used by Matejka et al. [28]. We define the expected frequency, ef_{ij}, for pattern p_i and user u_j:

$$ef_{ij} = \sum_{k=1}^{n} w_{jk} pf_{ik}$$

where w_{jk} is the similarity between u_j and u_k, and pf_{ik} is the frequency of pattern p_i for u_k.

Step 5: *Remove previously used patterns.* We then rank the patterns based on the expected frequency and remove patterns that the target user has been observed using.

Step 6 (Algorithm 3): *Select patterns for the user's most relevant task.* In the list of patterns ranked by expected frequency, to select patterns that are more relevant to the user, we select the top five patterns that belong to the most frequently used tasks by the target user.

Step 6 (Algorithm 4): *Select patterns of less relevant task.* In the list of patterns ranked by expected frequency, to select patterns that are less relevant to the user, we select top five patterns that are outside the user's most frequently used tasks (as defined in Algorithm 3).

Step 7: *Choose videos based on patterns.* For each chosen pattern, we first select all videos that contain that pattern. We then rank the selected videos based on their task similarity to the target user, and rank the top 10 by view count. Finally, we select the top-viewed video as the video for that pattern. This is the same method for video selection used in Approach 1 and 2, which is designed to guarantee the video is close to the user's typical workflows and of reasonable quality.

STUDY 2: WORKFLOW RECOMMENDATIONS

To evaluate the proposed algorithms, we conducted a study where we generated a personalized set of videos for participants, and sent them a survey where they could view the videos and rate the recommendations. This follows the methodology used in past work by Matejka et al. [28].

Participants and Procedure

We recruited 8 users that had actively used the software during the past 2 months (1 female, 7 male). With permission, we retrieved participants' natural log data for the past two months, from June 25 to August 25, 2017. We made 20 video recommendations in total for each user – five videos from each of the four algorithms described above. We filtered videos to be shorter than 5 minutes. The videos were presented to participants in random order.

For each video, participants were asked to rate to what extent they would agree with the following statements (1=Strongly Disagree, 5=Strongly Agree): (1) *I was familiar with the workflow (or workflows) shown in this video.* (2) *I may use the workflow (or workflows) shown in this video.* (3) *This video would be a good demonstration for someone who was unfamiliar with the workflow (or workflows) being shown.* The first question (*Familiarity)* was used to evaluate whether the workflows were novel to the user. The second question (*Relevance*) was used to evaluate whether the workflows were relevant to the user. The third question was used to evaluate the quality of the video, and the feasibility of using community generated videos for learning new workflows. Each question was followed by a justification text box.

Quantitative Results and Analysis

Table 5 shows the average rating of each recommendation algorithm on the two dimensions of relevance and familiarity. In general, our results indicate that participants found the recommended videos to be relevant (indicating that they would use the workflow), and familiar (indicating lower novelty). Given the low sample size (n=8) it is difficult to make definitive conclusions, however we do see potential trends of higher familiarity ratings for pattern-level recommendations than topic-level recommendations (p=0.08) and higher relevance ratings for the pattern-level recommendations as well (ns). Our interpretation of pattern-based algorithms generating more familiar and relevant recommendations is because similar users, as identified by pattern frequency similarity, are more likely to use similar patterns. Even if previously-used patterns are removed, the patterns recommended may be of a similar general task.

Algorithm		Relevance Rating	Familiarity Rating
Topic-Level	Most-Familiar Topic	4.13	3.70
	Less-Familiar Topics	4.10	3.58
Pattern-Level	Most-Familiar Topic	4.23	4.08
	Less-Familiar Topics	4.18	3.95

Table 5. Study 2 results.

While the LFT approaches showed some potential impact on the novelty of the recommendations, the novelty ratings were lower than we expected. This could be because we favored relevance in the design of the recommendation algorithms. For instance, in the video selection step, we selected videos based on the similarity between the target user's task usage and the task distribution of the videos, which can cause the recommended videos to be closer to the user's typical workflows. We revisit this issue in our discussion of future work.

In terms of the video quality, the ratings were generally positive, with an average rating of 3.5 for all the recommendations. In their free-form feedback, participants showed a strong preference for videos with audio.

Qualitative Analysis of User Feedback

Through a qualitative analysis of user feedback, we found that users may rate the videos as very familiar, even if they disclosed in their justification responses that they were only partially familiar with the workflow. Part of this issue is that each video may show more than one workflow or command pattern. If part of the video shows a general task that the user is familiar with, the user may rate the video as familiar, even if a subsequence of the video was novel.

Despite the low level of reported novelty, users did report positive attitudes towards the recommended videos and expressed that they would want to try out the workflows in the future. For example, P1 stated:

"I knew how to use all these features, but hadn't really thought of using them in combination this way before. The workflow will be useful in the future."

P2 expressed that he/she is partially familiar with the workflow recommended:

"I am partially familiar with the patch workflow but understand how to use extrude to make a groove to a box. I want to learn more about patch and the video gave a good description of how it can be used."

P3 was excited about one of the recommended workflows:

"Amazing workflow, I was never familiar with such an approach. Maybe because I've never used Remake. I would love to give it a go. This has a lot of uses and potential in the field I'm working, so I would definitely use it."

In conclusion, the algorithms show a strong potential for recommending learning resources that are relevant to the goals of the target user. The study reinforces the tradeoff between the two factors of relevance and novelty in our design space. We see opportunities for improving the algorithms and adjusting the design decisions to make more novel recommendations, which we will discuss as future work.

DISCUSSION AND FUTURE WORK

A diagram summarizing our approach is shown in Figure 7. The overall idea is to first use topic modeling to segment logs into mutually exclusive sets based on high-level tasks (Layer 1), and then to apply frequent pattern mining to each set, to identify finer-grained patterns of command usage (Layer 2). The resulting topics (*i.e.,* tasks) and patterns are then used as input to software support systems, *e.g.*, recommender algorithms, as we have demonstrated.

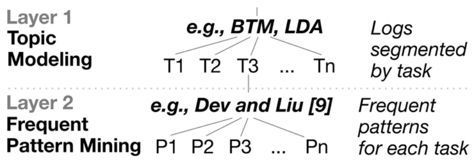


Figure 7. Summary of our hierarchical approach.

While we found that Bi-term Topic Modeling and Dev and Liu's algorithm for frequent pattern mining were effective, we see the hierarchical approach as being largely independent of these specific techniques, or the assumptions used to tune them to our dataset. In particular, a key finding from this work is that it is valuable to first use topic modeling to segment logs, and then to apply pattern mining to the resulting segments, because it prevents frequent user activities (*e.g.,* sketching activities for our application) from drowning out other distinct activities performed in the software.

The remainder of this section discusses opportunities for future work to develop and build on this approach.

Incorporating Heuristics Based on Expertise Levels

In our approach, we tried to minimize human input. Apart from the expert labeling of topics, the process is data driven. However, we see opportunities to incorporate heuristics to enable more meaningful workflow recommendations (an approach used at the command level in [28]). Though we did not report on it, we had experts evaluate the expertise levels of the 18 tasks produced by our topic modeling, and this data could be incorporated into a workflow recommender system (*e.g.,* to recommend intermediate sketching workflows to users who have been observed doing beginner sketching).

User In-the-Loop Recommender Systems

In this work, we did not apply filtering to control the quality of community-generated content, but we see the potential of integrating such quality-control methods (*e.g.,* machine learning methods to predict video quality [19]). Prior work has shown that "learner-sourcing" systems can harness input from learners to improve the quality of content over time (*e.g.*, ask learners to label activities performed in MOOC videos [17]). Similar approaches could be used to refine the recommendations made by a workflow recommender system, so that recommendations improve over time.

Generalizability

Though we developed our hierarchical approach for a specific 3D design application, the approach can be applied to other applications as well. Our approach is based on command log data, which is commonly logged in feature-rich software. More unique is that we also leverage data from a user-generated video repository, where the videos are supplemented with command log data. Software companies looking to apply our approach could curate such marked-up video repositories with existing tools, *e.g.,* Autodesk Screencast's public SDK [4]. Alternatively, prior work has demonstrated approaches to extract command data from existing video repositories [5, 16]. In this way, our approach can be generalized to other applications and software domains.

Limitations

A limitation of our work that could impact its generalizability is that we used heuristic or ad-hoc approaches to choose some parameters (*e.g.,* the cosine similarity threshold of 0.9, and the cutout cohesion score of 2), which would need to be adapted for other data sets. More broadly, the use of cosine similarity is a limitation as it comes from a non-probabilistic modeling approach, and thus it would be valuable to investigate probabilistic-based similarity metrics, such as Kullback-Leibler Divergence [25], in future work. When selecting videos to recommend, we also used an ad-hoc method to select videos that are similar to a user's typical workflow, which favored "Relevance" over "Novelty" in our design space. Future work could investigate more rigorous approaches to tuning the similarity threshold, or more generally modeling the similarity/novelty of workflow recommendations.

CONCLUSION

In this paper, we have proposed a hierarchical approach to classifying user workflows by first applying topic modeling to identify high-level tasks, and then applying frequent pattern mining to identify distinct command patterns for each task. An evaluation showed encouraging evidence that topic modeling can effectively categorize logs into meaningful high-level tasks. As well, the hierarchical approach appears to help identify a larger variety of distinct command patterns. Based on this approach, we proposed a design space of workflow-based recommender systems. An evaluation of four such algorithms was encouraging, and suggests that this approach has the potential to effectively support software users.

ACKNOWLEDGEMENTS

Thanks to Aishwarya Balamukundan and Eduardo Teixeira for help with data collection, to Bryce Heventhal for his domain expertise, to all our study participants for their time, to the reviewers for their valuable feedback and suggestions, and to Anhong Guo for his help and support.

REFERENCES

1. Rakesh Agrawal, Tomasz Imielinski, and Arun Swami. 1993. Mining Association Rules Between Sets of Items in Large Databases. In *Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data* (SIGMOD '93). ACM, New York, NY, USA, 207–216. http://dx.doi.org/10.1145/170035.170072
2. Rakesh Agrawal and Ramakrishnan Srikant. 1995. Mining Sequential Patterns. In *Proceedings of the Eleventh International Conference on Data Engineering* (ICDE '95). IEEE Computer Society, Washington, DC, USA, 3–14. http://dl.acm.org/citation.cfm?id=645480.655281
3. Eytan Adar, Jaime Teevan, and Susan T. Dumais. 2008. Large scale analysis of web revisitation patterns. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '08). ACM, New York, NY, USA, 1197-1206. https://doi.org/10.1145/1357054.1357241
4. Autodesk. Retrieved Dec 21, 2017. About the Screencast API and SDK. https://goo.gl/eNPbw4
5. Nikola Banovic, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2012. Waken: reverse engineering usage information and interface structure from software videos. In *Proceedings of the 25th annual ACM symposium on User interface software and technology* (UIST '12). ACM, New York, NY, USA, 83-92. https://doi.org/10.1145/2380116.2380129
6. Scott Bateman, Jaime Teevan, and Ryen W. White. 2012. The search dashboard: how reflection and comparison impact search behavior. In Proceedings of the SIGCHI Conference on Human Factors in *Computing Systems* (CHI '12). ACM, New York, NY, USA, 1785-1794. http://dx.doi.org/10.1145/2207676.2208311
7. David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent dirichlet allocation. *J. Mach. Learn. Res.* 3: 993-1022.
8. Boris Cuke, Bart Goethals, and Celine Robardet. A new constraint for mining sets in sequences. In *Proceedings of the 2009 SIAM International Conference on Data Mining*. 317–328. http://dx.doi.org/10.1137/1.9781611972795.28
9. Himel Dev and Zhicheng Liu. 2017. Identifying Frequent User Tasks from Application Logs. In *Proceedings of the 22nd International Conference on Intelligent User Interfaces* (IUI '17). ACM, New York, NY, USA, 263-273. https://doi.org/10.1145/3025171.3025184
10. Gensim Library. Retrieved July 10 2017 from https://radimrehurek.com/gensim/
11. Thomas L. Griffiths, Mark Steyvers, and Joshua B. Tenenbaum (2007). Topics in semantic representation. *Psychological review*, 114(2), 211.
12. Tovi Grossman, George Fitzmaurice, and Ramtin Attar. 2009. A survey of software learnability: metrics, methodologies and guidelines. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '09). ACM, New York, NY, USA, 649-658. https://doi.org/10.1145/1518701.1518803
13. Tovi Grossman and George Fitzmaurice. 2010. ToolClips: an investigation of contextual video assistance for functionality understanding. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '10). ACM, New York, NY, USA, 1515-1524. https://doi.org/10.1145/1753326.1753552
14. Tâm Huynh, Mario Fritz, and Bernt Schiele. 2008. Discovery of activity patterns using topic models. In *Proceedings of the 10th international conference on Ubiquitous computing* (UbiComp '08). ACM, New York, NY, USA, 10-19. http://dx.doi.org/10.1145/1409635.1409638
15. Juho Kim, Robert C. Miller, and Krzysztof Z. Gajos. 2013. Learnersourcing subgoal labeling to support learning from how-to videos. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems* (CHI EA '13). ACM, New York, NY, USA, 685-690. https://doi.org/10.1145/2468356.2468477
16. Juho Kim, Phu Tran Nguyen, Sarah Weir, Philip J. Guo, Robert C. Miller, and Krzysztof Z. Gajos. 2014. 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). ACM, New York, NY, USA, 4017-4026. https://doi.org/10.1145/2556288.2556986
17. Juho Kim, Philip J. Guo, Carrie J. Cai, Shang-Wen (Daniel) Li, Krzysztof Z. Gajos, and Robert C. Miller. 2014. 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). ACM, New York, NY, USA, 563-572. https://doi.org/10.1145/2642918.2647389
18. Benjamin Lafreniere, Tovi Grossman, and George Fitzmaurice. 2013. Community enhanced tutorials: improving tutorials with multiple demonstrations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '13). ACM, New York, NY, USA, 1779-1788. https://doi.org/10.1145/2470654.2466235
19. Benjamin Lafreniere, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2014. Investigating the feasibility of extracting tool demonstrations from in-situ video content. In *Proceedings of the SIGCHI*

Conference on Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA, 4007-4016. http://dx.doi.org/10.1145/2556288.2557142

20. Gilly Leshed, Eben M. Haber, Tara Matthews, and Tessa Lau. 2008. CoScripter: automating & sharing how-to knowledge in the enterprise. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '08). ACM, New York, NY, USA, 1719-1728. https://doi.org/10.1145/1357054.1357323
21. Wei Li, Justin Matejka, Tovi Grossman, Joseph A. Konstan, and George Fitzmaurice. 2011. Design and evaluation of a command recommendation system for software applications. *ACM Trans. Comput.-Hum. Interact.* 18, 2, Article 6. http://dx.doi.org/10.1145/1970378.1970380
22. Wei Li, Tovi Grossman, and George Fitzmaurice. 2014. CADament: a gamified multiplayer software tutorial system. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '14). ACM, New York, NY, USA, 3369-3378. https://doi.org/10.1145/2556288.2556954
23. Frank Linton, and Hans-Peter Schaefer (2000). Recommender systems for learning: building user and expert models through long-term observation of application use. *User Modeling and User-Adapted Interaction*, 10(2-3), 181-208.
24. Frank Linton, Andy Charron and Hans-Peter Schaefer (2000). OWL: A recommender system for organization-wide learning. *Educational Technology & Society*, 3(1), 62-76.
25. David J.C. MacKay (2003). *Information Theory, Inference, and Learning Algorithms* (First ed.). Cambridge University Press. p. 34.
26. Sylvain Malacria, Joey Scarr, Andy Cockburn, Carl Gutwin, and Tovi Grossman. 2013. Skillometers: reflective widgets that motivate and help users to improve performance. In *Proceedings of the 26th annual ACM symposium on User interface software and technology* (UIST '13). ACM, New York, NY, USA, 321-330. http://dx.doi.org/10.1145/2501988.2501996
27. Heikki Mannila, Hannu Toivonen, and A. Inkeri Verkamo. 1997. Discovery of Frequent Episodes in Event Sequences. *Data Min. Knowl. Discov.* 1, 3 (Jan. 1997), 259–289. http://dx.doi.org/10.1023/A:1009748302351
28. Justin Matejka, Wei Li, Tovi Grossman, and George Fitzmaurice. 2009. CommunityCommands: command recommendations for software applications. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology* (UIST '09). ACM, New York, NY, USA, 193-202. https://doi.org/10.1145/1622176.1622214
29. Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2011. Ambient help. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '11). ACM, New York, NY, USA, 2751-2760. https://doi.org/10.1145/1978942.1979349
30. Hartmut Obendorf, Harald Weinreich, Eelco Herder, and Matthias Mayer. 2007. Web page revisitation revisited: implications of a long-term click-stream study of browser usage. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (CHI '07). ACM, New York, NY, USA, 597-606. https://doi.org/10.1145/1240624.1240719
31. Jaimie Y. Park, Neil O'Hare, Rossano Schifanella, Alejandro Jaimes, and Chin-Wan Chung. 2015. A Large-Scale Study of User Image Search Behavior on the Web. *In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (CHI '15). ACM, New York, NY, USA, 985-994. https://doi.org/10.1145/2702123.2702527
32. Marco Pennacchiotti and Siva Gurumurthy. 2011. Investigating topic models for social media user recommendation. In *Proceedings of the 20th international conference companion on World wide web* (WWW '11). ACM, New York, NY, USA, 101-102. http://dx.doi.org/10.1145/1963192.1963244
33. Adam Perer and Fei Wang. 2014. Frequence: interactive mining and visualization of temporal frequent event sequences. In *Proceedings of the 19th international conference on Intelligent User Interfaces* (IUI '14). ACM, New York, NY, USA, 153-162. http://dx.doi.org/10.1145/2557500.2557508
34. Daniel Ramage, David Hall, Ramesh Nallapati, and Christopher D. Manning. 2009. Labeled LDA: a supervised topic model for credit attribution in multi-labeled corpora. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1 - Volume 1* (EMNLP '09), Vol. 1. Association for Computational Linguistics, Stroudsburg, PA, USA, 248-256.
35. Amit Singhal. 2001. Modern information retrieval: A brief overview. IEEE Data Engineering Bulletin, 24(4), 35-43.
36. SPMF Library. Retrieved July 10 2017 from: http://www.philippe-fournier-viger.com/spmf/
37. Gang Wang, Xinyi Zhang, Shiliang Tang, Haitao Zheng, and Ben Y. Zhao. 2016. Unsupervised Clickstream Clustering for User Behavior Analysis. In *Proceedings of the 34th Annual ACM Conference on Human Factors in Computing Systems* (CHI '16). ACM, New York, NY, USA, 225-236. https://doi.org/10.1145/2858036.2858107

38. Miaomiao Wen and Carolyn Penstein Rose. 2014. Identifying Latent Study Habits by Mining Learner Behavior Patterns in Massive Open Online Courses. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (CIKM '14). ACM, New York, NY, USA, 1983-1986. http://dx.doi.org/10.1145/2661829.2662033

39. Xiaohui Yan, Jiafeng Guo, Yanyan Lan, and Xueqi Cheng. 2013. A biterm topic model for short texts. In *Proceedings of the 22nd international conference on World Wide Web* (WWW '13). ACM, New York, NY, USA, 1445-1456. https://doi.org/10.1145/2488388.2488514

40. Yi Zhang, Jamie Callan, and Thomas Minka. 2002. Novelty and redundancy detection in adaptive filtering. In *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval* (SIGIR '02). ACM, New York, NY, USA, 81-88. http://dx.doi.org/10.1145/564376.564393