Switter: Supporting Exploration of Software Learning Materials on Social Media

Volodymyr Dziubak[1] **Patrick Dubois**[1] **Andrea Bunt**[1] **Michael Terry**[2]

[1]University of Manitoba,
Winnipeg MB, Canada
{vdziubak, umdubo26, bunt}@cs.umanitoba.ca

[2]University of Waterloo
Waterloo ON, Canada
mterry@cs.uwaterloo.ca

ABSTRACT

There is always something new to learn about feature-rich software, even for experienced users. This paper focuses on a specific type of learning activity that we refer to as *ad libitum exploration*. Based on an interview study with 11 experienced software users, we define ad libitum exploration as the process of routinely seeking new software knowledge, without necessarily having a specific problem to solve. To support this activity, we designed Switter, an alternative Twitter client embedded in a replica of Photoshop's user interface. Given a tweet referencing a tutorial, Switter highlights the interface elements mentioned in the tutorial in the interface replica. Switter also allows users to filter tweets by clicking tools in the interface replica. Through a weeklong field study with nine Photoshop experts, we found that Switter supports a range of software learning objectives, from focused exploration targeting known weaknesses, to the discovery of novel command combinations.

Author Keywords

Feature-rich software; software learnability; social media

ACM Classification Keywords

H.5.m. Information interfaces and presentation (e.g., HCI)

INTRODUCTION

Feature-rich applications like Photoshop and AutoCAD contain thousands of operations, and can be applied to countless problems. This versatility means that one can never truly master feature-rich software—there is always the opportunity to learn something new, whether it is a new workflow for a familiar task, or a completely new use case for the software.

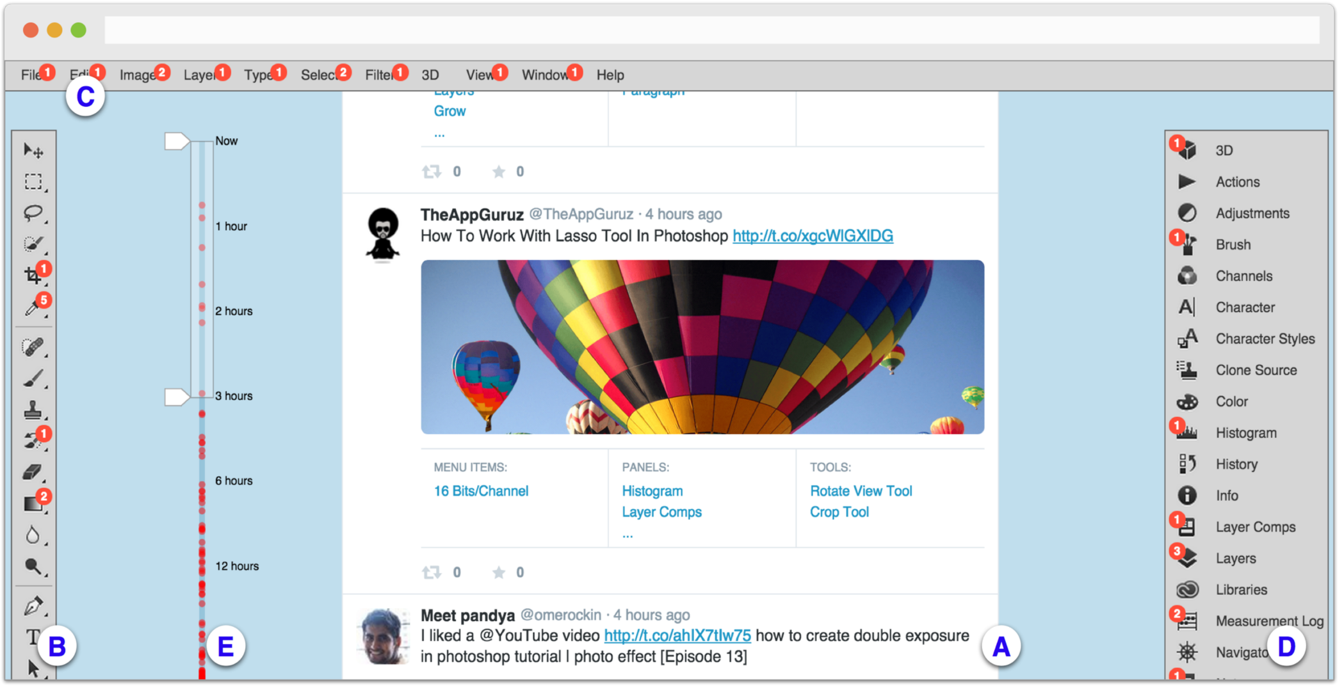
A rich history of prior work has examined how to improve the process of learning software (e.g., [2,8,16,23]), but has mainly focused on learning activities motivated by a specific task (e.g., [7,19]). For example, users often seek materials on how to change colors in a picture because they want to enhance their favorite photo, not because they want to learn Photoshop. The actual learning of Photoshop's features occurs as a side effect of completing the task [32]. We investigate a different activity for learning software, where users actively seek out new software knowledge due to general curiosity or desire to improve their skills. For example, a user might regularly look through a Photoshop forum hoping to pick up new tips and techniques that might be helpful in the future. We refer to such a learning activity as *ad libitum exploration* (Latin for "at one's pleasure").

In this work, we first characterize ad libitum exploration through an exploratory study with 11 experienced designers and artists. Based on our study results, we created Switter, an ad libitum-centric Twitter client that organizes tweets that refer to Photoshop tutorials within a web-based replica of Photoshop's interface (Figure 1). Below each tweet, Switter lists the Photoshop commands mentioned in the linked tutorial to help people assess whether the tutorial is worth their time. Switter also uses the command annotations to link tweets directly to the interface replica, enabling users to locate potentially interesting content by clicking on any user interface element.

To validate Switter's design, we conducted a weeklong field study with nine experienced Photoshop users. Our results showed that Switter helped participants to 1) filter, locate, and assess content which is likely of interest to them, 2) validate their existing knowledge, and 3) learn new skills and techniques. Collectively, these results suggest the value of tools that explicitly support ad libitum exploration backed by social media.

In summary, this paper makes the following contributions:

- We characterize ad libitum exploration of software learning materials through an initial study with expert designers and artists.
- We present Switter, an ad libitum-centric Twitter client that organizes tutorial-related tweets within a replica of the target software's UI (in our case - Photoshop).



TheAppGuruz @TheAppGuruz · 4 hours ago
How To Work With Lasso Tool In Photoshop http://t.co/xgcWlGXIDG

Meet pandya @omerockin · 4 hours ago
I liked a @YouTube video http://t.co/ahlX7tlw75 how to create double exposure in photoshop tutorial I photo effect [Episode 13]

Figure 1. Switter supports ongoing learning of software by embedding an augmented Twitter timeline (A) within a web-based replica of that application's interface (here, Photoshop). Given a tweet referencing a tutorial for the target application, Switter highlights the user interface elements mentioned in that tutorial (B, C, D), and lists these tools below the tweet itself (A), providing an at-a-glance, compact summary of the tutorial content. Switter also displays recent Twitter activity related to the software in a historical summary (E).

- We present results from a field study that validates Switter's design and demonstrates its utility for ad libitum exploration.

RELATED WORK

Research in software learning spans decades. In this section, we look at ad libitum exploration in the context of work that characterizes software learning. We then look at four common approaches to support software learning: command recommender systems; novel tutorial formats; systems that facilitate finding and selecting tutorials; and systems that embed learning aids within the application itself.

Characterizing Software Learning

Within the scope of general learning theory, we consider ad libitum exploration to be a form of self-regulated learning. Following Zimmerman's definition, self-regulated learners are "metacognitively, motivationally, and behaviorally active participants in their own learning" [33]. Prior work on software learning strategies points to evidence of such learners among users of software programs (e.g. [6,32]). For example, Rieman in his field study of software learning strategies [32] reported that some users were browsing the interface and documentation because of curiosity about features of the software. Dorn and Guzdal in their study of learning practices among web designers and web developers [6] also observed some users learning software due to curiosity and a desire to stay up-to-date. However, prior work on software learning has not investigated such task-free exploration in detail.

We also consider ad libitum exploration to be a type of *extended learning* as discussed by Grossman et al.'s survey on software learnability [13]. Their survey differentiates between *initial learning,* where novice users gain initial proficiency, and *extended learning,* where users' performance changes over time. Our work seeks to add to this discussion by focusing on a particular method of engaging in extended learning. Specifically, as we show later in the paper, it is not uncommon for seasoned users of an application to routinely search for new articles, videos, or tutorials that can help them improve their abilities. Importantly, this form of learning is rather open-ended: users are not seeking to learn a specific thing, but rather to generally improve their abilities, potentially in ways they have never considered.

In our work, we focus specifically on task-free self-regulated extended software learning. We look at its prevalence among a specific group of expert software users, investigate learning activities and objectives that characterize it, and suggest design requirements to support such type of software learning. We introduce a new term for this type of extended software learning activity ("ad libitum exploration"), as Rieman's term of "task-free exploration" [32] does not capture all of its characteristics.

Command Recommender Systems

Command recommender systems aim to help a user expand their command vocabulary [12] by highlighting new commands that the system believes are relevant to the user's work. Examples include OWL [26] and CommunityCommands [30], which apply collaborative filtering techniques on community usage logs to find potentially relevant commands for the user. QFRecs [17] identifies relevant commands by mining web documentation for logical command clusters [11]. Another example, Patina [29], overlays the interface with command usage heatmaps, revealing commands that are heavily used within the community. Command recommender systems provide excellent support for maintaining command awareness and expanding one's command vocabulary, but they do not demonstrate how commands can be used for specific tasks or how others use commands in their projects. Thus, recommender systems provide only limited support for ad libitum exploration.

Promoting Learning via Novel Tutorial Formats

Web-based tutorials are one of the most common learning sources. Given their ubiquity and utility, a significant number of research projects have focused on how to improve the process of learning software via tutorials, particularly for novice users. For example, Sketch-Sketch Revolution [8] provides scaffolding and stroke guidance in tutorials that focus on drawing techniques, enabling users to experience greater degrees of success than they might otherwise be capable of. TApps [23] uses selective automation to reduce the effort involved in completing a tutorial, while still allowing users to experiment on their own. Chronicle [15] provides interactive, annotated document histories enabling users to see the evolution of specific workflows. Researchers have also added gamification elements as a way to motivate tutorial completion [5,24].

Other work has sought to improve the utility of tutorials by leveraging community feedback. For example, TaggedComments [1] promotes and integrates the comments that users post to tutorials to highlight relevant community insight. FollowUs [21] augments tutorials with multiple demonstrations collected from other users who have completed the tutorial.

In general, the above approaches facilitate and simplify the use of an individual software learning resource, but do not provide a sense of the most recent trends and techniques.

Supporting Tutorial Selection

Given the large number of software learning resources available on the web, prior research has examined the problem of finding useful tutorial content. As an example, Ekstrand et al.'s system for selecting tutorials via web search [7] uses recently used commands and other application content to supplement the user's search query. Each result returned is annotated with the commands mentioned in the web page. This approach facilitates searching, but is not suitable for ad libitum exploration, when users often do not know what exactly they are looking for.

Another command-centric approach can be found with the Delta system [19], which supports selecting a workflow from a corpus by allowing users to compare the commands involved. Their approach, though, did not consider the fast-paced nature of social media. As we discuss later, one of the main requirements for an ad libitum-centric tool is the ability to filter large amounts of continuously updated content to identify those bits of information that are most relevant or meet the user's interests.

Other work has explored automated tutorial retrieval based on user activity within the software. The Ambient Help system [27] automatically selects learning resources according to the user's current interactions with the application, displaying the selected resources on a secondary monitor. Our work is similar in spirit to Ambient Help, but makes use of a community's ongoing appraisal of what is topical and relevant. Accordingly, our approach can help a user get outside their "comfort zone," since it is not keying off their current interactions with the software.

Application-Integrated Learning Resources

A wide range of systems has sought to more tightly integrate learning resources with the target application. Some have approached this integration problem from the point of view of individual tutorials, such as Tutorial Stencils [16], which highlight the commands required at each step of a tutorial within the application itself. Pause and Play [31], focuses on video tutorial pacing by, for example, using command invocations within the application to control the tutorial's progression.

Like Switter, others have explored ways to integrate social media and software applications. Specifically, TwitApp [25] embeds a Twitter client into a feature-rich application as a way to support software micro blogging. TwitApp focuses on providing rich application-level support for authoring tweets, with the primary aim of allowing collaborators to share and critique their ongoing work.

More similar to our approach are systems that link learning resources to individual user interface elements. ToolClips [14] extends the notion of traditional text-based tooltips by attaching short video demonstrations to individual tools. Subsequent work showed that such tool demonstrations could potentially be extracted automatically from screen-captured workflows [22]. Such an approach demonstrates usage of a specific tool, but it does not show the high-level tasks where the tool can be used.

Finally, IP-QAT [28] and LemonAid [3] support crowdsourced in-application Q&A, allowing users to attach questions, answers and tips directly to individual user interface elements. Intertwine [10] identifies commands and tools mentioned in the top-most tab of a web browser, and places stars next to those commands in the software's interface. These approaches extend the functionality and usability the target application, but do not help users explore and assess a broader range of posted learning materials.

EXPLORATORY INTERVIEW STUDY

Informally, we observed that some people (including one of the paper authors) seem to regularly check online resources (e.g. Reddit, Twitter) to locate new, potentially relevant software learning materials, such as tutorials. Notably, this activity does not represent a targeted search to assist with an existing task.

To determine whether this activity was more widespread, we conducted semi-structured interviews with 11 artists and designers (6 female). We recruited participants via snowball sampling, using the authors' personal contacts, and through notices posted on a university campus, Reddit, and Twitter. Our participants were between 18 and 48 years of age and had at least one year of experience with image-manipulation or graphics software (e.g., Adobe Photoshop and Illustrator) or other design software (e.g., Sketch[1]). Nine participants used the software professionally, while two used the software extensively as part of their current training. Interviews were conducted either in-person or via Skype, and lasted between 30 and 45 minutes. Participants were remunerated with a $15 gift card.

In our interviews, we asked participants what, if anything, motivates them to learn about the software they regularly use. We also asked participants to describe the specific learning strategies they use and how well their current strategies support their learning objectives.

Interviews were transcribed in full. Data from the transcripts were analyzed by creating affinity diagrams using a bottom-up inductive approach [4]. From these affinity diagrams, we held joint data interpretation sessions among the paper authors where we extracted common themes.

The Desire to Continue Learning

All of our participants emphasized their desire to continually learn new things, as the following two quotes illustrate:

> *[I've been using Photoshop] since I left school, which was when I was 15. So, 10 years [of experience] … I am learning all the time. You learn stuff you didn't realize you could do. (P3)*

> *I usually look at posts that… describe something I haven't done before, so maybe [a] new style or new tools I haven't accessed, or I might not have been really comfortable with them. So I am always looking for stuff that's pretty much new to me. And I also… look at the stuff that I've done before, [that's in] similar styles to mine. (P2)*

These quotes not only emphasize that the desire to learn is still strong after 10 years of experience (P3), but also highlight that participants seek out topics completely new to them (P2), or which enable them to compare others' methods of solving a problem to their existing practices (P2).

Across all interviews, participants expressed a variety of learning objectives including: staying up-to-date with the latest industry standards, improving the efficiency of their workflows, improving their end products, reinforcing existing skill sets, and uncovering new tools or capabilities. These findings are consistent with Lafreniere et al.'s analysis of comments users post to online tutorials once they have completed them. Their findings revealed a number of tutorial users that go beyond task-specific learning, such as seeking to expand one's skills set, or to shadow the techniques of other expert users [20].

Characterizing Continual Learning

We found evidence that participants regularly seek out resources that provide pointers to new instructional materials, which they hope will lead them in unexpected directions. However, participants also recognized that they may not find anything new, and thus stressed the need to filter content. We expand on these themes below.

Habitual Monitoring. For some participants, the learning process is an integral part of their daily routine. Participants periodically check their favorite "trusted" websites, such as official Adobe forums or Reddit, with the goal of staying aware of what is happening in their field:

> *[Design] Reddit is nearly daily. For work, I'd say it'd be every other day… I definitely like to… pop [in] and to see what's going on. (P5)*

Another popular trusted source is Twitter, where participants reported following key individuals for new tips and tricks:

> *On Twitter, for example, ...I follow people that are in [the] design community, that are also in my field. They regularly share links to interesting websites, or articles, or tutorials... and if it fits my interest, I will click on it and investigate further. (P1)*

These behaviors suggest a clear desire to stay current and to discover new techniques that may be useful to the participant.

Looking for the Unexpected. Our participants reported that curiosity and accidental discovery also play important roles in their learning process. Due to the large number of features in programs such as Photoshop, even our experienced participants were not sure they knew how all the available tools work. Some participants described specializing in certain aspects of the program they use, yet still being keen to discover new tools that might be relevant to their work:

> *Sometimes I will be stuck on a painting and I will start staring off into the space and I will look at the buttons and will be like 'hey, wait, I do not know what that does'. So, I will open up a new document and I will start playing with it. (P10)*

[1] https://www.sketchapp.com/

As another example of an individual seeking the unexpected, one participant reported watching tutorials on YouTube, in hopes of seeing something new through recommended videos:

> *And I start with a YouTube video of something I do know and then just seeing what else pops up. So that you learn stuff that you don't think to ask. (P11)*

Low Expectations for Learning. In some cases, participants expressed low expectations for learning something new, but still engaged in watching instructional videos, in case there was something they could use in their own work:

> *You see what people do in interesting ways and they take these tools and they use them in new ways and they post their weird creations... and I will look at that, and even though the method is not going to be useful to me, you never know. You might learn something from that method. (P5).*

The Need to Filter Content. Our interviews showed that with more experience, finding content for learning that is personally interesting and new becomes more and more difficult. The main challenge is that the more knowledge people have, the less likely it is they will learn something from a tutorial. As a consequence, participants spoke of the high cost of viewing a tutorial and the desire to ensure that they would benefit from it:

> *Tutorials are usually time-consuming, so you have to be sure that it's worth your time... and to determine that... I would really be in need of the skill [covered in the tutorial]. (P1)*

In summarizing our exploratory study, we found that experienced software users continually seek to learn and improve their abilities, but engage in ad libitum exploration with the knowledge that they may not always find something useful to them. The cost associated with finding new and relevant information raises the importance of assessing a resource's learning potential quickly and accurately.

SUPPORTING AD LIBITUM EXPLORATION

Reiterating our definition in the Introduction, we define ad libitum exploration as the process where users routinely seek information that may help them improve, deepen, or expand their skills and knowledge of a specific application. Building on the results of our study, we further note that users understand that they may not learn anything when engaged in this process, and thus desire information that helps them assess the learning potential of a given resource.

Supporting Ad Libitum exploration: Design Desiderata

Drawing on the findings from our exploratory study, we define the following set of design desiderata for systems that support ad libitum exploration:

- *Continuous flow of recent content.* To support the habitual nature of ad libitum exploration, the system should present new content that users can access on demand (e.g. when they are taking a break from their work). Ideally, this content should reflect recent trends to help people stay up-to-date with the field.
- *Serendipitous discovery*. The system should support the discovery of new content that diverges from a user's typical practices.
- *Filtering and browsing*. The system should allow users to browse and filter learning materials. For example, the presentation of an individual content item should be descriptive enough to help people decide whether watching or reading the content is going to be beneficial for them. The system should also include filtering capabilities that help the user target topics of interest.

Limitations of Current Strategies

In our study, we found a number of ways ad libitum practices could be improved. For example, social media, such as Twitter or Reddit, provide users with streams of links to new content, but often lack adequate information to help users determine whether it is worth viewing the full source material. Conversely, dedicated learning portals, such as the official Adobe forums, often provide high quality learning materials, but the pace of information flow on such portals is not as rapid as on social media, such as Reddit and Twitter.

We also note that search engines help people find instructional materials for specific problems, but are not necessarily conducive for helping users discover or browse materials that push one outside one's typical work practices.

To address these limitations, we designed and developed Switter, a system for supporting ad libitum exploration.

SWITTER

Switter is an alternative Twitter client for software-centric learning that allows users to browse tutorials for a specific software product (in our implementation, Photoshop). Switter (Figure 1) projects the content referenced by a tweet into a user interface replica, and provides a Twitter timeline in place of where the document would be found in the replicated application. This enables users to browse and explore tutorials broadcasted via Twitter using the target application's interface as a navigation aid. Our current implementation of Switter is a prototype that makes use of a Wizard-of-Oz backend for extracting commands from tutorials, and inserting tweets about Photoshop into a read-only Twitter timeline.

Switter's interface consists of three main components: the Twitter timeline (Figure 1, A), the interface replica (Figure 1, B-D), and the historical summary (Figure 1, E). We discuss the role of each interface component in the following subsections.

Twitter as a Data Source

To provide users with *a continuous flow of recent content* (our first design goal), we use Twitter as a source for discovering new tutorial-related information. This decision was grounded in the results of our exploratory study, where

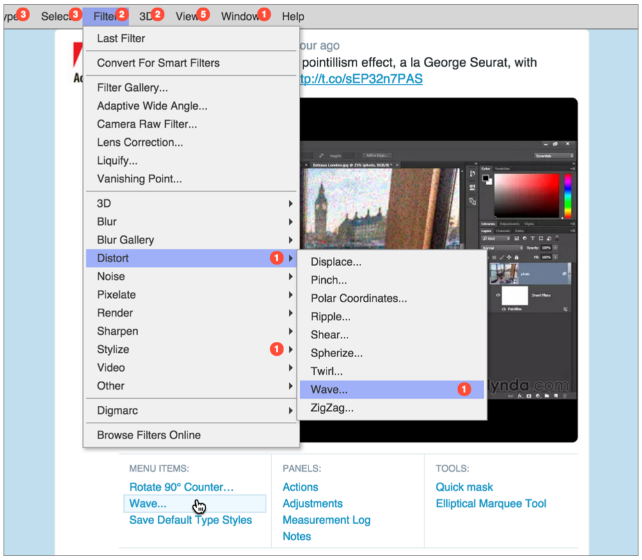

our ago
pointillism effect, a la George Seurat, with
tp://t.co/sEP32n7PAS

Figure 2: When the user clicks a command below the tweet, Switter highlights the referenced interface element in the interface replica and reveals its location.

our participants positioned Twitter as one of the trusted places they used to discover new relevant information.

To get a sense of the potential volume of tutorial-related information on Twitter, we collected and examined a stream of sequential tweets that contained the word "Photoshop" over a one-hour time period. During this single hour, about 30 tweets (of a 300 total tweets) linked to Photoshop learning resources. We believe this rate of information flow to be sufficient for providing users with fresh learning material in a continuous manner.

Projecting Tutorials onto the Software Interface

To address the need to filter and browse learning materials, we highlight the areas of the interface mentioned in the tutorial. This technique 1) provides awareness of the volume and breadth of commands used in the tutorials, and 2) enables users to browse tweets from an interface-centric perspective. For completeness, our replica contains the main menu (Figure 1, C), the toolbar (Figure 1, B), and the list of accessible modal panels (Figure 1, D).

Switter highlights a command or tool by placing a red dot next to it, with a number indicating the number of linked tutorials. For the purpose of the study, we manually extract commands from online content, as we focus on interaction and design. In the future this could be automated using recent advances in command extraction (e.g. [9,27,31]).

To support navigating and browsing tweets, the user can click on a tool or command in the interface replica. Switter responds by reducing the list of tweets to only show those that mention the item selected.

Switter also augments each tweet with the list of commands that it references. This supplemental information seeks to help people make decisions about the utility of the individual tutorial. When the user clicks on a menu item or a tool in the list below the tweet, Switter automatically reveals the referenced user interface element in the replica (see Figure 2).

Temporal Awareness via Historical Summary

Switter includes a historical summary of tweet activities (Figure 1, E) to provide both awareness and filtering capabilities. For awareness, each tweet's posted date is mapped to the respective point on the timeline. This allows users to get a sense of the volume of tutorials as well as their distribution over time. For example, a user can see many tweets were posted in the last three hours, providing cues as to whether or not the volume of new information is worth browsing. Users can also use the historical view to filter the tweets projected onto the interface replica by specifying a range of time.

FIELD EVALUATION

To test the utility of Switter, we conducted a weeklong field study. The main goal of the study was to see how people would adopt the tool, and to gain initial insight into Switter's ability to support ad libitum exploration.

Participants

We recruited nine designers and photographers (3 female) who use Photoshop as their main working tool. All of our participants were between 21 and 45 years old, and have been working with Photoshop for at least one year. We recruited participants through online postings on Reddit and via snowball sampling. Participants received a $75 gift card for their participation.

Procedure and Data Collection

Participants were asked to use the system at least once per day over a period of seven days. We did not give participants any specific task to perform, but rather asked them to browse through the content, to look for something that would catch their interest, or that could potentially teach them something new. We did not specify how much time they should spend in the system each day, nor how many tutorials to view.

We asked participants to fill out a short online journal entry at the end of each day to record their experiences with Switter that day. We also conducted two semi-structured interviews with each participant: one in the middle of the study (day 3) and one at the end of the study (day 7). Finally, we logged key interactions with the Switter interface.

We analyzed the qualitative data using the same analysis techniques as in our exploratory interview study.

Switter Content

We deployed Switter as a standalone web application. Throughout this proof-of-concept field study, we manually annotated the latest tweets pointing to Photoshop tutorials. To ensure a relatively realistic flow of information, we wrote a script to publish the content in this pool of annotated tutorials to Switter in a periodic manner, using randomization to assure irregularity of the content stream.

We ran the entire field study over a period of 11 days, with participants joining at different times. Consequently, Switter's initial state was not the same for all participants. These different initial states helped us understand Switter's utility in a variety of contexts (e.g., for late starters, they were presented with a backlog of tweets that they could sift through).

Throughout the study, two of the paper authors actively monitored Twitter for new tweets that linked to Photoshop tutorials. For each such tweet, we manually labelled the tweet with all commands referenced in the tutorial, and added the labelled tweet to a repository. A randomized script then gradually delivered Tweets from the repository to the participants. This procedure made Switter's data flow similar in nature to the continuous data flow of Twitter. Over the 11 days, Switter displayed 311 tweets with a median of 30 tweets per day (IQR=9).

Findings: Usage Logs and Daily Journals

To gain some insight into how much people used Switter, we calculated the number and duration of usage sessions. We define a usage session as an interval of activity within Switter longer than one minute, separated from other sessions by at least one hour of inactivity. We used the above heuristics to dismiss sessions that likely did not include true interaction. For example, if the user leaves the system open in the browser and occasionally hovers over the page while switching tabs, we did not count this as a session. For these reasons, we believe our summary statistics represents only a conservative report of engagement with Switter.

We observed a total of 56 sessions, with a median session duration of 24.2 minutes (IQR=27.5). Overall, the duration of user sessions was skewed towards longer times, with Q3=45min. In most cases, participants had one session per day. However, in some cases the system was used more extensively. For example, on most days, P6 had three interaction sessions and on one day, P3 had four sessions of interaction.

As part of the journal entries, we asked participants to indicate how many new things they learned that day and how many tutorials caught their interest. Despite daily email reminders, most participants forgot to make an entry at least once. Additionally, P8 encountered technical difficulties with their company's firewall. For these reasons, the total number of entries we received was 49 (as opposed to 63).

In 46 out of 49 journal entries, participants indicated learning at least one or two "new things", while only three entries indicated no learning. Interestingly, about half of the time people reported learning one or two new things, the number of tutorials they found interesting was effectively double that number (from three to five). On several occasions, participants reported seeing 10 interesting tutorials, while indicating they learned only one or two new things. These results suggest the difficulty in finding useful content, even against the backdrop of interesting content.

In terms of what participants reported learning, they described discovering underused tools, learning ways to combine several tools for neat effects, learning unknown techniques, and brushing up on their existing skillset. The following excerpts illustrate these findings:

> *The smudge or sponge tool are tools I rarely use so I learned how and why are other people using them. (P3)*

> *I was a bit weak at [the] pen tool, and today I was easily able to find a tutorial for [the] pen tool and had a good practice… I am more confident with [the] pen tool. Apart from [the] pen tool, I explored some style tutorials. I was familiar with the techniques, but discovered that there are some pretty good alternatives to get same style for the text with different techniques. (P7)*

> *I learned about deeper use of the spot healing brush. I gained a deeper understanding of mixing usage of the tool with the clone stamp. (P9)*

We note that the above findings describe participants' self-assessments of what they learned each day. Because we provided a diverse set of tutorial content, and our participants varied in their expertise and knowledge, we did not attempt to quantify their learning beyond self-reports.

Findings: Interview Data

In the interviews, we asked participants to describe their impressions of the system and their experiences using it. We also referenced the logs and journal entries, and asked participants to elaborate on certain behaviors and learning outcomes.

We structure our findings by considering how Switter supports learning via its ability to browse and filter content; how it can be used as a reference tool; and opportunities for improving these types of systems.

Browsing and Filtering Content

Overall, participants were positive about the system, indicating that it was very helpful in finding useful content:

> *It's much easier than Google. Like I said, if you just [added] a search bar, it's competition to Google now for tutorials. (P7)*

Participants indicated that Switter's features allowed for a number of new browsing and searching behaviors that might be hard to achieve with currently available tools. In what follows, we highlight participants' feedback on these capabilities.

Projecting Commands on the Interface Replica. Participants appreciated Switter's ability to project the commands and tools referenced in a tweet onto the Photoshop interface replica. This domain-specific rendering of the tweets helped them browse and filter the content referenced by the tweets non-linearly. The following quotes demonstrate this appreciation:

> *I haven't seen such [a] thing at all. I haven't imagined that there can be a system where you could use commands to filter tweets and learn from that. That part is really awesome. (P7)*

> *I really, really liked that, actually -- that it has the overlay [highlighted commands]. I was actually showing a couple of friends of mine… and they thought it was so cool, the layout. They liked overlay. (P1)*

Participants adopted a range of new browsing techniques that leveraged command-specific filtering and the projection indicators in the replica, which we describe next.

Engaging in Popularity-Driven Exploration. Participants reported that the interface projections guided browsing behavior by attracting their attention to commonly used tools and commands. For example, P3 indicated that the projections allowed her to identify commands frequently referenced in the tweets, which made her curious about what people use them for:

> *I just think it's really cool that you can click on everything and you see people using [commands] and I am, like "I wonder why, or how", you know? ...And I feel like this is more… for the curious mind, that is "ah, what is this? I want to learn more about this thing". And then you can... from there you click and it will show you this many possibilities. I do like that a lot. (P3)*

As hinted above, P3 would filter tweets by clicking on the specific tools that she found most interesting. In turn, command-level filtering satisfied her curiosity by exposing her to the variety of use cases of the selected tool.

Exploring to Address Weaknesses. While P3 used the popularity of highlighted elements to guide her exploration, a few other participants started by first identifying weaknesses in their skill set they wished to address:

> *So, I can just focus on my weaknesses here, like whenever I jump in a system, I just go to the pen tool and start practicing with it. I do not have to go through other stuff that I do not want... I can just focus on my skills. (P7)*

> *I am really bad at masking, so I always kept certain edges when I did masking. And I saw the tool listed as masking and I particularly got interested and just went there and checked the video when it's listed there. (P5)*

Comparing Alternatives. Our interviews revealed that Switter's command filtering capabilities could be used to draw comparisons between several commands, a use case we had not anticipated. For example, P9 described the ability of "jumping" between two commands as one of his favorite aspects of the system:

> *For instance, one of the tools I was learning about yesterday was the patch tool. That was the tool I haven't used before. And the thing that I kind of found when I was using it was it would be a good alternative to the clone stamp tool. And the clone stamp was the thing I used in the past most frequently for that type of work… One of the things that I found the most useful about the program was the ability to quickly jump between tutorials that use the patch tool and back to tools with the clone stamp tool, to see if there's overlap between the two, to kind of see which tool would be most effective in which situation. And being able to do that quickly was what allowed me to make that comparison. Because obviously, if I did not have something that allowed me to jump quickly... it's kind of difficult to see where that overlap is. Just because you kind of lose track of where you are at. (P9)*

What is notable about this use case is that the participant learned not by consulting a single resource, but by explicitly juxtaposing the content of *multiple* learning resources to compare and contrast alternative methods. Switter aided this process by helping them first locate this similar content, and then swiftly move between the resources.

Discovering Synergies. In addition to comparing workflows, we found that participants would also use Switter's filtering capabilities to learn which commands could be used in conjunction with another:

> *I was looking into ones that have burn and dodge, but then it was interesting to see what they use in conjunction with [burn and dodge] ... So that's kind of when I would look into, like, the menu items… It's cool to see combinations. Like, it's not always about the one. It's about how they fit together. (P1)*

At present, Switter lets users filter tweets based on single selections. However, several participants expressed interest in filtering tweets by multiple commands at the same time. We believe that adding such functionality might help users compare command capabilities and search for tutorials that illustrate specific tool combinations.

Filtering Tutorials using the Command Summaries. Participants also made use of the summary of commands Switter provides below each tweet. One participant preferred these summaries over interacting with the interface replica directly, which she sometimes found overwhelming:

> *It's a little bit overwhelming, you know? ...If I click on layers, there's a huge amount here… Whereas if I am looking at this one video and I click on the layer there, it takes me to that specific tool it's going to be using, in that context. (P6)*

As hinted at by this quote, clicking on commands in the summary list carries with it the advantage of teaching one where the command can be found in the interface.

Seeing the list of commands also enabled experienced users to quickly assess what is being covered in a tutorial. For example, several participants described how this information allowed them to essentially recreate the tutorial in their heads without having to look inside:

> *I see that they use "Desaturate," I see that they use "Gaussian Blur," and it makes sense. It's like, I do not need to go into the video and spend 15-30 minutes there, because [by] just looking, and "Oh, desaturation is what takes away the gloss from the photo." Like I know how to use the texture, or I know how to blend. I know WHY would they use "Gaussian Blur," I know why they would use desaturation. (P4)*

Uncovering Unexpected Usages. Conversely, when participants could *not* imagine how the list of commands could achieve a given result, they pursued the tutorial to learn a new workflow. In these cases, participants compared their existing workflow to the one in the tutorial to find out which one is more efficient:

> *So, it's pretty much just looking for a way to do things quicker than you are already doing. So, sometimes seeing the commands it's like "ah, that makes way more sense to be doing it this way instead of the way I've been doing that." (P1)*

Similarly, unexpected uses of a *specific* command or tool incited participants' curiosity. For example, P4 noted that looking for such "out-of-the-box" knowledge was something he was most interested in:

> *If someone, for example, comes and tells me that "I used my hammer to eat my noodles," I'd be very curious, like "how did you do that?" Because a hammer is for banging a nail in its head... It's something that's out-of-the-box knowledge that you get from people. (P4)*

Switter as a Reference Tool

While we originally requested that participants use Switter at least once a day, a number of participants integrated Switter into their existing workflow. For example, P6 indicated that he used Switter as a convenient place to look up how others use the tool he had just struggled with:

> *I was trying to use the clone stamp tool. So, I wasn't quite getting it the way I wanted it to, so I ended up in [Switter] and clicked on a clone stamp tool and simply filtered down to all of those videos using clone stamp tool, and that's a much easier way for me to find tutorials in context with certain things I am struggling with, you know? (P6)*

Findings: Opportunities for Improvement

Participant comments also provide insight into ways that Switter could be improved to better support ad libitum exploration.

Curating Incoming Tutorials. Many participants appreciated being constantly exposed to fresh learning materials. However, a few reported that the quality of the delivered tutorials often did not meet their expectations:

> *what deterred me the last few times, was that some of the content did not appeal to me at all [...] another one is like "how to make water drops" and the picture of water drops [...] Look at those water drops. No offense, but those are really bad water drops. (P1)*

This quote suggests that low-quality content might discourage some users from exploring new learning materials. In future design iterations, one could explore mechanisms to curate twitter-retrieved tutorials, so that only higher-quality tutorials are delivered to the users.

Improving Tutorial Summaries. Participants appreciated the command summaries below each tweet, but many wanted more detailed information about each tutorial. For example, some requested that command summaries reflect the order in which operations are performed in the tutorial. Some participants also wanted the tutorials labelled according to the higher-level skills covered, similar to the approach explored by Kim et al. [18]. Moving forward, the challenge will be both obtaining accurate labels and finding ways to provide this additional information without visual overload.

Re-finding Tutorials. Some participants reported finding a useful tutorial early in the study, but had difficulties re-finding it later. Participants also commented that they did not always have enough time to watch and follow a tutorial that caught their attention. These two needs suggest that it may be worthwhile to include search-based capabilities, or bookmarking functionality, to enable re-finding interesting content at a later date.

DISCUSSION AND FUTURE DIRECTIONS

Social media like Twitter provide a platform for users to easily share and discuss instructional materials for software. Our field study provides encouraging evidence that Switter's approach of projecting these tweets onto a replica of the application's interface helps support ad libitum exploration, by helping users browse and locate learning materials of interest. Our results also suggest that Switter's organization of these resources helps preserve the curiosity-driven component central to ad libitum exploration: users in our field study described a number of instances where they used the tool to uncover unanticipated and sometimes unorthodox bits of new knowledge.

In light of these promising initial results, we discuss a number of directions for future work.

Leveraging Social Media Affordances

In our current implementation, we primarily make use of Twitter's timeline capabilities. However, there are a number of other capabilities built into this platform that could be leveraged in a system like Switter. For example, one can "favorite" or "retweet" content with Twitter. These actions serve as signals that could be used to help users filter and browse tweets. For example, Switter could augment the interface replica by showing which commands link to tutorials that are being favorited or retweeted.

Switter could also make use of hash tags added to a tweet. For example, Switter could add a new menu to the application replica's menu bar, where the menu items are hash tags that open up to submenus containing all tweets with that hash tag. This capability would provide a means for the community to explicitly organize instructional materials within Switter's interface at the time of authoring a tweet.

Scalability of the Approach

During the study, Switter published around 30 new tutorials per day. This information flow was sufficient for the purpose of our study, but did not allow us to test how well our design scales to a larger volume of tweets. For example, our preliminary analysis suggests that there might be closer to 30 Photoshop tutorial tweets per hour. Our current design attempts to address this issue by providing a historical overview and time-based filtering of tweets. Nonetheless, how well this approach scales needs further research.

Generalizability of the Approach

In this work, we observed ad libitum software exploration for a specific user group – expert designers and artists. Although the selected user group is rather broad, in the future we plan to investigate the prevalence of ad libitum exploration among other user groups.

We chose Photoshop as Switter's target application because of its popularity among a wide range of design professionals: graphic and Web designers, digital artists, photographers, etc. Implementing similar design ideas for other command-driven feature-rich applications should be straightforward, but the current approach may not generalize to other types of feature-heavy applications, particularly ones in which users rely on macros to execute complex tasks.

Extending the approach to other social media platforms, such as Facebook or Reddit, is another open research problem. These platforms differ from Twitter in a number of ways, including their social dynamics and the way that they organize information. Consequently, future work should examine their potential for supporting ad libitum exploration.

Integration with Software

We implemented Switter as an independent web tool, rather than integrating it into Photoshop itself primarily for ease of prototyping. However, we also believe that Switter's independent format has its own benefits, for example, enabling users to browse the resources on devices that do not have Photoshop installed. Investigating the tradeoffs between in-application instrumentation and independent application is an area for future work.

Measuring Effects on Software Learning

While our participants reported learning one or two things each day, our study did not include a control condition to provide a baseline for comparison, nor did we explicitly test what they learned. Thus, one fruitful path for future work is to compare Switter to existing, general-purpose social media clients, with respect to their ability to support ad libitum exploration. From our study results, it is clear that Switter provides some clear benefits compared to existing clients (e.g., the ability to quickly toggle between sets of learning resources by clicking on commands), but it is worthwhile to quantify the impact of these features.

Switter also takes a qualitatively different approach to supporting software learning compared to other approaches, such as command recommender systems. We consider Switter complementary to these other approaches, but providing users with simultaneous access to different types of ad libitum exploration tools will likely provide useful insights to guide future research.

Tutorial Annotation

Our field study included a Wizard-of-Oz component by virtue of us hand-annotating each tweet with the commands used in the referenced tutorial. Prior work has examined ways to automatically extract commands from both text [9,17] and videos [27,31]. Incorporating automated extraction techniques into a system like Switter will introduce noise into the system, which will alter the user experience. However, there are alternative approaches to determining the commands mentioned in a tutorial. For example, if workflows are automatically captured within the application, as Chronicle does [15], precise command metadata is readily available. Command annotations could also be crowdsourced, for example, within an application like Switter.

Beyond command annotations, some participants desired more descriptive labels for the instructional content. For example, they wanted to know whether it described a particular technique, or the intended general audience for the material (e.g., an interface designer vs. graphic artist). These labels are further candidates for crowdsourcing approaches, given that automatically categorizing tutorials across dimensions (such as audience) is currently a challenging problem. Alternatively, one could provide explicit support to organize tweets by hash tags, as described above.

CONCLUSION

In this paper, we explored an expert software learning activity that we refer to as ad libitum exploration. We used an exploratory interview study to understand current practices with respect to this type of ongoing, curiosity-driven software learning. Based on our findings, we designed Switter, an alternative Twitter client that is embedded in a replica of a target application's interface. Our weeklong field study revealed a number of compelling use cases for Switter. Our field study also suggests opportunities for future research in ad libitum exploration.

ACKNOWLEDGMENTS

We would like to thank the National Sciences and Engineering Research Council (NSERC) and the University of Manitoba's Graduate Enhancement of Tri-Council Stipends (GETS) program for funding this research.

REFERENCES

1. Andrea Bunt, Patrick Dubois, Ben Lafreniere, Michael A. Terry, and David T. Cormack. 2014. TaggedComments : Promoting and Integrating User

Comments in Online Application Tutorials. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'14

2. John M. Carroll and Caroline Carrithers. 1984. Training Wheels in a User Interface. Communications of the ACM
3. Parmit K. Chilana, Andrew J. Ko, and Jacob O. Wobbrock. 2012. LemonAid: Selection-Based Crowdsourced Contextual Help for Web Applications. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'12
4. Juliet Corbin and Strauss Anselm. 2008. Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. SAGE Publications
5. Tao Dong, Mira Dontcheva, Diana Joseph, Karrie Karahalios, Mark Newman, and Mark Ackerman. 2012. Discovery-Based Games for Learning Software. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'12
6. Brian Dorn and Mark Guzdial. 2010. Learning on the Job: Characterizing the Programming Knowledge and Learning Strategies of Web Designers. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'10
7. Michael Ekstrand, Wei Li, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2011. Searching for Software Learning Resources Using Application Context. Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST'11
8. Jennifer Fernquist, Tovi Grossman, and George Fitzmaurice. 2011. Sketch-Sketch Revolution: an Engaging Tutorial System for Guided Sketching and Application Learning. Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST'11
9. Adam Fourney, Ben Lafreniere, Richard Mann, and Michael Terry. 2012. "Then Click OK!": Extracting References to Interface Elements in Online Documentation. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'12
10. Adam Fourney and Ben Lafreniere. 2014. InterTwine: Creating Interapplication Information Scent to Support Coordinated Use of Software. Proceedings of the 27th annual ACM symposium on User interface software and technology - UIST'14
11. Adam Fourney, Richard Mann, and Michael Terry. 2011. Query-feature Graphs: Bridging User Vocabulary and System Functionality. Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST'11
12. Saul Greenberg. 1993. The Computer User as Toolsmith: The Use, Reuse and Organization of Computer-based Tools. Cambridge University Press.
13. Tovi Grossman, George Fitzmaurice, and Ramtin Attar. 2009. A Survey of Software Learnability: Metrics, Methodologies and Guidelines. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'09
14. Tovi Grossman and George Fitzmaurice. 2010. ToolClips: an Investigation of Contextual Video Assistance for Functionality Understanding. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'10
15. Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2010. Chronicle: Capture, Exploration, and Playback of Document Workflow Histories. Proceedings of the 23nd annual ACM symposium on User interface software and technology - UIST'10
16. Caitlin Kelleher and Randy Pausch. 2005. Stencils-Based Tutorials : Design and Evaluation. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'05
17. Adnan Alam Khan, Volodymyr Dziubak, and Andrea Bunt. 2015. Exploring Personalized Command Recommendations based on Information Found in Web Documentation. Proceedings of the 20th International Conference on Intelligent User Interfaces - IUI'15
18. 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. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'14
19. Nicholas Kong, Tovi Grossman, Björn Hartmann, Maneesh Agrawala, and George Fitzmaurice. 2012. Delta: a Tool for Representing and Comparing Workflows. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'12
20. Ben Lafreniere, Andrea Bunt, Matthew Lount, and Michael Terry. 2013. Understanding the Roles and Uses of Web Tutorials. Proceedings of the 8th international AAAI conference on weblogs and social media - ICWSM'13
21. Ben Lafreniere, Tovi Grossman, and George Fitzmaurice. 2013. Community Enhanced Tutorials : Improving Tutorials with Multiple Demonstrations. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'13
22. Ben Lafreniere, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2014. Investigating the Feasibility of Extracting Tool Demonstrations from In-Situ Video Content. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'14

23. Gierad Laput, Eytan Adar, Ann Arbor, Mira Dontcheva, and Wilmot Li. 2012. Tutorial-Based Interfaces for Cloud-enabled Applications. Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST'12

24. Wei Li, Tovi Grossman, and George Fitzmaurice. 2012. GamiCAD: A Gamified Tutorial System For First Time AutoCAD Users. Proceedings of the 25th annual ACM symposium on User interface software and technology - UIST'12

25. Wei Li, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2011. TwitApp: In-Product Micro-Blogging for Design Sharing. Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST'11

26. Frank Linton, Andy Charron, and Debbie Joy. 2000. OWL : A Recommender System for Organization-Wide Learning. Educational Technology & Society, 62–76.

27. Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2011. Ambient Help. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'11

28. Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2011. IP-QAT: In-Product Questions, Answers, & Tips. Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST'11

29. Justin Matejka, Tovi Grossman, and George Fitzmaurice. 2013. Patina: Dynamic Heatmaps for Visualizing Application Usage. Proceedings of the SIGCHI Conference on Human Factors in Computing Systems - CHI'13

30. Justin Matejka and Wei Li. 2009. CommunityCommands: Command Recommendations for Software Applications. Proceedings of the 22nd annual ACM symposium on User interface software and technology - UIST'09

31. Suporn Pongnumkul, Mira Dontcheva, Wilmot Li, et al. 2011. Pause-and-Play: Automatically Linking Screencast Video Tutorials with Applications. Proceedings of the 24th annual ACM symposium on User interface software and technology - UIST'11

32. John Rieman. 1996. A Field Study of Exploratory Learning Strategies. ACM Transactions on Computer-Human Interaction - TOCHI'96

33. Barry J. Zimmerman. 1989. Models of Self-Regulated Learning and Academic Achievement. In Self-Regulated Learning and Academic Achievement (1st ed.), Barry J. Zimmerman and Dale H. Schunk (eds.). Springer New York, New York, NY, 1 – 25.