Uncovering Trajectories of Informal Learning in Large Online Communities Of Creators

Seungwon Yang, Carlotta Domeniconi, Matt Revelle, Mack Sweeney, Ben U. Gelman, Chris Beckley, and Aditya Johri
George Mason University
4400 University Drive, Fairfax, VA 22030, U.S.A.
syang20@gmu.edu, {carlotta, revelle}@cs.gmu.edu,
{msweene2, bgelman, cbeckley, ajohri3}@gmu.edu

ABSTRACT
We analyzed informal learning in Scratch Online – an online community with over 4.3 million users and 6.7 million instances of user-generated content. Users develop *projects,* which are graphical interfaces consisting of interacting programming *blocks*. We investigated two fundamental questions of *how we can model informal learning,* and *which patterns of informal learning emerge*. We proceeded in two phases. First, we modeled learning as a trajectory of cumulative programming block usage by long-term users who created at least 50 projects. Second, we applied K-means++ clustering to uncover patterns of learning and corresponding subpopulations. We found four groups of users manifesting four different patterns of learning, ranging from the smallest to the largest improvement. At one end of the spectrum, users learned more and in a faster manner. At the opposite end, users did not show much learning progress, even after creating dozens of projects. The modeling and clustering of trajectory patterns that enabled us to quantitatively analyze informal learning may be applicable to other similar communities. The results can also support administrators of online communities in implementing customized interventions for specific subpopulations.

Author Keywords
Learning analytics; informal learning; modeling; clustering; programming; Scratch; online community.

ACM Classification Keywords
H.3.3. Information Search and Retrieval: Clustering; K.3.1. Computers and Education: Computer Uses in Education.

L@S 2015, March 14 - 18, 2015, Vancouver, BC, USA Copyright 2015 ACM 978-1-4503-3411-2/15/03…$15.00
http://dx.doi.org/10.1145/2724660.2724674

INTRODUCTION
Thousands of massive open online courses (MOOCs) [17], Q&A forums [28], tutorial sites [31], and online communities of creators (OCOCs) [29] attract people interested in learning. MOOCs provide a formal learning environment with specific learning goals, rubrics to follow, and assessments on learner performance toward goals. Other online communities such as OCOCs offer platforms with tools and services to create content both individually and in collaboration with others. Although these settings do not provide formally structured instruction, they do provide participants the opportunity to learn important content and skills informally [5, 9, 14, 19].

Within formal contexts, such as MOOCs, assessment of learning outcomes is relatively straightforward as student progress can be measured against educational aims of the course. Consequently, MOOCs incorporate automatic and formal assessments to measure learner progress in order to provide feedback [4, 30]. In contrast, online communities that support informal learning often lack appropriate means to assess progress of their members.

In this paper, we aimed to analyze informal learning occurring in Scratch online [23, 24] by addressing the questions of *how we can model informal learning,* and *which patterns of informal learning may emerge*. Specifically, we focused on measuring and analyzing the three aspects of learning:

- Amount of learning
- Speed of learning
- Potential prior knowledge

Scratch users are mainly kids and teenagers who create and share their 'projects,' which are games, animations, arts, and other media content. Projects are constructed by importing multimedia files and editing/controlling them using a visual programming language called Scratch. The Scratch language consists of various programming 'blocks,' which are the software version of Lego™ bricks. Once created, a project can be shared with other users. Remixing – creating a project based on another's project – is also highly encouraged in the community. Although there may be exceptions (e.g., sophisticated arts project

may not require a large vocabulary), we believe that a wider spectrum of vocabulary block use identified from a user's original projects (non-remix) may be an indicator of learning. We developed a two-phase approach to answer our questions:

- Phase I: Model informal learning of a user as a trajectory of cumulative vocabulary block use, as s/he creates more projects over time. Visualize trajectories for all projects, as well as only the original projects.
- Phase II: Find and compare patterns of trajectories using a clustering approach to uncover subpopulations that share similar learning patterns [11, 18]. Examine samples from each cluster for detail analysis.

This study contributes to the learning community by exploring an approach to quantitatively measure informal learning from a large online community – which is difficult due to the nature of online communities (e.g., non-goal directed, less structured). It may also trigger administrators of online communities to implement targeted services for specific subpopulations.

After introducing related studies, we plunge into modeling and clustering the trajectories of programming block use. Then, we present the clustering results and examples in each cluster, followed by the discussions regarding the meaning of our results and limitations. We provide implications of this study, and conclude with an overall summary and future work.

RELATED STUDIES

Scratch Online

A core goal of Scratch is to make the creation and sharing of interactive content easy for users [20]. Within the Scratch programming environment as shown in Figure 1, users program with blocks, which map to programming constructs and allow users to manipulate data including media content (e.g., sounds, videos, images). Users can then share these interactive creations on the Scratch website [23]. The website is an online community where users may run and build on others' ideas and projects. Members can import the source code of any project into their own workspace and remix it.

In addition to the Scratch project's goal of creating and sharing programs, a fundamental element behind its creation has been to provide an alternate model for how the Web can be used for learning. The Scratch team envisioned learning outcomes related to programming and math skills as well as skills in design, creativity, communication, and so on. Learning is inherently a collaborative process that occurs across communities of practice. The Scratch community was created with the goal to foster learning among users through interaction and sharing of ideas and projects [15, 16].

Resnick et al. wrote about the Scratch in 'Programming for all' and their goal has always been to develop an approach to programming that would appeal to people who had not considered themselves programmers [20]. In May 2007, a website was launched to allow for more sharing of projects on the Web. Maloney et al. studied the use of Scratch at a Computer Clubhouse where urban youth between the age of 8-18 used Scratch [13]. The findings from this research indicated that users learned key programming concepts even in the absence of instructional interventions or experienced mentors. The ease of use for the Scratch platform and its interactivity made it attractive for users.

Informal Learning and Assessment

One of the alternate mechanisms that is accepted as central to revising the current model of education is emphasizing and improving learning that occurs in informal settings [2, 6]. Even when in formal learning environments, people gain from participation in informal learning activities [2, 8]. Consequently, informal settings and activities represent an important opportunity for engineering students to develop technical capabilities, social skills, strengthen their engineering identities, and cultivate habits necessary for life-long learning [3]. The lack of research on informal learning within engineering education was highlighted in the Educating the Engineers of 2020 report, which raised

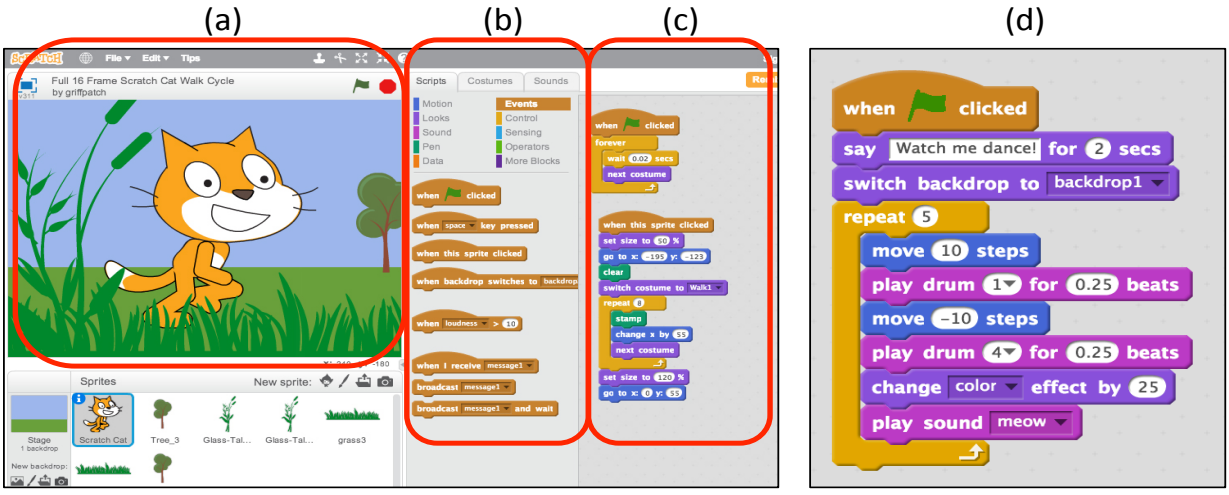

Figure 1. Scratch interface: (a) a project window, (b) blocks, (c) scripts, and (d) a script constructed with blocks.

the question, 'How can formal education be better integrated with informal and lifelong learning by engineering graduates?' [6].

The topic of assessment for online learning (e.g., MOOCs) has already found significant traction within the Learning at Scale community. For instance, Kulkarni et al. investigated ways to scale up peer assessment [12]. Peer assessment is a useful tool in the learning process, but it suffers from quality issues. The authors use machine learning techniques to scale this process whereby the quality is maintained and the burden of grading is not high. Brooks et al. propose a cluster-based interface that allows teachers to read, grade, and provide feedback on large groups of answers at once [4]. Wilkowski et al. looked at grading in MOOCs where the solutions are complex and qualitative items require subjective judgment [30]. They incorporated meta-evaluation, which evaluates the self-evaluation of students, and found that the assessments were accurate. These studies are interesting in that they investigated assessment issues occurring in large online courses. However, the primary setting for these studies was not on informal learning, but on MOOCs that provide formal learning.

Studies do exist for assessing informal learning. For example, Naturalistic Assessments [14] focuses on STEM learning taking place outside institutions. Naturalistic Assessment is focused on people's awareness of who knows how to do what. Carliner [5] proposes self-assessments, process portfolios, and certifications for measuring informal learning occurring in a professional environment. However, these approaches may not scale to large communities, are obtrusive to online community members, or require longitudinal studies to capture the cumulative nature of learning [19].

TRAJECTORIES OF VOCABULARY PROGRESS

Modeling Vocabulary Progress

The Scratch Dataset

The Lifelong Kindergarten Group at the MIT Media Lab created the Scratch dataset from the Scratch Online Community [24, 25]. It contains five years of data between 2007 and 2012, including 1 million users and their 1.9 million projects, grouped in five categories (Table 1). Demographics of users have not been published due to privacy concerns considering that the majority of Scratch users are kids and teens. Data files in each category accompanied their documentation, which describe variable fields, summary statistics, and omitted observations.

In this study, we selected 3,852 long-term users each with at least 50 original projects. In total, there were 618,721 projects of which 145,916 were original and 472,805 were remixed. Each project consists of sprites (characters in a project), imported media files, and programming blocks. Users can stack up blocks and specify variables (e.g., '10 steps', '0.25 beats') embedded in blocks to control sprites and media (Figure 1 (c) and (d)). In total, 170 different types of blocks are available.

Category	Description	Count*
Core	Data files describing the major objects and relationships captured by the Scratch website	18 (1)
Text & Code	User submitted texts useful for text and natural language processing analysis	8 (3)
Project Analytics	Detail data of blocks, drums, media, midi instruments, save history, and sprites of projects	6
Other Scratch Website	Scratch Media wiki dump, forum posts, and Text-based Games forum posts	3(3)
Code	Code used to generate the datasets from the MySQL database used by Scratch	1

Table 1. The Scratch dataset. *The numbers inside parentheses denote the number of data files in preparation (not available).

Assigning Block Weights

Some blocks are commonly used while others appear only rarely in the projects. The question then arises: should the different block types be equally considered when modeling users' progress? We assumed that users create simple projects at the beginning using a set of common blocks. As users create more projects and become familiar with the Scratch editor, they may explore blocks that they have not used before, gradually expanding their vocabulary set. Based on this assumption, we considered the presence of rarely used blocks in a project as a sign of a more advanced progress in learning about blocks, as compared to the presence of common blocks.

To account for this difference of the blocks, we assigned a weight to each block using *Inverse Document Frequency* (IDF) [27], which was computed from the entire set of original projects of all users in our selected data set:

$$w_{b_j} = \log_{10}\left(\frac{1+P}{1+P_{b_j}}\right) \quad (1)$$

where P is the total number of original projects, and P_{b_j} is the number of original projects containing a vocabulary block b_j $(1 \leq j \leq 170)$. We added '1' to both numerator and denominator in (1) as a smoothing term to avoid negative weights and division by zero. Thus, our IDF weight vector $\mathbf{W}$ has 170 elements, one for each corresponding vocabulary block:

$$\mathbf{W} = [w_{b_1} \quad \cdots \quad w_{b_j} \quad \cdots \quad w_{b_{170}}] \quad (2)$$

The highest weight value is 5.17, and is assigned to the blocks used most rarely, e.g., *changebrightnessshift*; very common blocks, such as *eventhatmorph_startclicked,*

receive a weight value that is close to zero (Figure 2). The total sum of weights is 402.94.

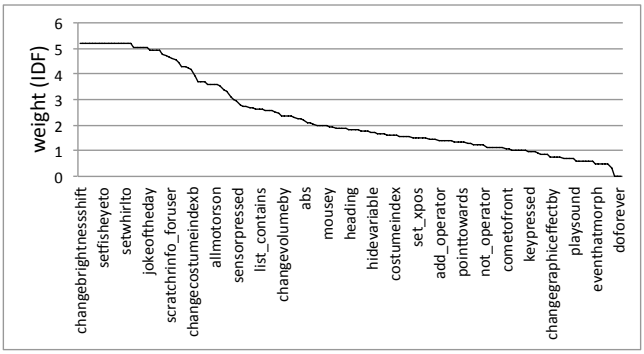

Figure 2. Vocabulary weights computed from IDF.

Modeling Learning Trajectories

We modeled the informal learning of a user as a trajectory of cumulative vocabulary block use. Procedures shown below are based on the analysis of the user's first 50 projects – including original and remix. Since our main focus was to examine the use of blocks as a user creates projects one by one, we used the project sequence, instead of actual time stamps, as the unit of trajectories.

For each user *u*, we proceed as follows (the user ID ranges from 1 to 3852).

1. Align all projects of a user *u* including original and remix in sequence from the earliest to the latest.
2. Construct a 50 x 170 matrix $\mathbf{P}_u$ using the first 50 projects and frequencies of their 170 blocks:

$$\mathbf{P}_u = \begin{bmatrix} f_{1,1} & \cdots & f_{1,j} & \cdots & f_{1,170} \\ \vdots & & \vdots & & \vdots \\ & \ddots & & \ddots & \\ f_{i,1} & \cdots & f_{i,j} & \cdots & f_{i,170} \\ & \ddots & & \ddots & \\ \vdots & & \vdots & & \vdots \\ f_{50,1} & \cdots & f_{50,j} & \cdots & f_{50,170} \end{bmatrix} \quad (3)$$

where $f_{i,j}$ is the frequency of block b_j $(1 \leq j \leq 170)$ in project i $(1 \leq i \leq 50)$.

3. Create a matrix $\mathbf{P}_c$ by cumulatively summing rows in $\mathbf{P}_u$ (e.g., the i^{th} row of $\mathbf{P}_c$ is the element-wise sum of the first i rows in $\mathbf{P}_u$).
4. Create a binary matrix $\mathbf{P}_b$ from $\mathbf{P}_c$ ('1' if frequency of an element > 0, '0' otherwise).
5. Compute a trajectory by applying weights on the elements of $\mathbf{P}_b$ and summing values in each row: $\mathbf{V}_u = (\mathbf{P}_b\mathbf{W}^T)^T$. The final result is a 50-dimensional vector:

$$\mathbf{V}_u = [v_{u,1} \quad v_{u,2} \ \ldots \ v_{u,i} \ \ldots \ v_{u,50}] \quad (4)$$

where $v_{u,i}$ is a cumulative sum of weighted vocabulary blocks for a user *u,* computed using the first *i* projects. It is a weighted binary count considering whether a block was ever used (frequency is ignored). We compute $\mathbf{V}_u$ for all 3,852 users to construct a matrix $\mathbf{T}_{all}$ that has 3,852 x 50 dimensions:

$$\mathbf{T}_{all} = \begin{bmatrix} \mathbf{V}_1 \\ \vdots \\ \mathbf{V}_u \\ \vdots \\ \mathbf{V}_{3,852} \end{bmatrix} \quad (5)$$

6. Create a vector $\mathbf{O}_u$, corresponding to a trajectory that uses only original projects, simply by adding another step after step 2 above: Replace the rows of remix projects in matrix $\mathbf{P}_u$ with vectors of zeros:

$$\mathbf{O}_u = [o_{u,1} \quad o_{u,2} \ \ldots \ o_{u,i} \ \ldots \ o_{u,50}] \quad (6)$$

where $o_{u,i}$ is the analogous to $v_{u,i}$ except that it is based only on the original projects. We compute $\mathbf{O}_u$ for all 3,852 users to construct a matrix $\mathbf{T}_{ori}$ that has 3,852 x 50 dimensions:

$$\mathbf{T}_{ori} = \begin{bmatrix} \mathbf{O}_1 \\ \vdots \\ \mathbf{O}_u \\ \vdots \\ \mathbf{O}_{3,852} \end{bmatrix} \quad (7)$$

The trajectory of vocabulary progress – *Original & remix* or *Original only* – of a user *u* can be plotted using the corresponding rows in $\mathbf{T}_{all}$ and $\mathbf{T}_{ori}$ (Figures 8-11).

Identifying Trajectory Patterns

After inspecting many trajectory graphs, which represent informal learning of blocks over the first 50 projects, we recognized their shapes vary significantly. In order to determine a small number of canonical trajectory patterns, we applied a clustering algorithm on $\mathbf{T}_{ori}$ since we were initially interested in the trajectory patterns from original projects. Using the trajectories in each cluster, we also collected corresponding trajectories from $\mathbf{T}_{all}$.

The trajectory graphs increase monotonically, and the maximum value they reach in the first 50 projects is 402.94 (most trajectories reached a value under 150). Since we were not concerned with outliers, we decided to use the K-means++ algorithm [1] as opposed to the more robust variant, K-medoids [26], which works well in the presence of outliers. Regular K-means is sensitive to initial cluster centroid seeds, which are randomly selected. K-means++ does a bit more intelligent selection by choosing the first centroid seed randomly, the second seed to be the farthest from the first, the third seed to be the farthest from the first and second, and so on. Therefore, the only source of randomness is the first choice of the first seed.

K-means-based algorithms require the number of clusters (*k*) as an input. To find an appropriate *k*, we plotted the total Within-Group Sum of Squared Error (SSE) [21] for increasing *k* values, as shown in Figure 3. A red rectangle box encloses the meaningful portion of the graph. Increasing the value of *k* from 3 to 4 reduces considerably the value of the total Within-Group SSE. However, further

increasing k affects the SSE values only minimally. Thus we chose $k = 4$.

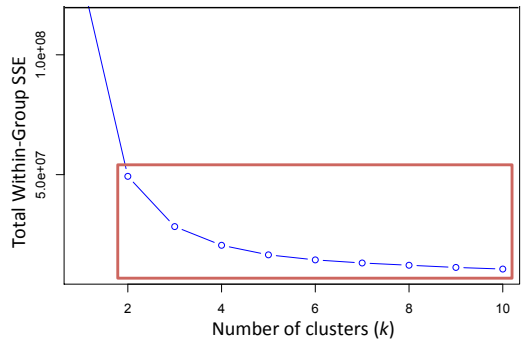

Figure 3. The total Within-Group SSE values by different number of clusters (k). $k = 4$ was selected.

We ran K-means++ 50 times on our matrix $\mathbf{T}_{ori}$, and then selected the model that had the least total Within-Group SSE. Figure 4 illustrates the SSE values achieved at each run of K-means++. Although the first seed centroid of K-means++ is randomly selected at each run, we observed that only a handful of different SSE values were reached. All the multiple runs, which gave the same smallest SSE value at convergence, corresponded to the same partition.

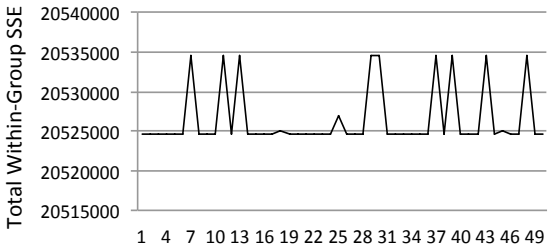

Figure 4. Total Within-Group SSE values for the 50 runs of K-means++.

Table 2 gives the sizes (number of users) of the resulting clusters.

Cluster A	Cluster B	Cluster C	Cluster D
1,304	1,250	941	357

Table 2. Size of clusters.

RESULTS

Figure 5 illustrates average trajectory patterns of users from clusters A-D. For each cluster, we first plotted an *Original only* graph (solid line) using the centroid vector of that cluster since a centroid in K-means++ clustering represents an average of all the members in that cluster. We then plotted the corresponding *Original & remix* graph (dotted line). We recognized interesting differences between graphs in each cluster and across clusters. Since the first 10 projects and the second 10 projects in the graph showed prominent characteristics, we attached regression lines and their slopes for those learning segments.

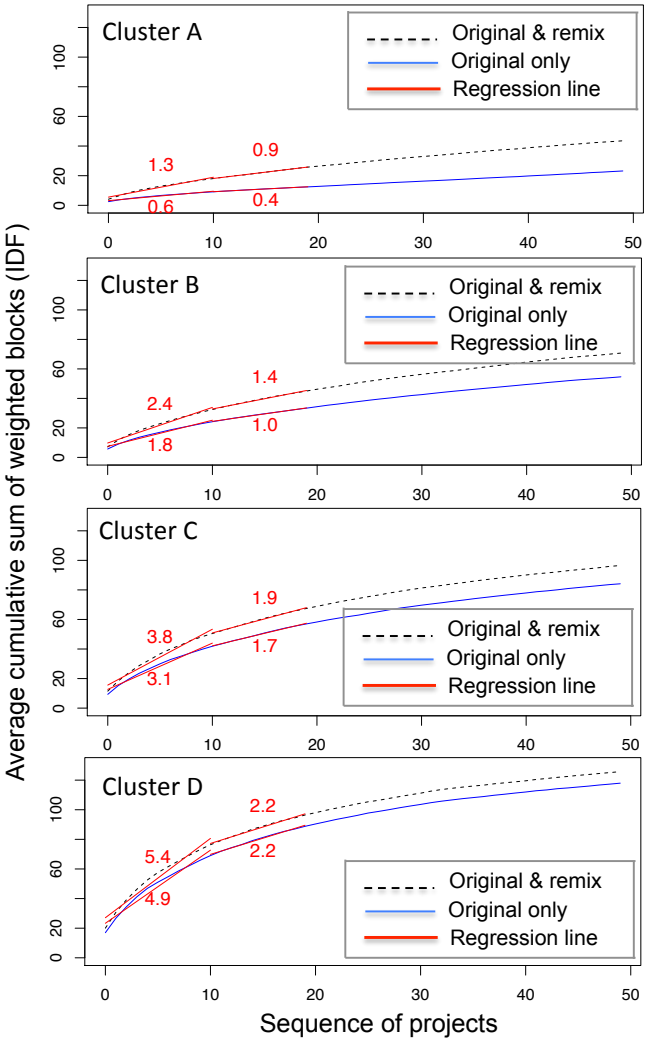

Figure 5. Average trajectory graphs for clusters A-D. Regression lines and slopes are shown for the first 10 and the following 10 projects (p-value << 2.2e-16).

Differences in Amount and Speed of Learning

Based on the value of the graphs at the 49th project, which is the index of the last project, we can measure the average 'amount' of learning of blocks in each cluster (Figure 5). The amount of learning in clusters A-D was proportionally related to the 'speed' of learning occurring in the corresponding clusters, as evidenced by the increasing slopes (from A to D) for the first 10 projects and the following 10. However, the amount and speed of learning were inversely related to the size of clusters in Table 2, which reflects the general trend in groups of learners.

The gap between the upper (*Original & remix*) and lower (*Original only*) graphs at a specific project index represents the corresponding cumulative vocabulary difference up to that point. This gap is smaller in D than in A. Also, the differences in slope between the two curves, measured within the first 10 projects and the following 10 projects, are smaller in D than in A. One reason for this might be that users in cluster D used diverse vocabulary even in their

early original projects. This interpretation is supported by the fact that the cumulative sum in the *Original only* graph (cluster D graph in Figure 5) reached 70 at the project index of 10, when the second-most-progressed group of users (cluster C graph in Figure 5) reached the same value at the project sequence of 35. Therefore, it might have been hard to find remix projects with previously unused vocabulary. Another possible explanation for the discrepancy might be that users in cluster D typically remixed projects similar to their original projects, leading to only a small difference between the graphs.

Differences in Initial Vocabulary

Another observation from Figure 5 is that the 'starting value' of the trajectory (*Original only*) at project sequence 0 is higher in cluster D (mean=17.0) than in A (mean=2.5).

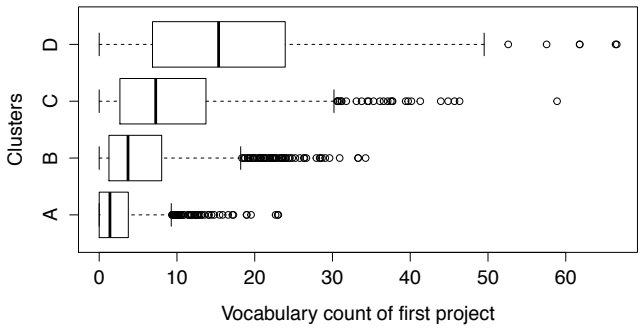

Figure 6. Boxplots summarizing the starting value for the first original projects in each cluster (count is 0 if a trajectory starts with a remix project).

The boxplots in Figure 6 illustrate this further. The median starting values of the trajectories from A to D are 1.4, 3.7, 7.3, and 15.4, showing an exponential increase. Even the outliers of A have lower values than those of D. This may be due to the differences in users' background knowledge. For example, users with prior programming experience already know how to use various vocabulary blocks. Thus, these users can apply a variety of blocks when creating complex Scratch projects. Other factors such as age and gender of users may contribute to this phenomenon as well. We further discuss on this in Discussion and Limitations section.

Characteristics of Block Use by Different Clusters

We computed the percentage of blocks used in the original projects for each cluster to understand the block usage trend among clusters (Figure 7). A total of 145,916 projects (A: 47,097; B: 48,270; C: 36,421; and D: 14,128) were analyzed and the percentages of block use were visualized along with the block names, which were organized in decreasing order of their IDF weights from left (higher weight, rare blocks) to right (lower weight, common blocks). 25 blocks with the highest weights were removed from the computation because they were not used at all.

As we may expect from the definition of IDF weighting, blocks with lower weights were more popular in general than those with higher weights. An interesting tendency was that the D graph (black solid line) was higher than the A graph (red dotted line) until they reached the 'doif' block, where the two lines crossed. After the 'doif' block, the A graph became higher than the D graph, indicating that the users of cluster D used rare blocks more often compared to the users of A. The B and C graphs were located in between the A and D graphs.

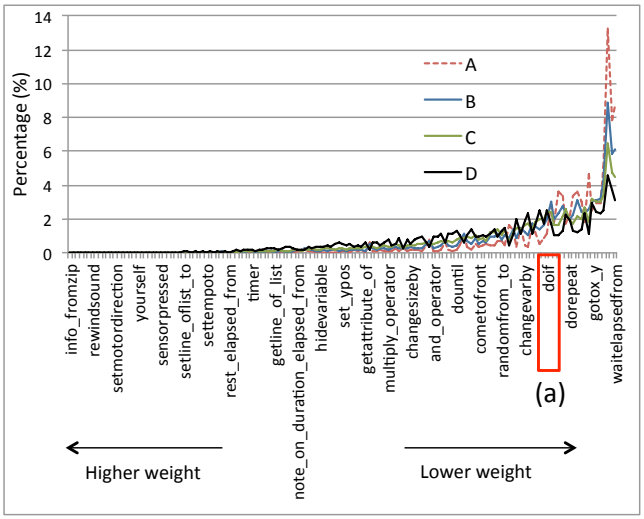

Figure 7. Block use (%) for different clusters. The A graph and the D graph cross at (a) 'doif' block.

Text Analysis of Project Descriptions and Comments

To get a glimpse of how the users in different clusters describe their projects and what kind of comments are posted for those projects, we extracted the 100 most frequent words from the project descriptions and comments. We then compared these 100 words across clusters to uncover unique words (Table 3).

Cluster	Unique Words	Count
A	poor, story, kirby, movie, sad, cookie, anime, car, baby, wolf, shadow, drawing	12
B	king, head, views, meow, dragon, nice	6
C	nyan, spam, watch, astro, darkraiworld, hill, battle, pizza, stick	9
D	color, bros, radas, player, tagger, darkb, mouse, liam, long, version, online, turbo, adventure, survived, scripts, ninja, projects, great, mhm, magic, level, geometry, button, bob, view, erk	26

Table 3. Unique words extracted from the project descriptions and comments in each cluster.

Table 3 shows that the cluster D has the largest number of unique words (26) by a considerable margin. In addition,

the unique words from cluster D may have come from games (e.g., player, mouse, adventure, survived, ninja, level, button) and programming (e.g., version, scripts, geometry). Users in cluster B and C create a large number of projects that are related to cats (See the unique words 'meow' in cluster B and 'nyan' in cluster C in Table 3). In fact, 'cat' was a popular topic in Scratch considering that searching with 'cat' returned over 7 million results. The popularity of 'dragon' in cluster B was probably due to over 2.26 million projects about a famous animation "Dragon Ball." 'battle' in cluster C was usually associated with games (e.g., RPG battle, Tank battle, battle training game). The word 'poor' in cluster A was often used in titles along with the name of animation characters (e.g., 'poor foxy', 'poor Pinkie Pie', 'poor snow man'). 'poor' was also used to comment on the quality of projects (e.g., 'poor translation'). The frequent use of word 'poor' may indicate that the users in cluster A are novices who frequently acknowledge the low quality of their projects in relation to those in other clusters.

	Emotion	Project Type	Other
Across A-D	crazy, awesome, love, best, funny, yay, happy, good, evil, cool, rock(s), bad, stupid, fun, weird	art, cat, pokemon, click, contest, music, sonic, super, mario, press, game(s), remix, space, sprites	wuz, random, pie, cant, tag(s), blue, song, waffles, add, stuff, scratch, green, epic, red, real, life, star, big, time, day, eat, man

Table 4. Common words extracted from clusters A-D (a total of 56 words).

Table 4 shows common words across all clusters. They might be divided into three groups: emotion, project type, and other. Combined together, these word groups illustrate a simplified snapshot of the Scratch community, where people create arts, games, animations, and express their emotions towards those content.

Example Trajectory Graphs

Cluster analysis provides us a macroscopic examination of the patterns in the trajectory graphs. For microscopic details of graph variations, we present two examples which are randomly selected from each cluster.

Figure 8 depicts typical patterns of trajectories from the least progress group in cluster A. In Figure 8 (a), the *Original only* graph stayed under 25 cumulative sum of vocabulary even though the user remixed many projects over time. The user in Figure 8 (b) remixed only one project later in the sequence. His/her cumulative sum of vocabulary increased gradually during the first 10 projects, then stabilized for another 30 or so. There was a small vocabulary jump around the 43rd project, but overall progress is slow.

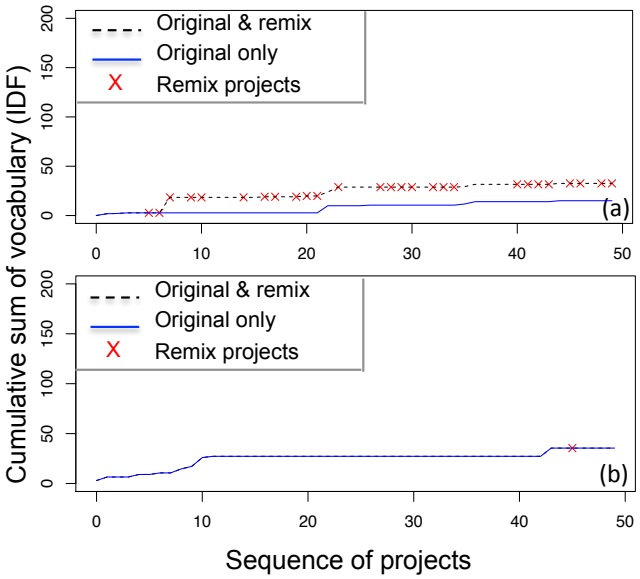

Figure 8. Examples from cluster A (the least progress group). Graphs in (a) and (b) stay under the cumulative sum value of 50.

Figures 9 and 10 illustrate example trajectories from cluster B and C respectively. In the first 10 projects, the two figures show noticeable differences: corresponding portions in Figure 9 (a) and (b) are rather flat; but those in Figures 10 (a) and (b) are increasing. The *Original & remix* graph (dotted line) in Figure 9 (a) has a slight jump at the 21st project due to new vocabulary in a remix project. However, the last two remix projects at 36th and 37th do not contribute to the vocabulary progress very much. Figure 9 (b) shows a sharp jump at the 8th project caused by the vocabulary increase in the original project, but then the progress slows down even after multiple remixes.

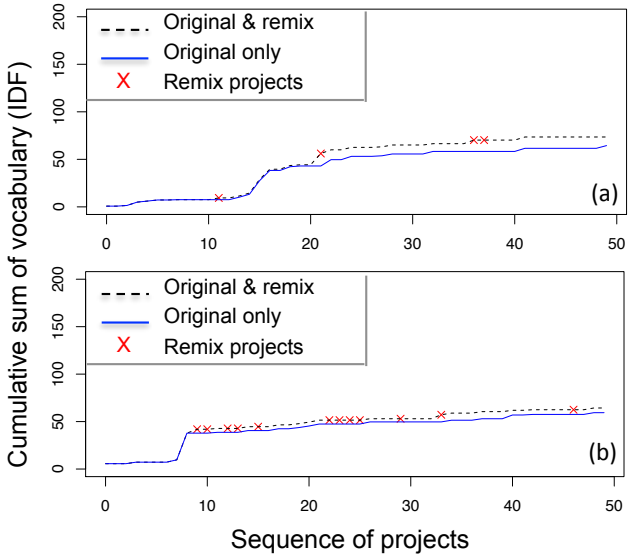

Figure 9. Examples from cluster B. (a) and (b) reach slightly over 50 cumulative sum of vocabulary in the end.

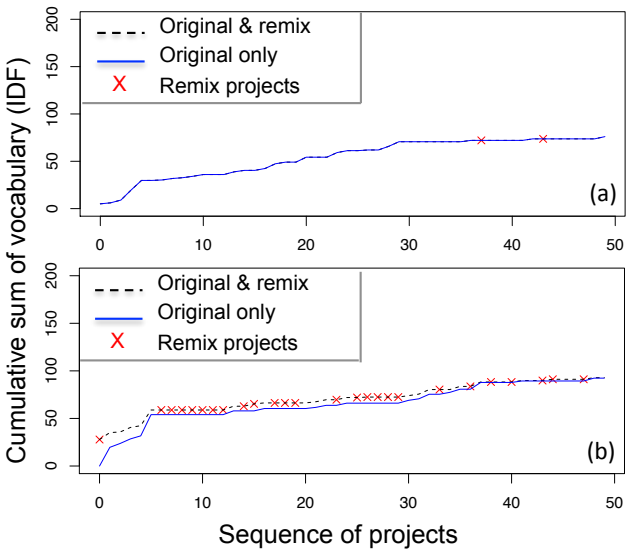

Figure 10. Examples from cluster C. (a) and (b) reach near 70-80 cumulative sum of vocabulary in the end.

The final cumulative values at project index 49 in Figures 9 (a) and (b) reach between 60 and 70. Figures 10 (a) and (b) have similar patterns, but Figure 10 (b) has many more remix projects which do not contribute much to the vocabulary increase. The final cumulative values in Figure 10 (a) and (b) reach between 80 and 90.

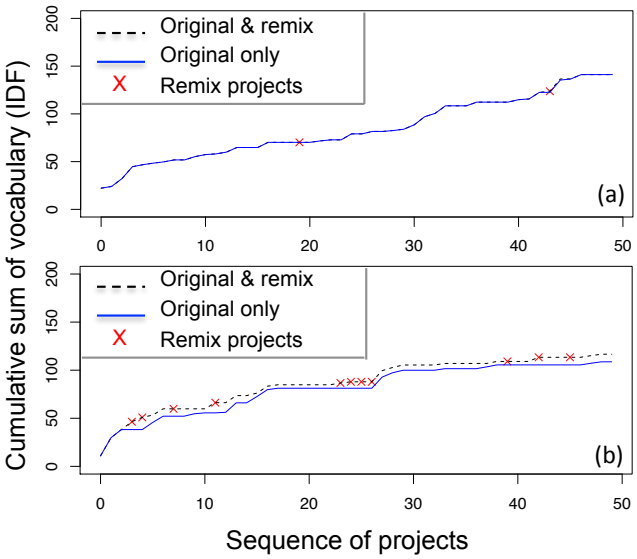

Figure 11. Examples from cluster D (the most progress group). (a) and (b) reach 110-140 cumulative sum of vocabulary in the end.

Figure 11 shows example trajectories from the most progressed group in cluster D. These trajectory graphs start increasing from the beginning and reach over the cumulative value of 50 before the 5th projects. This is a steep increase compared to the corresponding portions in Figures 5, 6, and 7. The graphs in Figure 11 (a) keep increasing until they reach close to 140. Graphs in Figure 11 (b) show a slight jump at the 27th project due to vocabulary increase in the original project. Another observation to note is that four remix projects occur consecutively right before this jump. They do not directly contribute to the vocabulary increase; however, it's possible that those remix projects indirectly helped learning new blocks and contributed to the vocabulary increase in the *Original only* graph.

DISCUSSION AND LIMITATIONS

We found that our approach – modeling and clustering of learning trajectories – was effective in answering our research questions. We could quantitatively measure and analyze the three aspects listed below, both at macroscopic (Figure 5) and microscopic levels (Figures 8-11):

1. The amount of learning using the cumulative sum at the end of a trajectory
2. The speed of learning using the slopes of regression lines of a trajectory
3. Potential prior programming knowledge using the first value of a trajectory

The clustering analysis unveiled four patterns of trajectories and corresponding subpopulations in our dataset – from the least progress (cluster A) to the most progress (cluster D). We observed that the quantities listed above in 1, 2, and 3 were positively correlated with each other. Users in cluster A (least progress) had the smallest amount of vocabulary learning, their learning speed was the slowest, and their starting vocabulary sum was only 2.5 on average (Figure 5 and 6). Referring to vocabulary weights in Figure 2, the vocabulary sum of 2.5 means the projects of users in A contained only one type of vocabulary (e.g., list_contains (IDF=2.5)), or two (e.g., set_xpos (1.5) and keypressed (1)). The trajectory of learning in cluster A shows slow increase in the first 10 (slope = 0.6) and the following 10 projects (slope = 0.4), and then stabilizes after the 20th project for the Original only graph. The larger gap between the two trajectory graphs may indicate that the users in A incorporated only a small number of new vocabulary blocks from the remix projects into their original projects.

Users in D showed characteristics of 'expert' learners – fastest learning, largest amount of vocabulary use, and a highest starting vocabulary sum of 17 (Figure 5-6). A vocabulary sum of 17, for example, means that their first projects contained several types of blocks: 'changecostumeindexb (3.8)', 'allmotorson (3.2)', 'sensorpressed (2.8)', 'mousey (2)', 'hidevariable (1.8)', 'add_operator (1.4)', 'not_operator (1.2)', and 'changegraphiceffectby (0.8).' If a vocabulary of smaller weights were used, many more different types of blocks might have been included in a project. The trajectory of their vocabulary growth shows a steep increase in the first 10 projects, where the slope of 4.9 indicates fast learning. At the 10th project, the trajectory reaches 70, and then it keeps increasing even after the 20th project.

Figure 7 and Table 3 also support that the users in D showed characteristics of expert learners. In Figure 7, users in D tend to use blocks with higher weight for their projects compared to the users in A. It might be interpreted in three ways: (1) users in D use blocks with higher weight in general compared to those in A, (2) the proportion of advanced users using blocks with higher weight was larger in D than in A, or (3) a combination of these two cases. From the unique words of users in D in Table 3, we can also postulate that these users create unique and sophisticated games which require advanced programming skills.

Limitations

Although we identified patterns of learning trajectories, we may not know the reasons behind the different levels of learning outcomes for users with certainty. For example, there can be several reasons for the use of only a small number of vocabulary blocks by the users in cluster A: the characteristics of their projects may not require a large list of vocabulary. This might be the case in art projects, since visualizing pictures only requires a single keystroke to advance to the next picture. Another possibility is that of very inexperienced or very young users who do not understand the usage of more sophisticated blocks (e.g., blocks with high IDF values in Figure 2). We also do not know the reasons for differences in starting values of the first original projects (Figure 6). We were limited by the lack of demographics and user profile data. In addition, the nature of the Scratch platform is flexible and not goal-directed; therefore, users are free to develop the projects they wish over time without concern for vocabulary acquisition.

In our dataset, we included only the long-term users who created at least 50 original projects, and analyzed their first 50 projects. This selection criterion may have limited our scope of analysis to focus on a phenomenon occurring only in a small sub-community of Scratch online. Thus, lowering the user selection/analysis criteria (e.g., original projects < 50, analyzing all projects of a user) may have led to more meaningful and generalizable results.

IMPLICATIONS

Uncovering patterns in trajectories of learning implies that there exist subpopulations in the community, each of which shares similar learning characteristics (i.e., amount, speed, and, potentially, prior programming knowledge). It may allow administrators and designers of online communities to apply interventions and implement services to better support underperforming subpopulations. More importantly, our approach is based on automated modeling and clustering, thus it is scalable and applicable to much larger online communities than Scratch. Since it was successful in quantifying learning in an informal setting, applying this approach to formal settings (e.g., MOOCs) may lead to more robust results and enable deeper analyses. In addition, the patterns of learning trajectories might be connected with the learner engagement patterns [11] to acquire richer information about the users and subpopulations.

CONCLUSION AND FUTURE WORK

In this study, we presented an approach to model informal learning, which takes place in the Scratch online community. We modeled a trajectory of learning by analyzing the cumulative vocabulary block use over the first 50 projects. The K-means++ algorithm was applied to a total of 3,852 users and their learning trajectories to uncover four canonical patterns and corresponding subpopulations. Average trajectory graphs from each cluster indicated the existence of 'elite' users, who learn more vocabulary terms more quickly, and potentially have more prior programming knowledge; whereas the least progress users learn only a small number of vocabulary terms at a slower pace than other groups, and they have almost no prior programming knowledge. The text analysis of the project descriptions and comments, as well as the analysis of block usage by users from each cluster, further evidenced the different characteristics of user groups in the Scratch community. A few examples from each cluster were also analyzed to account for the chance that average graphs might oversimplify patterns of learning trajectories.

Our approach is meaningful in that the modeling and clustering of trajectory patterns enabled us to quantitatively analyze informal learning both at a single-user level (microscopic) as well as at a cluster level (macroscopic). It may also be applicable to much larger communities regardless of whether they are structured (e.g., MOOCs) or less structured (e.g., Q&A). A potential implication of our work is the opportunity for targeted intervention and support for a specific subpopulation, which may lead to an improved experience for the community members.

We aim to extend our study in two ways. First, we will apply our approach to other types of online communities for learning (e.g., Q&A forums) and compare the results with the one presented in this paper. Second, we plan to design a recommender system based on a detailed analysis of blocks and project themes in each cluster, in addition to our approach in this paper. This system will promote learning of users by suggesting programming blocks and projects so that the learning trajectory of users can be aligned with the trajectory pattern of more advanced users within his/her own cluster, as well as from the most advanced cluster.

ACKNOWLEDGMENTS

We appreciate the Lifelong Kindergarten group at MIT for publicly sharing the Scratch datasets. This work is partly based upon research supported by U.S. National Science Foundation (NSF) Award#DUE-1444277 & EEC-1408674. Any opinions, recommendations, findings, or conclusions expressed in this material are those of the author(s) and do not necessarily reflect the views of NSF.

REFERENCES

1. Arthur, D. and Vassilvitskii, S. 2007. K-means++: the advantages of careful seeding. In *Proc. SODA '07*. 1027-1035.
2. Bell, P., Lewenstein, B., Shouse, A. W. and Feder, M. A. (Eds.). 2009. Learning Science in Informal Environments: People, Places, and Pursuits. Washington, D.C.: The National Academies Press. ISBN 978-0-309-11955-9.
3. Bransford, J. D. 2007. Preparing People for Rapidly Changing Environments. Journal of Engineering Education, 2007, 96(1), 1–3.
4. Brooks, M., Basu, S., Jacobs, C., and Vanderwende, L. 2014. Divide and correct: using clusters to grade short answers at scale. In *Proc. L@S 2014*. 89-98.
5. Carliner, S. 2012. How to evaluate informal learning. Newsletters published by the Association for Talent Development. Article retrieved from http://bit.ly/1tBwXUk.
6. Committee on the Engineer of 2020. (2005). Educating the engineers of 2020: adapting engineering education to the new century. Washington, D.C.: The National Academies Press. ISBN 0-309-09649-9.
7. Darling-Hammond, L. 2010. The Flat World and Education: How America's Commitment to Equity Will Determine Our Future. New York, NY: Teachers College Press. ISBN 978-0-8077-4963-0.
8. Dutta, D. 2010. Lifelong Learning Imperative in Engineering Workshop. Summary of workshop conducted in June 17-18, 2009. Arlington, VA U.S.A.
9. Eraut, M. 2004. Informal learning in the workplace. Studies in Continuing Education, 26(2), 247-273.
10. Khalid, S. and Naftel, A. 2005. Classifying spatiotemporal object trajectories using unsupervised learning of basis function coefficients. In *Proc. VSSN 2005*. 45-52.
11. Kizilcec, R. F., Piech, C. and Schneider, E. 2013. Deconstructing disengagement: analyzing learner subpopulations in massive open online courses. In *Proc. LAK 2013*, Suthers, D., Verbert, K., Duval, E., and Ochoa, X. (Eds.). 170-179.
12. Kulkarni, C., Wei, K. P., Le, H., Chia, D., Papadopoulos, K., Cheng, J., Koller, D., and Klemmer, S. R. 2013. Peer and self-assessment in massive online classes. ACM Transactions of Computer-Human Interaction. 20 (6), Article 33, 31 pages.
13. Maloney, J., Peppler, K., Kafai, Y.B., Resnick, M., Rusk, N. 2008. Programming by Choice: Urban Youth Learning Programming with Scratch. In *Proc. SIGCSE 2008*, 367-371. New York, NY: ACM Press.
14. Michalchik, V. and Gallagher, L. 2010. Naturalizing Assessment. Curator: The Museum Journal, 53: 209–219.
15. Monroy-Hernández, A. 2007. ScratchR: sharing user-generated programmable media. In *Proc. IDC 2007*. 167-168.
16. Monroy-Hernández, A. 2009. Designing a website for creative learning. In *Proc. WebSci 2009*: Society On-Line.
17. MOOCs Directory. Retrieved from http://www.moocs.co/.
18. Morris, B. and Trivedi, M. 2009. Learning trajectory patterns by clustering: Experimental studies and comparative evaluation. In *Proc. CVPR 2009*. Miami, FL, USA, 312-319.
19. Rennie, L. J. 2007. Chapter 7: Learning Science Outside of School. In Handbook of Research on Science Education, S. K. Abell and N. G. Lederman. (Eds.). 125-167. Mawah, NJ: Lawrence Erlbaum Associates.
20. Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K., Millner, A., Rosenbaum, E., Silver, J., Silverman, B., and Kafai, Y. 2009. Scratch: Programming for All. Commun. ACM 52, 11 (November 2009), 60-67.
21. R Script for K-means Clustering Analysis. Retrieved from http://www.mattpeeples.net/kmeans.html.
22. Salton, G., & Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. Information Processing and Management, 24(5), 513–523.
23. Scratch™ Online. Retrieved from http://scratch.mit.edu.
24. Scratch™ Online Statistics. Retrieved from http://scratch.mit.edu/statistics/.
25. Scratch Research Data Sharing Agreement. Retrieved from http://llk.media.mit.edu/scratch-data/.
26. Singh, S. and Chauhan, N. C. 2011. K-means v/s K-medoids: A Comparative Study. National Conference on Recent Trends in Engineering & Technology.
27. Sparck-Jones, K. 1972. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation. 28(1), 11-21.
28. StackExchange. Retrieved from http://stackexchange.com/.
29. Sylvan, E. A. 2010. Predicting social influence and project influence in Online Communities of Creators. In *Proc. ICLS 2010*, Kimberly Gomez, Leilah Lyons, and Joshua Radinsky (Eds.), International Society of the Learning Sciences, 455-457.
30. Wilkowski, J., Russell, D. M., and Deutsch, A. 2014. Self-evaluation in advanced power searching and mapping with Google MOOCs. In *Proc. L@S 2014*. 109-116.
31. W3Schools. Retrieved from http://www.w3schools.com/