Learning Transfer: does it take place in MOOCs?

An Investigation into the Uptake of Functional Programming in Practice

Guanliang Chen*, Dan Davis†, Claudia Hauff and Geert-Jan Houben
Delft University of Technology
Delft, the Netherlands
{guanliang.chen, d.j.davis, c.hauff, g.j.p.m.houben}@tudelft.nl

ABSTRACT

The rising number of Massive Open Online Courses (MOOCs) enable people to advance their knowledge and competencies in a wide range of fields. *Learning* though is only the first step, the *transfer* of the taught concepts into practice is equally important and often neglected in the investigation of MOOCs. In this paper, we consider the specific case of `FP101x` (a functional programming MOOC on edX) and the extent to which learners alter their programming behaviour after having taken the course. We are able to link about one third of all `FP101x` learners to `GitHub`, the most popular social coding platform to date and contribute a first exploratory analysis of learner behaviour beyond the MOOC platform. A detailed longitudinal analysis of `GitHub` log traces reveals that (i) more than 8% of engaged learners transfer, and that (ii) most existing transfer learning findings from the classroom setting are indeed applicable in the MOOC setting as well.

INTRODUCTION

The rising number of MOOCs enable people to learn & advance their knowledge and competencies in a wide range of fields. *Learning*, though, is only the first step; the *application* of the taught concepts is equally important, as knowledge that is learned but not frequently applied or activated is quickly unlearned [28, 6, 25].

Existing investigations into student learning *within* MOOC environments are commonly based on pre- & post-course surveys and log traces generated within those environments by the individual learners [15]. While student learning is indeed an important measure of success, we argue that another key measure is the amount of **learning transfer** [17] that is taking place: do learners actually utilize the newly gained knowledge in practice? Are learners expanding their knowledge in the area over time or do they eventually move back to their pre-course knowledge levels and behaviours? These are important questions to address in the learning sciences, and their answers will enable us to shape the MOOCs of the future based on empirical evidence.

The main challenge researchers face in answering these questions is the lack of *accessible*, *large-scale*, *relevant* and *longitudinal* data traces outside of MOOC environments. While learners can be uniquely identified within a MOOC platform, at this point in time we have no general manner of capturing their behavioural traces outside of these boundaries.

Not all is lost though. Social Web platforms (Twitter being the prime example) have become a mainstay of the Web. They are used by hundreds of millions of users around the world and often provide open access to some — if not all — of the data generated within them. While most of these platforms are geared towards people's private lives, in the past few years social Web platforms have also begun to enter our professional lives.

One such work-related social Web platform is `GitHub`[1]; it is one of the most popular *social coding* platforms worldwide with more than 10 million registered users. Hobbyists and professional programmers alike use `GitHub` to collaborate on programming projects, host their source code, and organize their programming activities. As `GitHub` was founded in 2007, we have potential access to log traces reaching several years into the past; moreover, its continuously increasing popularity will enable us to observe our learners over years to come. The potential of `GitHub` for behavioural mining has long been recognized by the software engineering research community where `GitHub` is one of the most popular data sources to investigate how (groups of) people code.

Thus, for MOOCs with a strong focus on programming concepts, we consider `GitHub` to be one of the most detailed and openly accessible sources of learners' relevant behavioral traces outside of the MOOC environment itself. Concretely, we analyze `FP101x`[2], an edX MOOC covering basic functional programming concepts. Of the 37,485 learners that registered for `FP101x` we matched

*The author's research is supported by the *Extension School* of the Delft University of Technology.
†The author's research is supported by the *Leiden-Delft-Erasmus Centre for Education and Learning.*

L@S 2016, April 25–26, 2016, Edinburgh, UK.

http://dx.doi.org/10.1145/2876034.2876035

[1] `https://github.com/`
[2] `https://www.edx.org/course/introduction-functional-programming-delftx-fp101x`

12,415 (33.1%) to their respective GitHub accounts, enabling a first large-scale analysis of the uptake of taught programming concepts in practice.

Here, we are foremost interested in exploring to what extent the course affects learners *after it has ended*. We are guided by the following three **R**esearch **Q**uestions:

RQ1 *To what extent* does the transfer of learned concepts take place?

RQ2 What *type of learners* are most likely to make the transfer?

RQ3 How does the transfer manifest itself *over time*?

Based on these guiding questions we have formulated seven research hypotheses which build on previous research efforts in work-place and classroom learning. In contrast to our work though, in these settings, the investigations are mostly based on questionnaires and interviews instead of behavioural traces. To the best of our knowledge, learning transfer has not yet been investigated in the context of MOOCs. Gaining deeper insights about the (lack of) learning transfer in MOOCs will lead to more informed discussions on the practical purposes and benefits of MOOCs. The main contributions of our work can be summarized as follows:

- We investigate to what extent learning transfer insights gained in work-place and classroom settings hold in the MOOC context. We find that the majority of findings are also applicable in the case of MOOCs.
- We introduce the use of *external* social Web-based data sources to complement learner traces within MOOC environments as a means to capture much more information about MOOC learners.
- We introduce GitHub as a specific large-scale data source to mine relevant longitudinal behavioural traces about learners before, during and after a programming-oriented MOOC.

BACKGROUND

Meaningful, robust educational experiences transcend rote memorization of facts and leave the learner empowered to take on new problems and practice novel ways of thinking. In tracking student activity from the learning context (edX) to a real-world, practical one (GitHub) over a period of three years, the present study observes the first two of the three criteria of *robust learning* as outlined in [17]: (i) application in new situations different from the learning context, (ii) retained over the long-term, and (iii) prepares for future learning. Gaining a better understanding of how students apply what they learn in online learning environments over an extended time frame enables instructors to design future courses that induce more robust learning.

Some researchers [27, 26] have begun to look beyond traces generated in online learning environments, by utilizing post-course surveys or conducting post-course interviews with MOOC students.

Although the early studies of transfer stemmed from educational issues, the majority of recent learning transfer research literature is concerned with work-place training in Human Resource Development (HRD) [7]. With the recent influx of student activity data generated from digital learning environments, we can now empirically measure not only the rate of transfer, but other contributing factors as well. That, in tandem with the established surveying strategies used by HRD, promises to fundamentally change the way we think about measurable learning outcomes.

Learning transfer is the application of knowledge or skills gained in a learning environment to another context [2]. While training situations in professional environments have a clear target context (the job), this is not the case with most academic learning situations. Students are generally taught a broad set of skills and knowledge which they may apply in countless ways. This deliberately broad definition encapsulates both *near transfer* (to similar contexts) and *far transfer* (to dissimilar contexts) [4] and avoids the subjective question of *how* similar or different the learning context is from the target context, as we are only concerned with whether the student transferred the learned skills or knowledge beyond the learning context.

Due to their rising popularity as a professional development tool and their roots as an educational resource, MOOCs serve as an ideal source of information to gain new insights on learning transfer. Studies have begun to discuss the learners' intention to apply what they've learned in MOOCs but do not continue to track student activity beyond the learning platform [12]. The present research aims to reoperationalize [10] the understanding of learning transfer given the emerging possibilities of user modeling and learning analytics from the current standard of *reported* learning transfer towards *observed* learning transfer.

Yelon & Ford [29] offer a key distinction in transfer that differentiates *open* and *closed* skills. Open skill training programs include "leadership and interpersonal skills training," and typical closed skill trainings include "various technical training and computer software training." This emerges as an important distinction. In a study in which Blume et al. [7] found post-training knowledge (PTK) and post-training self-efficacy (PTSE) to have similar correlations with learning transfer, PTK and PTSE for closed skills resulted in lower correlation coefficients than for open skills. Independent of performance, self-efficacy is a person's self-reported ability to successfully complete a future task [3]. Knowledge is measured as a result of a task—answering a quiz question correctly indicates possession of that knowledge [7].

Regarding the maintenance and persistence of learning transfer over time, Blume et al. analyzed how the *amount of time* (the "lag") between the end of training and the beginning of the transfer study affects learning transfer. They found that in studies with at least

some lag time between training and testing, learners exhibited significantly lower post-training knowledge and post-training self-efficacy than those that tested students immediately following training [7].

In their survey of training professionals from 150 organizations, [24] report that 62% of employees in their organization *"effectively apply what they learned in training"* to their job immediately, 42% after six months, and 34% after one year. Other studies directly survey students in gathering self-reported data about learning transfer [19]. Another manner by which researchers have measured transfer is through assessment questions following instruction that, in order for students to answer correctly, would have to apply what they learned to a new context or problem [20, 1]. The present study examines transfer as a more naturally occurring, un-elicited phenomenon that the learners undertake and exhibit on their own accord.

FP101X

Introduction to Functional Programming (or short `FP101x`) is a MOOC offered on the edX platform. The course introduces learners to various functional programming concepts; all programming is performed in the functional language *Haskell*.

The first iteration of the course ran between October 15, 2014 and December 31, 2014. As is common in MOOCs today, learners were invited to participate in a pre-course and a post-course survey containing questions on the motivation of the learners, the perceived quality of the course, etc. In August 2015 we approached a subset of learners for an additional post-course survey.

The course was set up as an xMOOC [23]: lecture videos were distributed throughout the 8 teaching weeks. Apart from lectures each week, exercises ("homeworks" and "labs") were distributed in the form of multiple choice (MC) questions. While homework questions evaluated learners on their understanding of high-level concepts and code snippets (e.g. *"What is the result of executing [...]?"*), labs required learners to implement programs themselves. To enable fully automatic evaluations, all lab work was also assessed through MC questions. Each of the 288 MC questions was worth 1 point & could be attempted once. Answers were due 2 weeks after the release of the assignment. To pass the course, $\geq 60\%$ of all MC questions had to be answered correctly.

Overall, 37,485 users registered for the course. Fewer than half (41%) engaged with the course, watching at least one lecture video. The completion rate was 5.25%, in line with similar MOOC offerings [18]. Over 75% of the learners were male and more than 60% had at least a Bachelors degree.

METHODOLOGY

We first outline and justify the seven research hypotheses upon which we ground our work. Next, we describe in detail how to verify them empirically based on course questionnaire data, edX logs and `GitHub` data traces.

Research Hypotheses

Based on prior work we can make the following hypothesis related to **RQ1**:

H1 *Only a small fraction of engaged learners is likely to exhibit learning transfer.*
While previous works, e.g. [24], note transfer rates of up to 60%, we hypothesize our rate to be much lower, due to the natural setting we investigate, the difficulty of the topic (closed skills) and the generally low retention rate of MOOCs.

A large part of existing literature has focused on the different dimensions of a learner that may be indicative of a high or low transfer rate. Thus, the following research hypotheses are all related to **RQ2**, which focuses on the *type* of learner exhibiting transfer.

H2 *Intrinsically motivated learners with mastery goals are more likely to exhibit learning transfer than extrinsically motivated learners.*
[22] found that, in academic settings, mastery goals are more consistently linked to transfer success than performance goals. This was measured by instructors guiding students through either mastery- or performance-oriented experimental conditions and comparing their assessment scores. In line with intrinsic motivation, mastery goals are characterized by a learner's intention to understand and develop new knowledge and abilities. Performance goals, extrinsically motivated, are those sought after in order to obtain positive judgements from others [9].

H3 *Learners expressing high self-efficacy are more likely to actively apply their trained tasks in new contexts.*
In other words, in both academic and professional settings, if you believe that you are able to do something, you are more likely to try it [13, 14, 16, 22].

H4 *Experienced learners (high ability levels) are more likely to transfer trained skills and knowledge in order to maintain and improve performance levels* [13].

H5 *Learners reporting a high personal capacity (time, energy and mental space) for transfer are more likely to actually exhibit learning transfer* [16].

H6 *Learners exhibiting a high-spacing learning routine are more likely to exhibit learning transfer than learners with a low-spacing learning routine.*
Here, high-spacing refers to a larger number of discrete learning sessions than low-spacing with few learning sessions each lasting a long time (i.e. "cramming") [21, 11, 5].

Finally, for **RQ3** we investigate the following hypothesis:

H7 *The amount of exhibited transfer decreases over time* [24].

From Hypotheses To Measurements

Table 1 shows an overview of the data sources used to investigate each research hypothesis.

	Pre CS	Post CS	edX Logs	GitHub Logs
H1			✓	✓
H2	✓	✓		✓
H3		✓		✓
H4	✓			✓
H5		✓		✓
H6			✓	✓
H7				✓

Table 1. Overview of the different data sources used to investigate each research hypothesis. *CS* refers to the conducted Course Surveys (before and after the course).

To explore **H1** we relate learners' performance during the course (as found in the edX logs) to their development activities on `GitHub`.

To determine the impact of learners' motivation on learning transfer (**H2**), we distinguish learners based on their answers to several pre/post-course survey questions we manually established as being motivation-related. To determine *intrinsic motivation* we identified six question-answer pairs including the following two[3]:

- What describes your interest for registering for this course?; Answer: My curiosity (in the topic) was the reason for me to sign up for this course [Pre CS, 5-point Likert]
- Express your level of agreement with the following statement.; Answer: Course activities piqued my curiosity. [Post CS, 5-point Likert]

Similarly, for *extrinsic motivation* we determined nine appropriate question-answer pairs, including:

- What describes your interest for registering for this course? Choose the one that applies to you the most; Answer: My current occupation motivated me to enroll in the course. [Pre CS, 5-point Likert]
- Considering your experience in this, how much do you agree with the following statement?; Answer: The course was compulsory for me [Post CS, Multiple choice]

Learners' belief in their ability to complete a task (**H3**), can be inferred based on a question asking the learners to express their level of agreement with a set of statements from the validated General Self-Efficacy Scale [8]:

- I can describe ways to test and apply the knowledge created in this course. [Post CS, 5-point Likert]
- I have developed solutions to course problems that can be applied in practice. [Post CS, 5-point Likert]
- I can apply the knowledge created in the course to my work or other non-class related activities. [Post CS, 5-point Likert]

The prior expertise of learners (**H4**) can both be inferred from survey questions as well as from the `GitHub` logs. The questions utilized are:

- Is your educational background related to (Functional) Programming? [Pre CS, 5-point Likert]
- Do you have professional experience in this field? [Pre CS, 5-point Likert]

The personal capacity (**H5**) of a learner is inferred based on two questions:

- Did any of the following negatively affect your participation in the course? [Post CS, 5-point Likert]
- Considering your experience in this course, how much did each of the technical issues affect your participation? [Post CS, 5-point Likert]

Responses to these questions allow learners to share which factors inhibited and distracted them from engaging with the course. Examples of responses to these questions range from personal problems, such as family obligations and medical issues, to technical trouble, such as slow Internet or hardware problems.

H6 considers the manner in which learners learn and can be inferred solely based on edX log traces which will be explained in more detail in the section below. Finally, **H7**, the extent to which functional programming is employed and applied by the learners over time can be inferred from `GitHub` logs alone.

edX Logs

For each learner, we collect all available traces (between October 1 and December 31, 2014), such as the learner's clicks & views, provided answers to MC questions as well as forum interactions. Using the `MOOCdb` toolkit[4] we translate these low-level log traces into a data schema that is easily queryable.

To investigate **H6**, for each learner the learning routine is determined based on their edX logs. We partition the learners into low-spacing and high-spacing types following [21]. Initially, all learners are sorted in ascending order according to their total time on-site. Subsequently they are binned into ten equally-sized groups. Within each group, the learners are sorted according to the number of distinct sessions on the site and based on this ordering divided into two equally-sized subgroups: learners with few sessions (low-spacing) and learners with many sessions (high-spacing). In this manner, learners spending similar amounts of time (in total) on the course site can be compared with each other.

GitHub Logs

We identify edX learners on `GitHub` through the email identifiers attached to each edX and `GitHub` account. A third of all learners that registered to `FP101x` are also active on GitHub: 12,415 learners in total. This is likely to be an underestimate of the true number of `GitHub` users (people generally have multiple email accounts), as we did not attempt to match accounts based on additional user profile information.

`GitHub` provides extensive access to data traces associated with *public* coding repositories, i.e. repositories

[3]Due to space constraints, only a subset of the identified question/answer pairs are shown.

[4]`http://moocdb.csail.mit.edu/`

visible to everyone[5]. GitHub is built around the git distributed revision control system, which enables efficient distributed and collaborative code development. GitHub not only provides relevant repository metadata (including information on how popular a repository is, how many developers collaborate, etc.), but also the actual code that was altered. As the *GitHub Archive*[6] makes all historic GitHub data traces easily accessible, we relied on it for data collection and extracted all GitHub data traces available between January 1, 2013 and July 21, 2015. We then filtered out all traces that were *not* created by our edX learners, leaving us with traces from 10,944 learners. Of the more than 20 GitHub *event types*[7], we only consider the PushEvent as vital for our analysis.

```
{
  "_id" : ObjectId("55b6005de4b07ff432432dfe1"),
  "created_at" : "2013-03-03T18:36:09-08:00",
  "url" : "https://github.com/john/
           RMS/compare/1c55c4cb04...420e112334",
  "actor" : "john",
  "actor_attributes" : {
        "name" : "John Doe",
        "email" : "john@doe.com"
  },
  "repository" : {
        "id" : 2.37202e+06,
        "name" : "RMS",
        "forks" : 0,
        "open_issues" : 0,
        "created_at" : "2011-09-12T08:28:27-07:00",
        "master_branch" : "master"
    }
}
```

Figure 1. Excerpt of a GitHub PushEvent log trace.

Every time code is being updated ("pushed" to a repository), a PushEvent is triggered. Figure 1 contains an excerpt of the data contained in each PushEvent. The most important attributes of the event are the created_at timestamp (which allows us to classify events as before/during/after the running of FP101x), the actor (the user doing the "push") and the url, which contains the URL to the actual *diff file*. While the git protocol also allows a user to "push" changes by another user to a repository (which is not evident from inspecting the *diff file* alone), this is a rare occurrence among our learners: manually inspecting a random sample of 200 PushEvents showed 10 such cases. A *diff file* shows the difference between the last version of the repository and the new one (after the push) in terms of added and deleted code. An example excerpt is shown in Figure 2. For each of the identified 1,185,549 PushEvents by our learners, we crawled the corresponding *diff file*, as they allow us to conduct a fine-grained code analysis. As a first step, we identified the additions and deletions a user conducts in each programming language based on the filename extensions found in the corresponding *diff file*. We consider code updates in the following nine functional languages as clear evidence for functional programming: *Common Lisp, Scheme, Clojure, Racket, Erlang, Ocaml, Haskell, F#* and *Scala*. We also log changes made in any of the other 20 most popular programming languages found on GitHub in the same manner. Any filename extension not recognized is first checked against a blacklist (which includes common filename extensions for images, compressed archives, audio files, etc.) and if not found, the change is classified as *Other*.

[5]Data traces about private repositories are only available to the respective repository owner.

[6]https://www.githubarchive.org/

[7]https://developer.github.com/v3/activity/events/types/

```
diff --git a/viewsA.rb b/viewsA.rb
index e37bca1..3ad75e4 100644
--- a/viewsA.rb
+++ b/viewsA.rb
@@ -26,6 +26,16 @@ def new
     @shift = Shift.new
   end
...
diff --git a/config/routes.rb b/config/routes.rb
index e576929..27ce68f 100644
--- a/config/routes.rb
+++ b/config/routes.rb
@@ -29,6 +29,7 @@
       put 'secondary'
...
```

Figure 2. Excerpt of a *diff file*. Two files were changed (*viewsA.rb* and *routes.rb*). The extension *.rb indicates code written in Ruby.

RESULTS

We first present some basic characteristics of FP101x, before delving into the analyses of our research questions and hypotheses.

FP101x Overview

We partition our set of all *registered* FP101x learners according to two dimensions: (i) learners with and without a GitHub account, and (ii) learners with and without prior expertise in functional programming. In the latter case, we consider only those learners that could be identified on GitHub. We define **Expert learners** as those who used any of our nine identified functional programming languages before the start of the course to a meaningful degree (i.e. more than 25 lines of functional code being added). The characteristics of these learner cohorts are listed in Tables 2 and 3.

When considering the GitHub vs. non-GitHub learners, we observe significant differences along the dimensions of engagement and knowledge:

- GitHub learners are on average **more engaged** with the course material (significantly more time spent on watching lecture videos and significantly more questions attempted).
- GitHub learners exhibit **higher levels of knowledge** (significantly more questions answered correctly).

Zooming in on the GitHub learners and their functional programming expertise, we find the differences to be enlarged: Expert learners have a higher completion rate (more than double that of non-Expert learners), attempt

to solve significantly more problems and are significantly more accurate in answering. Experts are also more engaged in terms of forum usage - 8% post at least once compared to 4% of the non-Expert learners.

	All Learners	GH Learners	Non-GH Learners
#Enrolled learners	37,485	12,415	25,070
Completion rate	5.25%	7.71%	4.03%
%Learners who watched at least one video	40.84%	50.58%	36.02%
Avg. time watching video material (in min.)	31.87	44.56 †	25.59 †
%Learners who tried at least one question	23.28%	31.94%	18.99%
Avg. #questions learners attempted to solve	22.07	31.29 †	17.51 †
Avg. #questions answered correctly	18.30	26.54 †	14.22 †
Avg. accuracy of learners' answers	16.36%	23.41% †	12.86% †
#Forum posts	8,157	3,726	4,431
%Learners who posted at least once	2.84%	4.27%	2.13%
Avg. #posts per learner	0.22	0.30 †	0.18 †

Table 2. Basic characteristics across all learners and their partitioning into GitHub (GH) and non-GitHub learners. Significant differences (according to Mann-Whitney) between GH and non-GH learners are marked with †($p < 0.001$).

	Expert Learners	Non-Expert Learners
#Enrolled learners	1,721	10,694
Completion rate	15.05%	6.53%
%Learners who watched at least one video	64.44%	48.35%
Avg. time watching video material (in min.)	69.61 †	40.53 †
%Learners who tried at least one question	48.69%	29.24%
Avg. #questions learners attempted to solve	57.86 †	27.02 †
Avg. #questions answered correctly	50.24 †	22.73 †
Avg. accuracy of learners' answers	37.96% †	21.06% †
#Forum posts	1,612	2,114
%Learners who posted at least once	7.55%	3.74%
Avg. #posts per learners	0.94 ‡	0.20 ‡

Table 3. Basic characteristics when partitioning the GitHub learners according to prior functional programming expertise. Significant differences (according to Mann-Whitney) between Expert and Non-Expert learners are marked with †($p < 0.001$) and ‡($p < 0.01$).

Finally, we note that we repeated this analysis on the subset of *engaged* learners only, where we consider all learners that attempted to solve at least one MC question or watched at least one video. While the absolute numbers vary, the trends we observe for the different partitions of learners in Tables 2 and 3 remain exactly the same.

Learning Transfer

Let us first consider the general uptake of functional programming languages. We can split each learner's `GitHub` traces into three distinct sequences according to their timestamp: traces generated *before*, *during* and *after* `FP101x`. We are interested in comparing the before & after and will mostly ignore the activities generated during `FP101x`.

Expert Learners

Overall, 1,721 of all `GitHub` learners have prior functional programming experience (our **Expert learners**). 1,165 of those are also *engaged* with `FP101x` (the remainder registered, but did not engage), leading to nearly a third (29.4%) of all engaged `GitHub` learners having pre-`FP101x` functional programming experience.

Most of our `GitHub` learners though are not continuously coding functionally: Figure 3 shows for each month of `GitHub` logs (January 2013 to July 2015) the *unique* number of `GitHub` learners programming functionally - while in 2013 less than 250 of our `GitHub` learners were active per month, by 2015 this number has increased to nearly unique 600 active users a month. Thus, the trend to functional programming is generally increasing. Most learners though are not actively using functional languages on a monthly basis.

How much functional code do our engaged Expert learners produce over time? An answer to this question delivers Figure 4: here, for each month, the functional coding activities (calculated as the additions made in functional languages as a fraction of all additions made in recognized programming languages) are averaged across all engaged Expert learners. Again we observe that over the years functional programming has become more popular. By September 2014 (right before the start of `FP101x`), on average more than 36% of coding activities are functional. What is surprising (and somewhat counter-intuitive) is the steady decline of functional activities after the end of `FP101x`. If we restrict our engaged Expert Learners to those 542 learners with functional traces before *and* after `FP101x` (Figure 5), the results are more in line with our expectations: functional programming is continuously gaining in popularity and a peak in activities is observed in the two months following `FP101x`[8]. Thus, 46.5% of engaged Expert learners did continue to program functionally after the end of `FP101x`.

Novice Learners

Most interesting to use are the **Novice Learners**: to what extent do learners that did not program (meaningfully) in functional languages before `FP101x` take it up afterwards? We find 522 such learners — 4.3% of all `GitHub` learners. If we restrict ourselves to engaged `GitHub` learners, we are left with 336 Novice learners (8.5% of all engaged `GitHub` learners). Figure 6 shows

[8]The drop in July 2015 is explained by the non-complete log coverage of July (the log ends on July 21, 2015).

the evolution of their functional programming usage over time: the uptake after the end of FP101x is substantial, on average more than 35% of all activities are conducted subsequently in functional languages! While there is no substantial increase after the initial uptake over time, there is also no significant drop. Since the average can only provide limited insights, we drill down to the individual user level in Figure 7: the usage of functional programming is highly varied; 50% of the Novice Learners use it for less than 10% of their programming activities, while some learners almost exclusively code in functional languages. Finally, we also consider which functional languages these Novice learners code in. Figure 8 shows that a month after FP101x ended (January 2015), Haskell contributions made up 48% of all contributions, but continued dropping to a low of 14.5% in June 2015. Scala on the other hand (the most popular functional language in industrial settings) slowly rises in popularity over time and by June 2015 makes up roughly half of the functional contributions. Other functional languages play less of a role. Conducting a similar analysis on our engaged Expert learners (not shown here), we find that on average across all months, 47% ($\sigma = 7.4$) of all functional activities are in Scala, whereas 24.0% ($\sigma = 5.5$) are in Haskell. The distribution of functional languages is stable over time. The only outliers can be found in the three months of FP101x, where Haskell contributions rise significantly.

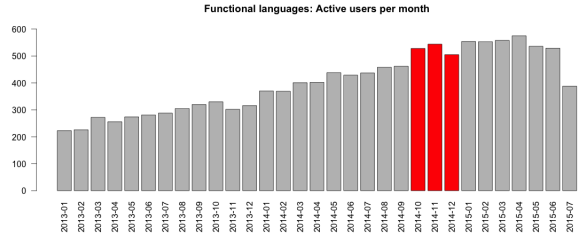

Figure 3. Number of unique users actively using a functional language. FP101x ran during the highlighted region.

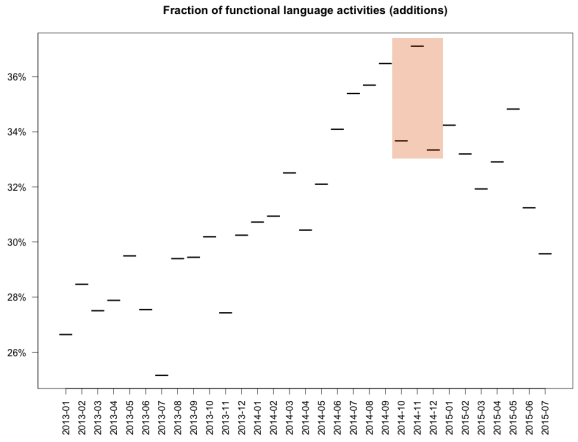

Figure 4. Fraction of functional programming activities among the 1,165 engaged Expert Learners. FP101x ran during the highlighted region.

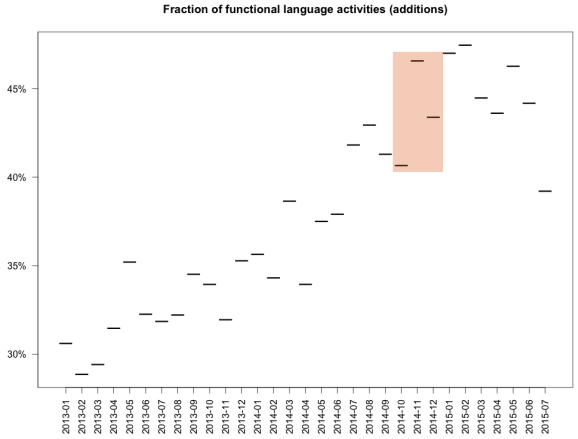

Figure 5. Fraction of functional programming activities among the 542 engaged Expert Learners with functional activities before & after FP101x. FP101x ran during the highlighted region.

Transfer Learning Hypotheses

On which learners should (or can) we investigate our seven research hypotheses? Ideally, we rely on all learners that engaged with the course and for whom GitHub traces are available. However, for Expert learners we are unable to determine the amount of transfer: since our analysis of functional coding is based on activities in functional languages (instead of a more fine-grained analysis of the type of functional concepts employed), we are not able to determine whether learners that programmed functionally before acquired knowledge in FP101x and applied it in practice (a direction of future work). Only for the engaged Novice learners can we be confident that FP101x actually impacted their programming practice and that the observed transfer is likely a result of FP101x.

Considering **H1**, we observe a **transfer rate of at least 8.5%** (i.e. among the 3,965 engaged GitHub learners we found 336 Novice learners that began programming functionally after FP101x). This percentage can be considered as a lower bound, as we (due to the reasons listed above) do not consider engaged Expert learners here. Only a minority (70) of the 336 engaged Novice learners did pass FP101x, indicating that transfer and pass rate are related but not synonymous. In fact, while the 70 Novice learners that successfully completed the course remained mostly active until the final course week (Figure 9), nearly 40% of all engaged Novice learners became inactive after week 1.

To investigate **H2, H3, H4, H5** and **H6**, for each hypothesis, we partition our 336 engaged Novice Learners who made the transfer according to the investigated dimensions (e.g. intrinsic vs. extrinsic motivation). Recall that the partitioning of the learners relies on their self-reported abilities in the pre- and post-course surveys. Similar to the retention rate, the return rate for

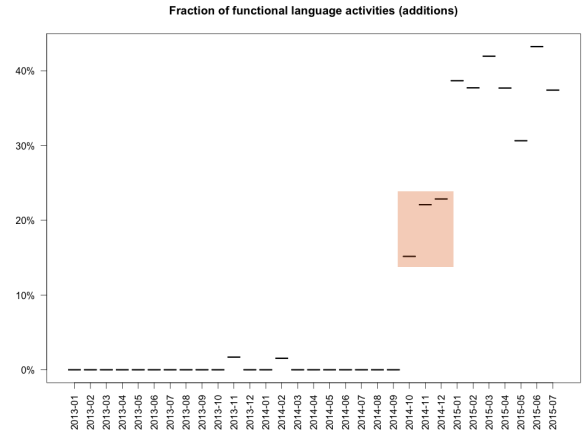

Figure 6. Fraction of functional programming activities among the 336 engaged Novice Learners with functional activities after `FP101x`. `FP101x` ran during the highlighted region.

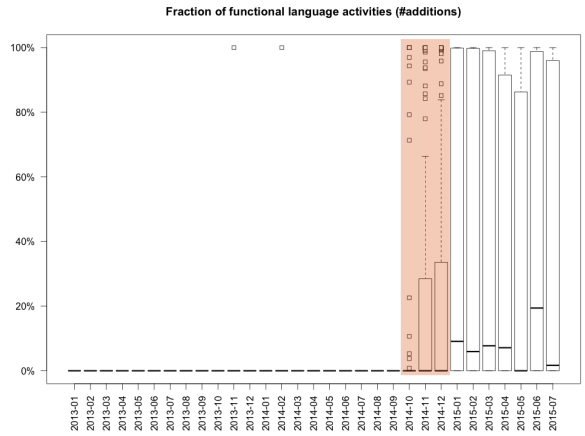

Figure 7. Distribution of functional programming activities among the 336 engaged Novice Learners with functional activities after `FP101x`. `FP101x` ran during the highlighted region.

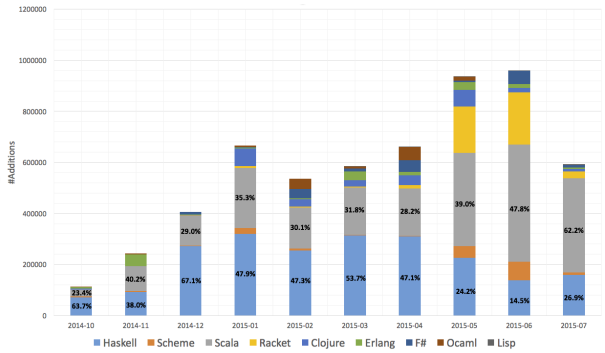

Figure 8. Functional languages used by the 336 engaged Novice Learners during and after `FP101x`. Best viewed in color.

such questionnaires is very low and many learners do not participate in these surveys for a variety of reasons. Table 4 shows the partitioning of our engaged Novice

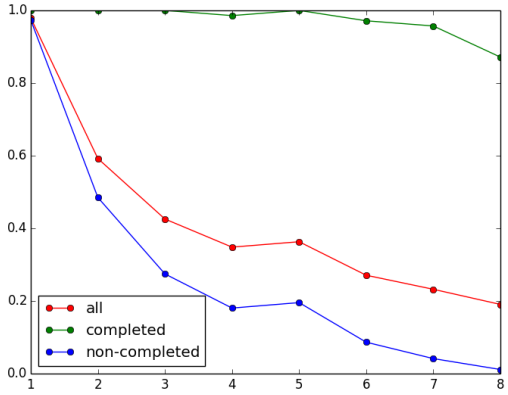

Figure 9. Fraction of the 336 engaged Novice learners remaining active in each course week. 70 Novice learners completed `FP101x` successfully, 266 did not complete it.

learners based on their survey data. The majority of learners cannot be assigned to a dimension due to a lack of data. Despite the low numbers, we do observe that the transfer learning hypotheses seem to hold in `FP101x` (for those learners for which it is possible to measure their effect): learners are more likely to make the transfer if (i) they are intrinsically motivated, (ii) have high self-efficacy, (iii) are more experienced programmers, and (iv) report a high personal capacity. Even though the number of learners we were able to investigate are small, we consider this as first evidence that transfer learning hypotheses also hold in the MOOC setting.

	Dimensions		N/A
H2 Motivation	Extr.: 12	Intr.: 28	296
H3 Self-efficacy	High: 23	Low: 5	308
H4 Experience	A lot: 42	Little: 25	269
H5 Personal capacity	High: 22	Low: 10	304

Table 4. Partitioning of the 336 Novice learners according to several dimensions. The last column shows the number of learners that could not be assigned (N/A) to a dimension.

Groups	Low spacing	High spacing
0	2	2
1	9	9
2	6	16
3	10	20
4	21	21
5	19	16
6	19	22
7	20	22
8	16	29
9	27	30

Table 5. The number of Novice Learners falling into spacing groups.

To answer **H6** (high-spaced learners are more likely to transfer), we binned all GitHub learners according to their total time and number of distinct sessions in the FP101x edX environment, as outlined earlier. This creates 10 groups, with learners in *Group 0* spending the least amount of time and learners in *Group 9* spending the most amount of time on the course site. Thus, each group contains those learners that roughly spent the same amount of time on the site. Further, within each group, learners are divided according to the number of distinct sessions. In Table 5 we report how many engaged Novice learners fell into each group and which part of the group — the high-spacing or the low-spacing one. While 187 engaged Novice learners are classified as high-spacing, 149 are classified as low-spacing. Thus, there is some indication that **H6** holds. However, the observed difference is rather small.

To conclude this section, we lastly consider **H7**. In contrast to the hypothesis (transfer decreases over time), we neither observe a significant decrease nor increase after the initial uptake as evident in Figures 6 and 7.

A Qualitative Analysis

We have found similarities and differences between transfer in classroom learning and our MOOC. Instead of speculating about the reasons for these differences, we designed a follow-up survey (containing 10 questions about learners' functional programming experiences before and after FP101x) and distributed it to subsets of GitHub learners in August 2015[9]. A second purpose of this questionnaire is to verify whether GitHub logs offer a good approximation of our learners' true behaviour. We partitioned the *engaged* GitHub learners into eight categories:

A **Novice** learners that **completed** the course but did **not transfer** (i.e. we did not observe functional GitHub traces after FP101x). #Survey responses: 131 (32% return rate).

B **Expert** learners that **completed** the course but did **not transfer**. #Survey responses: 15 (39%).

C **Novice** learners that **completed** the course and **transferred** (i.e. we observed functional GitHub traces after FP101x). #Survey responses: 11 (61%).

D **Novice** learners that **did not complete** the course, but **transferred**. #Survey responses: 1 (3%).

E **Expert** learners that **completed** the course and **continued programming functionally** (did they transfer?). #Survey responses: 20 (56%).

F **Expert** learners that did **not complete** the course but did **program functionally** after FP101x. #Survey responses: 8 (16%).

G **Novice** learners that were engaged in the course (but not completed) and did **not transfer**. #Survey responses: 93 (6%).

H **Expert** learners that were engaged in the course (but not completed) and did **not transfer**. #Survey responses: 4 (7% return rate).

How accurate are GitHub traces as approximation of learners' functional programming activities? Of those learners we had identified as Novices, 63% also self-reported as such. Of the learners we estimated to have some prior functional programming experience, 77% self-reported prior experience. In particular, the latter number is intriguing: based on our stringent methodology, we can be confident that all of our identified Expert learners did indeed functionally program before FP101x, though about a quarter self-reports otherwise. Of the learners we identified as having demonstrated learning transfer, 88% also self-reported as doing so. Of those we identified as not having demonstrated learning transfer, only 37% self-reported of not having applied anything they had learnt. An explanation for this discrepancy is based on the non-exclusive use of GitHub: while 73% indicated that they use GitHub for either work or personal coding projects, 65% use a Private/Employer's repository service, and 39% use BitBucket. While 73% is promising in that it accounts for nearly three quarters of all learners, we could only detect users who use the same email address for both their edX and GitHub account.

What are the main reasons for learners not to transfer their acquired functional programming skills? 80% of learners reporting a reason for not transferring their acquired skills report a lack of opportunities. Many learners go on to explain that the programming language standards in their work-place do not allow them to practice what they have learned. Another common sentiment is that it is difficult for some experienced programmers to suddenly change their ways. For example, when asked why they did not apply what they learned in FP101x to either work or personal projects, one respondent shared, "It takes time and effort to change old programming habits." And another shared a similar sentiment: "[It's] hard to think functionally after 25 years of imperative [programming] experience."

[9]All contacted learners had consented to additional contact.

CONCLUSIONS

We have investigated the extent of learning transfer in the MOOC setting and introduced the use of a social-Web based data source (i.e. GitHub) to complement the learner traces collected within MOOC environments. Focusing on one-third of FP101x learners we were able to link to GitHub, we made several important findings:

(1) Most transfer learning findings from the classroom setting translate into the MOOC setup; large discrepancies were only found for **H1**: the amount of observed transfer and **H7**: the development of transfer over time.

(2) The observed transfer rate in MOOCs is low. We found that 8.5% of engaged learners were indeed exhibiting transfer to varying degrees in our GitHub traces. We acknowledge that a substantial amount of programming occurs outside of GitHub (e.g. in private employer repositories). While the traces we gather offer many new insights by following learners beyond the MOOC platform for an extended period of time, considering one external data source alone is a limiting factor.

(3) The amount of transfer, operationalized as the fraction of functional coding is varying highly: about 50% of the learners transferring code less than 10% of the time functionally, while a small minority almost exclusively turns to functional languages.

(4) After the end of `FP101x`, learners making the transfer quickly identified the most industrially-relevant functional language at this moment (Scala). Over time their activities in Scala increased significantly, while their activities in Haskell (the language of `FP101x`) decreased. Overall though, after the initial uptake of functional programming, the fraction of functional activities (between 35%-40%) of all coding activities remained constant.

The limitations of the current study (only 33% of learners could be coupled to a `GitHub` account and our exploratory analysis has been conducted on the programming language type level) naturally lead to three directions for future work: (i) instead of focusing on the amount of code added per language, a more detailed analysis will determine the particular functional concepts employed and match them with the course material, (ii) programming languages are taught in a variety of MOOCs, it is an open question whether the same methodology is applicable across a variety of courses, and lastly, (iii) we will move beyond the `GitHub` platform and consider alternative external data sources.

REFERENCES

1. Adams, L. T., Kasserman, J. E., Yearwood, A., Perfetto, G. A., Bransford, J. D., and Franks, J. J. Memory access: The effects of fact-oriented versus problem-oriented acquisition. *Memory & Cognition 16*, 2 (1988), 167–175.
2. Baldwin, T. T., and Ford, K. J. Transfer of training: A review and directions for future research. *Personnel Psychology 41*, 1 (1988), 63–105.
3. Bandura, A. Self-efficacy: toward a unifying theory of behavioral change. *Psychological review 84*, 2 (1977), 191–215.
4. Barnett, S. M., and Ceci, S. J. When and where do we apply what we learn? a taxonomy for far transfer. *Psychological bulletin 128*, 4 (2002), 612–637.
5. Bjork, R. A. Memory and metamemory considerations in the training of human beings. *Metacognition: Knowing about knowing* (1994), 185–205.
6. Bjork, R. A., and Bjork, E. L. A new theory of disuse and an old theory of stimulus fluctuation. *From learning processes to cognitive processes: Essays in honor of William K. Estes 2* (1992), 35–67.
7. Blume, B. D., Ford, K. J., Baldwin, T. T., and Huang, J. L. Transfer of training: A meta-analytic review. *Journal of Management 36*, 4 (2010), 1065–1105.
8. Chen, G., Gully, S. M., and Eden, D. Validation of a new general self-efficacy scale. *Organizational research methods 4*, 1 (2001), 62–83.
9. Darnon, C., Butera, F., and Harackiewicz, J. M. Achievement goals in social interactions: Learning with mastery vs. performance goals. *Motivation and Emotion 31*, 1 (2007), 61–70.
10. DeBoer, J., Ho, A. D., Stump, G. S., and Breslow, L. Changing "course": Reconceptualizing educational variables for massive open online courses. *Educational Researcher 43*, 2 (2014), 74–84.
11. Donovan, J. J., and Radosevich, D. J. A meta-analytic review of the distribution of practice effect: Now you see it, now you don't. *Journal of Applied Psychology 84*, 5 (1999), 795–805.
12. Fidalgo-Blanco, Á., Sein-Echaluce, M. L., García-Peñalvo, F. J., and Escaño, J. E. Improving the mooc learning outcomes throughout informal learning activities. In *TEEM '14* (2014), 611–617.
13. Ford, K. J., Quinones, M. A., Sego, D. J., and Sorra, J. S. Factors affecting the opportunity to perform trained tasks on the job. *Personnel Psychology 45*, 3 (1992), 511–527.
14. Hill, T., Smith, N. D., and Mann, M. F. Role of efficacy expectations in predicting the decision to use advanced technologies: The case of computers. *Journal of Applied Psychology 72*, 2 (1987), 307–313.
15. Ho, A. D., Chuang, I., Reich, J., and Coleman *et al.*, C. A. Harvardx and mitx: Two years of open online courses fall 2012-summer 2014. *SSRN 2586847* (2015).
16. Holton III, E. F., Bates, R. A., and Ruona, W. E. Development of a generalized learning transfer system inventory. *Human resource development quarterly 11*, 4 (2000), 333–360.
17. Koedinger, K. R., Corbett, A. T., and Perfetti, C. The knowledge-learning-instruction framework: Bridging the science-practice chasm to enhance robust student learning. *Cognitive Science 36*, 5 (2012), 757–798.
18. Koller, D., Ng, A., Do, C., and Chen, Z. Retention and intention in massive open online courses. *Educause Review 48*, 3 (2013), 62–63.
19. Lim, D. H., and Johnson, S. D. Trainee perceptions of factors that influence learning transfer. *International journal of training and development 6* (2002), 36–48.
20. Lockhart, R. S., Lamon, M., and Gick, M. L. Conceptual transfer insimple insight problems. *Memory & Cognition 16*, 1 (1988), 36–44.
21. Miyamoto, Y. R., Coleman, C. A., Williams, J. J., Whitehill, J., Nesterko, S. O., and Reich, J. Beyond time-on-task: The relationship between spaced study and certification in moocs. *SSRN 2547799* (2015).
22. Pugh, K. J., and Bergin, D. A. Motivational influences on transfer. *Educational Psychologist 41*, 3 (2006), 147–160.
23. Rodriguez, O. The concept of openness behind c and x-moocs (massive open online courses). *Open Praxis 5*, 1 (2013), 67–73.
24. Saks, A. M., and Belcourt, M. An investigation of training activities and transfer of training in organizations. *Human resource management 45*, 4 (2006), 629–648.
25. Thorndike, E. L. *The psychology of learning*. Educational Psychology. Teachers College, Columbia University, 1913.
26. Veletsianos, G., Collier, A., and Schneider, E. Digging deeper into learners' experiences in moocs: Participation in social networks outside of moocs, notetaking and contexts surrounding content consumption. *British Journal of Educational Technology 46*, 3 (2015), 570–587.
27. Wang, Y., Paquette, L., and Baker, R. A longitudinal study on learner career advancement in moocs. *Journal of Learning Analytics 1*, 3 (2014), 203–206.
28. Willingham, D. T. What will improve a student's memory? *American Educator 32*, 4 (2008), 17–25.
29. Yelon, S. L., and Ford, K. J. Pursuing a multidimensional view of transfer. *Performance Improvement Quarterly 12*, 3 (1999), 58–78.