Optimizing an Educational Game Using UpGrade: Challenges and Opportunities

Nirmal Patel
Playpower Labs
Gujarat, India
nirmal@playpowerlabs.com

Dhrushit Raval
Playpower Labs
Gujarat, India
dhrushit@playpowerlabs.com

Vivek Fitkariwala
Playpower Labs
Gujarat, India
vivek@playpowerlabs.com

Derek Lomas
Delft University of Technology
Delft, The Netherlands
j.d.lomas@tudelft.nl

ABSTRACT

UpGrade is an open-source tool for A/B testing in educational software. We this study, we used UpGrade to run large scale online experiments in an educational game. Our experiments were aimed at increasing the student engagement. We experimented with various features of the game such as question difficulty, game narrative, feedback style, etc. to find out which conditions produced optimal outcomes. One after another, we conducted 3 different experiments. All of these experiments were created and monitored through UpGrade. We faced several issues during the implementation of these experiments within UpGrade. We discovered that buggy programming logic in the educational software can produce invalid experiment enrollments in UpGrade. We also found out that without tracking version of the educational software, it is possible to get noise in the experimental data. We present several recommendations to avoid these pitfalls. Results of our experiments are not discussed in this paper.

Author Keywords

Online Experiments; A/B Testing; Educational Games; Product Optimization

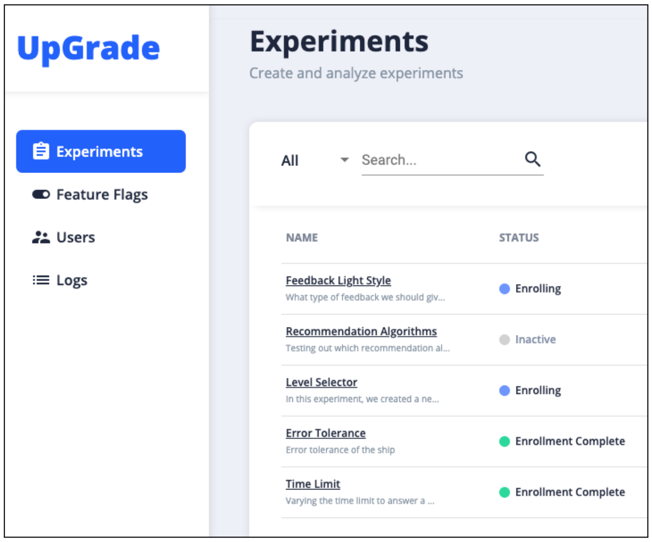
CCS Concepts

•**Human-centered computing** → **Human computer interaction (HCI);** *Haptic devices;* User studies; Please use the 2012 Classifiers and see this link to embed them in the text: https://dl.acm.org/ccs/ccs_flat.cfm

INTRODUCTION

Online experiments provide us an evidence-based framework to improve software applications [1]. We can run experiments in the web, desktop, and mobile applications and optimize various metrics such as user engagement, click through rate, and, in the context of the educational software, student learning. It is possible run large-scale online randomized control trials (RCTs) in online learning applications to make student learning more efficient and effective [4].

Running large-scale RCTs in online educational software is often complicated and expensive. Each experiment requires a dedicated software development effort at the start and end of the study. This makes it hard for the educational software companies to leverage the great value that A/B testing can provide to them. An open-source A/B testing tool called UpGrade has been designed to get around these issues and make A/B testing in education more affordable and accessible [3].

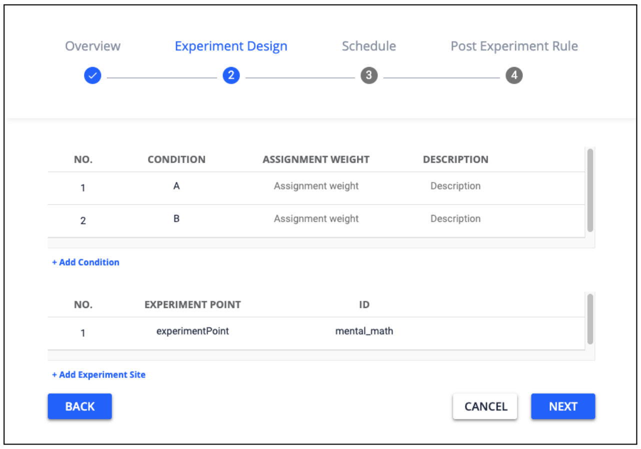


NAME	STATUS
Feedback Light Style What type of feedback we should giv...	● Enrolling
Recommendation Algorithms Testing out which recommendation al...	● Inactive
Level Selector In this experiment, we created a ne...	● Enrolling
Error Tolerance Error tolerance of the ship	● Enrollment Complete
Time Limit Varying the time limit to answer a ...	● Enrollment Complete

Figure 1. Experiments in the UpGrade web application.

UPGRADE

UpGrade is a web application that allows researchers to run experiments in the digital educational products. It allows users to create a number of simple experiments that can run in parallel (See Figure 1). Students in the digital platform get enrolled in the running experiments, and randomly get assigned to one of the experimental conditions. Once a student is assigned to a condition, the condition remains the same until the experiment ends. At the end of the experiment, based on a researcher selected rule, users either keeps getting the condition they were



NO.	CONDITION	ASSIGNMENT WEIGHT	DESCRIPTION
1	A	Assignment weight	Description
2	B	Assignment weight	Description

NO.	EXPERIMENT POINT	ID
1	experimentPoint	mental_math

Figure 2. Creating an experiment in the UpGrade web application.

assigned to or revert back to a 'default' non-experimental experience. UpGrade also implements a set of consistency rules for group random assignments. Using these rules, researchers can do random assignment at the classroom, school, or district level.

Experiment Points and Experiment Sites

To use UpGrade to conduct experiments within the educational software, we have to integrate the software with the UpGrade server. This integration takes place at different Experiment Points in the code. Experiment Points are actual places in the application code where the software sends the user information to the UpGrade server, and requests a condition assignment. For example, a line 280 in a file RenderLessonViewer.java can be an Experiment Point. At this line, the software will connect to the UpGrade server to find out which version of the lesson viewer should be shown to the user. UpGrade server will respond with a condition assignment. If UpGrade server returns a condition assignment that the Experiment Point does not understand, it will revert back to the 'default' non-experimental experience.

Experiment Sites are the places in the user interface where the experiments are going on. For example, a lesson in an online program can be an Experiment Site. At this Site, you can run an A/B test to compare the effectiveness of two different versions of the same lesson. In UpGrade, Experiment Sites are defined by a combination of an Experiment Point and a unique identifier. If there is only one unique identifier for the Experiment Site, then it can be defined only by the corresponding Experiment Point.

Experiment Creation and Deployment

UpGrade web application provides a user interface to create, modify, and monitor experiments. Creating an experiment in UpGrade requires the researcher to configure various properties of the experiments (See Figure 2). An experiment in UpGrade can have two or more conditions. Each condition is assigned a weight according to which it will be randomly sampled. A single experiment can run on one or more Experiment Sites in the digital platform. You have to define all of the Experiment Sites of the experiment by writing down their corresponding Experiment Points and unique identifiers. The experiments can be configured to start and end automatically or manually. Once an experiment starts, the Experiment Points listed in the experiment will start receiving condition assignments.

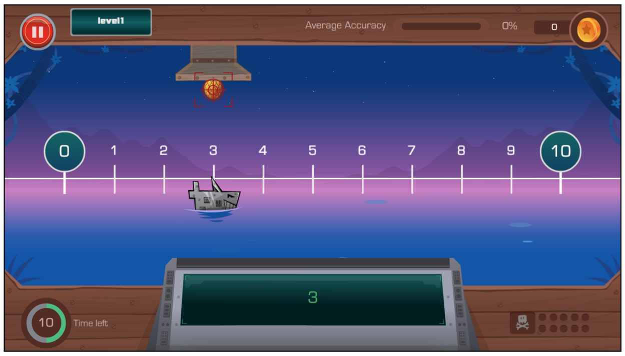


Figure 3. Numberline estimation task in the Battleship Numberline. The game tells the student that there is a robo-pirate at position 3 in the numberline, and the student to estimate 3 on the numberline.

Assignments and Enrollments

UpGrade assigns users to conditions in all of the running experiments. It is possible that a some users do not experience the assign condition and leave before they experience it. To distinguish condition assignment and actual condition experience, UpGrade requires the educational software to call a markExperimentPoint() function in its SDK when users actually experience their assigned conditions. This action enrolls the user in the corresponding experiment.

BATTLESHIP NUMBERLINE STUDY

We used UpGrade to run 3 different experiments in an online game called Battleship Numberline. The game is hosted on an educational games website BrainPOP, and students from different parts of the world can freely access the game. The game is designed to improve children's understanding of numberline estimation. When the game play starts, students are asked to estimate a number's position on the numberline. If they estimate it correctly, they get to explode a pirate ship. For every explosion, students get coins. When the game loads, students are presented a menu to choose the content area they want to practice. Once they pick one of them (students can pick between Fractions, Whole Numbers, Decimals, and Mixed Bag), the game keeps presenting randomly chosen questions from an item bank. At any point, students can choose to go back to the menu. This game has been subjected to multiple experiments in past [2]. Our experiments in this study aimed at improving student engagement in the game, which was defined as $log(seat\, time * number\, of\, questions)$.

UpGrade Experimentation Process

To run the experiments within Battleship Numberline using UpGrade, we started by setting up the Experiment Points in the code. Once they were ready, we were free to run experiments at those Experiment Points anytime without requiring

Experiment Point	Description
setTimeLimit	Set the time limit to answer a given question (accepted conditions: any positive integer)
setItemDifficulty	Set the difficulty of the item by varying the size of the target, bigger sized targets are easy to hit, smaller ones are more difficult (accepted conditions: any positive integer)
setMissionMode	Turn the 'mission mode' in the game on or off (accepted conditions: 'on' and 'off')
setFeedbackStyle	Change the style of feedback between positive only and positive and negative combined (accepted conditions: 'positive' and 'positivenegative')

Table 1. Experiment Points in the Battleship Numberline

any additional software development effort. The UpGrade architecture enabled us to rapidly run a series of experiments. We found this to be a great improvement over our previous method of running online RCTs, where we had to do software development and quality assurance testing for every single experiment.

To use UpGrade to conduct experiments in the Battleship Numberline, we implemented 4 different Experiment Points in the game (See Table 1). Each experiment point corresponded to exactly one Experiment Site in the user interface.

Experiment 1: Time Limit and Difficulty

The objective of this experiment was to find the optimal combination of time limit and question difficulty that produced the most amount of student engagement. This was a 4x4 multi-factor experiment. The time limit levels were 10, 20, 30, and 60 seconds. The question difficulty was controlled by changing the size (or effectively the length) of the target. The unit of the size was the percentage of the numberline. For this factor, the levels were 5, 10, 15, and 20 percent of the numberline (See Figure 4).

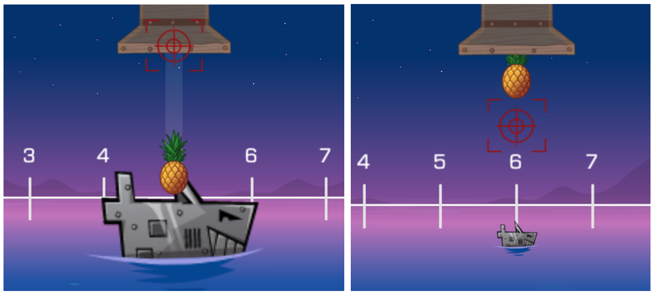
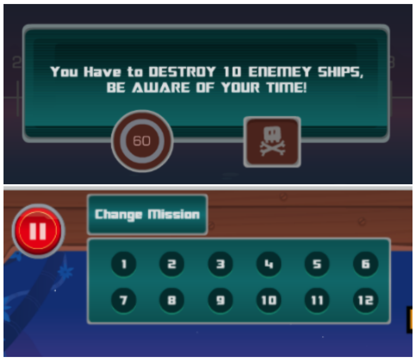


Figure 4. Experiment 1 where we varied time limit and question difficulty. The question in the left is less difficult versus the question in the right.

We used experiment points setTimeLimit and setQuestionDifficulty in this experiment. To implement this multi-factor experiment in UpGrade, we created two single-factor experiments and ran them in parallel.



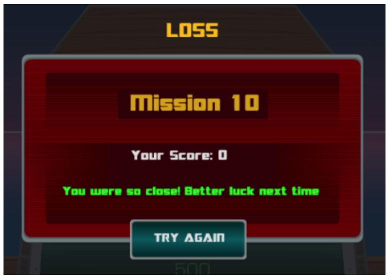
You Have to DESTROY 10 ENEMEY SHIPS,
BE AWARE OF YOUR TIME!

Figure 5. Experiment 2 where we introduced the challenge mode. In the challenge mode, we showed students a prompt telling them that they were on a mission (top). We also let them see how many missions were there in the game (bottom).

Experiment 2: Mission Mode

To increase the engagement of the students further after Experiment 1, we decided to introduce a 'mission mode' in each of the content areas of the game. In this experiment, we used the optimal parameters of the Experiment 1 that produced the highest student engagement with the game.

It has been observed in an older version of the Battleship Numberline that challenge can lead to more engagement [2]. To test that hypothesis in the new version of the game, we created an A/B test that assigned students to either a mission mode or a non-mission mode. In the mission mode, students were told that they were on a mission to complete a certain number of challenges (See Figure 5). In the non-mission mode, students were given questions indefinitely until they went back to the menu.



LOSS

Mission 10

Your Score: 0

You were so close! Better luck next time

Figure 6. Experiment 3 removed the above prompt from the Experiment 2. This prompt was shown to the students in the Experiment 2 when they incorrectly answered 4 questions in a row.

Experiment 3: Feedback Style

To further improve Experiment 2, we eliminated the negative feedback in the mission mode. In the Experiment 2, when users incorrectly answered 4 questions in a row, we showed them negative feedback (See Figure 6) and restarted their progress. In this experiment, we stopped showing users negative feedback and they did not lose their progress. Rest of the configuration of the game remained same as the Experiment 2.

Data Collection

UpGrade allowed us to export the condition assignment data from the web portal. This data contained a mapping between student IDs and their experimental conditions. The students' activity data were exported from our game's database. Afterward, both of these datasets were combined for the experimental data analysis.

RESULTS

Since our game was hosted on BrainPOP, which is quite popular among K-12 classrooms in the US, we saw a fair amount of usage of our game. The high usage of our game combined with the flexibility of UpGrade allowed us to quickly run and analyze experiments one after another. In the Experiment 1, we found out that the time limit of 20 seconds and the question difficulty of 10 percent led to the most engagement. In Experiment 2, we discovered that the Mission Mode did not produce more engagement.

Experiment	N	Start Date
1	83070	5 May 2020
2	23135	17 June 2020
3	1366	16 July 2020

The experiment enrollments decreased over time because of the end of the school year. We hope to present the results of all of the experiments in a later article.

IMPLEMENTATION CHALLENGES

We faced several novel implementation challenges when running the described experiments. We realized that if researchers are unaware of these potential pitfalls, they can end up reporting incorrect findings. All of these pitfalls can be avoided by improving the educational software and its integration with the UpGrade server.

Invalid Enrollments

In our implementation, Experiment 2 received several thousand invalid enrollments because of a buggy game program. The invalid enrollments occurred because students enrolled in the Experiment 1 were mistakenly enrolled again in the Experiment 2. The game programmer had written down a logic that enrolled the game users in all of the experiments running on the UpGrade server. When we created Experiment 2 in UpGrade, we immediately started getting enrollments in it from the users who were getting enrolled in the Experiment 1. To resolve this issue, we stopped bulk enrolling users in all of the experiments at the same time. Instead, we started enrolling the users in the experiment when they actually experienced a condition from that experiment.

Application Version Tracking

During Experiment 1, we made several small fixes and changes in the game. In the data logs of our game, we did not track these changes using SemVer or any other versioning scheme. Due to this, in the experimental data of Experiment 1, we did not know which users saw which variation of the Experiment 1. After we released the fixes, some of the users kept using the cached version of our game. This caused a data mix-up that was impossible to resolve without a version tracking scheme. To fix this issue, we started tracking the version of our game using the SemVer versioning scheme. This allowed us to see that our users were playing the cached version of our game for up to 4 days after the new version was released.

Testing Data

Although UpGrade provides a way to manually assign users to conditions for quality assurance testing purpose, it is very easy to get test or demo users enrolled in a live experiment. For example, right now, if anyone opens Battleship Numberline on the BrainPOP platform for a demo or testing purpose, they cannot avoid getting enrolled in an experiment. To avoid this test data mix-up in web based learning software, we recommend using a URL parameter that can indicate that the current use of the application is for a testing or a demo purpose.

UpGrade Implementation Recommendations

Based on the implementation challenges that we faced, we provide several recommendations to that will help ensure the validity of the UpGrade experiments:

1. Only run one experiment at a time. Avoid running multiple experiments together, unless they are part of a single multi-factor experiment.
2. In the educational software code, enroll users in the experiments when they actually experience the experimental conditions, not before.
3. Track versions of your educational software in the data logs, and keep them aligned with your experiments.
4. During an experiment, try to ensure that a newly released change in the software reaches to all of your users. If some of the users still keep using the older version, use version tracking to subset the experimental data.
5. For web based application, provide a URL parameter that makes users experience the experimental conditions, but does not enroll them in the experiment.

OPPORTUNITIES

UpGrade provides a lot of opportunities to researchers for rapidly running educational experiments at scale. Once the learning software is equipped with Experiment Points, researchers are free to run experiments without requiring any additional software development effort. This makes experimentation more efficient and affordable. Since UpGrade is open-source, it can be enhanced to have more features such as multi-factor experiments, within subject experiments, Multi-Armed Bandit based optimization, pre-registration of the studies, etc. Overall, we find that UpGrade brings affordability and efficiency in the educational experimentation, and so, it is bound to increase the speed at which we can do learning science and impact student learning at scale.

CONCLUSION

In this paper, we showed how we ran experiments in an online educational game using UpGrade, an open-source tool for A/B testing. We described several novel implementation challenges that we faced, and presented a list of recommendations for researchers and developers to avoid the these challenges. We hope that our insights help researchers ensure validity of their experimental analyses.

REFERENCES

[1] Ron Kohavi, Roger Longbotham, Dan Sommerfield, and Randal M Henne. 2009. Controlled experiments on the web: survey and practical guide. *Data mining and knowledge discovery* 18, 1 (2009), 140–181.

[2] J Derek Lomas, Kenneth Koedinger, Nirmal Patel, Sharan Shodhan, Nikhil Poonwala, and Jodi L Forlizzi. 2017. Is difficulty overrated? The effects of choice, novelty and suspense on intrinsic motivation in educational games. In *Proceedings of the 2017 CHI conference on human factors in computing systems*. 1028–1039.

[3] Steve Ritter, April Murphy, Stephen Fancsali, Derek Lomas, Vivek Fitkariwala, and Nirmal Patel. 2020. UpGrade: An open source tool to support A/B testing in educational software. *L@S Workshop on A/B Testing at Scale* (2020).

[4] John C Stamper, Derek Lomas, Dixie Ching, Steve Ritter, Kenneth R Koedinger, and Jonathan Steinhart. 2012. The Rise of the Super Experiment. *International Educational Data Mining Society* (2012).