Shih, J. L. et al. (Eds.) (2019). Proceedings of the 27th International Conference on Computers in Education. Taiwan: Asia-Pacific Society for Computers in Education

Leveraging LSTM in the fine-grained analysis of the Incubation Effect in Physics Playground

May Marie P. TALANDRON-FELIPE[ab*] & Ma. Mercedes T. RODRIGO[a*]
[a]*Ateneo De Manila University, Philippines*
[b]*Central Mindanao University, Philippines*
*maymarie.talandron@upou.edu.ph, mrodrigo@ateneo.edu

Abstract: Incubation Effect (IE) refers to the phenomenon where one gets stuck in a problem-solving activity, decides to take a break, and afterwards revisits the unsolved problem and eventually solves it. While studies on IE were all limited to traditional classroom activities, this research aimed to continue the study of IE in the context of a computer-based learning environment and find features that would predict the incidence of revisiting an unsolved problem and its positive outcome. A prior IE model was developed using a logistic regression but the hand-crafted features used were from aggregated data and do not reflect specific characteristics of students' actions. Further analysis was conducted in this study and used a deep learning technique which significantly improved the performance of the IE model. In order to interpret the learned features of the neural network, a combination of dimension reduction, visualization technique, and clustering were used. It was found that the coarse-grained features are consistent with the fine-grained features but action level features were also discovered which provided more evidence that there was an improvement on how students tried to solve the problem after incubation.

Keywords: Incubation Effect, Physics Playground, LSTM

1. Introduction

Taking a break from a series of failed attempts to solve a problem may facilitate the solution process as shown in prior work (Fulgosi & Guilford, 1968; Gilhooly, Georgiou, & Devery, 2013; Penaloza & Calvillo, 2012; Sio & Ormerod, 2015). This momentary break is known by the name incubation (Sio & Ormerod, 2015). During some incubation periods, an internal mental process associates new information with past information to generate solution ideas (Medd & Houtz, 2002). In the context of education, students who get stuck in a problem-solving activity may temporarily engage in another task, after which, they return to the original problem and find a solution. When the student solves the problem after incubation, the phenomenon and its positive result is called the Incubation Effect (IE).

IE is divided into three phases (Gilhooly et al., 2013): (a) pre-incubation phase, (b) incubation phase, and (c) post-incubation phase. The pre-incubation phase is when a student attempts to solve a problem and gets stuck. When the student decides to take a break from the problem-solving task and engages in another task, it signals the beginning of the incubation phase. The post-incubation phase happens when the student goes back to the original problem and tries to solve it again. The benefits of incubation prompted researchers to incorporate breaks into educational activities which were shown to have positive results (Lynch & Swink, 1967; Medd & Houtz, 2002; Rae, 1997; Webster, Campbell, & Jane, 2006). Prior studies (Ellwood, Pallier, Snyder, & Gallate, 2009; Fulgosi & Guilford, 1968; Gilhooly et al., 2013; Penaloza & Calvillo, 2012; Sio & Ormerod, 2015) investigated specific factors that could lead to successful incubation in the context of classroom tasks and suggested that engaging in a different activity may produce a better outcome. On the other hand, (Penney, Godsell, Scott, & Balsom, 2004) claimed that engaging in a task with similar nature would promote priming which allows students to realize the correct solution to the problem but (Segal, 2004) said that the task during incubation has no effect on its outcome.

The incidence of incubation effect has also been investigated (Martinez et al., 2016; Talandron, Rodrigo, & Beck, 2017; (Talandron & Rodrigo, 2018) in the context of a computer-based environment called Physics Playground (PP) which is a two-dimensional game that is designed for high school students to better understand concepts in Physics. (Martinez et al., 2016) found evidence that taking a break helped some students to solve a problem in which they were previously stuck. For example, if the student cannot solve level 3 in playground 3, the student can leave it and try to solve other levels then just go back to it again later. To further explore IE on PP, Talandron, Rodrigo, and Beck (2017) attempted to model IE and examined possible factors that predict the successful outcome of incubation. However, the study was limited to hand-crafted features from aggregated data which means that individual attempts comprising the 3 phases of IE were not analyzed in a fine-grained level and the actual activity during incubation were not taken into consideration. Thus, further analysis is needed to look at the large amount of unused data from the previous studies and come up with better features and model.

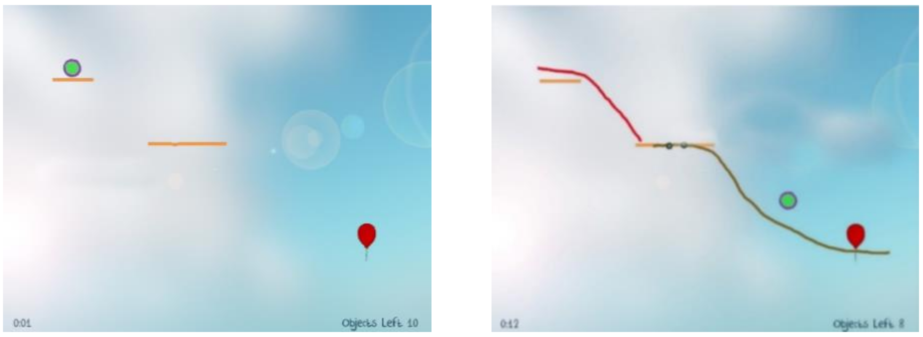
The main objective of the study is to conduct a fine-grained level modeling of the incubation effect among students playing Physics Playground and further understand the factors that predict IE. Student interaction logs contain a rich amount of data that can be utilized to extract information on student actions and behaviors in each attempt. Specifically, this study would like to answer the following:

(RQ1) What fine-grained features predict incubation effect? What are the features that more likely give a positive result during the post-incubation phase? What's the difference between the actions of the students to solve level X during pre-incubation and during post-incubation?

(RQ2) How will the extracted features perform against the hand-crafted features (Talandron et al., 2017) in predicting incubation effect? How does the model using hand-crafted coarse-grained features perform versus the model with fine-grained learned features?

2. Physics Playground

Physics Playground (PP), formerly known as Newton's Playground, is a two-dimensional computer-based game designed for students in the secondary level to better understand the concepts of qualitative Physics. The game simulates how the physical objects operate in relation to Newton's laws of motion: balance, mass, conservation and transfer of momentum, gravity, and potential and kinetic energy (Shute & Ventura, 2013). The game has different problems with varying levels of difficulty and solutions. The main objective of each problem is to guide a green ball to a red balloon. To solve each level, the players must draw objects (i.e., ramp, lever, pendulum, springboard) using the computer mouse and these objects become part of the game environment. Figure 1 (a) shows an example level of PP which requires a ramp to lead the ball to the balloon. All objects drawn obey the basic rules of physics relating to gravity and Newton's three laws of motion (Shute & Ventura, 2013). Once the player draws a ramp, the ball will then follow its path until it reaches the red balloon as shown in Figure 1 (b).



(a) A level that requires a ramp (b) Sample solution to a ramp problem

Figure 1. A sample level in Physics Playground.

When the student solves a level, he/she receives either a gold or silver badge. A badge is awarded if the student solves the level – a gold badge if the problem was solved using at or below a par number of objects determined by the game designers; otherwise, a silver badge is given.

3. Methods

3.1 Data

The analyses for this study used a dataset collected from a total of 176 high school students in the Philippines: 29 from a public junior high school in Baguio City (School 1); 31 from a private university also in Baguio City (School 2); 56 from a private university in Cebu (School 3); and 60 from a private university in Davao City (School 4). These students were considered average in terms of their academic performance.

The students were given an orientation to introduce them to Physics Playground and to explain the game mechanics. Before playing, they were asked to answer a pre-test which was comprised of 16 multiple-choice type questions about simple machines and laws of Physics in relation to the learning objectives of PP. Then the students were given about 2 hours to play Physics Playground where their interactions with the game were automatically recorded into a log file. They were allowed to choose the problem they would like to solve, they could leave the problem, and return to it at a later time. After the session, the students answered a post-test which was also based on the topics covered in PP.

While playing PP, student's interactions with the game were automatically recorded along with each action's time stamp. The actions recorded were divided into 4 categories: Menu Events, Level Events, Play Events, and Agent Events. Menu Events refer to interactions when the player is in the main menu of the game while Level Events are actions related to each individual level within a playground. Play Events are the player's interactions within the PP environment once the player started to play a specific level. Agent Events refer to the interactions of and with the objects or simple machines drawn by the player to solve the level. These include the level, start time, end time, objects drawn, badge, etc. where we can derive other information as in prior work (Martinez et al., 2016; Palaoag, Rodrigo, & Andres, 2015; Palaoag, Rodrigo, Andres, Andres, & Beck, 2016; Talandron et al., 2017) such as attempt duration, number of restarts and revisits, sequence of levels, number of badges earned.

3.2 Fine-Grained vs. Coarse-Grained Features

To distinguish coarse-grained (Talandron et al., 2017) and fine-grained analysis, operationalizing IE in the PP interaction logs were done in a hierarchical manner. Figure 2 shows the levels of analysis as well as the relationship of the entities. The coarse-grained level analysis involved features from levels 1 to 3 of the diagram and the fine-grained analysis will include features on levels 4 and 5.

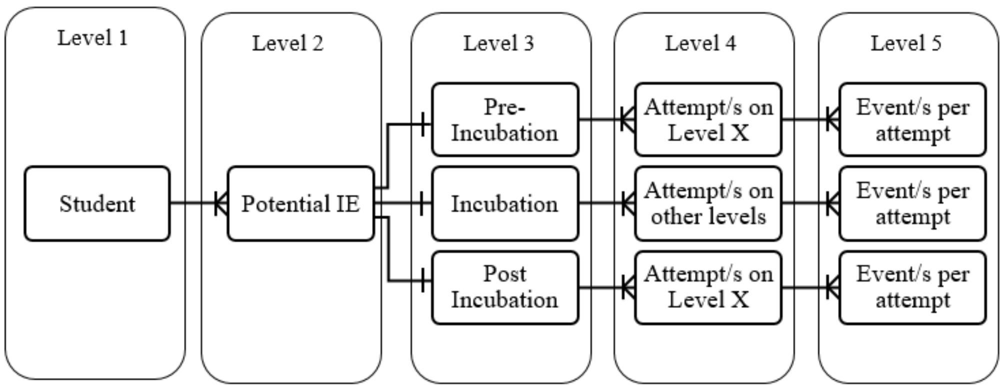


Figure 2. The levels of analysis.

The data was filtered to only include the interaction logs of students who exhibited potential IEs. Unnecessary columns were then removed and only included the problem ID, series of actions taken to solve the problem, and the result which indicates whether the student solved the problem or not. The canonical solution to each problem which was the basis of the problem type was also integrated into the logs.

3.3 Modeling

To prepare the data for modeling using LSTM, it is essential to structure the data for the specification of the timestep and batch size such that the timestep corresponds to the number of actions to solve a problem and the batch size as the number of problems or attempts per student. Another step was the transformation of the data type from string to numeric. String data must be encoded as numbers to be used as input or output for machine learning and deep learning models and the *Scikit-learn* library has provided the tool to do this. *Sklearn's LabelEncoder* module finds all classes and assigns each a numeric id starting from 0. For the output labels, *np.utils.to_categorical* was used to convert array of labeled data (from 0 to nb_classes-1) to one-hot vector.

The model using LSTM was developed using Keras, a high-level neural networks API on top of TensorFlow with Python as the underlying programming language. To realize the objectives, this study focused on the following training tasks:

- Given a sequence of attempts for a series of problems, classify each attempt based on the following:
    - son – solved new (successfully solved the problem at first attempt)
    - unn – unsolved new (failed to solve the problem at first attempt)
    - sops – solved a problem that has already been previously solved
    - unps – failed to solve a problem that has already been previously solved
    - sopu-nb – no incubation, replayed and solved a previously unsolved problem
    - unpu-nb – no incubation, replayed and failed to solve a previously unsolved problem
    - **sopu – revisited and solved a previously unsolved problem (IE-True)**
    - **unpu – revisited and failed to solve a previously unsolved problem (IE-False)**

The "sopu" corresponds to IE-True while "unpu" corresponds to IE-False. The analyses were focused on these 2 classes. The term "replay" means the player re-played the same level consecutively without interval while "revisit" means a "break" or interval occurred before the same level is played again.

The input features include the time of the action, the problem ID, and the series of actions in the attempts to solve the problem. The output label was the result of each attempt which was coded based on the 8 classes described in this section. For this task, a one-layer deep LSTM was used. To predict the output label, the hidden state at the last timestep was passed through a fully connected layer and a subsequent *softmax* layer. The batch size used was the number of attempts per student which was 91 and timestep was set to 100 so that the network would consider all the actions for each attempt as it backpropagates when calculating gradients for weight updates. The final number of epochs was set to 200. The values of other hyperparameters followed the configuration described in section 3.3. A student level cross-validation was used which was to ensure that a student's data was only either on the training set or the testing set.

Another issue that had to be addressed was class imbalance. Majority (71%) were first attempts on a new problem and the remining 29% were divided into 3 types: 13% replay previously solved problems, 9% revisit previously unsolved problem (potential IE), 7% replay unsolved problems (no incubation). This was addressed using the *sklearn.utils. class_weight. compute_class_weight* from the *Scikit-learn* library which computes for the appropriate weight based on the given training data. The computed values were stored in a dictionary which was then implemented during training. A student-level cross validation was done to ensure that each student's data was either on the training set or the testing set.

3.4 Analysis of Features

The application of t-SNE (Maaten & Hinton, 2008) and X-means clustering algorithm as described in Wang et al. (2017) was done to map the input samples with the learned features. To answer the research questions previously stated, the data were analyzed as follows:

RQ1: What fine-grained features predict incubation effect? The learned representations are expected to be features that predict performance. The features derived from the input data was visualized using t-SNE as well as the prediction results. Then, X-means algorithm was used to cluster the data points in order to identify distinct groups. Quantitative analysis was conducted on each cluster based on the features that could be derived from the LSTM input data to find significant differences or effect of these features on each cluster.

RQ2: How will the extracted features perform against the hand-crafted features in predicting incubation effect? The performance of the model and the coarse-grained features presented in Talandron, Rodrigo, and Beck (2017) were compared to the fine-grained model and the extracted fine-grained features.

4. Results and Discussion

4.1 The Fine-Grained Model

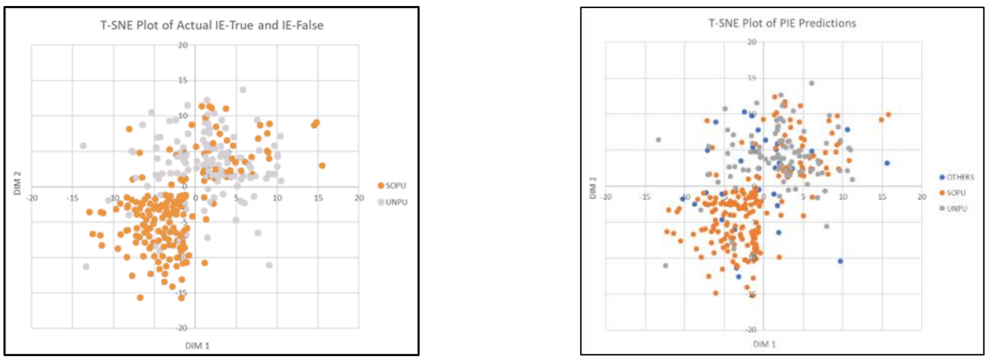
LSTM was used to model IE in a fine-grained level as described in section 3.3. A zero-padding technique was applied in order to achieve a uniform number of attempts per student which was used as the batch size during training and the same technique for the number of actions per attempt which was used the number of timesteps. The input vector has a total of 773,500 rows (85 students with 91 attempts each and each attempt with 100 actions). Model performance was measured based on the confusion matrix which is shown in Table 2 (recall = 91.62%, precision = 82.55%, f-score = 86.84%, kappa = 0.821).

Table 1

Fine-Grained IE model confusion matrix

Actual	Predicted		
	IE-True *(sopu)*	IE-False *(unpu)*	Others
IE-True *(sopu)*	175	2	14
IE-False *(unpu)*	15	95	22
Others	22	36	3251

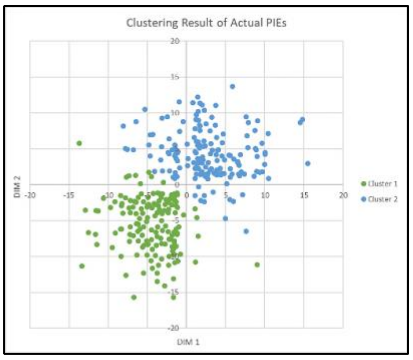
Using t-SNE, both the actual and the predicted IE-True and IE-False instances were visualized on a two-dimensional plot. Several runs of t-SNE were conducted with varying values for perplexity and in order to decide on the most appropriate value, the one with the highest t-SNE nearest neighbor accuracy was selected. In this result, the perplexity of 20 yield the highest t-SNE nearest neighbor accuracy of 81%. The input data used in t-SNE includes all the features that could be derived from the LSTM input data and that could have been learned by the neural net such as time, problem, actions taken to solve the problem, duration of incubation, duration for each attempt, problem difficulty, productivity, and problem type. The t-SNE output reduced these features into two dimensions which were then the basis for the t-SNE plot. Figure 3 shows the t-SNE plot of IE-True (*sopu*) and IE-False (*unpu*) based on the actual (a) and prediction results (b).



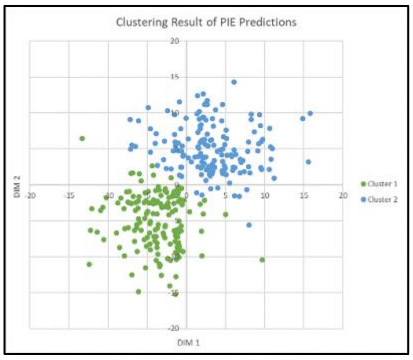
(a) actual IE-True (*sopu*) and IE-False (*unpu*) (b) predicted IE-True (*sopu*) and IE-False (*unpu*)

Figure 3. The t-SNE plot of IE-True (*sopu*) and IE-False (*unpu*).

Since the model's recall was at 91.62%, it was expected that their t-SNE plot should look similar. More importantly, both plots show two apparent groups, one on the third quadrant of the plot which was composed of almost all IE-True instances, and one on the first quadrant which includes majority of IE-False but was mixed with some IE-True. In order to get the distinct clusters, we applied x-means clustering on the t-SNE results and then plot the clustering results on the t-SNE data. This was done on both the actual and predicted IE-True and IE-False as shown in Figure 4(a) and Figure 4(b), respectively.



(a) Clustering result of actual Potential IEs



(b) Clustering result of the predictions

Figure 4. The clustering results of both actual IEs and predicted IEs

Based on the clusters of prediction results and referencing Figure 3(b) on Figure 4(b), Cluster 1 is predominantly composed of SOPU predictions (IE-True). A quantitative analysis was done for all the features derived from the LSTM input data in order to extract distinct features for the clusters. As previously mentioned, these features include the time of the revisit, incubation duration, problem difficulty, student's productivity, the problem type, and the different actions done to solve the problem.

4.1.1 Time of Revisit and Problem Difficulty

It was found that more potential IEs (revisit) occurred during the last 30 minutes of the session. However, when investigated in terms of how productive revisits were at each 30-min interval, it was found that more revisits at the later time resulted to IE-False. A t-test was conducted to compare the difference between the time of revisit in cluster 1 vs those in cluster 2. There was a significant effect of the time of revisit on the clusters at the $p<0.05$ level [F (1, 321) = 7.01, p = 0.008]. The IE incidence for each cluster in 30-minute bins is also shown in Figure 5. Since instances in cluster 1 were mostly successful IEs, it can be inferred that potential IEs in the early part of a time-limited session are more likely to be beneficial.

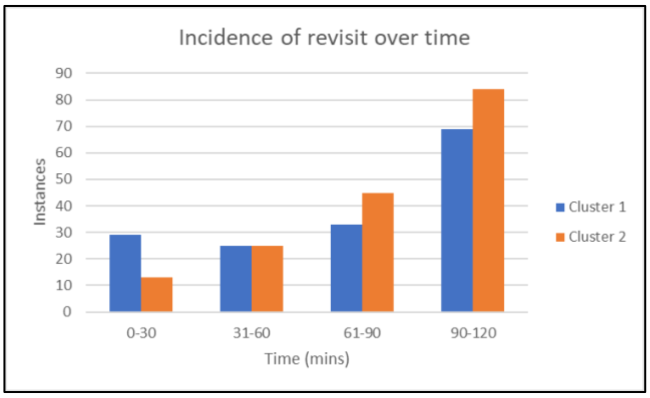


Figure 5. Predicted revisit instances at each time interval from clusters 1 and 2

Also, from the coarse-grained IE model (Talandron et al., 2017), problem difficulty was a significant feature such that revisiting a problem with lower difficulty rate more likely results to IE-True. This was also analyzed in a fine-grained level and the difference was significant on cluster 1 (mean = 39.36%, sd = 16.64%) and cluster 2 (mean = 45.39%, sd = 17.33%) at the p<0.05 level [F (1, 321) = 10.06, p = 0.002].

4.1.2 Duration of Incubation

The incubation duration between clusters were compared and a significant difference was found between cluster 1 (mean = 10.95, sd = 17.99) and cluster 2 (mean = 16.06, sd = 20.81) at the p<0.05 level [F (1, 321) = 5.52, p = 0.02]. This finding is also consistent with the coarse-grained analysis where it was found that a lengthy incubation could lead to IE-False (Talandron et al., 2017) and specifically, incubation duration that was more than 40 minutes resulted to IE-False (Talandron and Rodrigo, 2018).

4.1.3 Productivity Rate

Another notable feature in predicting IE-True in the coarse-grained analysis (Talandron et al., 2017) was the student's productivity which means how well the student was performing at the time of revisit. Similarly, a comparison was made in the fine-grained level analysis to see how this feature affected the clusters. This was computed as the number of problems solved over all attempts made at the time of revisit. The difference was significant between cluster 1 (mean = 64.48%, sd = 17.89%) and cluster 2 (mean = 56.97%, sd = 19.75%) at the p<0.05 level [F (1, 321) = 12.30, p < 0.001]. This means that having been productive by the time of revisit would more likely lead to a positive revisit outcome. Figure 6 shows the potential IE instances in clusters 1 and 2 and the student's productivity rates in 25% bins.

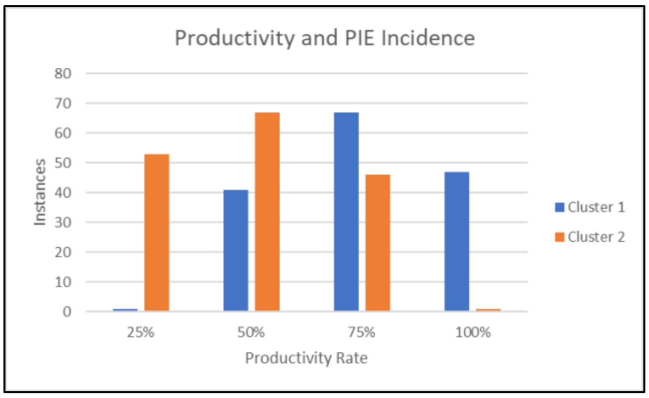


Figure 6. Productivity of IE instances from clusters 1 and 2

4.1.4 Problem Type

The problem type is a binary feature which indicates whether the attempt preceding the revisit had a similar solution to the problem during the revisit. There were 208 potential IEs preceded by an attempt on a problem with a similar solution and 125 (60.10%) resulted to IE-True. In terms of its effect on clusters 1 and 2, the Chi-Square test of independence was conducted and found a significant effect, c^2 (1, N=323) =39.55, p<0.001, between the clusters and the problem type. This means, being preceded by a problem with a similar solution was considered as a significant factor in predicting IE-True.

4.1.5 Actions

There's a total of 22 events or actions from the logs and some of these actions can be found on other games such as 'click', 'pause', 'hover/preview tutorial', 'watch tutorial', 'erase'. Other actions are very specific to Physics Playground such as 'collision', 'diver', 'draw pin', 'lever', 'pendulum strike', 'springboard', 'ramp', 'stacking', 'stacking warning', 'draw freeform'.

The analysis for these actions was divided into three: 1) an analysis on the common game actions between cluster 1 and cluster 2; 2) an analysis on the common game actions between the pre-incubation period and post-incubation period; and 3) an analysis on solution specific actions during the pre-incubation period and post-incubation period to explore whether there has been a change on the approach to solve the problem. For the common game actions, a comparison on the frequency of these actions between the clusters were analyzed and a significant effect, at p<0.05 level, was found on the incidence of pause, hover tutorial, and erase between cluster 1 and cluster 2 as shown in Table 3.

Table 2

Comparison of common game actions between clusters 1 and 2

Actions	Cluster 1 (mean)	Cluster 2 (mean)	F	p
Erase	10.89	6.55	7.77	<0.01
Hover Tutorial	0.41	0.11	4.68	<0.05
Pause	0.39	1.21	143.42	<0.001

Based on this, we can infer that higher incidence of erase in cluster 1 means better awareness when incorrect actions were made, lower incidence of pause can be an indication that they were more confident of what they're doing and higher incidence of hover tutorial means they are making sure that they are drawing the object correctly. Further, a comparison of these actions between the pre-incubation and post-incubation period was conducted to see if there was a difference before and after incubation. The difference was significant for pause between cluster 1 pre-incubation (mean = 0.08) and cluster 1 post-incubation (mean = 0.03) at the p<0.05 level [F (1, 381) = 109.19, p < 0.001]. It was also significant for erase, cluster 1 pre-incubation (mean = 0.03) and cluster 1 post-incubation (mean=0.05) at the p<0.05 level [F (1, 381) = 6.72, p = 0.009]. Third, solution specific actions were compared between pre-incubation and post-incubation and a significant difference, at the p<0.05 level, was found for ramp and springboard as shown in Table 4.

Table 3

Comparison of solution-specific actions between clusters 1 and 2

Actions	Pre-incubation (mean)	Post-incubation (mean)	F	p
Ramp	0.02	0.07	14.72	<0.001
Lever	0.015	0.017	0.085	0.77
Springboard	0.007	0.025	7.87	0.005
Pendulum	0.010	0.014	0.88	0.35

With the significant difference on the frequency of these actions during the pre-incubation period and post-incubation period, it can be inferred that incubation has made an impact on how the students try to solve the problem during the revisit.

4.2 Coarse-Grained Model vs Fine-Grained Model

The findings were compared to establish consistency of results as well as the necessity of fine-grained analysis to improve model performance. All the significant features from the coarse-grained analysis(Talandron et al., 2017) are consistent with the extracted features in the fine-grained analysis which are the time of revisit, incubation duration, productivity before revisit, and problem difficulty. Aside from these, new fine-grained features were discovered during the analysis which are the problem type, common game actions and solution-specific actions which provided evidence that incubation improved students' actions in their attempt to solve a previously unsolved problem after incubation.

The fine-grained model performed better in all metrics used (precision, recall, f-score, kappa) compared to the coarse-grained model as shown in table 5. There was a notable increase of 31.82% in precision and 5.97 in kappa.

Table 4

Comparison of model performance

IE Model	precision	recall	f-score	kappa
Coarse-grained model (Talandron et al., 2017)	50.73%	89.61%	64.78%	0.224
Fine-grained model	82.55%	91.62%	86.84%	0.821

This improvement could be attributed to the problem-level or attempt-level data that the fine-grained level analysis was able to use. The difference between the student's actions before and after incubation was a significant factor on the model's prediction performance.

5. Conclusion, Limitations, and Future Work

This study investigated and modeled the incubation effect phenomenon among students playing an educational game in a fine-grained level. The initial investigation found that students' IE success rates matched their non-IE success rates, implying that IEs may indeed benefit students who are stuck (Martinez et al., 2016). This study continued the work and answered the following:

RQ 1: What fine-grained features predict incubation effect?

The significant features found were time of revisit (low), duration of incubation (low), problem difficulty (low), student's productivity at the time of revisit (high), similarity with the preceding problem. In terms of game actions, the following were discovered: erase (high), pause (low), hover tutorial (high). For problem specific actions, improvement in the student's drawing of ramp and springboard were observed.

RQ 2: How will the extracted features perform against the hand-crafted features in predicting incubation effect?

The coarse-grained features (Talandron et al., 2017) that were manually engineered were consistent with the extracted features but action-level features were also discovered providing more evidence on the improvement of students' actions to solve the problem after incubation. Moreover, a notable increase of 31.82% in precision (82.55% vs 50.73% of the coarse-grained model) and 5.97 in kappa (0.821 vs 0.224 of the coarse-grained model) were achieved when compared to the previous model which used aggregated data.

This study contributes to researches using computer-based learning environments in studying phenomenon of a similar construct with IE since interaction logs of test subjects can be recorded automatically and hence more accurately. Second, findings quantified the pedagogical practice where teachers instruct students who are stuck at a problem to skip it and go back to it at a later time. It showed that incubation can be an effective technique in solving problems where activities performed during the break are similar or related tasks and the features extracted from this study could be translated to design features that could be used in other educational games to improve students' performance. Third, this study contributed to the growing applications of deep learning on educational data specifically by analyzing interaction logs from a computer-based learning environment to improve predictive models of behaviors and phenomena in the context of education. It has also shown that combining deep learning with other machine learning techniques such as dimensionality reduction, visualization, and clustering, can pave way to understand the learned features of a neural network.

Based on the limitations of the data collection method which used stealth assessment, another experiment can be recommended which would involve a control group and an experimental group where one group is instructed and allowed to incubate and other is not to further study the benefits of incubation versus no incubation in an experimental setup. Second, the features that predict IE could be translated into a game design such as rules or game mechanics of an educational application or software to further study the integration of IE as a pedagogical strategy.

Acknowledgements

The authors would like to thank Mr. Joshua Martinez and Mr. Jun Rangie Obispo for their valuable contribution in the initial investigation of incubation effect in Physics Playground which paved way for this study and Prof. Valerie Shute for her assistance on questions about Physics Playground specifically about the canonical solutions. We also would like to thank the Ateneo Laboratory for the Learning Sciences and the Ateneo de Manila University Department of Information Systems and Computer Science. This study was made possible through a grant from the Philippines' Department of Science and Technology Philippine Council for Industry, Energy and Emerging Technology Research and Development entitled "Stealth assessment of student conscientiousness, cognitive-affective states, and learning using an educational game for Physics."

References

Ellwood, S., Pallier, G., Snyder, A., & Gallate, J. (2009). The Incubation Effect: Hatching a Solution? *Creativity Research Journal*, *29*(1), 6–14. https://doi.org/10.1080/10400410802633368

Fulgosi, A., & Guilford, J. P. (1968). Short-term Incubation in Divergent Production. *American Journal of Psychology*, *81*(2), 241–246.

Gilhooly, Georgiou, G., & Devery, U. (2013). Incubation and creativity: Do something different. *Thinking & Reasoning*, *19*(2), 137–149.

Lynch, M. D., & Swink, E. (1967). Some effects of priming, incubation and creative aptitude on journalism performance. *Journal of Communication*, *17*(4), 372–382.

Maaten, L. van der, & Hinton, G. (2008). Visualizing data using t-SNE. *Journal of Machine Learning Research*, *9*(Nov), 2579–2605.

Martinez, J. C., Obispo, J. R. C., Talandron, M. M. P., & Rodrigo, M. M. T. (2016). Investigating the Incubation Effect among Students playing Physics Playground. *24th International Conference on Computers in Education*, 1–8. Mumbai, India: Asia-Pacific Society for Computers in Education.

Medd, E., & Houtz, J. C. (2002). The effects of facilitated incubation on fourth graders' creative writing. *Educational Research Quarterly*, *26*(2), 13.

Palaoag, T. D., Rodrigo, M. M. T., & Andres, J. M. L. (2015). *An Exploratory Study of Persistence Markers within a Game-based Learning Environment*. Presented at the 23rd International Conference for Computers in Education, China.

Palaoag, T. D., Rodrigo, M. M. T., Andres, J. M. L., Andres, J. M. A. L., & Beck, J. E. (2016). Wheel-Spinning in a Game-Based Learning Environment for Physics. *International Conference on Intelligent Tutoring Systems*, 234–239. Springer.

Penaloza, A. A., & Calvillo, D. P. (2012). Incubation provides relief from artificial fixation in problem solving. *Creativity Research Journal*, *24*(4), 338–344.

Penney, C. G., Godsell, A., Scott, A., & Balsom, R. (2004). Problem variables that promote incubation effects. *The Journal of Creative Behavior*, *38*(1), 35–55.

Rae, C. M. (1997). The creative power of doing nothing. *Writer*, *110*(7), 13–15.

Segal, E. (2004). Incubation in Insight Problem Solving. *Creativity Research Journal*, *16*(1), 141–148.

Shute, V., & Ventura, M. (2013). *Stealth assessment: Measuring and supporting learning in video games*. MIT Press.

Sio, U. N., & Ormerod, T. (2015). Incubation and Cueing Effects in Problem-Solving: Set Aside the Difficult Problems but Focus on the Easy Ones. *Thinking and Reasoning*, *21*(1), 113–129.

Talandron, M. M. P., & Rodrigo, M. M. T. (2018). Let's Take a Break: Analysis of the Incubation Effect Among Students Using a Learning Game for Physics. *Proceedings of the 26th International Conference on Computers in Education*. Presented at the 26th International Conference on Computers in Education, Philippines.

Talandron, M. M. P., Rodrigo, M. M. T., & Beck, J. E. (2017). Modeling the Incubation Effect Among Students Playing an Educational Game for Physics. *Artificial Intelligence in Education: 18th International Conference on Artificial Intelligence in Education Proceedings*, 371–380. https://doi.org/10.1007/978-3-319-61425-0

Wang, L., Sy, A., Liu, L., & Piech, C. (2017). Learning to Represent Student Knowledge on Programming Exercises Using Deep Learning. *10th International Conference on Educational Data Mining*. Presented at the International Conference on Educational Data Mining, Wuhan, China.

Webster, A., Campbell, C., & Jane, B. (2006). Enhancing the creative process for learning in primary technology education. *International Journal of Technology and Design Education*, *16*(3), 221–235.