August 2018

Modeling Student Behavior: Analysis of Student Answers from ASSISTments

Diana Doherty
Worcester Polytechnic Institute

Modeling Student Behavior: Analysis of Student Answers from ASSISTments

A Major Qualifying Project Report:
Submitted to the Faculty of the
WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the
Degree of Bachelor of Science

Submitted By:
Diana Doherty

Date:
August 10, 2018

Report Submitted to:
Professor Joseph Beck

Acknowledgement

I would like to thank the following individuals and organization for their support and assistance throughout my project:

- **Professor Joseph Beck** from Worcester Polytechnic Institute for giving me the opportunity to work on this project and for his guidance and contribution to this project.

- **Professor Douglas Selent** from Worcester Polytechnic Institute for giving me the tools to test and extend his project on producing solution paths to students' incorrect answers.

- **Anthony Botelho** who familiarized me with Neural Network resources and extracted the data needed for this project from the ASSISTments database.

- **Worcester Polytechnic Institute,** for providing me with the educational opportunities needed to succeed in this project.

Abstract

This project explores an approach for analyzing problem level data received from an intelligent tutoring system, ASSISTments. Through data processing techniques, a dataset representative of student answering patterns is constructed. This data is fed into various machine learning algorithms to model student competency. The output from one such algorithm, an LSTM neural network, is extracted to generalize across success metrics, which the original model was not built to predict. Such a model could be used to determine a threshold for student competency and detect when students need help early. Instructors can then act on this information and follow through with prevention techniques before the student fails.

Table of Content

Introduction

The technologically-driven era simplified human labor through automation and increased accessibility with electronic records. However, technological reach did not simply end at influencing factory strategies or storage of public records; it extended way past into other regions, such as agricultural, medical and educational industries. Once the use of technology appeared in an industry, its application then continued to be improved upon boundlessly. When looking more closely into one such field, the incorporation of education with technology, while being more recent than other fields, is making life changing progress.

One of the goals in current research in the educational industry is to tailor lessons to individuals by accounting for their skills and learning preferences. In optimizing the learning model for the individual, lessons can be conducted more effectively and efficiently. With this goal in mind, laptops and tablets have made their way into classrooms, ebooks gave way to easier note taking, and educational platforms became the organizational force behind connecting the physical student to their classroom responsibilities. The distribution of computers to students allowed for the assumption that every student has access to the internet. This assumption made it possible for instructors to utilize intelligent tutoring systems in their teaching curriculum. The personalized tutoring aims of the intelligent tutoring systems are particularly helpful in situations where an instructor is not readily available, such as in larger class sizes, take-home work, and online classes.

ASSISTments is an online intelligent tutoring system. By using this application, instructors can manage their classrooms by utilizing or modifying existing problems or creating their own ones to then assign to their students. Real time feedback of student responses on both per problem and assignment levels are supplied to the instructors, giving them more time to reflect on student work, instead of requiring their time to grade and formulate the report. On the other hand, students get immediate feedback on the correctness of their answer, allowing them to self-reflect on their mistakes, and change their approach as needed. To guide struggling students, teachers can provide assists to the problems of their choosing by equipping them with explanations or guiding hints towards the correct answer. A regular hint might remind to multiply before adding, while a bottom out hint will reveal the actual answer. Depending on the assignment settings, the usage of hints could account for a wrong answer or, if partial credits are enabled, give some amount of recognition for the attempts.

An assignment may be one of three structures: a problem set, a placement, or a skill builder. A problem set consists of varying number of problems, and its completion is marked by the completion of all problems contained within it. A placement tests student skills such that, in case of incorrect answers, the student must complete problems testing prerequisite skills. Finally, in a skill builder a student masters the whole assignment, and, presumingly, learns the skill that assignment is testing, when they answer three problems correctly in a row, without a use of a hint. The students will continue answering questions until they get to a correct answer, and will continue being fed questions until they achieve mastery of the assignment. The student cannot skip questions. A question is marked as correct only if the first action was an attempt that resulted in the right answer. If the student used hints or did not provide with the right answer during the first attempt, the problem is marked as incorrect. The skill builder problems are assigned in a random fashion to students, such that each student gets their own set of problems, potentially creating an assignment that is unique to the student. A large pool of problems that test one or a set of skills within an assignment is required. Temple problems exist to ensure that teachers do not spend time coming up with or searching for questions to quench the need for the large number of problems. A template problem substitutes random numbers into allowed spots. An example of a template problem is the following: “What is the solution to the expression below? a + b * c”. From this template, numbers are generated to replace the variables, thereby creating new problems. A newly generated problem might look like 10 + 10 * 5. Easily a new problem might be generated to look like 15 + 17 * 6. Therefore, when testing the skill of PEMDAS, students will get multiple tries on similar problems in case their first attempts are wrong.

Problem level data is a byproduct of classrooms that use intelligent systems. In ASSISTments, for each student working on an assignment, problem level information gets stored: the answer given, correctness of that answer, time spent answering, number of attempted answers, and hint usage. With the application of data mining and machine learning techniques, this information can be used to model student learning and current understanding of concepts. Student past or current answers can model student behavior to help predict future performance. Application of a specific student’s answering sequences through the model will allow for early detection of potential success or failure. If warning signals of failure are manifested, instructors can efficiently shift their focus on executing prevention plans to specifically those students who might fail before the failure occurs. By being able to anticipate and better plan failures, instructors can be more prepared to give struggling students opportunities to not fall behind.

Methodology

Goals

This project explores an approach for analyzing problem level data received from an intelligent tutoring system, ASSISTments. Through data processing techniques, a dataset representative of student answering patterns is constructed. This data is fed into various machine learning algorithms to model student competency. The output from one such algorithm, an LSTM neural network, is extracted to generalize across success metrics, which the original model was not built to predict. Such a model could be used to determine a threshold for student competency and detect when students need help early. Instructors can then act on this information and follow through with prevention techniques before the student fails.

To accomplish the goal of creating such a model, the following points were considered:

1. A simple observation of student answers reveals that some incorrect answers are more common than others. For those uncommon answers, it is important to assess what separates them from the more common ones. Furthermore, if there exists a correlation between the strange answers and student success, it will be important to recognize what separates good and bad answers.
2. In taking student response oddity, methods of incorporating it in a predictive model must be analyzed.
3. Other aspects of student behavior should be incorporated in a predictive model. Finding the attributes which would yield better scores must be extracted.
4. A recurrent neural network that models student behavior by accounting for the above points must be implemented.

Concepts

Prior to the application of data analysis on modeling student success, data must be molded into a proper format through data cleaning. Data cleaning is a process of correcting incomplete or inaccurate records. However, the time spent on achieving data that is free from error does not necessarily reward with more accurate predictions. Studies have shown that the volume of the data, rather than the quality, is a better predictor of outcomes.[1] Therefore, it is important to derive more flexibility out of the existing data, by identifying material that is not directly visible

[1] Neil T. Heffernan, Korinn S. Ostrow, and Yan Wang, How Flexible is Your Data? A Comparative Analysis of Scoring Methodologies across Learning Platforms in the Context of Group Differentiation (Journal of Learning Analytics, 2017), 2.

in the data. This paper will focus on expanding characterization of student behavior into two concepts: stability and competency.

Stability measures how constant and resistant to the change of problems and assignment student responses are. The stability of a student answer can be measured in two different ways: the types of mistake or commonality of the mistake. If the skill the assignment is testing for is not learned or the student has a misconception about it, it is a fair assessment to say that the student might make persistent mistakes. When observing a student going through an assignment testing PEMDAS, if the student does not understand that multiplication occurs before addition despite being to the right of the addition, as the student continues answering questions, his future answers will embody this misunderstanding. The lack of knowledge of a skill can materialize in stability of the answers. The second definition of stability focuses on the commonality of the response. When constructing a probability of getting a specific answers, the more common answers to a problem attribute to commonality. If the student continuously makes mistakes that the majority of other students make, he is stable. If the student answers questions in an uncommon manner, the student is failing to be common, and is thus unstable.

Competency measures student skill and future success. Competency can be evaluated by analyzing student responses and be used as predictive labels for modeling student behavior. In terms of skill builder assignment in ASSISTments, competency can be determined in the achieved mastery of a problem set. Students who are able to complete three consecutive questions correctly will master the assignment, and students who give up before doing so will not. However, this measurement does not distinguish the struggling students from those who easily are able to achieve this mastery. Therefore other methods of assessing competency exist. The mastery of an assignment within the next k questions could be a determining measurement as well. Students who take less problems to master an assignment are more competent in that skill prior to the achieved mastery of that assignment. They are students who are not struggling in learning that skill. In case of early termination before mastery of the assignment, a stopout occurs. A student who give up prior to the tenth problem, stopout. A student who give up after the tenth problem, wheelspins. Wheelspining is a situation where a student completes ten problems without achieving mastery. More precisely, a student who completes three consecutive problems correctly on their tenth problem, does not wheelspin. However, a student who completes three consecutive problems correctly on their eleventh problem, does wheelspin.

Stability of mistakes is less erratic, and easier to understand. By teaching students the appropriate skills, they can fix all of their past mistakes. Similarly, a student making common mistakes, while there might not be a specific skill set that ties to his mistakes, he might be falling for common tricks. On the other hand, a student making different mistakes or those that no one else has made is more difficult to understand, and might be less likely to succeed. Competency can also use other measuring attributes as a predictive metric. In looking at timing information, it is possible to conclude how much effort a student is putting into the given answer. If the time it took for the student to answer a problem was too short, the student did not spend as much time on the problem as is required, and therefore did not put enough effort to succeed. If the student spent too long to answer a problem, the student was struggling and could not figure out how to solve the problem. Therefore, there should exist a length of time such that the student both had the time to thoroughly work through the problem and did not struggle while going through the process. Such a time will vary depending on the student, but on average, there must exist the time interval where student behavior is to be expected and normal. A competent student's responses should, ideally, fall somewhere on that optimal time interval. Hint usage is another of such attributes that ties in with competency. A student understanding the concept should not need to use any hints, or uses them minimally, and not as frequently.

Oddity in Answers

A student's raw answer can be analyzed in terms of stability. An odd answer will be marked as unstable and an expected answer will be stable. It is important to note that not only the correct answers are marked as stable; some incorrect answers can be stable as well. Stability of a wrong answer can be determined by two factors: the thought process the student went through to conclude to their answer and the commonality of that answer.

The Process of Getting to an Answer

The first idea of analyzing incorrect answers observes the process by which the student got to their answer. If the path that the student took to get to that answer seems off and very different from the correct answer, the student most likely does not understand the skills required to complete the problem and the assignment.

The Method

In his research on student mistakes, Douglas Selent concluded that 92% of the problem level hints from the 2012-2013 data were bottom out, containing only the answer to the problem.[2] Therefore, hints become a way to simply skip a problem, and do not advice on how to learn the skill required to complete it. Buggy messages address this issue. Instructors have the option to predict common wrong answers to a problem, and input a customized message depending on the answers. This message appears immediately after a student answers a problem in the manner the

[2] Douglas Selent, Creating Systems and Applying Large-Scale Methods to Improve Student Remediation in Online Tutoring Systems in Real-time and at Scale (Worcester Polytechnic Institute, 2017), 49.

instructor specified. However, this functionality is rarely used as it requires manual effort from the instructors.[3]

With a goal of solving this problem and improving intelligent tutoring systems in mind, Douglas Selent developed a machine learning algorithm that takes in student answers and raw problems as inputs that derives the student process of getting to their solution. The outcome of the machine learning algorithm was used to create customized hint messages to help students fix the cause behind their incorrectness. To evaluate the approach, the ASSISTments data from 2012-2013 was used. Furthermore, the algorithm encompases some of the built in features of ASSISTments, and is therefore a great tool to use to evaluate the first definition of stability.

The machine learning algorithm consists of five parts:

1. For each template, every incorrect answer for every problem is derived.[4] Template level information is constructed because a specific answer to a problem might have multiple solution paths. This might be problematic when picking the right solution path to the incorrect answer out of a derived set of paths. Choosing the simplest path might not always yield the right result because the numbers in the problem might, by chance, work with the simplest, but the least probable method.[5] Using template level information removes the ambiguity across multiple problems.
2. From the derived incorrect answers, the algorithm constructs a solution path to the actual incorrect answer, keeping count of the number of steps for the solution path.[6]
3. All of the constructed paths are saved to disk due to memory constraints.[7]
4. Regardless of which problem they came from, every path was merged into a single list. The goal of this step is to pick only those expressions that generalize across problems.[8]
5. After the paths have been generalized, some incorrect answers might still have more than one path. The path that applies to the most number of incorrect answers is picked, making it be the final solution path to an incorrect answer.[9]

Strengths

The most vital strength of this solution path processing algorithm is its ability to not depend on a vast array of students. Because paths are generalized across not only problems within a template, but also across many templates, it is possible to have an observation to student answers without needing to have large number of per problem data. This is particularly useful because of the

[3] Ibid., 50.
[4] Ibid., 53.
[5] Ibid., 51.
[6] Ibid., 60.
[7] Ibid., 63.
[8] Ibid., 74.
[9] Ibid., 80.

template problems, since they generate many problems, and thin out the number of students answering specific questions. Therefore, analysis of a student answer on a specific problem is almost independent of other students answering that same problem. When assuming that the student performed wrong operations to get to their answer, it is guaranteed that this algorithm will derive a path.[10] Also, the algorithm is designed to be optimized using the ASSISTments database, which is the source of data for this MQP.

Weaknesses

Despite its optimization, the algorithm is computationally expensive. Running the program on a dataset (which will be referred to as Assignment 5946) containing 2,681student records took roughly five hours to finish generating paths. Furthermore, out of the total records 1,236 answers generated solution paths, while, for the remaining 1,445 answers, no paths were generated. A similar outcome occurred when running the algorithm on another dataset (Assignment 7148), with the minority of the answers (1,394) having paths generated, but with a majority (2,546) of answers with no generated paths. While the algorithm promises that paths will be generated under certain, seemingly loose, conditions, in practice it appears that for the majority of answers, the algorithm cannot generate paths. In addition to time, the algorithm takes up a lot of space. Therefore, in order to limit the endless search for the solution paths, the algorithm is forced to stop executing only a couple of iterations into the search.[11] This works relatively well on short assignments, but could not be generalized across problems that are more complicated. So, there are only select problems for which the algorithm works for. The last weakness is the algorithm does not account for typographical errors. A student accidentally typing 12 instead of 21 should not be an underivable solution path. The number of records for which paths could not be generated for could reduce if typographical errors were considered.

Commonality of Answers

The idea behind observing the commonality of answers stems from the assumption that the odd answers will naturally separate from the expected answers. The expected answers, more students would submit, and the unexpected will be submitted by one student or a select few.

The Method

A commonality of an answer relies on other answers given for a specific problem. Therefore, a good measurement of commonality is to aggregate the frequencies of each individual answer, and compare it over total number of answers. This method gives a probability, or the likelihood, of a given answer. The more common answers will have a higher probability than the uncommon answers. However, when manipulating numbers with low probabilities, and therefore low value,

[10] Ibid., 59.

[11] Ibid., 59.

there exists a risk that they would disappear to zero. To solve this potential problem, a log-likelihood function is applied to the measurement of answer commonality. The natural log of the probabilities preserves the value at which the maximum value of the likelihood is reached, while, at the same time, mapping the probability values to larger and more convenient values.

Strengths

Unlike the solution path processing algorithm, the calculation of log-likelihood of answers is significantly faster. When running the likelihood calculations on the same two datasets as the previous method, the Assignment 5946 dataset finishing computing on average of 10 trials in 6.54 seconds, while the Assignment 7148 one 9.48 seconds. The time to calculate the log-likelihood is almost insignificant. Furthermore, in that time, while for more than half of the answers solution paths could not be generated, log-likelihood was calculated for all answers. This method does not rely on the problem type or the problem structure and can therefore generate a stability score for all answers to any non short or long answer problem.

Weakness

A major weakness of the log-likelihood method is its dependency on having non-trivial per problem data in order to be effective. The log-likelihood, since it looks at the raw answers themselves, could not be aggregated on a per template level. Therefore, while some problems might have two or three students answering it, others might have twenty. This causes a problem when calculating the probabilities. An odd answer for the problem with a low number of students answering it will have a higher probability, implying that it is less odd, than an odd answer from a problem with large number of responders. While, in some cases that observation might be true, the two problems cannot be compared without there existing a bias.

Comparing the Two Stability Methods

Goal

The solution path is generalized not only across problems within a template, but also across all problems within an assignment. Therefore, this method accounts for common mistakes within the whole assignment. As a result, the solution path can capture the student error in learning the skills the assignment is testing, such that, if an assignment tests fraction addition, the algorithm can detect the most common student errors related to not mastering the skill of adding fractions. Answer commonality, on the other hand, only accounts for one problem.

The strengths and weaknesses of both of the stability calculation methods complement one another. While the solution path method does not depend on the number of problem level responses, it is very slow in computing the rules and it cannot always generate them. While the log-likelihood method is fast and can be applied to a less restrictive set of problems, it strongly depends on having a large number of responses per problem. If the two methods are in high correlation, then both approaches yield the same result of describing student answers. In this case, it is possible to interchange either of the methods and pick the method that is best suited for the data. If, however, the two methods disagree, then one might be better than the other at predicting future performance, or each method picks up a different aspect of the answer, such that picking both might have them enhance the each other. Therefore, it is important to discover the relationship between the two methods.

Process

Since the solution path measurement is nominal while the commonality measurement is numerical, the two methods cannot be compared in their raw states. Therefore, the nominal data will be transformed in various ways to numerical data in order to examine which method will result in a highest correlation. For the following analysis, the same answer commonality will be used, and only the solution path transformations altered.

Answer commonality was computed by counting the number of times an answer occured within a problem, that number converted into the frequency of the answer (count per answer / count of total answers per problem) and then taking the log of that frequency.

In order to test the reliability of each transformation on solution paths to be generalized across assignments, multiple assignments needed to have been tested. However, since the execution of Selent's algorithm was time costly, it was utilized on two assignments: Assignment 5946 and Assignment 7148.

After the solution path algorithm was applied to the assignments, as accounted for in the algorithms weakness, for some answers rules were generated, while for others, they were not.

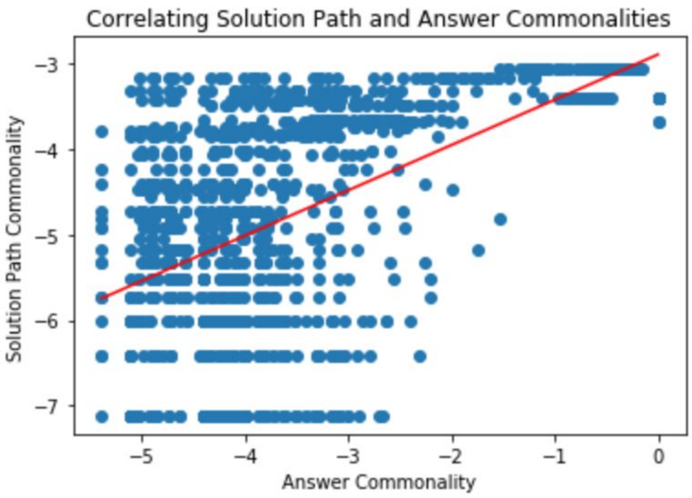
	Assignment 5946	**Assignment 7148**
Number of Rules Generated	1236	1394
Number of Rules Not Generated	1445	2546

Figure 1. Solution Paths Generated

While for more than half of the answers the solution paths could not be generated, the answer commonality accounts for all of the answers and their frequencies, even those that did not have solution paths. In other words, if for problem one, there were two answers, one submitted three times and another two, but the solution path was only generated for the first answer, the first answer's answer commonality would be log(3 / 5). Had the answer commonality included only the answers for which the solution paths were generated for, the first answer's answer commonality would have been log(3 / 3). This was done to evaluate whether either of the methods could substitute the other, accounting for both of the methods' weaknesses. The impact of such a loss in data by the solution paths generation method on the answer commonality will be analyzed.

Solution Path Commonality versus Answer Commonality

Solution path commonalty was generated by the same method as answer commonality was. For each answer, identical solution paths were grouped together, and the log of their frequency calculated. By observing the relationship between answer commonality and the likelihood of a solution path, it can be discovered if answer commonality also can be generalized across problems despite being calculated on a per problem basis.



Pearson's Correlation: 0.488

Figure 2. Assignment 5946. Correlation between Solution Path and Answer Commonalities

The red line is the linear least-squares regression, the best fitted straight line, for the two methods, revealing the correlation of 0.488 between the two methods. This correlation signifies a positive medium strength of association, such that when the the likelihood of a solution path is high, the likelihood of an answer is likely to be high as well.

The solution path with the highest solution path commonality value for Assignment 5946 appears to correspond with a higher answer commonality value (a value between 0 and -1.5). In looking more closely into that solution path, or others that might exhibit a similar containment within a range of answer commonalities, it is important to see what sets them apart from other solution path commonalities that range across many values of answer commonality.

Prior to retrieving the incorrect solution paths, the correct solution paths need to be extracted for easier comparison between them to the incorrect solution paths.

Correct Solution Paths	Count
(c + (a * b))	1107
(b - (a * c))	129

Figure 3. Assignment 5946. All Correct Solution Paths

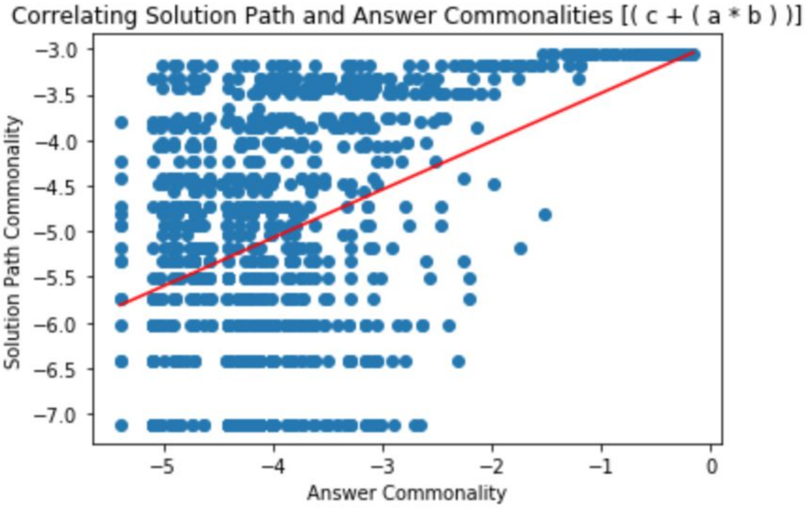
This assignment appears to have been testing PEMDAS. It tested if students multiplied before adding or subtracting despite the multiplication coming after in the sequence of operations. Therefore, the possible mistakes might include not adhering to the PEMDAS rule and adding c to a, or subtracting a from b, before performing the multiplication.

Incorrect Solution Path	Count of Solution Path	Description	Correct Solution Path
(b * (a + c))	58	Adding before multiplying (Not knowing PEMDAS)	(c + (a * b))
(a * b)	52	Only multiplying. Forgot to add	(c + (a * b))
(b + (a + c))	45	Adding all numbers	(c + (a * b))
((b - b) - (b - (a * c)))	41	Negative of correct answer	(b - (a * c))
((b / b) + (c + (a * b)))	40	Off by one error	(c + (a * b))

Figure 4. Assignment 5946. Top Five Incorrect Solution Paths

The top incorrect solution path produced, (b * (a + c)), coincides with logical assumption that rules of PEMDAS were not being followed. The other top incorrect mistakes, while are harder to explain, can still be described in words.

Because there are significantly more problems testing addition rather than subtraction before the multiplication, majority of the top incorrect solution paths refer to that problem template, potentially skewing the correlation. Therefore, the two templates have been separated to observe if there exists a better correlation.



Pearson's Correlation: 0.435

Figure 5. Assignment 5946. Correlation between Solution Path and Answer Commonalities for Solution Path (c + (a * b))

Incorrect Solution Path	Count of Solution Path	Description	Correct Solution Path
(b * (a + c))	58	Adding before multiplying (Not knowing PEMDAS)	(c + (a * b))
(a * b)	52	Only multiplying	(c + (a * b))

(b + (a + c))	45	Adding all numbers	(c + (a * b))
((b / b) + (c + (a * b)))	40	Off by one error	(c + (a * b))
(b * (a * c))	38	Multiplying all numbers	(c + (a * b))

Figure 6. Assignment 5946. Top Five Incorrect Solution Paths (c + (a * b))

After removing the second template, the correlation weakened. A potential explanation could be that the two correct solution paths share a similar incorrect solution path that helped increase the overall correlation. Alternatively, the second correct solution path might have a stronger association between the two methods.

The solution path (b * (a + c)), has the easiest explanation for the error, and has the tightest range of -1.6 to -0.15. This behavior best symbolizes the assumption that the more frequent answers coincide with the most explanatory and expected answers. However, solution path (a * b), despite seeming like an odd solution, appears rather frequently, and has a varying answer commonality value (from -1 to -5). This solution path contradicts the assumption.

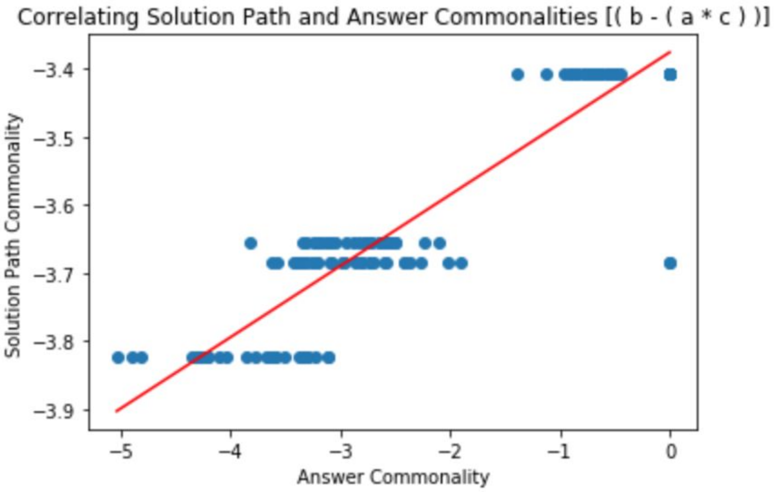
The reasons for why solution path (a * b) is common yet the answers derived from which do not have a set stability could be due to the weakness of the answer commonality approach. Since some problems have more students than others, for a similarly odd answer, the log-likelihood value will be different, and produce a varying degree of oddness. An odd answer with a high log-likelihood value of -1 might have less students answering that problem, skewing the strangeness of it to be less than it actually is.

A closer look at the two extremes of solution path (a * b):

Problem ID	**Problem Text**	**Incorrect Answer**	**This Incorrect Answer Count**	**Total Incorrect Answers Count**	**Log-Likelihood Value**
32984	1 + 9 * 8	72	13	43	-1.196251
46443	2 + 2 * 6	12	1	151	-5.017280

Figure 7. Assignment 5946. Two Extremes (c + (a * b))

While it is true that there are less number of students who answered problem 32984 than problem 46443, there are still a significantly higher number of students adhering to the solution path (a * b) for problem 32984. A possible reasoning behind this could be that the solution path (a * b) offered, only for problem 32984, an off by one error as well, which, from Figure 6, is a common wrong solution path for this correct solution. Had those factors not been in place, potentially, the two answers would have had a closer in value answer commonality value.



Pearson's Correlation: 0.910

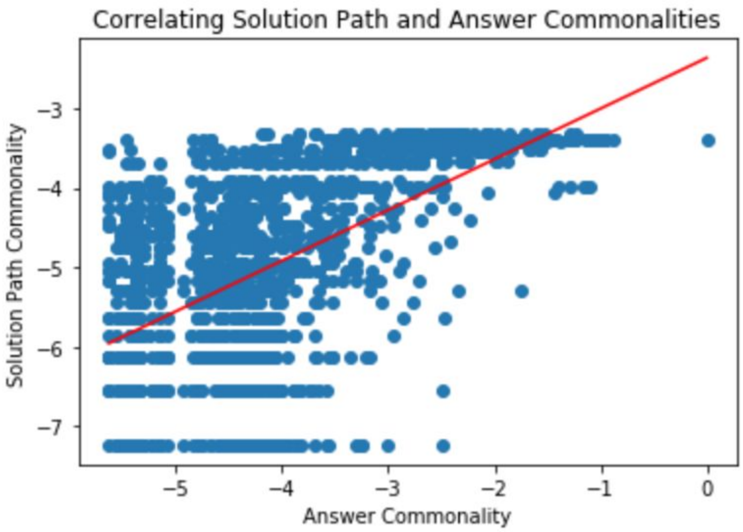
Figure 8. Assignment 5946. Correlation between Solution Path and Answer Commonalities for Solution Path (b - (a * c))

Incorrect Solution Path	Count of Solution Path	Description	Correct Solution Path
((b - b) - (b - (a * c)))	41	Negative of answer	(b - (a * c))
((b - a) * c)	31	PEMDAS error	(b - (a * c))
(a * c)	30	Not subtracting the first number	(b - (a * c))
((a - b) * c)	27	PEMDAS error and subtracting wrong	(b - (a * c))

Figure 9. Assignment 5946. Top Five Incorrect Solution Paths (b - (a * c))

For correct solution path (b - (a * c)), both the solution paths and answer commonalities are in high agreement. A possible reasoning behind this could be that since there are less students answering these problems, there are less incorrect solution paths. Furthermore, these solution paths are simpler, and yield a lower disorder in the answers. One of the solution paths generated for the addition template is (((b - c) - (c + c)) + (b + (a * c))). This solution path is long and complicated, and makes little sense in practice, yet it applied to eleven incorrect answers. Solution paths such as this might be the cause behind a lower correlation value for the other template.

While there exists a high correlations for the subtraction template, since there are more answers for the addition template, the overall correlation is significantly brought down in value. Potentially, the comparison between the solution path and answer commonality does not not generalize well across different templates, especially when there is an uneven distribution of student answers. To conclude if this observation is true, another assignment was analyzed.



Pearson’s Correlation: 0.552

Figure 10. Assignment 7148. Correlation between Solution Path and Answer Commonalities

Similarly to Assignment 5946, there appear to be many solution paths spanning a wide range of answer commonality values, with a similar moderately strong correlation value between the two methods.

The top solution path will be inspected.

Correct Solution Paths	Count

((c - b) / a)	1394

Figure 11. Assignment 7148. All Correct Solution Paths

Since there exists only one correct solution path, the data for this assignment will be looked at as a whole, without any splits.

Incorrect Solution Path	**Count of Solution Path**	**Description**	**Correct Solution Path**
((b / b) - ((b - c) / a))	51	Off by one of the negative of switching subtracted numbers	((c - b) / a)
((c - a) - b)	47	Subtracting all numbers	((c - b) / a)
((b - c) / a)	47	Switching subtracted numbers	((c - b) / a)
(c - b)	46	Forgot division	((c - b) / a)
(((c - a) / a) - (b / a))	41	???	((c - b) / a)

Figure 12. Assignment 7148. Top Five Incorrect Solution Paths

After the fourth most popular solution path, the solution stops being comprehensible when describing the error. As noted in Assignment 5946, a lower comprehensibility makes more sense with a lower answer commonality range. However, the top solution path commonalities are spread across a high variance of answer commonalities.

To further analyze why such a discrepancy exists, the two extremes of the solution path ((b / b) - ((b - c) / a)) were observed:

Problem ID	**Problem Text**	**Incorrect Answer**	**This Incorrect Answer Count**	**Total Incorrect Answers Count**	**Log-Likelihood Value**
49643	8y + 4 = -20	-2	69	237	-1.233954
49604	9y + 8 = 44	5	1	66	-4.189655

Figure 13. Assignment 7148. Two Extremes ((b / b) - ((b - c) / a))

For problem 49643, a = 8, b = 4 and c = -20. While the proposed solution path does produce the incorrect answer -2, there exists another, more intuitive path: (4 - 20) / 8, from a rule ((c + b) / a). The solution path appears to be odd despite being so common. Potentially, when accounting for problems such as 49643, more intuitive solution paths might be applied, but because ((b / b) - ((b - c) / a)) captures more answers by luck, it was chosen by the machine learning algorithm. When this solution path is compared against problem 49604, the result is (8 + 44) / 9 = 5.77778. This answer is not in the list of answers given to this problem, therefore, the two problems cannot be compared as one solution path, and should be split.

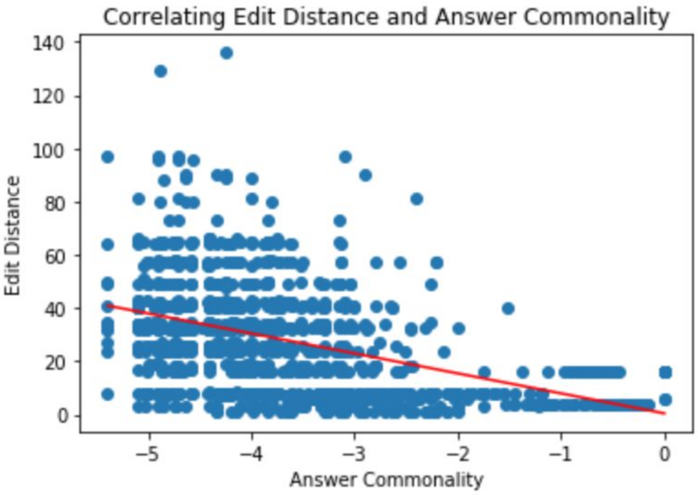
As a summary, comparing the solution path with answer commonality gives favorable correlations. However, some problems exist which might, in practice, bring down the correlation: the paths generated are either too complex to be legitimate ways by which the student got to their answer or too generalized to fit to problems for which a more intuitive path exists.

Edit Distance

Edit distance is a similarity measurement between two strings. It generates the least costly transformations needed to mutate one string into another. Four such transformations exist: addition of new characters, deletion of characters, substitution of characters, and transposition.

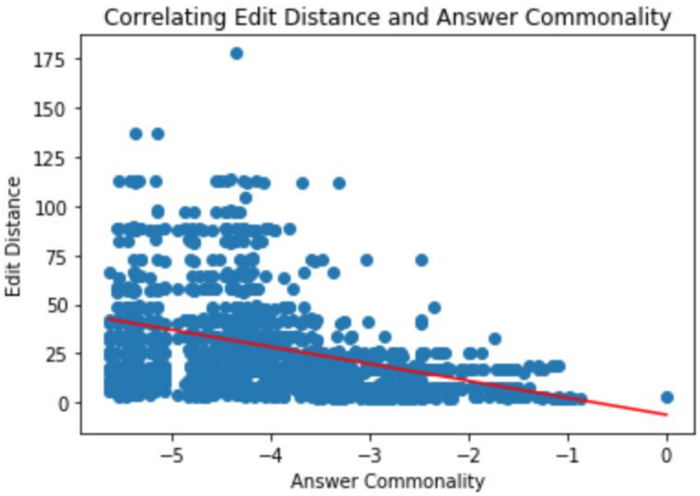
Constant Cost of One

Initially, the cost for each transformation was set to one, such that there is no difference between the type of transformation. The edit distance between correct and incorrect solution paths were calculated.



Pearson’s Correlation: -0.447

Figure 14. Assignment 5946. Correlating Edit Distance and Answer Commonality



Pearson’s Correlation: -0.387

Figure 15. Assignment 7148. Correlating Edit Distance and Answer Commonality

The baseline correlation for the edit distance metric is significantly lower than the solution path commonality. Despite the low correlation, the direction of the correlation is coherent with the expectation: the further away the incorrect answer is from the correct, the higher is the edit distance and therefore the less probable the answer is.

When the cost for transformation is one, multiple factors, such as the strangeness of a substitution, is not accounted for. It is more intuitive that a mixup from a multiplication to a division is to occur than a mixup from a subtraction to a multiplication were to occur. In the first scenario, it is possible to multiply as opposed to divide when dealing with percentages. It is more difficult to imagine a justification for the second scenario. Some transformations might be a signal of oddity. Therefore, it is important to readjust cost and weights for the edit distance before disregarding it as an invaluable comparison.

Analyzing Transformations

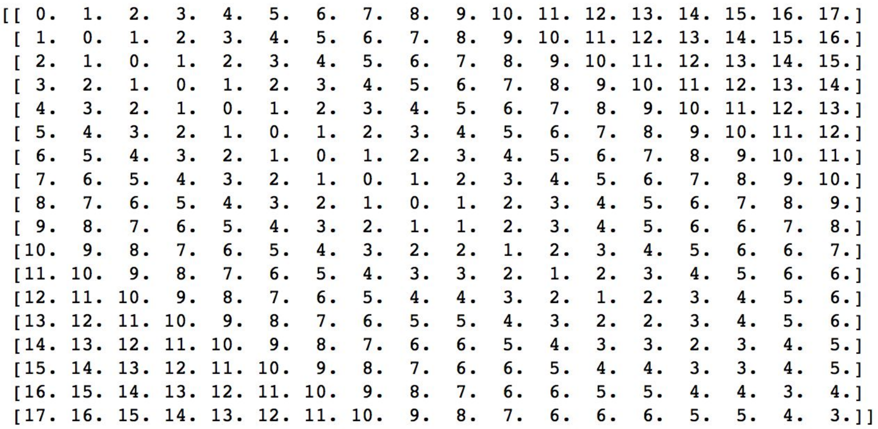
The first step is to look at each transformation and how it relates to the answer commonality. First an example will be followed through before being applied to the whole dataset. When calculating the edit distance, a matrix of transformations is formed. The value at the lower right edge of the matrix is the edit distance.

Assignment 7148 Problem 49596

Correct solution path: ((c - b) / a)

Incorrect solution path: ((c - a) - b)

Calculating the edit distance yields the following matrix:



```
[[ 0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11. 12. 13. 14. 15. 16. 17.]
 [ 1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11. 12. 13. 14. 15. 16.]
 [ 2.  1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11. 12. 13. 14. 15.]
 [ 3.  2.  1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11. 12. 13. 14.]
 [ 4.  3.  2.  1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11. 12. 13.]
 [ 5.  4.  3.  2.  1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11. 12.]
 [ 6.  5.  4.  3.  2.  1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10. 11.]
 [ 7.  6.  5.  4.  3.  2.  1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
 [ 8.  7.  6.  5.  4.  3.  2.  1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9.]
 [ 9.  8.  7.  6.  5.  4.  3.  2.  1.  1.  2.  3.  4.  5.  6.  6.  7.  8.]
 [10.  9.  8.  7.  6.  5.  4.  3.  2.  2.  1.  2.  3.  4.  5.  6.  6.  7.]
 [11. 10.  9.  8.  7.  6.  5.  4.  3.  3.  2.  1.  2.  3.  4.  5.  6.  6.]
 [12. 11. 10.  9.  8.  7.  6.  5.  4.  4.  3.  2.  1.  2.  3.  4.  5.  6.]
 [13. 12. 11. 10.  9.  8.  7.  6.  5.  5.  4.  3.  2.  2.  3.  4.  5.  6.]
 [14. 13. 12. 11. 10.  9.  8.  7.  6.  6.  5.  4.  3.  3.  2.  3.  4.  5.]
 [15. 14. 13. 12. 11. 10.  9.  8.  7.  6.  6.  5.  4.  4.  3.  3.  4.  5.]
 [16. 15. 14. 13. 12. 11. 10.  9.  8.  7.  6.  6.  5.  5.  4.  4.  3.  4.]
 [17. 16. 15. 14. 13. 12. 11. 10.  9.  8.  7.  6.  6.  6.  5.  5.  4.  3.]]
```

Figure 16. Assignment 7148. Edit Distance Matrix for Problem 49596

The edit distance between the correct and the incorrect solution paths is the value in the lower right corner: three. In order to determine the transformation steps taken to arrive to that edit distance value, the following method was used:

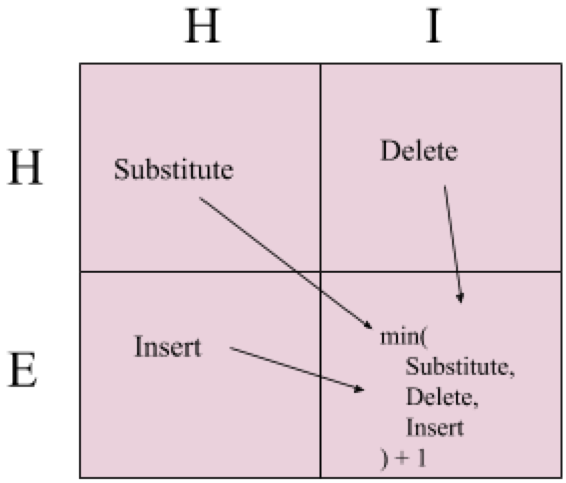


Figure 17. Edit Distance Transformations

Using the following strategy, the transformations for Assignment 7148 Problem 49596 were constructed to get the following matrix of transformations:

...	...	...	...	...
...	--------s---s-	--------s---s-i	--------s---isi-	--------s---isi-i
...	--------s---s-d	--------s---s-s	--------s---s-is	--------s---isi-s
...	--------s---dsd-	--------s---s-ds	--------s---s-s-	--------s---s-s-i

...	--------s---dsd-d	--------s---dsd-s	--------s---s-s-d	--------s---s-s--

Figure 18. Assignment 7148. Transformation Matrix for Problem 49596

Since the transformation matrix is large, only the lower right corner is shown in Figure 18. Similarly to the edit distance, the lower right cell contains the transformation needed to take place in order to go from the incorrect to the correct answer in the most optimal path: --------s---s-s--. The dash symbolizes no transformation, the s symbolizes substitutions, the i is insertion and d is deletion.

In this example, the substitutions are:

a → b
- → /
b → a

In order to measure how much each transformation corresponds with the log-likelihood value, each transformation total was calculated.

For each answer, the transformation that appeared more frequently than others was used to symbolize that record. For example, if a student answered a question whose transformation was --d--i--ss, since there are more substitutions, his answer was marked as being substitution dominant. The following was the result of each record.

	Assignment 5946	**Assignment 7148**
Substitution	199	238
Deletion	953	1050
Insertion	84	106
Transposition	0	0

Figure 19. Dominating Transformations Count Table

For every answer commonality log-likelihood value, the number of the dominant transformations was counted.

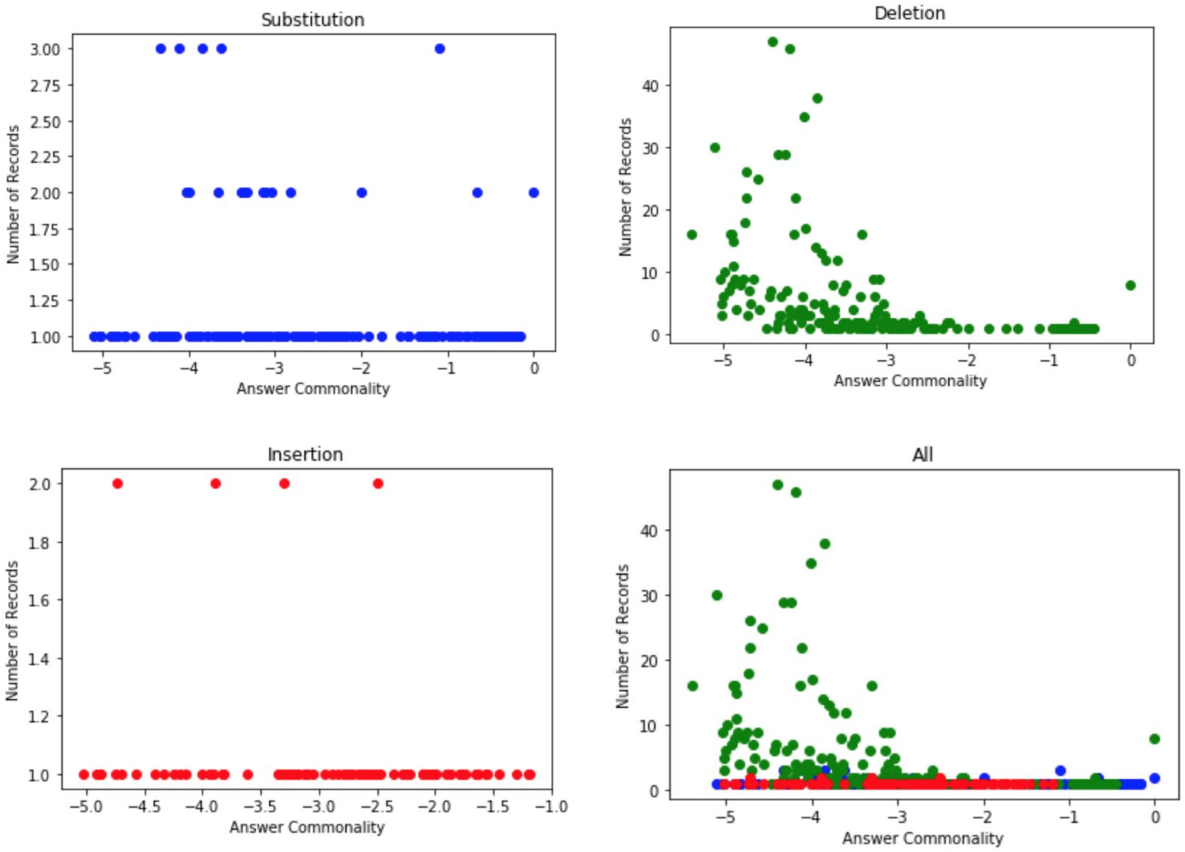
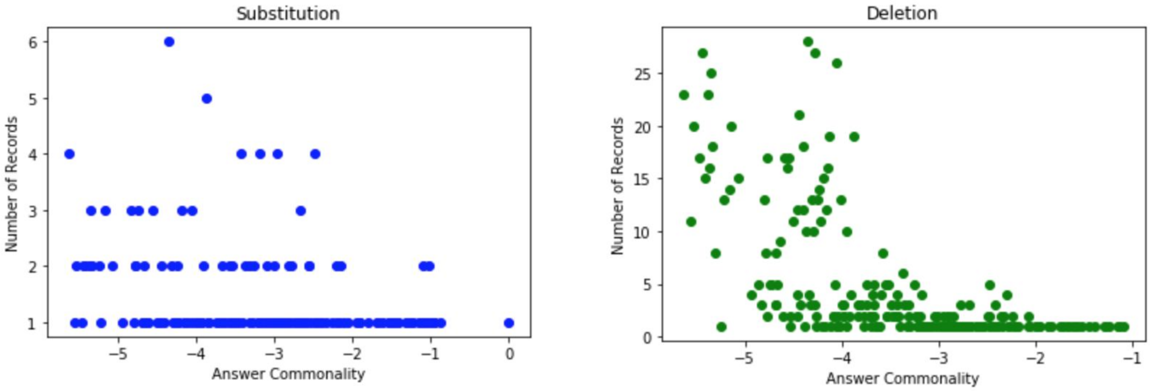


Figure 20. Assignment 5946 Dominating Transformations Graphs



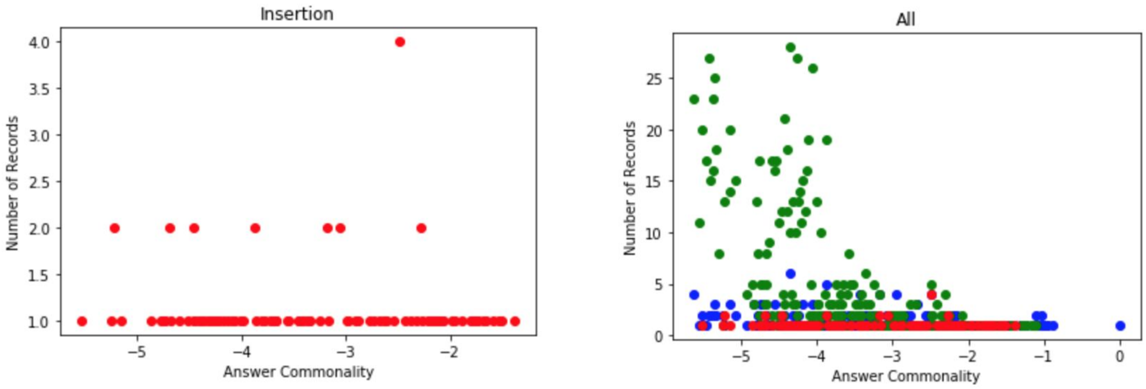


Figure 21. Assignment 7148 Dominating Transformations Graphs

From figures 20 and 21, it can be concluded that insertion does not have any oddity, substitution has an unclear representation and deletion symbolizes oddity in both of the assignments. Therefore, the cost for deletion and possibly substitution should be greater, as there are more records with lower log-likelihood value.

Cost of Substitution	**Assignment 5946 Correlation**	**Assignment 7148 Correlation**
1	-0.447	-0.387
5	-0.439	-0.397
10	-0.439	-0.397

Figure 22. Adjusting Cost of Substitution

With the increase of cost Assignment 5946 is getting worse, while Assignment 7148 is getting better. Therefore, the result of the increase in the cost for substitution is inconclusive. Strangely, the correlations stop changing when the cost increases from five to 10.

Cost of Deletion	**Assignment 5946 Correlation**	**Assignment 7148 Correlation**
1	-0.447	-0.387
5	-0.453	-0.385
10	-0.454	-0.385

Figure 23. Adjusting Cost of Deletion

While Assignment 5946 appears to be improving slightly, the change is not significant. Assignment 7148, on the other hand is doing slightly worse as the cost of deletion increases.

Insertion is skipped because it was least likely to yield correlation improvements. Since both deletion and substitution did not result fruitfully, it is not necessary to test modification of insertion. In its raw form, adjusting weights of transformations does not improve the correlation.

Transformation in Text

From Figure 19, the deletions transformation is significantly more predominant than other transformations. This is because to transform from the incorrect solution path that is much longer to the correct solution path, the incorrect solution path must go through multiple steps of the deletion process. It is faulty to assume that the characters being deleted, if they are not found in the text, to be weighed the same as the characters not found in the text. If a question asks for the student to multiply, but the student adds, that should be marked as a sign of oddity.

Deletion of operations found in the text were weighed less than deletions of operations not found in the text. Variables such as a, b and c, were counted as being part of the problem text.

Cost of Operations Not Found in Problem Text	Cost of Operations Found in Problem Text	Assignment 5946 Correlation	Assignment 7148 Correlation
1	1	-0.447	-0.387
5	1	-0.426	-0.346
10	1	-0.410	-0.325
1	5	-0.444	-0.385
1	10	-0.442	-0.383

Figure 24. Adjusting Cost of Deleting Operations Found in Problem Text

Modifying the cost of deleting operations found in problem whether positively or negatively does not improve correlation values. More interestingly, when the cost of operation not found in problem is increased, the correlation drops more than it does for the increase of cost of operations found in the problem. A decrease in both situations signifies that the modification of the deletion cost would not benefit the correlation in any way.

Substitution Weight Modification

The idea behind this weight adjustments is that some substitutions are more odd than others. It is more natural to mess up dividing instead of multiplying when, for example, dealing with percentage calculations. It is more odd, however, of the multiplication is replaced by subtraction.

Counting up the most common substitutions revealed:

Actual Text	Correct Text	Count
‘ ’	‘c’	261
‘ ’	‘b’	160
‘+’	‘*’	145
‘c’	‘b’	109
‘ ’	‘+’	106
‘b’	‘c’	103
‘ ’	‘*’	102
‘*’	‘+’	96
...	...	...
‘-’	‘*’	15
‘-’	‘+’	5
...	...	...
‘/’	‘+’	1

Figure 25. Assignment 5946. Common Substitutions

Actual Text	Correct Text	Count
‘’	‘c’	575
‘’	‘/’	264
‘a’	‘b’	201

‘ ’	‘b’	176
..	..	..
‘c’	‘b’	119
‘b’	‘c’	110
..	...	...
‘-’	‘/’	71
..	...	...
‘+’	‘-’	40
‘+’	‘/’	34
‘/’	‘-’	28
...	...	...
‘*’	‘-’	18

Figure 26. Assignment 7148. Common Substitutions

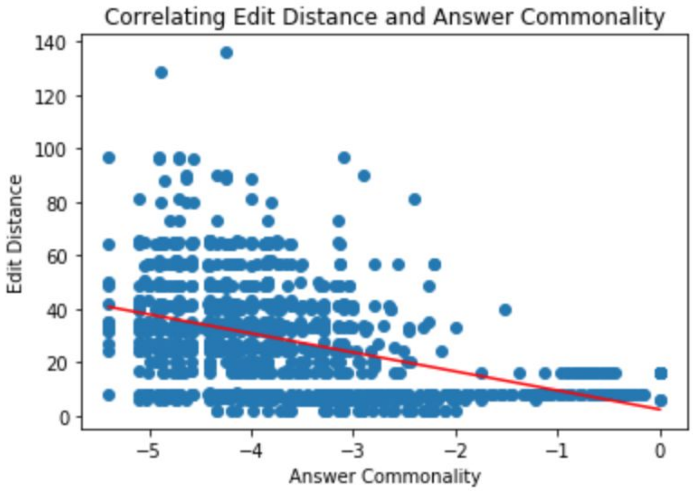
From the common substitutions and the assumption that extra parentheses and spaces should not cost much, a substitution table of costs was created. Because the variable based substitutions are problem specific, their weights between each variable were kept the same, but inflated slightly to appeal with the increase in other substitution.

	(	)	*	/	+	-	a	b	c
(	0	1	1	1	1	1	1	1	1
)	1	0	1	1	1	1	1	1	1
*	1	1	0	5	10	15	1	1	1
/	1	1	5	0	15	10	1	1	1
+	1	1	10	15	0	5	1	1	1
-	1	1	15	10	5	0	3	1	1
a	1	1	1	1	1	1	0	3	3

b	1	1	1	1	1	1	3	0	3
c	1	1	1	1	1	1	3	3	0

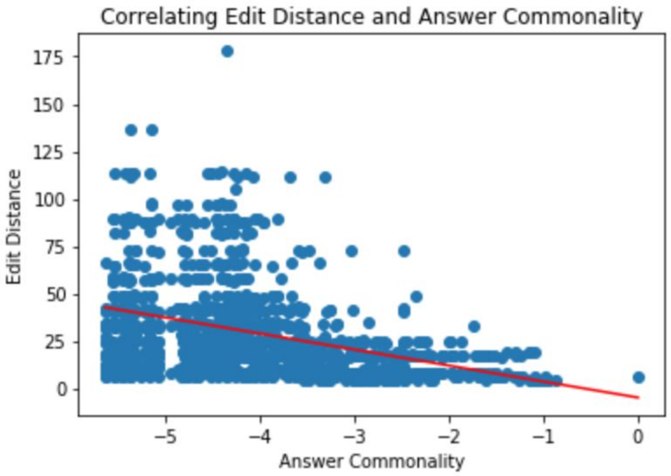
Figure 27. Substitution Cost Table

Using the costs in Figure 27 to modify the substitution weights, a new edit distance was computed. It was then compared against the answer commonality.



Pearson's Correlation: -0.432

Figure 28. Assignment 5946. Correlation After Substitution Cost Modification



Pearson's Correlation: -0.384

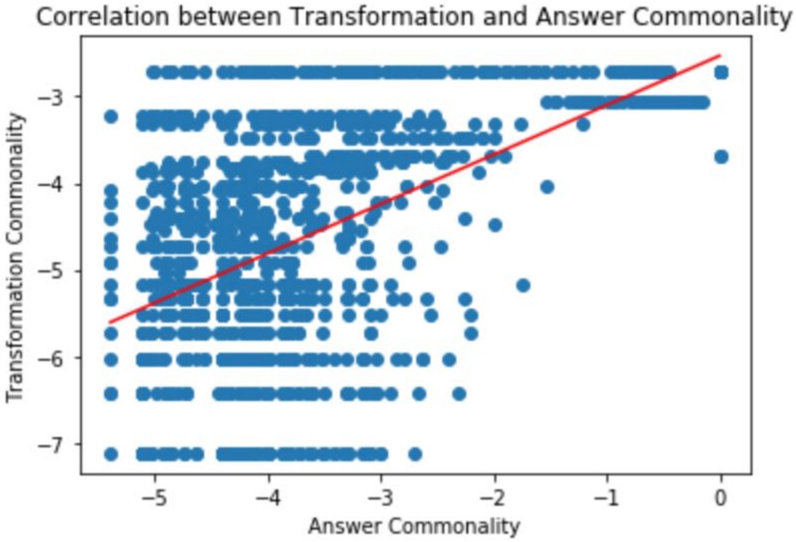
Figure 29. Assignment 7148. Correlation After Substitution Cost Modification

Compared to the raw edit distance correlation value of -0.447 for Assignment 5946, and -0.387 for Assignment 7148, the modifications to the edit distance provide to be inefficient in comparing to answer commonalities. Resulting values are still worse off than raw one weight cost. Judging by the graph, incorrect solution paths that are close to the correct solution path can be both common and uncommon. The further the edit distance grows, the less common the answers grow. Therefore, edit distance is not the best way to compare answer commonality and solution paths.

Transformation Commonality

Since the edit distance values were significantly less fruitful than commonality of solution paths, a deeper look at commonality in the context of edit distance is required.

Transformation commonality was generated by computing transformations for each answer using the most effective edit distance calculations (were a cost of each transformation was uniformly one). A transformation indicates the necessary steps required to reach the correct solution path from the incorrect one. For each answer, identical transformations were counted, and the log of their frequency calculated.



Pearson's Correlation: 0.492

Figure 30. Assignment 5946. Correlation between Transformation and Answer Commonality

While not only an improvement from raw edit distance, the transformation commonality also fairs better than the solution path commonality, correlation of which could not be beaten by edit distance. This signifies that commonalities better compare between one another, and that edit distance might prove to be more valuable than is observed from first glance. Figure 30, also reveals a set of transformations that are bounded by high answer commonalities.

Transformation	Count	Answer

		Commonality Range
'dddddddddddddd-------------dd----'	83	-5.00 to 0
'iiiiii-------ii--'	82	-5.02 to -1.19
'--s-s-----s-s----'	58	-1.54 to -0.16
'dd-dddd--dd--------------'	49	-5.39 to -2.53
'--s-------s-s----'	45	-5.11to -1.22

Figure 31. Assignment 5946. Top Five Transformations

The top two transformations are spread out across a wide variety of answer commonality values. A closer look at those values reveals the following:

Child ID	**Correct Rule**	**Correct Answer**	**Incorrect Rule**	**Incorrect Answer**	**Answer Commonality**
33013	(b - (a * c))	-1132	((b - b) - (b - (a * c)))	1132	0.0
33014	(b - (a * c))	-442	((b - b) - (b - (a * c)))	442	0.0
33018	(b - (a * c))	-85	((b - b) - (b - (a * c)))	85	0.0
33019	(b - (a * c))	-76	((b - b) - (b - (a * c)))	76	0.0
33025	(b - (a * c))	-993	((b - b) - (b - (a * c)))	993	0.0
56730	(b - (a * c))	-1908	((b - b) - (b - (a * c)))	1908	0.0

Figure 32. Assignment 5946. Transformations 'dddddddddddddd------------dd----' High Answer Commonality Value

In the most common case, the transformation 'dddddddddddddd------------dd----' is manifested in one solution path: ((b - b) - (b - (a * c))). When applied, this solution path gives the a negative of the correct answer. While those incorrect answers do not seem odd, their answer commonality is 0, signifying that they were the only response for the problem. As mentioned earlier, this is a weakness of answer commonality. While those answers do not seem so odd, they should not be valued so absolutely common, as they might be potentially skewing the correlation (either favorably or not).

Child ID	Correct Rule	Correct Answer	Incorrect Rule	Incorrect Answer	Answer Commonality
32970	(c + (a * b))	283	((b / b) + (c + (a * b)))	284	-4.406719
32983	(c + (a * b))	-393	((a - b) + (c + (a * b)))	399	-4.189655
32987	(c + (a * b))	84	((b / b) + (c + (a * b)))	85	-4.143135
32988	(c + (a * b))	12	((b / b) + (c + (a * b)))	13	-4.060443
32996	(c + (a * b))	293	((b / b) + (c + (a * b)))	294	-4.110874
46442	(c + (a * b))	22	((b / b) + (c + (a * b)))	23	-4.007333

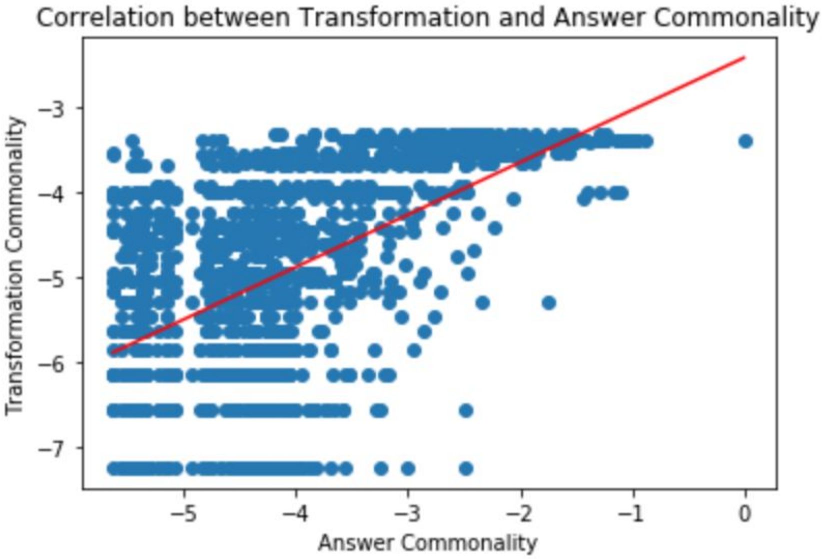
Figure 33. Assignment 5946. Transformations 'ddddddddddddd------------dd----' Low Answer Commonality Value

Now, for the same transformation, new solution paths are revealed: off by one error and non descriptive mistake. Here, the answer commonality has more data and seems more reasonable than in the previous case. Despite being a common transformation, however, the answer commonality is rather low. Therefore, had there been more information for the problems in Figure 32, the correlation would have been weaker.

Child ID	Correct Rule	Correct Answer	Incorrect Rule	Incorrect Answer	Answer Commonality
46443	(c + (a * b))	14	(b * (a + c))	24	-0.180998
32967	(c + (a * b))	60	(b * (a + c))	100	-0.602175
32968	(c + (a * b))	117	(b * (a + c))	192	-1.174120
32969	(c + (a * b))	57	(b * (a + c))	252	-1.442384

Figure 34. Assignment 5946. Transformations '--s-s-----s-s----' Answer Commonality Value

The PEMDAS error re-appears again, and it is contained within the same boundary. Successfully, the transformation commonality captures that more common transformations have a higher answer commonality value.



Pearson’s Correlation: 0.541

Figure 35. Assignment 7148. Correlation between Transformation and Answer Commonality

Unlike the Assignment 5946, transformation commonality has a lower Pearson’s correlation than the solution path’s correlation of 0.552. This signifies that there is not a concrete answer to which of the commonality methods are better at portraying the correlation.

Transformation	**Count**	**Answer Commonality Range**
----dsddddd--dddd--dddd------dd--	51	-4.19 to -1.23
--------s---s-s--	47	-5.47 to 0
----s---s--------	47	-3.54 to -0.88
ii-------iiiii--	46	-3.86 to -1.39
dd--------dsddddd--dddddd----dd--	41	-5.63 to -1.57

Figure 36. Assignment 7148. Top Five Transformations

The most common transformation appears to be rather complex, and, from Figure 36, to range a wide values of answer commonality. With exception for transformation ‘--------s---s-s--’, the less complex the transformation is, the tighter the answer commonality range becomes.

Child ID	Correct Answer	Incorrect Answer	Answer Commonality	Answer Count	Total Answers
49596	5	8	-1.535330	56	260
131041	2	10	0.000000	1	1
49598	6	25	-2.824774	7	118
49601	-4	-20	-4.677491	2	215

Figure 37. Assignment 7148. Transformations '--------s---s-s--' Answer Commonality Value

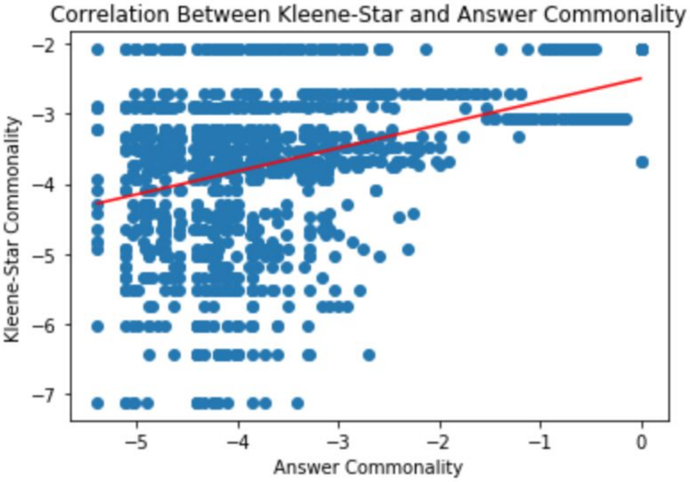
Aside from the problem 131041, there are enough total answers to make the answer commonality not a trivial value. Despite this knowledge, sometimes the path to an incorrect answer is just more common within certain problem sets than others.

Tuning

Transformation commonality is computed by comparing the exact raw edit distance transformations. This approach is quite restricting, and could not capture the generalization some substring of the pattern could be making. Some tuning measurements should be taken to assure the best correlation.

Kleene-Star

The Kleene-Star method merges all of the instances of a repeating transformation. More precisely, a transformation of iii----d, would capture the pattern of insert, followed by no change, and then a delete, giving a pattern of i-d. Similarly, a transformation of i-ddddd would produce the pattern i-d. The idea behind this method is to allow a comparison between strange answers with lots of deletion versus the strange answers with less deletions. And to see if there exists such a pattern, such that no matter the number of occurrences, it will be a sign of strangeness.



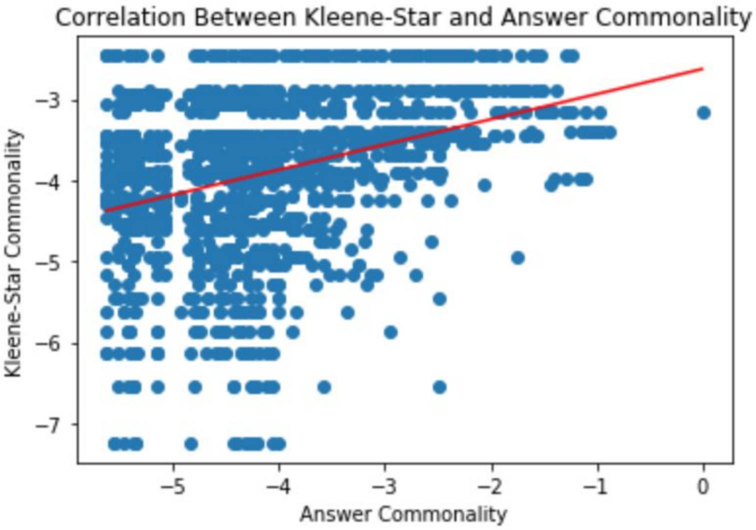
Pearson’s Correlation: 0.337

Figure 38. Assignment 5946. Kleene-Star Tuning Correlation

Kleene-Star	Count
d-d-	155
i-i-	82
d-d-d-	69
d-d-d-d-	67
-s-s-s-s-	58

Figure 39. Assignment 5946. Kleene-Star Top Five Transformations

From the results, it can be concluded that this method of generalization does not positively generalize the transformations. From the raw transformation, Figure 31 shows that the most common transformation was 'ddddddddddddd------------dd----', with a count 83. And the relationship with the transformation and answer commonality was such that for this transformation, the answer commonality was rather high. Kleene-star generalized the count to 155, and with it captured transformations that were more spread out across the answer commonality range, significantly decreasing the correlation.



Pearson's Correlation: 0.336

Figure 40. Assignment 7148. Kleene-Star Tuning Correlation

Kleene-Star	Count
-dsd-d-d-d-	119
i-i-	78
-dsd-d-d-d-d-	74
-dsd-dsd-d-	66
-s-s-s-	60

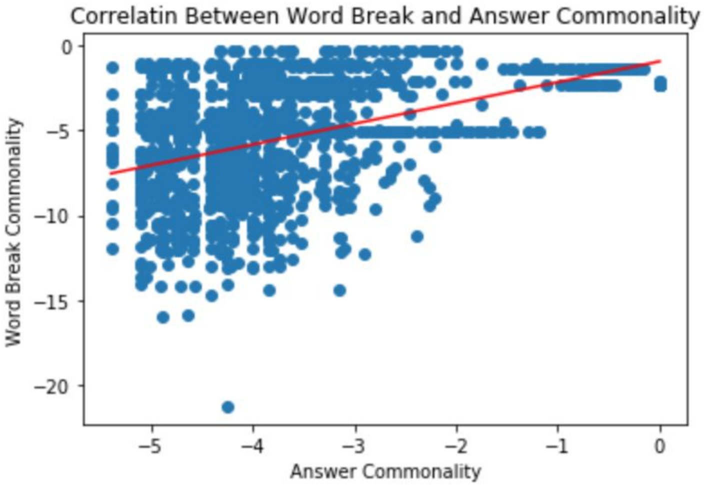
Figure 41. Assignment 7148. Kleene-Star Top Five Transformations

Similarly to the previous assignment, the top Kleene-Star transformation is identical to the application of Kleene-Star to the top raw transformation, but with the count more than doubling from 51 to 119.

Consistently, the correlation between the Kleene-Star commonality and answer commonality dropped from the transformation commonality and answer commonality. Therefore, this form of a generalization is ineffective. A possible assumption could mean that the rarity of lengthy transformations are associated with a lower answer commonality.

Word-Break

The Word-Break method splits transformations by active instances, such that areas of no change become the delimiter. Following the transformation of iii----d, when applied to the Word-Break method, the pattern split would result in iii and d. The two word fragments contribute to the overall frequencies for the assignment. For each student response, the probability of each word is summed to produce the final Word-Break commonality value. The idea behind this method is to zoom into a small section of a transformation and compare between the strangeness of that section. This will allow for a more generalized comparison, as it captures an odd raw transformation into pieces that might comprise of odd and expected patterns, to better evaluate the transformation.



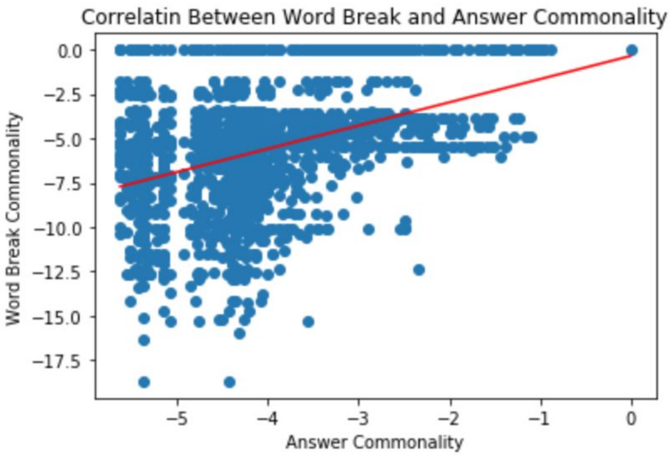
Pearson's Correlation: 0.433

Figure 42. Assignment 5943. Word Break Tuning Correlation

Word Break	Count
s	874
dd	858
dddd	702
dddddd	305
dsd	211

Figure 43. Assignment 5943. Word Break Top Five Transformations

With the Word-Break tuning method, the Pearson's correlation experienced a drop. Since a solution path becomes an aggregation of all words, each point on the scatter plot becomes smoother and more spread out. In the raw transformation, there were less variability and so patterns between that commonality and the answer commonality was easier observed.



Pearson's Correlation: 0.400

Figure 44. Assignment 7148. Word Break Tuning Correlation

Word Break	Count
s	1390
dd	734
dddd	729
dddddd	459
dddddddd	457

Figure 45. Assignment 7148. Word Break Top Five Transformations

Similarly to the previous assignment, the Word-Break tuning method decreased the Pearson's correlation from the raw transformation value. The top four word transformations from the two assignments are identical. The Word Break appears to better generalize across assignments, but does so at the cost of slight accuracy loss.

N-Grams

N-Grams split the raw transformation by character length, where N is the length of each segment, as opposed to content. Applying a 2-Gram split to the transformation of iii----d, results in segments: ii, i-, --, -d. The frequencies of each segment was then recorded and the log of which summed to signify the N-Gram value. This method aims to test if the content tuning of the previous approaches are effective.

N-Gram	Correlation
1-Gram	0.317
2-Gram	0.400
3-Gram	0.440
4-Gram	0.465
5-Gram	0.476
6-Gram	0.483
7-Gram	0.487
8-Gram	0.490
9-Gram	0.492
10-Gram	0.492

Figure 46. Assignment 5943. N-Gram Correlations Table

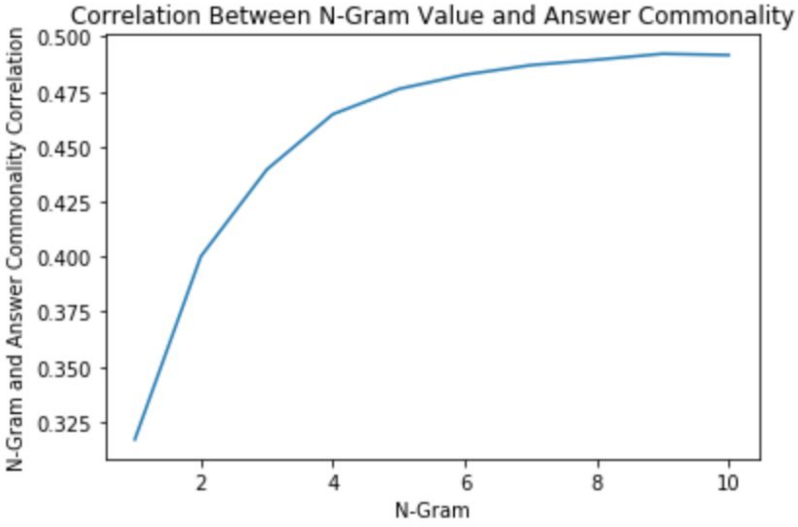


Figure 47. Assignment 5943. N-Gram Correlations Chart

The correlation grows, slowly leveling off with the increase of N. The maximum correlation reached is 0.492, which is the value for the raw transformation. When the 9-Gram is reached, there is almost no difference between the split length of the gram and the complete transformation, therefore the correlation is so high. Therefore, there is no improvement in a length split of the transformation.

N-Gram	Correlation
1-Gram	0.317
2-Gram	0.409
3-Gram	0.422
4-Gram	0.431
5-Gram	0.432
6-Gram	0.437
7-Gram	0.440
8-Gram	0.441
9-Gram	0.444
10-Gram	0.447
11-Gram	0.449
12-Gram	0.449

Figure 48. Assignment 7148. N-Gram Correlations Table

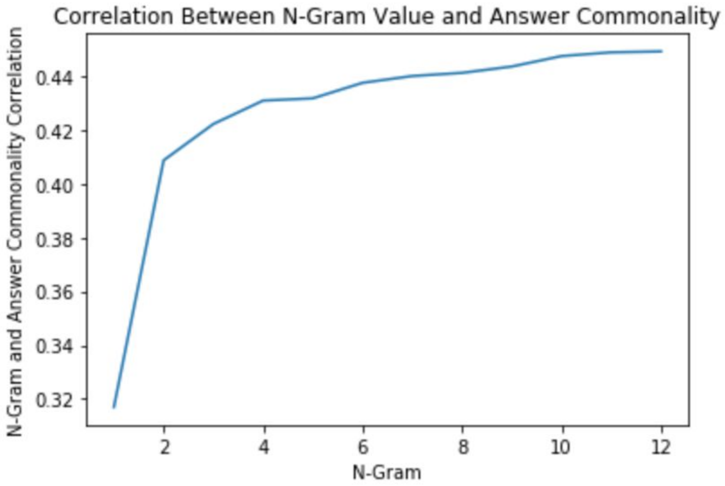


Figure 49. Assignment 7148. N-Gram Correlations Chart

As with the previous assignment, the correlation grows, reaching the maximum correlation of 0.449, which is lower than the 0.541 value for the raw transformation. When the maximum correlation is reached, the split gram, despite being almost identical to the full transformation, does not equate the complete transformation. Therefore, there is a difference between a high valued N-Gram and the original raw transformation, and this difference is not an improvement to the correlation.

Summary

Since there are many approaches for analyzing solution paths, it is important to compare the methods, to determine which of the methods worked best.

Type of Edit Distance	Assignment 5946 Correlation	Assignment 7148 Correlation
Cost 1	-0.447	-0.387
Cost of substitution 5	-0.439	-0.397
Cost of deletion 5	-0.453	-0.385
Deletion of operations not found in the problem text with cost of 5	-0.426	-0.346
Deletion of operations found in the problem text with cost of 5	-0.444	-0.385
Substitutions with costs from the costs table	-0.432	-0.384

Figure 50. Edit Distance Summary Table

Edit distance looked at the least costly transformation from the incorrect solution path to the correct one. While the increase of cost of substitution increases the correlation of Assignment 7148 by 0.1, it has the same strong effect on Assignment 5946 in the worst direction. Similarly, when cost of deletion is 5, Assignment 5946 correlation is improved, but Assignment 7148 correlation becomes worse. Therefore, despite the different manipulations to improve the comparison between solution paths and answer commonality, the universal cost of one for all transformation yields the best correlations for both of the Assignments: -0.477 for Assignment 5946, and -0.387 for Assignment 7148.

Type of Transformation Commonality	Assignment 5946 Correlation	Assignment 7148 Correlation
Whole Sequence	0.492	0.541
Kleene-Star	0.337	0.336
Word-Break	0.433	0.400
12-Grams	0.492	0.449

Figure 51.Transformation Commonality Summary Table

In the Transformation Commonality approach, the least costly transformation from the incorrect solution path to the correct one a commonality is calculated and compared against answer commonality. The tuning steps taken to maximize the comparison proved to not be an improvement from the original, whole sequence commonality.

Name	Assignment 5946 Correlation	Assignment 7148 Correlation	Pros	Cons
Solution Path Commonality	0.488	0.552	- Fastest to compute - Good performance	- Commonality depends on number of students being analyzed
Edit Distance	-0.447	-0.387	- Stand alone and does not depend on number of students being analyzed	- Has a comparatively low correlation from the two commonality approaches.
Transformation Commonality	0.492	0.541	- Good performance	- Commonality depends on number of students being analyzed - Slowest to compute

Figure 52. Solution Path versus Answer Commonality Summary Table

Answers Without Solution Paths

The answers for which solution paths were not generated should not be left without analysis. There exists a moderate correlation between the generated solutions paths and the answer commonality index. If a similar relationship exists between the records without solution paths and answer commonality, then a stronger connection would exist between the two different types of stabilities.

The first observation is the number of records for each answer commonality and how that compares between records with and without solution paths. A hypothesis exists that the answers for which the solution paths could not be generated, are odd, and therefore uncommon among students.

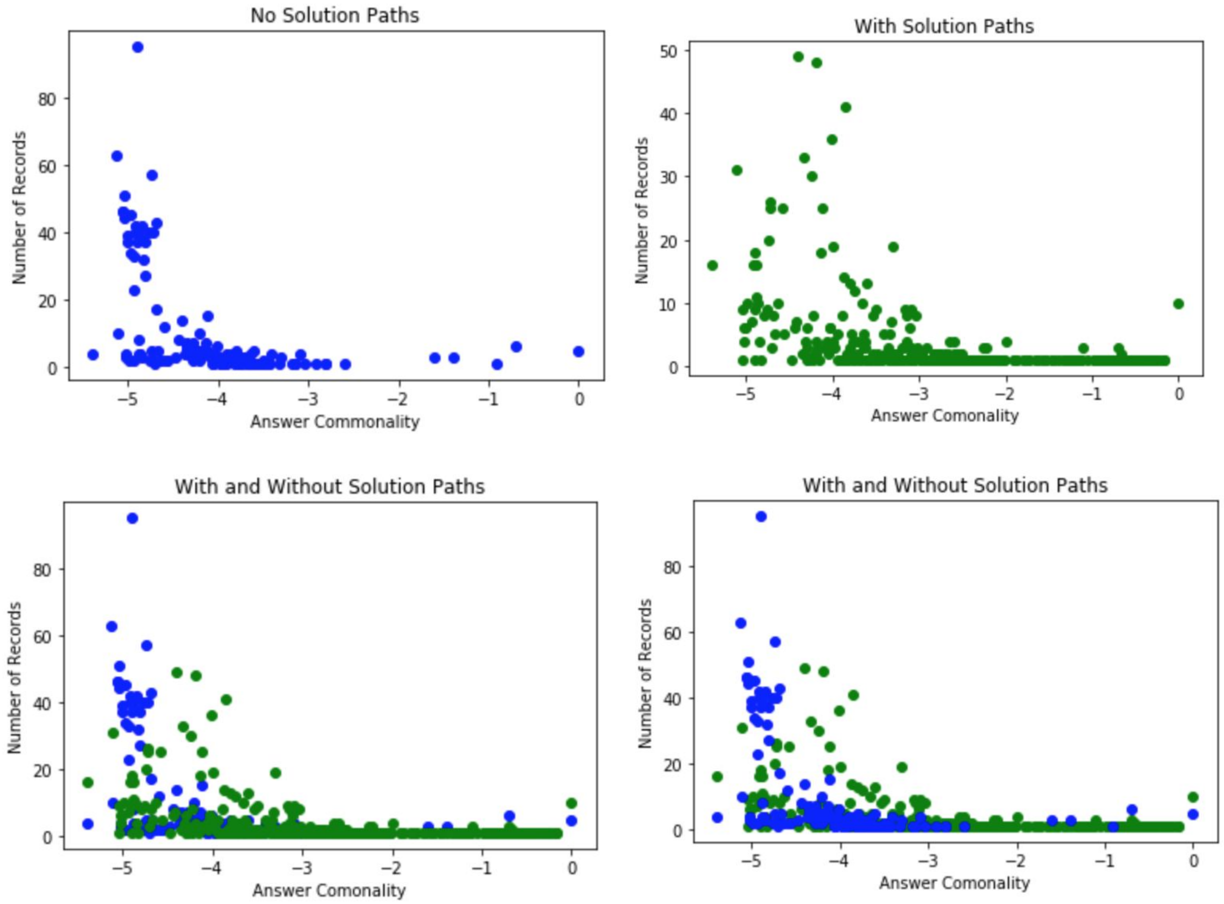


Figure 53. Assignment 5946. Generated vs Not Generated Solution Paths

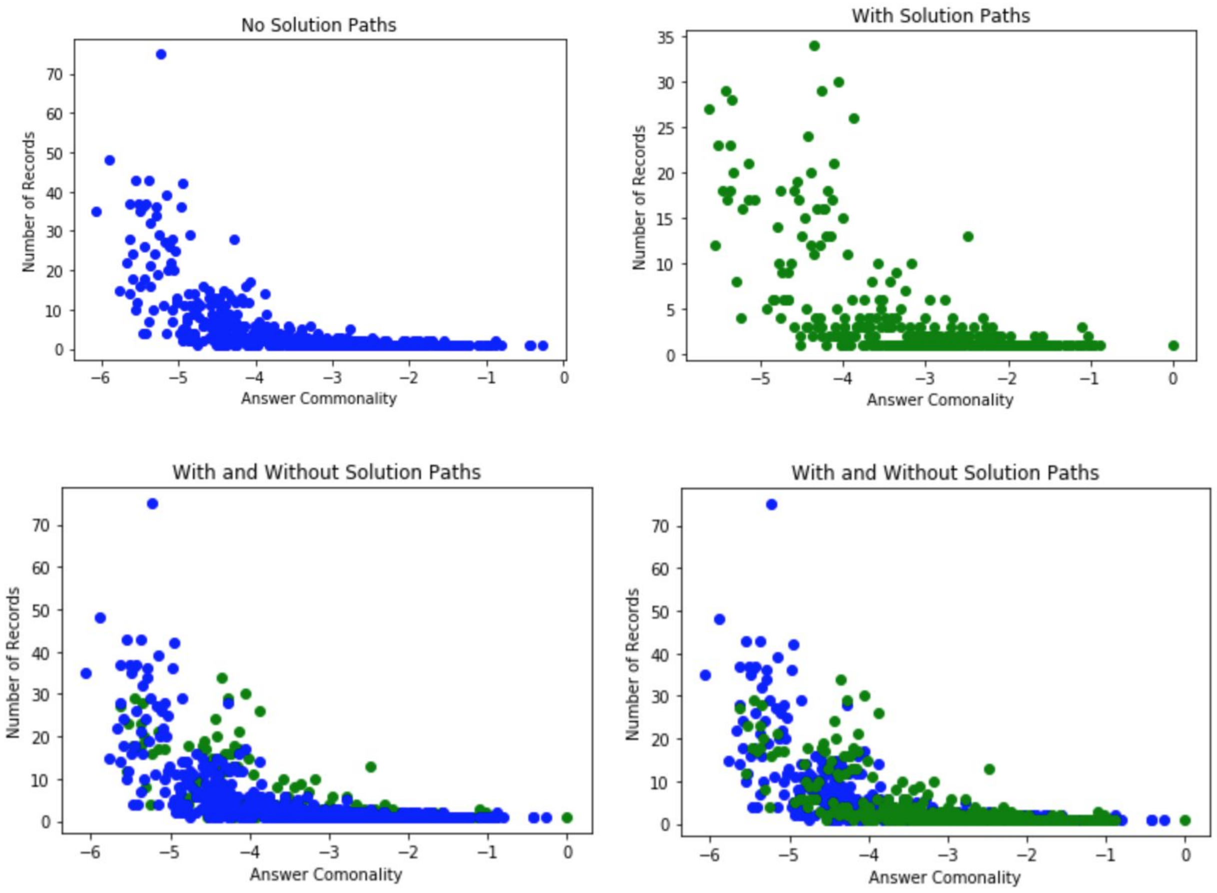


Figure 54. Assignment 7148. Generated vs Not Generated Solution Paths

When displayed on the same graph, the two record types overlap to the extent of, when one type is plotted before the other, the second type that is plotted almost completely covers the first. Therefore, the hypothesis is proven false. A deeper observation into actual values needs to occur to conclude why solutions paths could not be generated for common answers.

Child ID	Problem Text	Solution Path	Correct Answer	Possible Solution Path	Incorrect Answer	Answer Commonality
33015	38 - 8 x 28	a - b * c	-186	Typographic mistake	-168	0.000000
33017	26 - 17 x 19	a - b * c	-297	???	-303	-0.693147
33020	18 - 46 x 43	a - b * c	-1960	Typographic mistake	-1969	0.000000

33024	45 - 4 x 2	a - b * c	37	Off by one/ Typographic mistake	38	-0.916291
33024	45 - 4 x 2	a - b * c	37	Typographic mistake	27	-1.609438
33024	45 - 4 x 2	a - b * c	37	Typographic mistake	3	-1.609438
33024	45 - 4 x 2	a - b * c	37	Off by one/ Typographic mistake	36	-1.609438

Figure 55. Assignment 5946. Answers with No Solution Paths

From this sample of records with no solution paths, there are a lot of, what appear to be, typographic mistakes. In problem 33024, a student could have assumed they typed up an answer correctly, and hit submit without double checking their answer. This error is rare, but because a low number of students answered that problem, the answer probabilities score was high. It is by accident that the answer commonality marked this student response as normal. On of the theoretical benefits of stability measurement by observation of how the student got to the wrong answer is its ability to catch cases such as this. However, in practice, the current machine learning model misses a portion of normal answers.

In problem 33017, the given student response is one out of two. It appears odd, but since that problem has little data, the answer commonality method marked it as normal.

It is uncertain why the algorithm could not generate solution paths to problems 33024 and 33024 since they could be considered as off by one errors. Off by one errors, by the algorithm are marked as b/b + [correct solution path] or b/b - [correct solution path].

Child ID	**Problem text**	**Solution Path**	**Correct Answer**	**Possible Solution Path**	**Incorrect Answer**	**Answer Commonality**
131021	6c + 11 = -1	(c - b) / a	-2	(c + b) / a	1.666	-2.484907
49697	c/7 + 2 = -1	(c - b) * a	21	(c - b)	-3	-0.934309
49646	c/4 + 5 = 6	(c - b) * a	4	(c - b)	1	-1.090548

49695	x/9 + 4 = 3	(c - b) * a	-9	(c - b)	-1	-2.036882
49649	y/-2 + 7 = 1	(c - b) * a	12	-(c - b) / a	-3	-2.881069

Figure 56. Assignment 7148. Answers with No Solution Paths

Understandably, the solution path to the answer 1.666 of problem 131021 could not be generated. This is because the algorithm matches calculated answer exactly to the incorrect answer. Because of the differing rounding points were used, the solution path could not be found.

It is unclear as to why the answers to other problems did not have solutions since the possible solution path is even simpler than the correct solution path.

Impact of Number of Students on Correlation Magnitude

There exists a moderate correlation between the solution path and the answer commonality. However, these values were only computed when there approximately 1000 students who completed an assignment (e.g., 1236 students for Assignment 5946, and 1394 students for Assignment 7148). One question is how few students are needed to compute answer commonality accurately: are 1000 required or could we get by with fewer? Finding out the number of students needed to accurately compute answer commonality would inform us as to when to use each technique.

To determine the number of students needed, we experimented with ranges of 15.6 for Assignment 5946 and 18.85 for Assignment 7148, up to 100% of the data set, with 80 values in total tested. Since the exact students used are randomly selected, we ran 20 trials for each number of students. The trials conducted recorded the furthest from the best correlation during the twenty trials. When comparing the commonality approach, the best correlation for Assignment 5946 was the transformation commonality at 0.492 and for Assignment 7148 was the solution path commonality at 0.552. When comparing the edit distance approach, the best correlation -0.447 for Assignment 5946 and -0.387 for Assignment 7148.

The transformation and solution path commonality approaches were calculated within each trial sample, such that the commonality value was not influenced by students outside of the sample. The edit distance on the other hand, was calculated prior to the sample generation because the edit distance is not influenced by other students. Both of the methods were done to observe which method was more static against the change in student numbers.

Key:

🟦 Furthest Transformation Commonality

🟩 Furthest Solution Path Commonality

🟨 Furthest Edit Distance

🟥 True Correlation

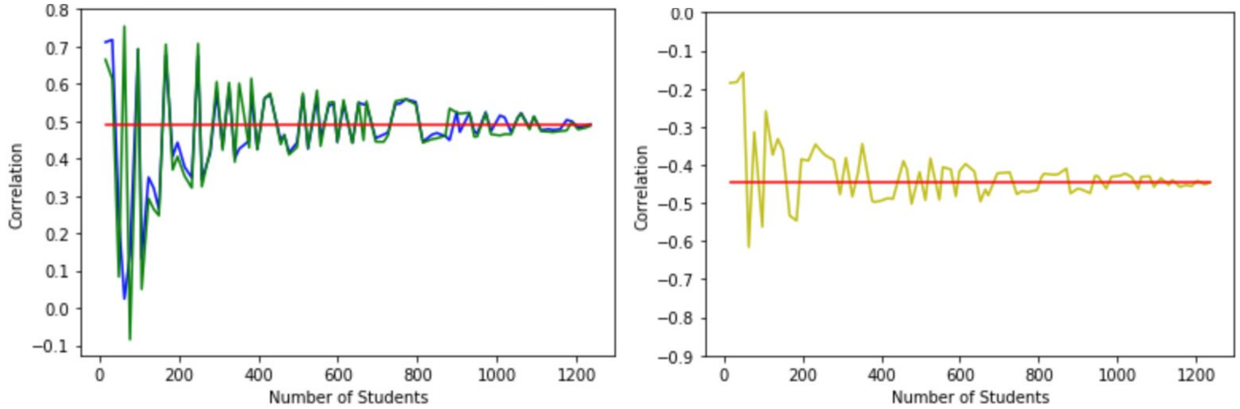


Figure 57. Assignment 5946. Impact of number of students on Correlation

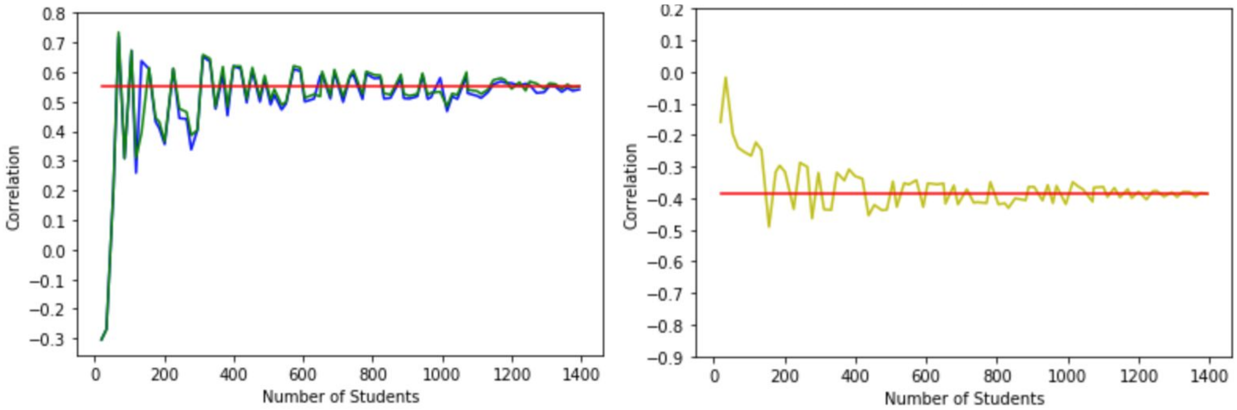


Figure 58. Assignment 7148. Number of Students Effect on Correlation

While the commonality features yield better correlation when the number of students is high, it is more susceptible to change as the number of students decreases. While different commonality features provide higher correlation for different assignment, they follow a similar pattern for worst case scenarios. Therefore, they could easily be interchanged for one another. Since the calculation of transformation commonality requires the computation of edit distance, and then the commonality value, it is computationally more expensive than the solution path commonality. When dealing with large number of students, edit distance does not need to be calculated as the solution commonality should provide high enough of a result.

From Figure 57, the commonality features start to differ by more than 0.1 around 400 students, while the edit distance that occurs around 200 students. From Figure 58, the commonality features start to differ by more than 0.1 around 300 students, while the edit distance that occurs around 150 students. For both of the assignments, the edit distance is more stable than the commonality features. The commonality features share a similar weakness with the answer commonality: when there is little data, the commonality value begins to signify less. If an answer is odd, but it is the only answer to a problem, it is marked as normal; if a solution path is odd, but there are little of other solution paths to compare to, it is also marked as normal. The edit distance is independent of other students and is therefore a better substitute for answer commonality for lower number of students. The threshold value for when the commonality approach becomes worse than the edit distance is around 300 students.

Future Work

For a more in depth research into the findings, certain characteristics of the approach can be improved upon. The solution path algorithm, on top of being slow to process, was also unable to process more than half of the existing answers. Steps could be taken to assure that the two weaknesses do not make as much of an impact as they do with the current algorithm.

To address the issue of speed, the algorithm's depth of search could be limited. Currently the algorithm has decided that a solution path of (((b / b) + (((c / a) - (b / a)) + ((c / a) - (b / a)))) + ((c / a) - (b / a))) is the best candidate for an answer of -11 to question 49607 from Assignment 7148: solve for x when 8x + 2 = -30. The solution path asks to perform a series of steps that do not make much sense in the context of the problem. However, this solution path applies to seven other answers, and was therefore chosen as optimal. A simpler solution could be the accidental reversal of added numbers: (-30 + 8) / 2. While this proposed solution path looks odd, it is significantly shorter and can be described in words. By dropping the long solution paths, shorter solution paths such as the proposed one might be chosen as the best candidate for the answer. On the other hand, if the answer were to not have a solution path, it would be added to the already long list of answers for which solution paths could not be generated. This, however, is not a negative outcome. An odd answer for which no solution path could be generated, is theoretically, stranger than the answers for which solution paths could be generated. Currently, the Answers Without Solution Paths section, not only has odd answers, but also more expected ones. If the number of the expected answers were to decrease, then increasing the number of answers without solution paths should not be information costly.

Considering a fewer set of operations to narrow search can also help decrease the time spent on generating possible solution paths. Currently, all operations are fed into the algorithm and allowed to be used in the formulation of solution paths. While it is necessary to allow for an easy path in case a student divided where he should have multiplied, this grows to be less and less expected deeper into the solution path generation. Therefore, first and foremost, the solution path to the correct answer must be generated. The operations from each of those solution paths, should be prioritized more when constructing solution paths for incorrect answers. The operations not included in the correct solution path should either be removed in the deeper search, or not prioritized. A likelihood estimates for each operation can be added to assess different next operation costs within the search, thereby limiting the next set of operations.

To decrease the number of expected answers from the answers without solution paths, new operators should be added. These operators will aim to generate mistakes faster and will only be used once per solution path. Some of the answers include a negative of the correct answer, or a negative of a common wrong answer. Currently, the algorithm solved those problems by attaching ((b - b) / b) to the beginning and subtracting the actual solution path from it. This generation, however, can become faster if a new operation *(-1) were to be added to the algorithm. Another set of answers has an off by one error, which is solved by a (a / a) +/- [rest of the solution path]. The current approach is long, and as shown in Figure 56, occasionally does not capture all of the off by one errors. A new operation to shorten the solution is +/- 1. This operation will allow for the capture of expected answers that are mistakenly grouped by the algorithm as not possible to have a solution path for.

A couple of template solutions could also be generated and applied to problems for a faster search. If an assignment tests for knowledge of PEMDAS, the algorithm can be programmed to explicitly look for specific problems, such as a PEMDAS error, first. However, this approach limits the independence of the algorithm to work by itself, and should therefore be used if the assignments being observed test a common set of skills.

Since the current algorithm looks at the raw answers, attempting to generate a solution path to it, typographic mistakes are missed. Figure 55 shows the potential typographic mistakes: off by one left or right keys on keyboard, and accidental reversal of keys when dealing with answers with more than two characters (i.e. 42 instead of 24). The algorithm could be improved upon by allowing it to specifically catch typographic mistakes.

By decreasing the length of solution paths and the number of expected answers without solution paths, the algorithm can be made faster and more precise. Addition of new operations, a more intelligent next level search and the decrease in the search space can contribute to achieving that improvement. Methods such as Approach Maps can be used to optimize path generation for the open ended logical problems by intelligently filtering out unproductive regions that cannot lead to the correct solution path.[12]

The work on correlating solution path data to the answer commonality can be expanded upon to include correct answers. If answer commonality were to account for the frequencies of correct answers, the difficulty of the problem would be included within the answer commonality. This will allow for the analysis of easy problems not having many incorrect answers, yet also giving leniency to difficult problems with many mistakes.

Predicting Student Behavior

With the techniques to represent student answers discussed, the next step is to incorporating this knowledge as well as other aspects of student behavior in a predictive model. The predictive model can then be used to describe student competency. To accomplish this goal, a Neural Network will be utilized.

Deep Neural Network

Modeled after biological neural circuits within the brain, a Neural Network is a learning system that can approximate mathematical functions. The Neural Network is comprised of neuron-like nodes that are interconnected with individually weighted links that symbolize a strength in connection between the nodes. A simple Neural Network is composed of two layers, the input and output layers, where each layer is composed of nodes. A complexity can be added to a Neural Network by adding one or more extra layers, called hidden layers, between the input and output, making a Deep Neural Network.

A Neural Network consumes the input features, such that each feature is associated with one node within the input layer. For each node, the feature value gets individually multiplied by all of the links and sent to the appropriately connected next nodes. From then on, each node calculates the weighted sum of all of its inputs, adds a bias, and sends the value through an activation function that transforms the value and decides whether the node fires to its linked nodes or not. This process continues until the output layer.

A Neural Network can be trained in a supervised, or unsupervised manner. For supervised learning, the desired output, or target, is known, while for unsupervised, there is no known outcome and the model learns to group similar elements and find patterns. For the prediction of student competency, the supervised learning method will be used.

[12] Michael Eagle and Tiffany Barnes, Exploring Differences in Problem Solving with Data-Driven Approach Maps (International Conference on Educational Data Mining, 2014).

Initially, all weights within a network are assigned randomly. Once the input is fed into the network, they are multiplied by weights, and activated by nodes deeper in the network, to the output layer. The result of the output layer is compared to the known target and an error is computed. The error is then back propagated into the network, and each node gets assigned a blame for the error. The weights are adjusted accordingly, and a new epoch begins.

Long Short Term Memory Neural Network

A typical Neural Network does not maintain information from previous outputs. When looking at student behavior, however, knowing the student's past performance can be vital information in predicting the future. Recurrent Neural Networks (RNN) allow for information persistence through the loops comprised within the network. A Long Short Term Memory Neural Network (LSTM), is a type of an RNN that is capable of remembering dependencies from arbitrary lengths of time ago.

Process

The first step to building a Neural Network is to acquire the data to be analyzed. Upon gathering the data, it must go through an extraction process where key features are calculated from the existing information. This transformed data can then be fed into a Neural Network and activated to begin training. Parameters of the network can then be modified for a more accurate model.

The Data

Taken from the ASSISTments database of the 2016 to 2017 school year, the data contains skill builder assignments starting from October 1st, 2016 and ending before June 1st, 2017. To ensure that each assignment is not optional and has a non-trivial amount of data associated with it, assignments with less than ten students and with less than 70% completion rate were excluded. The data consists of different levels of granularity. At assignment-level, features such as student mastery, and number of problems started are known. At problem-level, features such as problem correctness, number of hints used, and number of attempts are known. At action-level, features such as action type, and action time are known. An action can be one of nine unique types, however only answer, hint, answerhint, and scaffold were used: an answer is an attempt at a problem; a hint type is an use of a hint; an answerhint is a bottom-out hint, which is a hint that reveals the answer to the problem; a scaffold is a split of a a question into various subproblems. Therefore, each row within this dataset is one action completed by a student when answering each problem. The original dataset has 2.35 million rows, but after the filtering of the data by the action type, the dataset has 1.06 million rows.

Existing Features

Feature Name	Description
Descriptive Features	
user_id	Student identifier
problem_id	Problem identifier
assignment_id	Assignment identifier
action_time	Timestamp of the action
Action Level Features	
action_name	The type of action taken: answer, hint, answerhint, and scaffold. The feature is one-hot encoded when used as an input to the Neural Network.
correct	The correctness of the attempt: correct, incorrect, not an answer action. The feature is one-hot encoded when used as an input to the Neural Network.

Figure 59. Existing Features Table

Calculated Features

Previous work has been done into incorporating the existing features from the ASSISTments database into a student competency model. The article Incorporating Rich Features into Deep Knowledge Tracing, improved upon previous work by expanding the complexity of problem level features. Successfully, the AUC score went up from 0.831 to 0.858.[13] Therefore, better accuracy can be achieved through the calculation of new features. Therefore, to expand upon prior work, the dataset processed is not only more granular (is on action level as opposed to only being problem level), but will also contain more calculated features.

Problem level information can shape the steps taken during feature generation from the raw data. For more effective probability related calculations, there must exist a large enough number of

[13] Anthony Botelho et al.,
Incorporating Rich Features into Deep Knowledge Tracing (Cambridge, Worcester Polytechnic Institute, 2017), 4.

students to base the probabilities on. The threshold number for a good probability was picked to be 100 students.

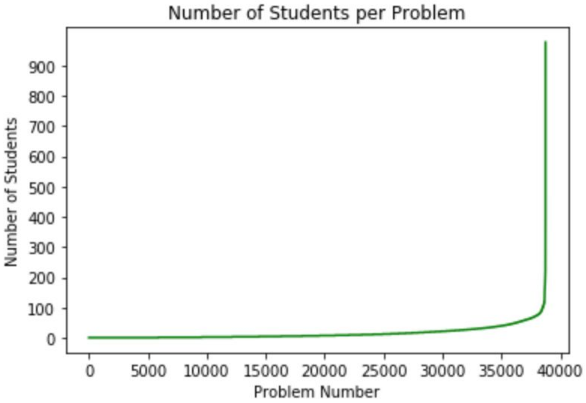


Figure 60. Number of Students per Problem

Problem level count of unique students revealed a low number of students per problem, such that 5884 problems had only one student answer. When applying the threshold, 247 problems out of 38,745 had more than or equal to 100 students answer them, this is 0.64 % of all problems. This makes up, assuming that each student completed a certain problem only once, 35,183 unique for a problem students out of 584,031 students, accounting for only 6.02 %. This reveals that there are a significant amount of problems with little students completing them. A cause for these low numbers is the existence of template problems. Because the problems are generated from a template and randomly assigned to a student, it is unlikely that students will be solving the same problem.

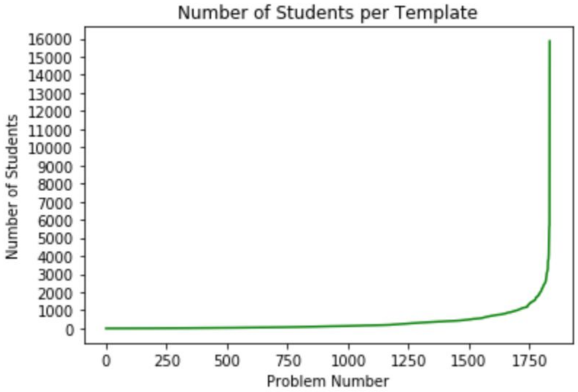


Figure 61. Number of Students per Template

Looking at template level reveals that when applying the 100 student threshold, 30,780 problems out of 38,745 this is 79.44 % of all problems. The number of student for template level analysis significantly increase: 553,737 out of 584,031 unique per problem students, or 94.81 %, are captured. Therefore, when possible it is important to generalize problem level information to the template level information.

From the raw features collected from the dataset, the following features were computed for each row.

Feature Name	Description
Probability Features	
probability_action	The probability of an action to occur in a template.
probability_action_action_count	The probability of an action to occur in a template given the action count. If the action is first, the action is compared to all other first actions. If the action is not first, the action is compared to all other actions that were not first.
probability_answer	The probability of an answer to occur in a problem. The value is 0 if the action is not an answer attempt.
probability_answer_action_count	The probability of an answer to occur in a problem given the action count. If the action is first, the action is compared to all other first actions. If the action is not first, the action is compared to all other actions that were not first. The value is 0 if the action is not an answer attempt.
log_likelihood_cumulative_answer	Cumulative log-likelihood of answer probabilities. With each new answer action made, the value gets updated by adding the log of the answer's probability to the previous log_likelihood_cumulative_answer value.
Time Features	
normalized_time	Time z-scored across templates
Previous Actions Transformation Features	
previous_3_actions	A recording of the three previous actions taken in a problem. A value of null is utilized for every previous action that does not exist. An sample of this feature would look like 'null_null_answer.' The feature is one-hot encoded when used as an input to the Neural Network.

current_and_past_2_actions	A recording of the two previous actions and the current action taken in a problem. A value of null is utilized for every previous action that does not exist. An sample of this feature would look like ‘null_answer_answer.’ The feature is one-hot encoded when used as an input to the Neural Network.
used_penultimate_hint	Indicates cumulative use of a penultimate, or second to last, hint in a problem. The value for this feature is 0 until a penultimate hint is used, and 1 for all rows in a problem on and after a penultimate hint is used.
used_bottom_out_hint	Indicates cumulative use of a bottom-out hint in a problem. The value for this feature is 0 until a bottom-out hint is used, and 1 for all rows in a problem on and after a bottom-out hint is used.
attempt_count	Indicates cumulative use of an attempt in a problem. The value for this feature is 0 until an answer attempt was made, and it is incremented by 1 for all rows in a problem on and after an attempt is made.
hint_count	Indicates cumulative use of a hint in a problem. The value for this feature is 0 until an hint was requested, and it is incremented by 1 for all rows in a problem on and after a hint is requested.
problem_count	Indicates cumulative number of problems completed in an assignment. The value for this feature is 1 until a new problem is started, and it is incremented by 1 for all rows in an assignment on and after new problem is started.

Figure 62. Calculated Features Table

Student response oddity will be measured with the four probability and the log-likelihood feature. Other features will be used to assure that the LSTM would transfer time related information, from one record to another.

Output Features

To model student competency, two output features were selected: current assignment wheelspin and current assignment stopout. Current assignment wheelspin indicates whether a student mastered the assignment they are currently working on inclusively within ten problems. A value of 1 indicates wheelspin on the current assignment, and a value of 0 indicates no wheelspin. Current assignment stopount indicates whether a student quit working on the assignment before completing ten problems. A value of 1 indicates stopount on the current assignment, and a value of 0 indicates no stopount.

Establishing Model Success

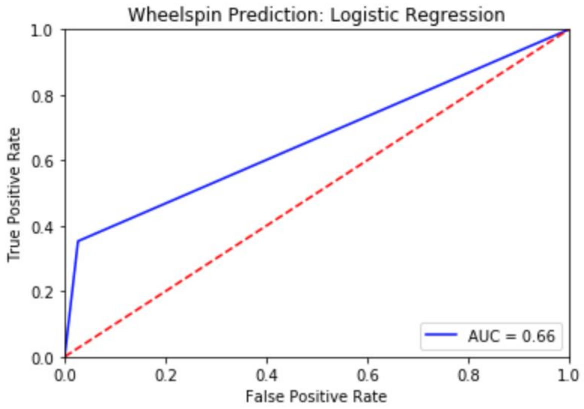
	Value: 1	Value: 0
Wheelspin	809985	245603
Stopout	997679	57909

Figure 63. Output Features Class Distribution

There exists an imbalanced class distribution among the output features. In this case, a simple evaluation of model accuracy proves to be a wrong approach as a model that always predicts wheelspin would have a high accuracy of $809985 / (809985 + 245603) * 100 = 76.73\%$, and an accuracy of $997679 / (997679 + 57909) * 100 = 94.51\%$ for stopout.

For a more meaningful evaluation of the model, two metrics will be used: AUC and Kappa. AUC is a measurement of the area under the ROC curve, which is defined by the model's True Positive and False Positive Rates. The True Positive Rate is the rate at which the model correctly classified an outcome while False Positive Rate is the rate at which the model predicted an outcome incorrectly. Kappa measures how well the model classifies, taking into account class distributions.

To establish a baseline AUC values for the given dataset to compare the LSTM network to, three machine learning methods were conducted: Logistic Regression. Naive Bayes, and Decision Tree. A five fold cross validation was constructed such that a single user belonged to only one fold.



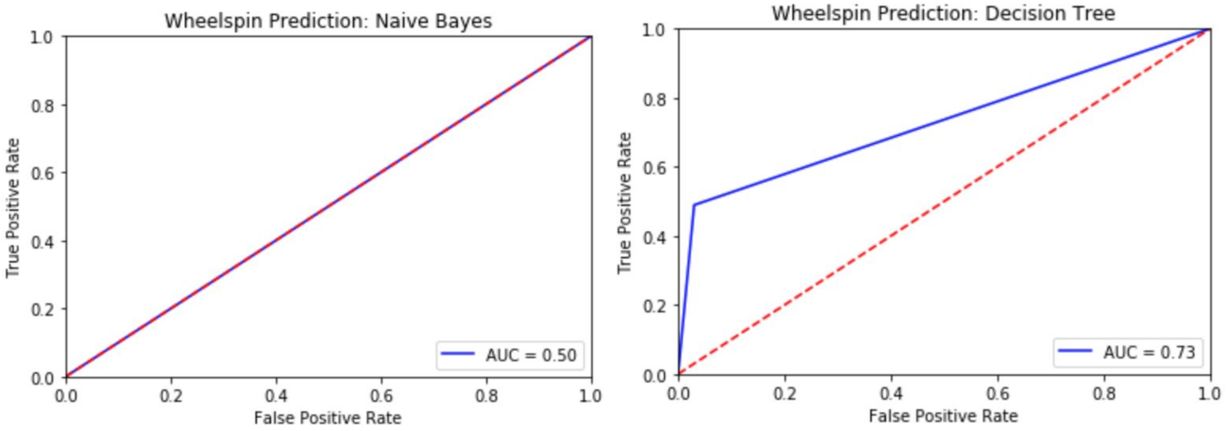
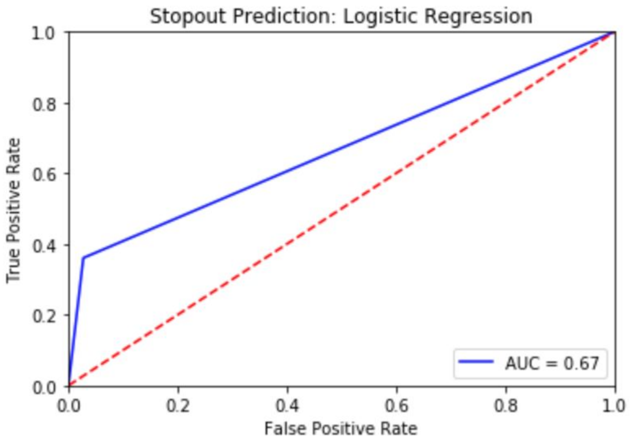


Figure 64. ROC Curves of Machine Learning Models in Predicting Wheelspin

While the highest value of AUC stands at 0.73 with the Decision Tree model, the Naive Bayes model weighed the more abundant class higher, and began predicting only one class as the output. Since a Naive Bayes model assumes that all features are independent from one another, the algorithm fails to produce accurate results when the data is dependent. Since feature selection techniques were not used on the dataset, for a more accurate result, independent input features need to be preselected prior to training.



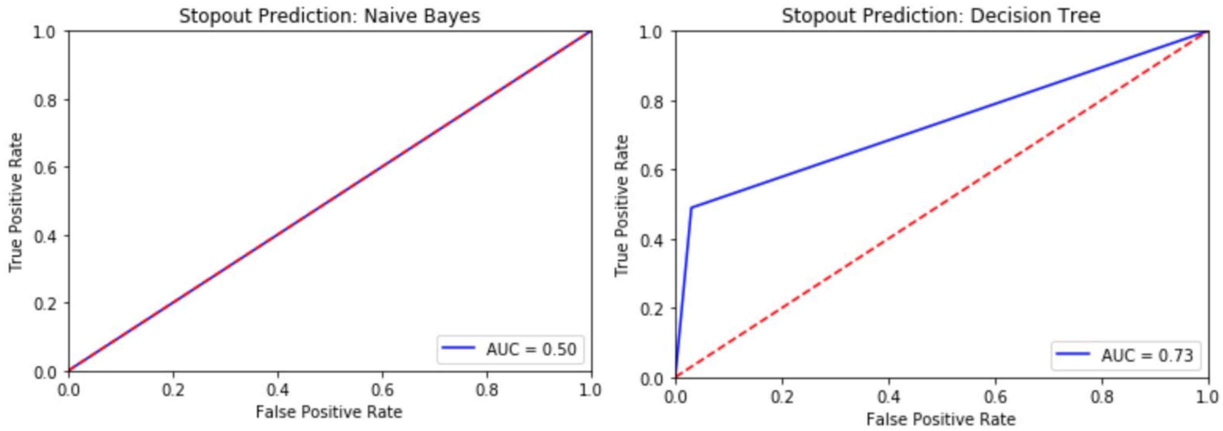


Figure 65. ROC Curves of Machine Learning Models in Predicting Stopout

Similarly to Wheelspin, maximum Stopout AUC is 0.73 with the Decision Tree and the Naive Bayes model does not perform well on this imbalanced dataset. Similarly to wheelspin, for a higher Naive Bayes score a preprocessing step of feature selection needs to be done.

The success of the LSTM can be measured by its perform against the Decision Tree model.

Network Structure

Both current assignment wheelspin and stopout output features were trained on the same model structure. This was done as an expansion on the article Modeling Student Competence: a Deep Learning Approach, that argued that different output features extract from the same descriptor: student competency.[14] Therefore, it becomes more efficient to maintain a single model for multiple outcomes. To ensure a competitive AUC value with the simple machine learning models, the LSTM network consists of a total of six layers, four of which are hidden.

Input Layer: 71 Neurons
Hidden LSTM Layer 1: 64 Neurons and leaky_relu as the activation function
Hidden LSTM Layer 2: 32 Neurons and leaky_relu as the activation function
Hidden LSTM Layer 3: 16 Neurons and leaky_relu as the activation function
Hidden LSTM Layer 4: 4 Neurons and leaky_relu as the activation function
Output Layer: 1 Neuron, a dropout percentage of 0.5 and sigmoid as the activation function.

Softmax cross entropy with logits, an error function for mutually exclusive outputs, is used as the cost method for learning the network weights.

[14] Modeling Student Competence: a Deep Learning Approach, 1

Model Accuracy

When training a model, the input features are fed through the network. An epoch consists of the forward and backward propagation of the entirety of the training data once through the network. With each new epoch, the weights in the network are better adjusted to fit the training. A problem of overfitting can occur in training when the model begins to represent the training data too closely such that it cannot be generalized to the testing data. Underfitting, can too occur when a network does not model the training nor the testing data. While more epochs ensures a higher training score, a problem of overfitting can occur; too little epochs can cause underfitting. To ensure that the LSTM model performs at its peak, both training and testing accuracies were observed.

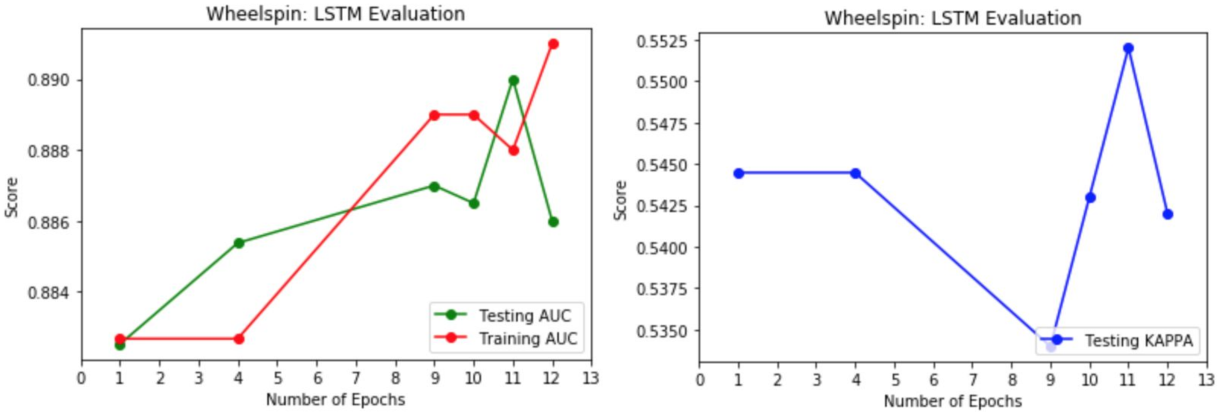


Figure 66. LSTM Evaluation: Wheelspin

When the maximum epoch is set to one, the training performs better than test, and the network can benefit from a longer training time. By the second epoch, the test begins to continuously increase and perform better than the training dataset. When the training overtakes test results, the network begins to overfit the training data and should therefore not be trained for so long. When the maximum epoch is set to 20, the model does not train for more than 12 epochs. The KAPPA score resembles the testing data AUC graph. And, while both peak at 11 epochs, that area is unstable and performs worse than training more often than not.

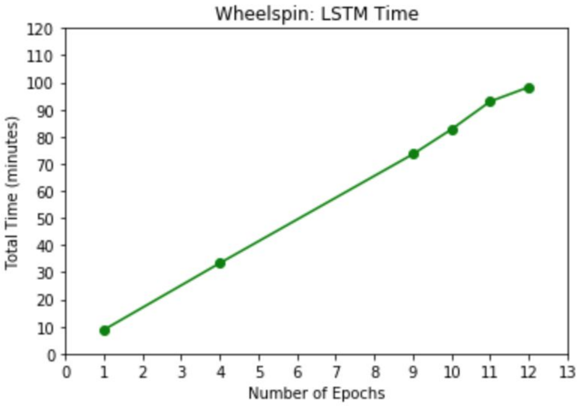


Figure 67. LSTM Time: Wheelspin

Despite the increase in the number of epochs, the testing AUC increases only at a slight rate. However, the time taken for the model to learn is a linear function where more epochs implies a longer time. Here, a tradeoff of slightly better accuracy and a significant increase in time can be observed. However, because the network begins to overfit around the seventh epoch, it will be more time and accuracy efficient to cut off the network before the seventh epoch.

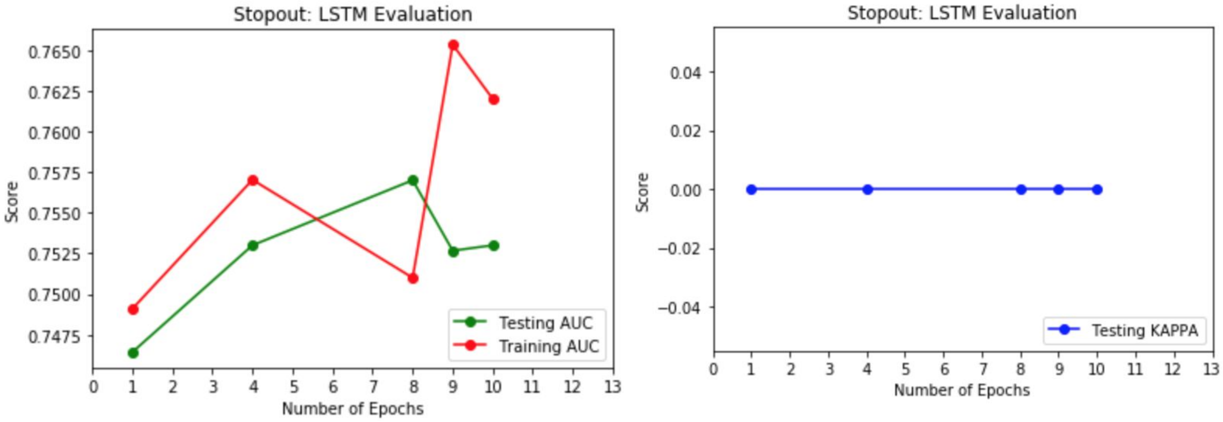


Figure 68. LSTM Evaluation: Stopout

From the beginning, the training data AUC score is higher and generally stays higher than it is for the test data. With that knowledge and the fact that the KAPPA score is always 0, signifies that the model is underfitting and cannot represent the data well. Therefore, the same network structure cannot represent both assignment metrics as well as it can just one of the features.

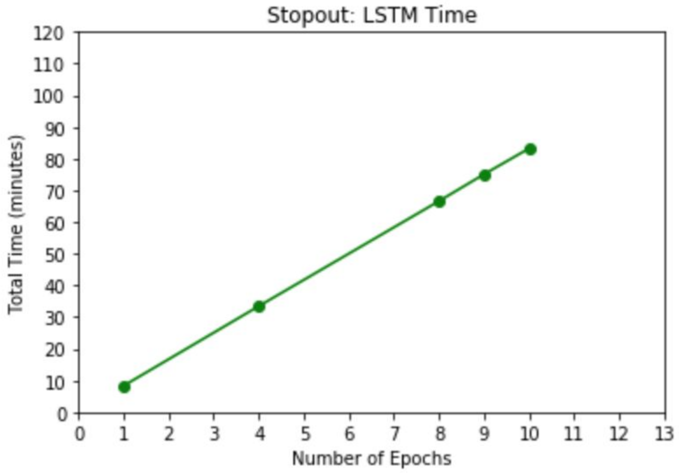


Figure 69. LSTM Time: Stopout

Once again the time function between each epoch is linear. Despite the non optimal model, the AUC score still continues increase as the number of epochs increase, until the ninth epoch. Therefore, a tradeoff between better AUC score and time needs to occur when building the final model.

Analysis

For both the wheelspin and stopout models, the last LSTM layer should construct something about student behavior that aids in predicting the outcomes. Since the network structure is the same, but the weights are different, the generalizability of each model can be tested by letting it predict the other outcome. Therefore, for both of the models, the training output of the fourth hidden layer was retrieved, flattened and fed to train the second model to predict the alternative outcome. Similarly, the test output was taken from the fourth hidden layer, flattened and sent to the prediction function of the second model. As a summary, the first model, referred to as the original model, takes in the input features and predicts either current assignment wheelspin or stopout. The model is trained on four epochs as this number of epochs does not sacrifice the AUC score but saves time. After flattening the output sequence from the original model, the shape of (843384, 4) is feed into the second model, referred to as the generalized model, for prediction of an alternative output feature.

Two models were tested as the generalization model: Decision Tree and Sequence of Dense Layers. The two models were evaluated by observing their capacity to pick up the output of the original model to predict both outcomes. In other words, for a original wheelspin model, the second model was trained twice: once for the wheelspin outcome and once for the stopout outcome. This was done to assess whether the model can learn by its ability to match the AUC score of the original model.

Decision Tree

In the Establishing Model Success section, out of the three simple models, the Decision Tree model came out superior in AUC evaluation of the imbalanced classes within the dataset. Therefore, it is used as the best representation of the simple models.

Experiment Number	Input Features	Model	Predicting	AUC
One	Original Features	Decision Tree	Wheelspin	0.730
Two	Original Features	LSTM	Wheelspin	0.885
Three	LSTM-Wheelspin Hidden Layer	Decision Tree	Wheelspin	0.755
Four	LSTM-Stopout Hidden Layer	Decision Tree	Wheelspin	0.544
Five	Original Features	Decision Tree	Stopout	0.730
Six	Original Features	LSTM	Stopout	0.753
Seven	LSTM-Wheelspin Hidden Layer	Decision Tree	Stopout	0.500
Eight	LSTM-Stopout Hidden Layer	Decision Tree	Stopout	0.500

Figure 70. Generalizing Success Metrics: Decision Tree

The model fails to make sense of the hidden layer output and predicting its intended output feature. This can be seen in the big difference between experiment two and three for wheelspin and six and eight for stopout. Furthermore, not only does the Decision Tree Model not express its original outcome well, it also does not learn the alternative outcome at all. Experiment seven has an AUC of 0.500, implying that the generalization was not successful in building a model that can learn. Similarly, in experiment four, when a stopout model is attempting to generalize wheelspin, the AUC is a low 0.544. Interestingly, from the comparison between experiment one and three for wheelspin, the Neural Network extracted features that helped the Decision Tree improve by 0.025. A simple machine learning model cannot extend the outcome of the hidden layer to express something more meaningful.

Sequence of Dense Layers

For a more complex model, two dense layers, each followed by a dropout layer were trained.

Input: Array of shape (843384, 4)
Dense Layer: 64 Neurons and relu as the activation function
Dropout Layer: keeping 0.5
Dense Layer: 64 Neurons and lrelu as the activation function
Dropout Layer: keeping 0.5
Dense Layer: 1 Neurons and sigmoid as the activation function

Binary cross entropy is used as the cost method for learning the network weights with rmspop as the optimizer.

Experiment Number	Input Features	Model	Predicting	AUC
One	Original Features	LSTM	Wheelspin	0.885
Two	LSTM-Wheelspin Hidden Layer	Dense Layers	Wheelspin	0.886
Three	LSTM-Stopout Hidden Layer	Dense Layers	Wheelspin	0.685
Four	Original Features	LSTM	Stopout	0.753
Five	LSTM-Wheelspin Hidden Layer	Dense Layers	Stopout	0.704
Six	LSTM-Stopout Hidden Layer	Dense Layers	Stopout	0.750

Figure 71. Generalizing Success Metrics: Decision Tree

The model does well in picking up the hidden layer output and predicting its intended output feature. This can be seen in the improvement from the Decision Tree of the AUC difference between experiment one and two for wheelspin and four and six for stopout. In fact, the wheelspin hidden layer (experiment two) slightly improved the AUC score from the raw features of experiment one. Despite the improvements, the stopout input generalizing to wheelspin still loses greatly from 0.885 in experiment one to 0.685 in experiment three. This could be due to the superiority of wheelspining as an output to generalize stopout or due to the significantly higher AUC score of the original wheelspin model. The success of hidden layer output to generalize to alternative outputs implies that its content models student competency, and that the constructed features were descriptive enough to allow for such a conclusion.

Future Work

In the dataset analyzed, only 0.64% of all problems have more than or equal to 100 students worth of data. The low number of students per problem could be circumvented when calculating the probabilities of actions by generalizing the feature by the problem template. However, the same tactic cannot be used when measuring probabilities of answers, as each answer is unique to the context of the problem. With the majority of problems not having enough data, the answer probability features (probability_answer, probability_answer_action_count, log_likelihood_cumulative_answer) could be more skewed than helpful. To combat this problem, more data can be collected, from years prior to the 2016-2017 school year. With more data, there will be more students answering problems, thereby ensuring that answer probability features are more accurate. Alternatively, more time could be invested on Douglas Selent's solution path generation to be able to extend to problems for which there are little student data, and substitute for answer probability features.

While the wheelspin model significantly outperforms the simple machine learning techniques, the deep stopout model is only marginally better than the Decision Tree. Therefore, future work can be conducted on improving the base LSTM model for predicting current assignment stopout. Furthermore, the concept of wheelspin and stopout can be extended to new LSTM network structures in predicting next assignment wheelspin and stopout. Each of those models could then go through a process of generalization to observe if there exists a set of student behaviors expressed in the last LSTM layers of each model that can determine outcomes other than the ones that the model was initially trained on.

Conclusion

This MQP evaluated and processed raw data of students answering questions from the ASSISTments database. Data analysis techniques were applied on incorrect student answers to determine their representation. Each having strengths and weaknesses, answer probability and the solution path methods were correlated to determine their high level of replaceability with the other. With this finding, the more fitted method, answer probability, for a large dataset spanning the school year of 2016-2017 was calculated along with other features. There features were used to train different machine learning models to predict current assignment wheelspin and stopout. The simple machine learning models established a base score to successfully tune a more complex, LSTM network. To ensure that the LSTM model learned to describe student competency, the output of the last hidden LSTM layer was fed into another model whose aim was to generalize the competency metrics: wheelspin and stopout. The original wheelspin model performed better in generalizing stopout than the original stopout model was at generalizing wheelspin, losing only a slight bit of accuracy over the original stopout model.

References

Barnes, Tiffany and Michael Eagle. *Exploring Differences in Problem Solving with Data-Driven Approach Maps*. International Conference on Educational Data Mining, 2014

Botelho, Anthony, Neil T. Heffernan, Xiaolu Xiong, Liang Zhang and Siyuan Zhao. *Incorporating Rich Features into Deep Knowledge Tracing.* Cambridge: Worcester Polytechnic Institute, 2017.

Heffernan, Neil T. , Korinn S. Ostrow, and Yan Wang. *How Flexible is Your Data? A Comparative Analysis of Scoring Methodologies across Learning Platforms in the Context of Group Differentiation*. Journal of Learning Analytics, 2017.

Modeling Student Competence: a Deep Learning Approach.

Selent, Douglas. *Creating Systems and Applying Large-Scale Methods to Improve Student Remediation in Online Tutoring Systems in Real-time and at Scale.* Worcester Polytechnic Institute, 2017.