A shorter version of this paper is to be published in Ashley & Ikeda (Eds.) *Proceedings of the 8th International Conference on Intelligent Tutoring Systems*. Berlin: Springer-Verlag. (2006) **http://www.its2006.org/**

Title:
Knowledge Engineering for Intelligent Tutoring Systems: Using machine learning assistance to help humans tag questions to skills based upon the words in the questions

Authors:
Kevin Kardian & Neil T. Heffernan III
Worcester Polytechnic Institute, Worcester, MA 01602 USA
nth@wpi.edu
(508) 831-5569

Abstract:
Building a mapping between items and their related knowledge components, while difficult and time consuming, is central to the task of developing affective intelligent tutoring systems. Improving performance on this task by creating a semi-automatic skill encoding system would facilitate the development of such systems. The goal of this project is to explore techniques involved in text classification to the end of improving the time required to correctly tag items with their associated skills.

1 Introduction

One of the more difficult problems for creating intelligent tutoring systems has to do the knowledge engineering for a given domain. Some intelligent tutoring systems have at their heart a matrix that maps their questions (called "items" in the psychometrics lingo) to a set of associated *knowledge components* (KCs) (we use the term "knowledge component" to emphasis that the tags might represent concepts, skills or strategies need to solve problem, and not simply procedural skills) (Koedinger et al, 2004; Barnes, Bitzer & Vouk, 2005). In the psychometrics community they call the matrix representation of this information, a *Q-matrix* (Tatsuoka, 1995), while Heffernan has used the term *Transfer Models* (Croteau, Heffernan & Koedinger, 2004) and they allow you to predict which items you would expect to see transfer of learning between. While there are many uses for these Q-matrices, building these matrixes is hard work that might be made easier by computer. In the ASSISTment Project (Razzaq, Feng, Nuzzo-Jones, Heffernan, Koedinger, et al, 2005), we have recently done three different 6-8 hour-long "coding sessions", in which two subject matter experts (one of whom was the second author) were given 200-400 items and asked to make up KC's and tag each item with up to 3 KC.[1] This procedure was done with paper cut-outs of all the items. When the session was

[1] This Q-Matrix representation is very simple, and we assume that a student must know all the skills associated with an item in order to get that item correct. For comparison, CTAT tags answers with skills, rather than simply tagging questions with skills. The CTAT representation is richer, and has its roots in the rule-based model tracing framework which allow for the tracing of different solution strategies. It is probably the case that this paper's results would also have value in the CTAT representation, but this paper is confined to talking about this simple representation and semantics of Q-Matrices.

over, there were 80-100 piles of items, and 20 hours of data-entry to build the Q-matrix. We want to put this whole process into the computer. Inspired by Rose et al (2005), the purpose of this paper investigate the possibility that the computer might be able to assist the author by suggesting what skills go with a given question by looking at the words in the question. Rose et al (2005) reported that "that even in cases where the predictions cannot be made with an adequate level of reliability, there are advantages to starting with automatic predictions and making corrections, in terms of reliability, validity, and speed of coding." We feel that having this assistance might also help in maintaining these matrices, as new items need to be added and coded to the system by human that might not be the original coders. Furthermore, such assistance *might* lead to more accurate Q-matrices as the system suggests a code that they human might have overlooked for a given item.

The goal of this project is to attempt to further investigate semi-automatic skill coding to an end of improving the time required to tag an item with one or more skills from a very large number of possible skills. This paper does not attempt to do any empirical analysis to measure coding time, and instead is first investigating the idea applying Rose et al. idea to our dataset. Specifically, this paper explores the accuracy of calculating several of the most likely skills, instead of only one. Furthermore, the accuracy associated with each skill individually is discussed in an attempt to gain a better understanding of what makes an item more or less difficult to classify. The worth associated with imposing a hierarchical model, beginning with a substantially less specific skill set as a basis for a more specific skill classification, is also investigated.

2 Motivation and Background

The US Dept of Education funded the ASSISTment project to build cognitively vaild diagnostic assessment systems, which tutor while they assess. For this, we need to tag thousands of items with a fine grained mapping. We use this mapping to report to teachers several of the top skills that their students need the most assistance on. For these reasons we want to build this tool that will help coders tag items with skills fast and more reliably. In Razzaq et al (2005) we report on the fact that we know students are learning from the computers, and that we can do a reasonable accurate job of assessing students.

3 Data set and General Methodology

The question texts that we used as data come from two sources. First, we started with about 280 released 8th grade math test items from the Massachusetts Comprehensive Assessments Systems (MCAS) state test. These items are avail on the Massachusetts's Department of Education web site.[2] The rest of the instances were questions that written at Worcester Polytechnic Institute and at Carnegie Mellon by graduate students as part of the assembly of a tutoring system for the test items. For each "original" MCAS item,

[2] http://www.doe.mass.edu/mcas/testitems.html

members of the ASSISTment Project (Razzaq et al, 2005) wrote between three and five of these "scaffolding" questions, which attempted to break down solving the item into solving a few easier questions.

Our original data had 1258 data instances, where many of the question text were tagged with more than one skill. Due to the difficulty of knowing how to evaluate our classifier, we decided to focus only on questions tagged with a single skill. This exclusion left us with 878 question text instances.

Each of the skills, the distribution of which is shown below in Figure 1, was used in all data calculations. Note that the skill with the highest number of occurrences, which happened to be named "Pattern Finding", made up less than 10% of the total number of instances, so we would hope to get classification accuracy at least higher than 10%.

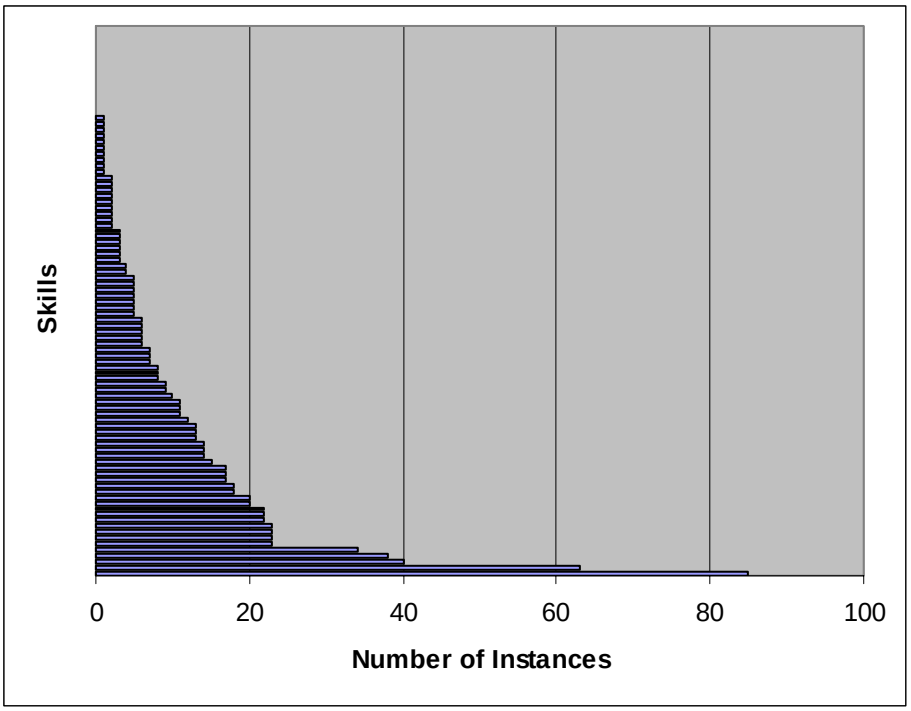

Figure 1: Distribution of Skills based on Number of Instances

Each instance was assigned one or more tags from the "April" transfer model, which contains 78 different skills. Each skill within the April model can also be mapped to exactly one skill from the MCAS5 transfer model, which contains 5 more general skills.

We used the Mallet toolkit (McCallum, 2002) to assist in text classification. Mallet includes several utilities for manipulating data and supports multiple text classification algorithms.

All trials were run using a NaïveBayes classification algorithm. This was chosen over the other algorithms suggested by Rose et al (2005) because neither VotedPerceptron nor SVM are supported by Mallet. Other algorithms were considered, but implementing a

hierarchical classification model proved simplest with NaiveBayes. In Mallet, the NaiveBayes trainer is implemented by splitting each question text into a feature vector, with one word per feature. Each feature is assigned a weight during training so that instances can be classified based on their comparison to the feature vectors derived from the training set.

There were a total of two separate experiments we report on to attempt to understand the difficulties in classifying questions based on question texts.

3.1 Experiment 1: Classification of question text into skills

The first was a study of the advantages associated with selecting more than one tag for a given instance. For this experiment, the items were classified using 90% of the data for the training set. The total data set was divided at random in every trial, and the accuracy was gathered for the top N choices for each instance in the testing set. Every test was run a total of five times, and the averages are reported below in table 1.

N	Accuracy	Standard Deviation
1	0.4096	0.0084
2	0.5194	0.0239
3	0.5663	0.0212
4	0.6005	0.0234
5	0.6369	0.0125
6	0.6738	0.0099
7	0.6875	0.0160
8	0.7098	0.0125
9	0.7130	0.0103
10	0.7303	0.0142

Table 1: Accuracies for Top N Choices: If we ask MALLET to pick one skill from the list of 78, 40% of the time it will pick the correct skill.

Though we are mainly interested in comparing improvements with different methods, the correct way to interpret this 41% accuracy is that this is the probability that you would classify any item correctly if you took that item at random from our data set. Note that this number is different, and higher, than the value you would expect to get if you first picked a skill at random and then selected an item at random from that skill.

3.1a Discussion of Experiment #1: Advantages Associated with Selecting Multiple Tags

We think, from an HCI perspective, it is reasonable to suggest several choices from the user. If it were possible to limit the choices presented to a user when they are tagging new items with reasonable accuracy, one could expect a significant decrease in the amount of time taken to enter new items. With as many as 78 different skills to choose from, narrowing down the selection can be accomplished effectively without only presenting one recommended skill to the user. This section discusses the possible improvements that stem from selecting the top two or more skill choices for a given item.

The accuracies shown in table 1 indicate the chance that the correct classification for a given item is one of the top N choices generated. These data indicate a substantial improvement over the initial accuracy by adding one or two additional skill selections. However, if the top 5 skills or more are selected, each additional skill selection seems to add between 2% and 3% accuracy. The goal is to narrow the selection of choices to as few as possible while still providing an accuracy that is high enough to assist the user in tagging items. Based on these results, it is apparent that the greatest benefits are achieved through selecting a small number of top choices.

3.2 Experiment #1: Analysis of Accuracy based upon the number of instances: Method and Results

The first experiment also included a qualitative analysis of the classification accuracies associated with each of the individual skills. We classified the items using 50% of the data for the training set and outlined the individual classification accuracies for each of the 78 skills. A different ratio of training set to testing set was used in this part of the experiment because a 90% split resulted in too many of the skill having no instances present in the testing set, which made gathering data on those skills impossible. This experiment was performed 5 times, and the data were averaged and compared to the number of instances representing each skill in the data set. Some of the skills have a more reliable representation than others; it is obvious that skills with very few instances in the data set do not yield high classification accuracy. Despite the improved ratio, some of the skills still did not have a single instance in the testing set. As a result, any skill with fewer than 7 instances was not considered at all in the analysis; this was chosen as the cutoff because it is the lowest number that still had skills with at least one instance in the testing set in each of the five trials. The remaining skills are shown below in figure 2; the average standard deviation for this set was 0.16869.

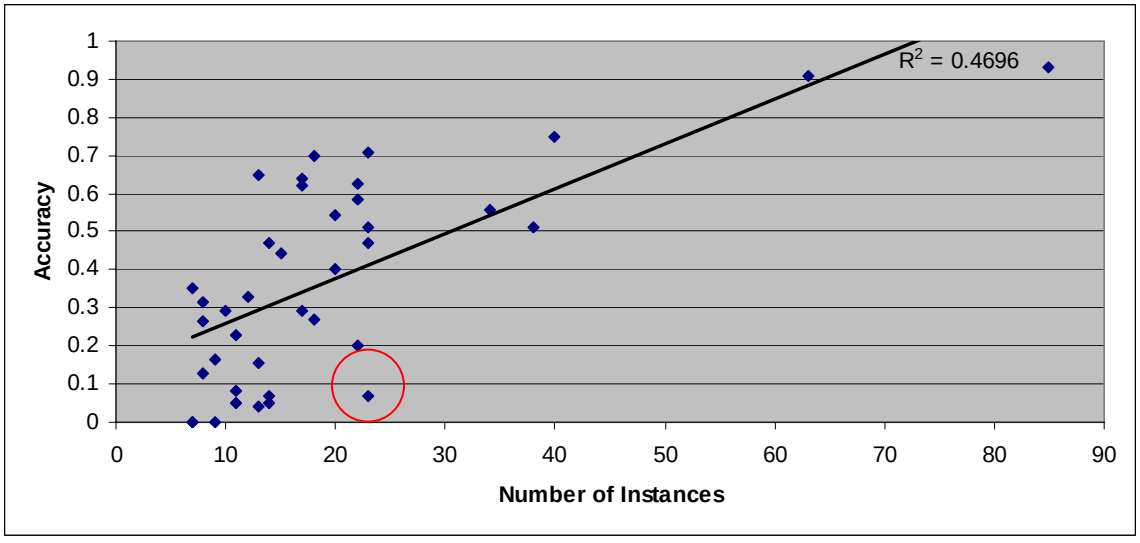

Figure 2: Accuracy vs. Number of Instances

In this part of the experiment, the average accuracy over the entire data set was approximately 42%. After all skills with less than 7 instances were eliminated from the data, the weighted average was approximately 51%.

3.2a Experiment #1: Analysis of Accuracy based upon the number of instances: Discussion

A better understanding of why some skills can be classified with higher accuracy than others may provide opportunities for improvement of the classification accuracy of the entire data set. The most general observation regarding the data was that a higher accuracy is related to a greater number of instances.

Skills with 17 instances or less, in general, did not show an accuracy that was lower than the accuracy for the entire data set. Conversely, the skills with the greatest number of instances showed relatively high accuracies in testing. The three skills with the greatest number of instances, Pattern Finding, Probability, and Symbolization-Articulation, yielded results of 0.9294, 0.9071, and 0.7510 respectively, well above those of the entire data set (about 42%). The fact that we observed a correlation between number of instances and classification accuracy is not surprising.

Not all skills with high instances counts showed such promising results, however. Equation Solving, circled in red in Figure 2, which ranked eighth for greatest number of instances, showed consistently poor performance. This is indicative that something about the question text in these types of questions makes them difficult to differentiate from other skills. An inspection of these question texts revealed a wide variety of actual question topics that are associated with this skill. Furthermore, many words that appear in these question texts can be logically associated with other skill sets and indeed appear in question texts from other skills. This association may have contributed to the comparatively low accuracy when classifying this skill.

Furthermore, some skills with comparatively small numbers of instances showed surprisingly high classification accuracy. In particular, the *"Linear Area Volume Conversion"* skill achieved an accuracy of 65%, which was about 23% above the accuracy for the total data set, while having fewer than 15 instances. The following are several examples from the data set for this skill:

1. Two rectangles, ABCD and WXYZ, are shown above. The measure of each side of WXYZ is 5 **times** the measure of each corresponding side of ABCD. Which statement is true of the areas of these two rectangles?

2. If we assumed that the length of an edge of the blue **cube** equals to 1, then, based on the problem condition, what is the length of an edge of the red **cube** going to be?

3. Now, let's find the surface **area** of a red **cube**. What is the surface **area** of the red **cube** if we know that the length of an edge of the red **cube** equals to 2?

4. Which of the following operations will give us the number of **times** the surface **area** of the red **cube** is larger that the surface **area** of the blue **cube**?

5. Which of the squares shown above has sides that are twice as long as the sides of **square** A shown on the left?

6. The original problem states that the **area** of square A is 4 **square** units. So, what is the **area** of **square** B?

Note that there are keywords associated with these question texts, which may have accounted for the increased accuracy. Each of the thirteen instances for this skill contained one or more of the words "times", "area", "square", or "cube".

Similar trends are seen for the Inequality Solving skill, which had produced an accuracy of about 62% with only 17 instances. All of the 17 items associated with this skill actually contain the word "inequality". In summary, many of the skills that were associated with high accuracy had appeared to have keywords that we often used in the question text, which makes sense.

4 Experiment #2: Trying to use hierarchical classification to improve performance

Our second experiment was to build a hierarchical classifier that we could compare performance with the classifier described in Experiment #1. Our method is based upon Rose et al 2005, who had the insight that by first classifying items in a small number of categories and later classifying those into categories that are nested with the broader categories a higher accuracy and could be achieved. To do this, we used a 90% training split, as in the first experiment. Our hierarchical model first selects a broad category from the MCAS5 (so termed because it has 5 categories of Algebra, Geometry, Probability, Number Sense, and Measurement) and then classifies it into one of the April Transfer Model skills. As is described by Rosé et al (2005), the broad category is not selected with absolute certainty, but the results for selecting a single skill hierarchically are still an improvement over selecting one of April Transfer Model skills directly. Each trial was run five times, and the averages are reported along side those of a direct classification (from table 1) below in table 2.

N	Basic Classification	Hierarchical Classification	Standard Deviation
1	0.4096	0.4519	0.0136
2	0.5194	0.5207	0.0166
3	0.5663	0.5722	0.0194
4	0.6005	0.5745	0.0235
5	0.6369	0.6137	0.0385

Table 2: Basic Classification vs. Hierarchical Classification

4.1 Experiment #2: Discussion: Issues with Hierarchical Classification

From Table 2, we see in bold, that the hierarchal classification had higher accuracy when asked to pick a single best skill. However, we decided to test the application of hierarchy for providing the user with two or more skills. As you can see in Table 2, a hierarchical classification is more effective when the best item is selected, and is even an improvement when the top two choices are selected (accuracy over the top 2 means that the instance was classified *correct* if it one of it's top two prediction was the correct classification). However, when the top three or more tags for a given item are selected, a direct classification into the April Transfer Model is either (approximately) as accurate as or more accurate than a hierarchical classification. Furthermore, it is evident that the rate of improvement in the performance of the hierarchical classification model decreases sharply, relative to the basic classification, if more than the top three options are selected. This is probably due to the accuracy of the initial selection from the MCAS5 transfer model; the effectiveness of hierarchical classification can be severely limited by the accuracy of the top tier when many options are selected from the lower tier possibilities. This suggests that a hierarchical model would serve as an effective part of a semi-automatic skill coder, but would work most effectively if supplemented in some way. For instance, we could use the best one or two guesses from the hierarchal classifier, and then pick 3 or 4 choices from the basic classifier. An alternative approach would be to use the confidence of the initial classifier that classified all items onto one of 5 categories to inform selection of the best classification at the next classification hierarchal of 78 skills. This would enable the top five choices to come from different parts of the MCAS5.

5 Conclusion

In conclusion, it appears that we can use text-based NaiveBayes classification somewhat effectively with an accuracy rate of about 40% when picking one of 78 skills, and an accuracy of about 51% when picking one of 39 adequately represented skills. This appears to be the basis for an effective aid for the people responsible for coding these items. We think it is reasonable that we could provide coders the top five skills as suggestions, and it turns out that in 2/3 of the cases, the system could suggest the correct coding. We speculate our surprising (to us) accuracy might be related to the fact that having 78 skills mean that you can divide up these instances in a large groups of highly distinct item types. We also investigated using hierarchal classification, and got some improvements.

For future work we are going incorporate this classifier into a tool for human coders to use, and we can experiment to see if we can speed coding time and accuracy using these classification.

Acknowledgments

This research was made possible by the US Dept of Education, Institute of Education Science, "Effective Mathematics Education Research" program grant #R305K03140, the Office of Naval Research grant #N00014-03-1-0221, NSF CAREER award to Neil Heffernan, and the Spencer Foundation. All the opinions in this article are those of the authors, and not those of any of the funders.

This work would not have been possible without the assistance of the 2004-2005 WPI/CMU Assistment Team including Mingyu Feng, Andrea Knight, Ken Koedigner at CMU, Abraao Lourenco, Michael Macasek, Goss Nuzzo-Jones, Kai Rasmussen, Leena Razzaq, Steven Ritter at Carnegie Learning, Carolyn Rose at CMU, Terrence Turner, Ruta Upalekar, and Jason Walonoski.

We would also like to acknowledge the assistance of the MALLET team at UMASS-Amherst including Andrew McCallum and Andrew Tolopko.

References

Barnes,T., D. Bitzer, & M. Vouk. (2005) **Experimental analysis of the qmatrix method in knowledge discovery.** *Proceedings of the 15th International Symposium on Methodologies for Intelligent Systems* 2005, May 25-28, 2005, Saratoga Springs, NY.

Croteau, E., Heffernan, N. T. & Koedinger, K. R. (2004) Why Are Algebra Word Problems Difficult? Using Tutorial Log Files and the Power Law of Learning to Select the Best Fitting Cognitive Model. In J.C. Lester, R.M. Vicari, & F. Parguacu (Eds.) *Proceedings of the 7th International Conference on Intelligent Tutoring Systems.* Berlin: Springer-Verlag., p. 240-250.

Koedinger, K. R., Aleven, V., Heffernan. T., McLaren, B. & Hockenberry, M. (2004). Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behavior by Demonstration. In J.C. Lester, R.M. Vicari, & F. Parguacu (Eds.) *Proceedings of 7th Annual Intelligent Tutoring Systems Conference*. Berlin: Springer-Verlag. Page162-173.

McCallum, A. & Kachites (2002). "MALLET: A Machine Learning for Language Toolkit." http://mallet.cs.umass.edu.

Razzaq, L., Feng, M., Nuzzo-Jones, G., Heffernan, N.T., Koedinger, K. R., Junker, B., Ritter, S., Knight, A., Aniszczyk, C., Choksey, S., Livak, T., Mercado, E., Turner, T.E., Upalekar. R, Walonoski, J.A., Macasek. M.A., Rasmussen, K.P. (2005). The Assistment Project: Blending Assessment and Assisting. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press

Rose, C, Donmez, P., Gweon, G., Knight, A., Junker, B., Cohen, W., Koedinger, K., & Heffernan, K. (2005). **Automatic and Semi-Automatic Skill Coding With a View Towards Supporting On-Line Assessment**. In C.K. Looi, G. McCalla, B. Bredeweg, & J. Breuker (Eds.) *Proceedings of the 12th Artificial Intelligence In Education*, 555-562. Amsterdam: ISO Press.

Tatsuoka, K. (1995) Architecture of knowledge structures and cognitive diagnosis: A statistical pattern classification approach. In P. Nichole, S. Chipman & R. Brenonn (Eds.), Alternative Diagnostic Assessment. Hillsdale, NJ: Erlbaum, 1995.