Combining techniques to refine item to skills Q-matrices with a partition tree

Michel C. Desmarais
Polytechnique Montreal
michel.desmarais@polymtl.ca

Peng Xu
Polytechnique Montreal
peng.xu@polymtl.ca

Behzad Beheshti
Polytechnique Montreal
behzad.beheshti@polymtl.ca

ABSTRACT

The problem of mapping items to skills is gaining interest with the emergence of recent techniques that can use data for both defining this mapping, and for refining mappings given by experts. We investigate the problem of refining mapping from an expert by combining the output of different techniques. The combination is based on a partition tree that combines the suggested refinements of three known techniques from the literature. Each technique is given as input a Q-matrix, that maps items to skills, and student test outcome data, and outputs a modified Q-matrix that constitutes suggested improvements. We test the accuracy of the partition tree combination techniques over both synthetic and real data. The results over synthetic data show a high improvement over the best single technique with a 86% error reduction on average for four different Q-matrices. For real data, the error reduction is 55%. In addition to the substantial error reduction, the partition tree refinements provide a much more stable performance than any single technique. These results suggest that the partition tree is a valuable refinement combination approach that can effectively take advantage of the complementarity of the Q-matrix refinement techniques. It brings the goal of using a data driven approach to refine the item to skill mapping closer to real applications, although challenges remain and are discussed.

1. INTRODUCTION

Defining which skills are involved in a task is non trivial. Whereas task outcome is observable, skills are not. This layer of opacity leaves a world of possibilities to define which skills are behind task performance, and no obvious evidence to know if the proposed definition is correct or not. Means to provide such feedback would be highly valuable to teachers and designers of learning environments, and we find numerous recent efforts towards this end in the last few years. They are reviewed in section 2.

We developed an approach that takes the outpout of a combination of techniques to detect likely errors of task to skills mappings given by experts. We investigate the combination of three data-driven techniques [3, 2, 7] based on a partition tree algorithm that creates binary partitions. See also [6] for a more detailed comparison of the performance of these thee techniques.

The performance of the partition tree approach is tested over synthetic and real data. But even in the case of real data, the approach to grow the partition tree trains on synthetic performance data generated from a set of Q-matrices that are similar to the Q-matrix to refine. This procedure is chosen because only synthetic data provides a large enough training set, and because it also provides ground truth labelling of latent variables.

In the rest of this text we use the term *items* to refer to questions or tasks that can be part of a formative or summative assessment, or exercises within an e-learning environment. Skills can be the mastery of concepts, factual knowledge, or any ability involved in item outcome success. However, the models reviewed here assume a static student skills state, as opposed to the Knowledge Tracing model and its derivatives [11], for example, which rely on dynamic data. We return to this limitation in the Discussion.

The different techniques to validate a Q-matrix are first described, followed by the description of the approach, the experiments, and the results.

2. Q-MATRICES AND TECHNIQUES TO VALIDATE THEM FROM DATA

A mapping of item to skills is termed a Q-matrix. An example of a 11 items and 3 skills Q-matrix is given beside. It corresponds to the Q-matrix labelled QM 1 in the results section below. From this example, item 4 requires skill 1 only, whereas item 11 requires skills 1 and 2. If all specified skills are required to succeed the item, the Q-matrix is labeled **conjunctive**. If a any of the required skill is sufficient to the item's success, then it is labeled **disjunctive**. The **compensatory** version corresponds to the case

Q-matrix QM-1

	Skill		
Item	1	2	3
1	1	1	0
2	1	0	1
3	1	0	1
4	1	0	0
5	1	1	0
6	1	1	0
7	1	0	1
8	1	0	1
9	1	0	0
10	1	0	0
11	1	1	0

where each required item increases the chances of success in some way. Conjunctive Q-matrices the most common and all matrices of the experiments here are of this type.

The conjunctive/disjunctive distinction is also referred to as AND/OR gates. Skills models such as DINA (Deterministic Input Noisy AND) and DINO (Deterministic Input Noisy Or) make reference to this AND/OR gates terminology.

The DINA model [10] defines the probability of success to an item as a function of whether the skills required are mastered, and of two parameters, the *slip* and *guess* factors. Mastery is a binary value based on the conjunctive framework: if all required skills are mastered then the value is 1, else it is 0. Slip and guess parameters are values that generally vary on a $[0, 0.2]$ scale. The probability of success to an item j by a student i is thereby defined as:

$$P(X_{ij}=1 \mid \xi_{ij}) \;=\; (1-s_j)^{\xi_{ij}} g_j^{1-\xi_{ij}}$$

where ξ_{ij} is 1 if student i masters all required skills of item j, 0 otherwise. s_j and g_j are the *slip* and *guess* factors.

Two techniques for Q-matrix validation surveyed here rely on the DINA model, whereas the third one relies on a matrix factorization technique called ALS (Alternative Least Squares), or more precisely ALSC for the *conjunctive* version of the technique. We briefly review each technique below.

2.1 Technique 1: MinRSS

Chiu defines a method that minimizes the residual sum of square (RSS) between the real responses and the ideal responses that follow from a given Q-matrix [2] under the DINA model. The algorithm adjusts the Q-matrix by first estimating the mastery of each student, then choosing the item with the worst RSS over to the data, and replacing it with a q-vector that has the lowest RSS, and iterates until convergence. We refer to this technique as MinRSS .

2.2 Technique 2: MaxDiff

The method defined by de la Torre [3] searches for a Q-matrix that maximizes the difference in the probabilities of a correct response to an item between examinees who possess all the skills required for a correct response to that item and examinees who do not. It also relies on the DINA model to determine item outcome probability, and on an EM algorithm to estimate the slip and guess parameters. Probability differences represents an item discrimination index: the greater the difference between the probability of a correct response given the skills required and the probability given missing skills, the greater the item is discriminant. As such, we can consider that the method finds a Q-matrix that maximizes item discrimination over all items. We refer to this technique as MaxDiff .

2.3 Technique 3: Conjunctive alternate Least-Square Factorization (ALSC)

The Conjunctive alternate Least-Square Factorization (ALSC) method is defined in [7]. Contrary to the other two methods, it does not rely on the DINA model as it has no slip and guess parameters. ALSC decomposes the results matrix $\mathbf{R}_{m\times n}$ of m items by n students as the inner product two smaller matrices:

$$\neg\mathbf{R} = \mathbf{Q}\,\neg\mathbf{S} \tag{1}$$

where $\neg\mathbf{R}$ is the negation of the results matrix (m items by n students), $\mathbf{Q}$ is the m items by k skills Q-matrix, and $\neg\mathbf{S}$ is negation of the the mastery matrix of k skills by n students (normalized for rows columns to sum to 1). By negation, we mean the 0-values are transformed to 1, and non-0-values to 0. Negation is necessary for a conjunctive Q-matrix.

The factorization consists of alternating between estimates of $\mathbf{S}$ and $\mathbf{Q}$ until convergence. Starting with the initial expert defined Q-matrix, $\mathbf{Q}_0$, a least-squares estimate of $\mathbf{S}$ is obtained:

$$\neg\hat{\mathbf{S}}_0 = (\mathbf{Q}_0^{\mathrm{T}}\,\mathbf{Q}_0)^{-1}\,\mathbf{Q}_0^{\mathrm{T}}\,\neg\mathbf{R} \tag{2}$$

Then, a new estimate of the Q-matrix, $\hat{\mathbf{Q}}_1$, is again obtained by the least-squares estimate:

$$\hat{\mathbf{Q}}_1 = \neg\mathbf{R}\,\neg\hat{\mathbf{S}}_0^{\mathrm{T}}\,(\neg\hat{\mathbf{S}}_0\,\neg\hat{\mathbf{S}}_0^{\mathrm{T}})^{-1} \tag{3}$$

And so on until convergence. Alternating between equations (2) and (3) yields progressive refinements of the matrices $\hat{\mathbf{Q}}_i$ and $\hat{\mathbf{S}}_i$ that more closely approximate $\mathbf{R}$ in equation (1). The final $\hat{\mathbf{Q}}_\mathbf{i}$ is rounded to yield a binary matrix.

Note that $(\neg\mathbf{Q}_i^{\mathrm{T}}\,\neg\mathbf{Q}_i)$ or $(\neg\hat{\mathbf{S}}_i\,\neg\hat{\mathbf{S}}_i^{\mathrm{T}})_i$ may not be invertible, for example in the case where the matrix $\mathbf{Q}_i$ is not column full-rank, or the matrix $\mathbf{S}_i$ is not row full-rank. This is resolved by adding a very small Gaussian noise before attempting the matrix inverse.

2.4 Other techniques

We chose the three techniques described above as the candidates to combine refinements that can potentially provide more accurate suggestions than any of the individual ones, but any other equivalent technique could also be combined in the same fashion instead of the three chosen ones here. Potential candidates could be, for example, a technique based on a Bayesian approach by DeCarlo et al. [5], and recent techniques that rely on time information [13, 12]. Yet another recent approach relies item text [8] to establish the mapping of items to skills.

Although the results obtained through a combination of techniques may vary as a function of the specific techniques chosen, the general principle remains valid for all possible combinations. And there is no reason to believe that the particular combination of the current study is better or worse than other potential combinations.

2.5 General validation principle

The general idea behind the validation of Q-matrices is to introduce a perturbation to a matrix and run a refinement technique that takes the perturbed matrix and test data as input, and outputs a set of refinements. In all, 8 cases can occur and they are listed in table 1. The 8 cases are a combination of the original cell value, perturbation, and value proposed ($2 \times 2 \times 2$).

The outcome of a proposed value from the refinement technique is considered correct if it corresponds to the original value before the perturbation, and incorrect otherwise. We

Table 1: Refinement outcomes

	Perturbation		Refinement	
	Value before	Value after	Value proposed	Outcome
		Perturbed cell		
(1)	0	1	0	correct (TP)
(2)	1	0	1	correct (TP)
(3)	0	1	1	wrong (FN)
(4)	1	0	0	wrong (FN)
		Non Perturbed cell		
(5)	0	0	0	correct (TN)
(6)	1	1	1	correct (TN)
(7)	0	0	1	wrong (FP)
(8)	1	1	0	wrong (FP)

also refer to the signal detection terminology with respect to perturbations to introduce further classification of the error types:

- **True Positives (TP)**: perturbed cell that was correctly changed
- **True Negatives (TN)**: non perturbed cell left unchanged
- **False Positives (FP)**: non perturbed cell incorrectly changed
- **False Negatives (FN)**: perturbed cell left unchanged

3. COMBINING TECHNIQUES WITH A PARTITION TREE

Each of the technique described above uses a different algorithm to provide a potentially improved Q-matrix. In that respect, their respective outcome may be complementary, and their combined outcome can be more reliable than any single one. This is the first hypothesis and objective of our study. Furthermore, some algorithms are more effective in general, but may not be the best performer in all context. Identifying in which context an algorithm provides the most reliable outcome is another objective of combining these techniques. We will see that the first hypothesis is confirmed in the results of the partition tree labeled (1) and the second is also confirmed by the results of partition tree (3).

3.1 Partitioning tree

To implement the partition tree combination of the three techniques, we chose the `rpart` package for this purpose [19].

The `rpart` package builds classification models that can be represented as binary trees. The tree is constructed in a top-down recursive divide and conquer approach. At each node in the tree, cases are split into two groups based on their attribute value.

3.1.1 Tree building

Attribute selection is done on the basis of Gini index in `rpart`. The Gini index [16] can be calculated as :

$$\text{Gini}(D) = 1 - \sum_{j=1}^{n} p_j^2$$

where n is the number of classes and p_j is the relative frequency of class j in dataset D. If attribute A is chosen to be a split on dataset D into two subset D_1 and D_2, then the Gini index for attribute A is defined as:

$$\text{Gini}_A(D) = \frac{|D_1|}{|D|}\text{Gini}(D_1) + \frac{|D_2|}{|D|}\text{Gini}(D_2)$$

Once we get the Gini index to add attributes we can calculate a Delta reduction for each attribute:

$$\triangle\text{Gini}(A) = \text{Gini}(D) - \text{Gini}_A(D)$$

The attribute that creates the largest reduction can be chosen as a splitting point in the decision tree.

3.1.2 Classification with the tree

In our case, attributes are sometimes numeric, such as factors, and sometimes binary, such as cell values in the Q-matrix. And the class is binary since it is also a Q-matrix cell value. At each point of decision from the root node of the tree to a leaf node, a choice is made to go left or right based on the splitting point of each node. The nodes in the partition trees of this experiment are the output of the techniques (suggested values) and the factors considered (they are described in the next section).

Once a leaf node is reached, classification is based on the majority vote of the cases that fall under that leaf node: if the training set contained more case labeled '0', this is the proposed value, else it is a '1'.

3.2 Factors considered

The partition tree relies on each technique's output, the Q-matrix refinement proposition, and on a number of factors that may provide information about the most reliable technique refinement in a given context. The factors considered to be relevant are the following:

- **Skills per row**. Items can require one or more skills. The skills per row indicates the number of skills required.
- **Skills per column**. The sum of the skills per columns is an indicator of how often this skill is measured by the different items of the Q-matrix.
- **Stickiness**. If a technique systematically proposes a change to a cell of the Q-matrix, no matter what the perturbation is, this is an indication that this particular change to the original Q-matrix is an artifact of the structure of the Q-matrix and the algorithm. We call this property the *stickiness* of a cell of the matrix and it is measured by the proportion of times the value of the cell is incorrectly changed over all perturbations.

  Recall that we train the partition tree over synthetic data for which the ground truth is known. We can therefore reliably identify incorrect changes. This is detailed below.

3.3 Training of the partition tree

The partition tree is trained on data that contains the following set of attributes:

- $\texttt{original}_{(j,k)}$: value of cell (j,k) in the original matrix. This is the target class of the partition tree and it corresponds to "Value before" in table 1.
- $\texttt{MaxDiff}_{(j,k)}$, $\texttt{MinRSS}_{(j,k)}$, $\texttt{ALSC}_{(j,k)}$: the three values proposed as refinements by the respective technique in place of the original value. For every record, at least one of these must be different from the original one, or else it is a perturbated cell record. This corresponds to "Value proposed" in table 1, one for each refinement technique.
- $\texttt{RS}_{Q_i,j}$, $\texttt{CS}_{Q_i,k}$: the number of skills per row and column attributes (see section 3.2). These factors are per Q-matrix, Q_i, and per row j and column k.
- $\texttt{SF}_{\text{MaxDiff}(Q_i,j,k)}$, $\texttt{SF}_{\text{MinRSS}(Q_i,j,k)}$, $\texttt{SF}_{\text{ALSC}(Q_i,j,k)}$: the stickiness factors of the cell, one for each matrix and technique.

The training data is generated through a perturbation process. Each cell of a Q-matrix is perturbated, in turn and one at a time, to create a new training record containing the above attributes. However, non perturbated cells that are left unchanged by all refinements techniques, cases (5) and (6) in table 1, are left out of the training data because they were assumed to be uninformative.

The size of the data set to train the partition trees over is very large. For the permutations of a single Q-matrix, the number of perturbated and non perturbated cells ranges from approximately 50,000 to 250,000.

Training of the partition tree for expert Q-matrices with synthetic data. Whereas for synthetic data, we can generate a large array of Q-matrices and ample training and testing data, real data poses a challenge in that respect. Typically, for a single data set, we have only a few expert Q-matrices, and often a single one is available. For a 3 skills $\times$ 11 items matrix, only 33 single perturbations are possible to train a partition tree. Furthermore, and unlike synthetic data, we do not know what are the valid refinements in the Q-matrix. A "sticky" cell might be a valid refinement, and so can some of the perturbations that are presumed incorrect.

To get around these issues, the training of the partition tree is conducted over synthetic data where the ground truth is know and where we can use a large span of matrices similar to the expert one. Similarity to the Q-matrix to refine is achieved by random permutations the cells of the original Q-matrix. For each Q-matrix, a total of 1000 Q-matrices are generated through this permutation process. Item outcome data for 400 simulated students is also generated. The R package CDM and the `sim.din` function [15] is used for generating synthetic student item outcome data, using 0.2 slip and guess factors.

4. REAL DATA AND Q-MATRICES

The primary source of real data for our study, from which the synthetic data is also mimicked, is the well known data set on fraction algebra problems from Tatsuoka [17] (see table 1 in [4] for a description of the problems and of the skills). The data contains complete answers of 536 students to 20 questions items, but only a subset of 11 items are used by the Q-matrices in the current study. It corresponds to the set of common items to the different Q-matrices of the experiment.

Table 2: Four Q-matrices over 11 items of Tatsuoka's data set on student item outcome

	Number of skills	Number of items	Number of cases	Description
QM 1	3	11	536	Expert driven. Skill 1 shared by all items. From [9]
QM 2	5	11	536	Expert driven. From [3]
QM 3	3	11	536	Expert driven. Single skill per item. [15]
QM 4	3	11	536	Data driven, SVD-based.

The original Q-matrix of this data set contains 8 skills and, as mentioned, 20 items. However, a number of variations of this matrix have been proposed and studied with a smaller number of skills and items [9, 3, 15]. We also chose to focus on this smaller skills set since they offer three very different expert-defined Q-matrices over the same set of items. Moreover, a smaller set of skills allows us to better establish the validity of the approach on a simpler problem, leaving for later the demonstration of whether it scales correctly to larger sets. The Q-matrices are described below.

Four Q-matrices are considered. Three of them have been studied in the literature and one is defined by ourselves. Their main attributes are reported in table 2 and the actual Q-matrices are shown in figure 1 (except for QM 1 which is introduced in section 2).

	Skills of										
	QM 2					QM 3			QM 4		
Item	1	2	3	4	5	1	2	3	1	2	3
1	1	1	1	1	0	0	1	0	1	1	0
2	1	1	1	1	1	0	0	1	1	0	1
3	0	0	1	0	0	0	0	1	0	1	0
4	1	1	1	1	0	1	0	0	1	0	0
5	1	1	1	1	0	0	1	0	1	0	0
6	1	1	0	0	0	0	1	0	0	0	1
7	1	0	1	1	1	0	0	1	1	0	1
8	1	0	1	0	0	0	0	1	0	1	1
9	1	0	1	1	0	1	0	0	1	0	0
10	1	1	1	1	0	1	0	0	1	0	1
11	1	1	1	1	0	0	1	0	1	0	0

Figure 1: Q-matrices 2, 3, and 4.

As mentioned, all Q-matrices are derivatives of the Tatsuoka [17] 20 item set. QM-1, QM-2 and QM-3 are available from the CDM package. All data sets have 3 skills, except for data set 2 which has 5 skills. Data set 3 is the only one

with a single skill per item. Matrix QM 4 was created for the purpose of this study, using the three largest singular values and the items to skills **V** matrix of the SVD decomposition of the Tatsuoka data mentioned above.

Therefore, while these four Q-matrices all share the same 11 items, they vary by the number of skills, item monoticity or not, whether a skill is common to all items, and whether they are driven from data or driven from expert analysis of item skills involved.

5. GENERAL PROCEDURE SUMMARY AND METHODOLOGICAL NOTES

To ease the understanding of the general process of the experiments, and at the expense of introducing some redundancy, figure 2 summarizes the main steps and dependencies. The top greyed box illustrates the process to generate the data for partition trees training, and the synthetic data for performance evaluation. The bottom greyed box illustrates the two test procedures for real and synthetic data. We explain the figure below and fill in some details as well.

Data generation. For each of the four Q-matrices (QM_i), the data generation process (1) 1000 permutations (2). Duplicates are kept if any. For each permutation, synthetic test outcome data of 400 simulated students is created with the CDM utility `sim.din` (3). Finally, each QM is perturbated, and that Q-matrix is fed to each of the three techniques to generate training data for the partition tree described in section 3.3 (4).

Test over real data. The experiment to assess the performance over real data takes three sources of input: the Q-matrices (1), the fraction algebra data set of Tatsuoka as described in 4 (6), and finally a partition tree (5) trained from data generated (4). It outputs a set of refinements from the different partition trees and for each of the three techniques as well (7). Finally, the refinements are compared with the original Q-matrices in (1).

Test over synthetic data. For assessing the performance over synthetic data (9), the process is similar, with the main difference that refinements are based on the synthetic test outcome data generated in (3) instead of real data. And the comparison is not done over the Q-matrices in (1), but instead over the permuted Q-matrices in (2), which constitute the ground truth as they are used to generate the data.

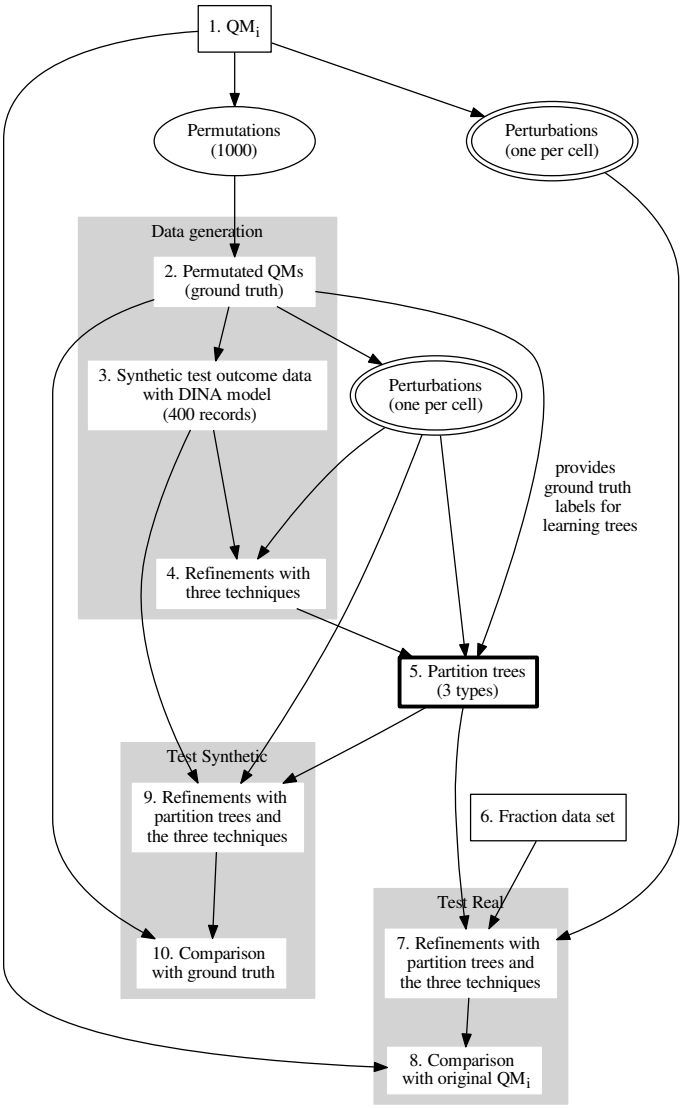

Figure 2: General validation procedure for each Q-matrix (QM_i). See section 5 for details.

5.1 Data set size, cross-validation, and the assumption of correctness of expert Q-matrices

As shown in figure 2, synthetic test outcome data (3) is used for both the training of the partition trees and testing over synthetic data. This large data set (see sect. 3.3) leaves little space for over fitting of the partition trees, and therefore the cross-validations bring very small differences in performance: accuracy/RSS error reduction is the same between a cross-validated and a non cross-validated performance assessment at the 0.01 level reported in the results below.

However, for real data, the size of the testing data set is much smaller. It varies between 366 (QM-2) and 561 (QM-3), because the test data is based solely on the permutations of the four Q-matrices. But because the test procedure uses partition trees trained from synthetic data, there are no bias issues and cross validation is not required here.

Note also that, for real data, the expert-defined Q-matrix is not necessarily consistent with the (unknown) ground truth. Nevertheless, we consider this Q-matrix as valid and the evaluation of the proposed refinements are made by comparing refinements with expert-defined Q-matrices, as though

Table 3: Results for synthetic data

QM	Technique			Partition tree		
	MinRSS	MaxDiff	ALSC	(1)	(2)	(3)
Accuracy of perturbated cells						
1	0.81	0.47	0.82	0.81	0.88	**0.95**
2	0.07	0.26	0.36	0.52	0.53	**0.83**
3	0.96	0.49	0.95	0.99	**1.00**	**1.00**
4	0.90	0.49	0.85	0.90	0.92	**0.96**
$\overline{\mathbf{X}}$	0.69	0.43	0.75	0.81	0.83	**0.93**
Accuracy of non perturbated cells						
1	0.97	0.56	0.44	0.97	0.91	**0.99**
2	**0.99**	0.53	0.50	**0.99**	**0.99**	**0.99**
3	0.95	0.26	0.74	0.95	0.94	**0.99**
4	0.97	0.56	0.44	0.97	0.97	**1.00**
$\overline{\mathbf{X}}$	0.97	0.48	0.53	0.97	0.95	**0.99**
F-score						
1	0.88	0.51	0.58	0.88	0.90	**0.97**
2	0.13	0.35	0.42	0.68	0.69	**0.90**
3	0.96	0.34	0.83	0.97	0.97	**1.00**
4	0.93	0.52	0.58	0.93	0.94	**0.98**
$\overline{\mathbf{X}}$	0.72	0.43	0.60	0.87	0.87	**0.96**

Table 4: Results for real data

QM	Technique			Partition tree		
	MinRSS	MaxDiff	ALSC	(1)	(2)	(3)
Accuracy of perturbated cells						
1	0.39	0.17	0.52	0.39	0.36	**0.67**
2	0.35	0.09	0.56	0.60	0.62	**0.64**
3	0.27	0.09	0.36	0.61	**1.00**	0.88
4	0.42	0.11	0.58	0.42	0.48	**0.61**
$\overline{\mathbf{X}}$	0.36	0.12	0.51	0.51	0.62	**0.70**
Accuracy of non perturbated cells						
1	0.45	**0.68**	0.56	0.45	0.38	0.60
2	0.93	0.93	0.28	0.94	0.94	**0.97**
3	0.64	0.83	0.42	0.69	0.76	**0.78**
4	0.55	**0.89**	0.32	0.55	0.52	0.51
$\overline{\mathbf{X}}$	0.52	0.68	0.32	0.62	0.62	**0.68**
F-score						
1	0.42	0.27	0.54	0.42	0.37	**0.63**
2	0.50	0.17	0.37	0.73	0.74	**0.77**
3	0.38	0.16	0.39	0.64	**0.86**	0.83
4	0.48	0.20	0.42	0.48	0.50	**0.56**
$\overline{\mathbf{X}}$	0.45	0.20	0.43	0.57	0.62	**0.70**

they were the ground truth. We should keep in mind that the performance score may be negatively biased if this assumption was false, but for the purpose of comparing the relative techniques performance among themselves, and if we assume that all techniques are equally affected by this bias, then it makes no difference to our relative results.

6. PERFORMANCE MEASURES

To measure the performance of the proposed refinements, we use the difference between the original Q-matrix and the proposed refinement of a technique. We use the classification of correct and incorrect refinements introduced in table 1. Cells that are neither perturbated nor incorrectly suggested as refinements by any of the technique are ignored in the analysis (the *true negatives* of table 1, TN). This is the case of the large majority and it also is consistent with the training of the partition tree for which they are also filtered out.

Recovery of a perturbated cell to its original value can be considered as a *recall* measure, whereas the non perturbated cells that are left unchanged can be considered as a *precision* measure. In that respect, we define a performance measure that combines precision and recall of the refinement technique into a single F-score measure:

$$\begin{aligned} \text{F-score} &= 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \\ &= 2 \times \frac{\text{Acc}_{\neg P} \times \text{Acc}_P}{\text{Acc}_{\neg P} + \text{Acc}_P} \end{aligned}$$

where Acc_P and $\text{Acc}_{\neg P}$ are respectively the accuracy measure of the proposed refinements for the perturbated and non perturbated cells. This measure gives equal weight to both types of accuracies and avoids a bias in favour of the accuracy of the non perturbated cells which can considerably outweigh in number the single perturbated cell, even after filtering out non-perturbated cells that are left unchanged.

7. RESULTS

The results are reported in tables 3 and 4. The format of these tables first described below.

7.1 Description

The respective results of the four Q-matrices (column **QM**) in table 2 are reported. They correspond to a single run (real data can vary a few percentage points by run, but it is practically stable for synthetic data due to the large number of cases). The accuracy of refinement for perturbated and non perturbated cells are reported separately, followed by the F-score which combines both types of accuracy. The averages of the four matrices for each of these these three performance measures is also reported as $\overline{\mathbf{X}}$.

The accuracy and F-score of each individual technique is reported under columns **MinRSS** , **MaxDiff** , and **ALSC**.

The three columns under **Partition tree** correspond to the performance as a function of different factors used for building the tree:

(1) MinRSS + MaxDiff + ALSC. Only the output of the three refinement techniques is considered.

(2) MinRSS + MaxDiff + ALSC + SR + SC. The number of skills per row (SR) and skills per column (SC) of the target cell are taken into account in addition to the output of each technique. If some technique performs better under some combination of SR and SC, this tree will be able to take these factor into account.

(3) MinRSS + MaxDiff + ALSC + SR + SC + Stickiness.MinRSS + Stickiness.MaxDiff + Stickiness.ALSC. The tendency of a cell to be a false positive for the MinRSS and ALSC methods are added. The Stickiness factor with MaxDiff is omitted here because it did not yield improvements.

7.2 Synthetic data

The results for synthetic data in 3 show large differences between the different matrices and across the individual techniques.

The MinRSS method is clearly superior in terms of general accuracy, except for the 5-skills Q-matrix where it can only identify the perturbated cell 7% of the time, and which brings its average below the ALSC technique. However, because it introduces fewer *false positives* (incorrect refinements) than other techniques, it outperform the other two methods on the F-Score.

On average, the ALSC technique is good at identifying the perturbated cell with a 75% average, but it also tends to introduce more false positives and consequently obtains a lower global F-score than MinRSS .

Another noticeable result is that the results for QM 3 are very good, in particular for the partition trees which have perfect performance (rounding at the second decimal). This is likely attributed to the fact that it defines a single-skill mapping.

Turning to the main questions addressed in this study, the results of partition tree (1), which uses only the three techniques' output, is equal or better on all scores than any individual one. This confirms the initial hypothesis for synthetic data. Furthermore, the inclusion of factors (partition trees (2) and (3)) also substantially improves all scores, confirming the other hypothesis that some techniques perform better under a combination of factors and that the partition tree is effectively able to take advantage of this information. The stickyness factor is by far the most effective.

7.3 Real data

The results over the real data reported in table 4 show the same trends as the synthetic data, but bring less pronounced improvements. They also support both hypothesis.

We do find an exception with the non perturbated cells where the MaxDiff accuracy is above the partition trees (1) and (2) and close to (3). This is mainly due to the fact that more "false positives" are generated by the MinRSS and ALSC techniques for real data than for synthetic data, whereas the MaxDiff technique outputs very few changes in both contexts. That observation is consistent with the results in [6].

The balance between true positives and true negatives illustrates why the F-score should be the reference: a perfect score could be obtained over the accuracy of non perturbated cells if no changes are always suggested, but that would make such refinement technique useless.

Therefore, turning to the F-scores, the tendencies are highly consistent with the synthetic data. The F-score of the best performer, 0.41 of MinRSS , is improved to 0.55 with the combination of the three techniques, and to 0.66 when all factors are included in the partition tree.

8. DISCUSSION

The results of the above experiments show that the combination of Q-matrix refinement techniques using a partition tree can bring substantial improvements over the best performance of the individual techniques. For synthetic data the average best F-score of the MinRSS technique, 0.72, is improved to 0.96, and for real data it is raised from 0.41 to 0.66. These results represent a 86% and 55% error reduction for the F-score of the synthetic and the real data respectively (error reduction $= 1 - (1 - F')/(1 - F)$, where F is the initial F-score and F' is the improved F-score).

In practical terms, if the best technique finds an error in a Q-matrix 5 out of 10 times, an error reduction of 40% represents an increase from 5, to 7 out of 10 times, and the same ratio applies to false errors reduction. And these figures rest on the assumption that we would know which technique is the best, whereas according to table 4's results the best technique varies across Q-matrices.

Another positive note on the results is that the partition tree F-scores are more stable across Q-matrices and are systematically better than any individual technique when all factors are taken into account (partition tree 3). This regularity incurs that, at least in the space of Q-matrices surveyed, one can safely choose partition tree refinements without concerns that, maybe, another technique could deliver better refinements for a specific Q-matrix.

In spite of these encouraging results, limitations and issues remain.

One limit is that the results are from a single 11 items set, and from a single domain. We can reasonably believe that the results vary across contexts and more investigation is required to assess this variability.

Another limitation is the models investigated in the current study use *static* student data: they assume that skill mastery does not change for a single student. This assumption is false for most data gathered in learning environments, where students take on exercises as they learn and are being assessed throughout the learning process. This type of data can be labeled as *dynamic* item outcome data because a student will be in different states of skills mastery as learning occurs.

In order to effectively use the existing techniques of Q-matrix refinement, we would need to be able to detect the moment when the state of skill mastery changed. Failure do do so would create noise in the data and impair the effectiveness of these techniques. Fortunately, substantial progress has been done in the recent decade or two towards detecting the moment of learning, such as the large body of work on Bayesian Knowledge Tracing and Tensor factorization (for eg. [1, 18]). We can also cite the work of [14] who refer to a time-varying skills matrix for students and test their approach on synthetic data. But apart from this recent

contribution, little work has been done on using this type of data for refining a Q-matrix, and we can only expect existing techniques to under perform with dynamic student data.

9. ACKNOWLEDGEMENTS

This research was funded under the NSERC Discovery grant of the first author.

10. REFERENCES

[1] R. S. Baker, A. B. Goldstein, and N. T. Heffernan. Detecting learning moment-by-moment. *International Journal of Artificial Intelligence in Education*, 21(1):5–25, 2011.

[2] C.-Y. Chiu. Statistical refinement of the Q-matrix in cognitive diagnosis. *Applied Psychological Measurement*, 2013.

[3] J. De La Torre. An empirically based method of Q-Matrix validation for the DINA model: Development and applications. *Journal of educational measurement*, 45(4):343–362, 2008.

[4] L. T. DeCarlo. On the analysis of fraction subtraction data: The DINA model, classification, latent class sizes, and the Q-Matrix. *Applied Psychological Measurement*, 35:8–26, 2011.

[5] L. T. DeCarlo. Recognizing uncertainty in the Q-Matrix via a bayesian extension of the DINA model. *Applied Psychological Measurement*, 36(6):447–468, 2012.

[6] M. C. Desmarais, B. Beheshti, and P. Xu. The refinement of a Q-matrix: Assessing methods to validate tasks to skills mapping. In *7th Educational Data Mining Conference*, pages 208–311, 2014.

[7] M. C. Desmarais and R. Naceur. A matrix factorization method for mapping items to skills and for enhancing expert-based Q-Matrices. In *6th International Conference, AIED 2013, Memphis, TN, USA*, pages 441–450, 2013.

[8] C. Goutte, G. Durand, and S. Léger. Towards automatic description of knowledge components. In *Proceedings of the 8th International Conference on Educational Data Mining*, page (to apppear), Madrid, Spain 2015.

[9] R. A. Henson, J. L. Templin, and J. T. Willse. Defining a family of cognitive diagnosis models using log-linear models with latent variables. *Psychometrika*, 74(2):191–210, 2009.

[10] B. Junker and K. Sijtsma. Cognitive assessment models with few assumptions, and connections with nonparametric item response theory. *Applied Psychological Measurement*, 25(3):258–272, 2001.

[11] K. R. Koedinger, A. T. Corbett, and C. Perfetti. The knowledge-learning-instruction (KLI) framework: Toward bridging the science-practice chasm to enhance robust student learning. Technical report, Carnegie-Mellon University, Human Computer Interaction Institute, 2011.

[12] J. Nižnan, R. Pelánek, and J. Řihák. Mapping problems to skills combining expert opinion and student data. In *Mathematical and Engineering Methods in Computer Science*, pages 113–124. Springer, 2014.

[13] J. Nižnan, R. Pelánek, J. Řihák, et al. Using problem solving times and expert opinion to detect skills. In *Proceedings of the 7th International Conference on Educational Data Mining*, pages 433–434, London, UK 2014.

[14] S. Oeda, Y. Ito, and K. Yamanishi. Extracting latent skills from time series of asynchronous and incomplete examinations. In *Proceedings of EDM 2014, The 7th International Conference on Educational Data Mining*, pages 367–368, London, UK 2014.

[15] A. Robitzsch, T. Kiefer, A. George, A. Uenlue, and M. Robitzsch. Package CDM. 2012.

[16] L. Rokach. *Data mining with decision trees: theory and applications*. World scientific, 2007.

[17] K. Tatsuoka, U. of Illinois at Urbana-Champaign. Computer-based Education Research Laboratory, and N. I. of Education (US). *Analysis of errors in fraction addition and subtraction problems*. Computer-based Education Research Laboratory, University of Illinois, 1984.

[18] N. Thai-Nghe, L. Drumond, T. Horváth, A. Nanopoulos, and L. Schmidt-Thieme. Matrix and tensor factorization for predicting student performance. In A. Verbraeck, M. Helfert, J. Cordeiro, and B. Shishkov, editors, *CSEDU 2011 - Proceedings of the 3rd International Conference on Computer Supported Education, Volume 1, Noordwijkerhout, Netherlands, 6-8 May, 2011*, pages 69–78. SciTePress, 2011.

[19] T. Therneau, B. Atkinson, B. Ripley, and M. B. Ripley. Package rpart, 2014.