LEARNING LARGE $Q$-MATRIX BY RESTRICTED BOLTZMANN MACHINES

CHENGCHENG LI, CHENCHEN MA AND GONGJUN XU

UNIVERSITY OF MICHIGAN

Estimation of the large $Q$-matrix in cognitive diagnosis models (CDMs) with many items and latent attributes from observational data has been a huge challenge due to its high computational cost. Borrowing ideas from deep learning literature, we propose to learn the large $Q$-matrix by restricted Boltzmann machines (RBMs) to overcome the computational difficulties. In this paper, key relationships between RBMs and CDMs are identified. Consistent and robust learning of the $Q$-matrix in various CDMs is shown to be valid under certain conditions. Our simulation studies under different CDM settings show that RBMs not only outperform the existing methods in terms of learning speed, but also maintain good recovery accuracy of the $Q$-matrix. In the end, we illustrate the applicability and effectiveness of our method through a TIMSS mathematics data set.

Key words: Cognitive diagnosis models, $Q$-matrix, Restricted Boltzmann machines.

Cognitive diagnosis models (CDMs) are popular statistical tools widely applied to educational assessments and psychological diagnoses, which have been receiving increasingly more attention in the past two decades. In many modern assessment situations, examiners are concerned with specific attributes that the examinees possess, and thus, a simple overall score is no longer sufficient to depict the whole picture of the candidates. As a result, a finer evaluation of the examinees’ attributes is desired. CDMs are such tools. They model the relationship between the test items and the examinees’ latent skills, which is helpful in assessment design and post-assessment analysis of the examinees’ latent attribute patterns. CDMs have seen vast applications in multiple scientific disciplines, including educational assessments (Junker and Sijtsma, 2001; von Davier, 2008; García et al., 2014), psychiatric diagnosis of mental disorders (Templin and Henson, 2006; de la Torre et al., 2018), epidemiological and medical measurement studies (Wu et al., 2016).

Many CDMs can be viewed as restricted latent class models that directly model the response probabilities as functions of discrete latent attributes. A common goal of cognitive diagnoses is to learn the examinees’ latent attributes, such as personalities or skills, based on their responses to a combination of specially designed test items. The $Q$-matrix plays a critical role in CDMs. It specifies the dependency structure between the test items and the latent attributes. Knowing the $Q$-matrix accurately is important because it is indispensable to cognitive diagnoses. Besides, the $Q$-matrix itself can be used to categorize the test items and enable efficient design of future assessments. However, in reality, many existing assessments do not even have the $Q$-matrix explicitly specified. Even the assessment providers specify the $Q$-matrix when designing the assessment, the specification may still be inaccurate. In many cases, one test item may potentially be linked to multiple attributes, but usually only the most direct and apparent ones are identified in the pre-designed $Q$-matrix. Therefore, it is of paramount importance to develop methodologies to efficiently learn the $Q$-matrix from the observational responses.

Various approaches have been proposed in the literature to learn the $Q$-matrix. Those methods can be generally classified into two categories, validation of the existing $Q$-matrix
and direct estimation of the $Q$-matrix from the observational data (Liu et al., 2012; Chen et al., 2015; Xu and Shang, 2018; Chung and Johnson, 2018; Chen et al., 2018; Culpepper, 2019). However, most of the existing estimation methods for the whole $Q$-matrix in general suffer from huge computational cost and are not scalable with the size of the $Q$-matrix; they either break down or are extremely computationally expensive even when the $Q$-matrix is moderately large. The high computational cost stems from the large number of configurations of the $Q$-matrix. If we view each binary element of the $Q$-matrix as a unique parameter, then the number of different configurations would grow exponentially with the size of the $Q$-matrix. In many applications, the number of latent attributes being tested is large, leading to a high-dimensional space for all possible latent attribute patterns. It is not uncommon that the number of potential attribute patterns is large, sometimes even larger than the sample size, making the estimation even more difficult. Such examples can be found in many applications, such as educational assessments (Lee et al., 2011; Choi et al., 2015) and the medical diagnosis of disease etiology (Wu et al., 2016); for instance, Sect. 4 presents a data set from the Trends in International Mathematics and Science Study (TIMSS), which has 13 binary latent attributes and $2^{13} = 8192$ attribute patterns while only 757 examinees. On the other hand, the number of items being tested may also be large in many applications. One example is the TIMSS mathematical test which often have more than 100 test items. Another example is the ADM admissions test, which is given twice a year and is used as an entrance test to universities and colleges, containing a total of 200 items (González and Wiberg, 2017). Therefore, it remains an open and challenging problem to learn the large $Q$-matrix from the observational data.

Borrowing the idea from the deep learning literature, we propose to use the restricted Boltzmann machines (RBMs) to learn the large $Q$-matrix. An RBM is a generative two-layer neural network that can learn a probability distribution over a collection of inputs (Smolensky, 1986). Among these inputs, some are observed variables while the others are latent variables that we do not observe, which matches the restricted latent class CDM setting. The weight matrix $W$ in RBMs determines the relationship between the observed variables and the latent variables. By learning this weight matrix $W$ under the framework of RBMs, we show that the structure of the $Q$-matrix in CDMs can be inferred accordingly. Although this is similar to the maximum likelihood learning approach, by tapping on RBMs, fast learning of the large $Q$-matrix can be achieved.

Our main contributions are that we identify the relationships between CDMs and RBMs, and propose a new way of learning the large $Q$-matrix efficiently. As far as we know, our proposed method is among the first ones in the literature that is scalable with the size of the $Q$-matrix (with computational cost of $O(J \times K)$) while at the same time retains high estimation accuracy. For example, comparing to Xu and Shang (2018) which attains an estimation accuracy of 71.2% in the GDINA setting with five independent latent attributes using 2000 observations, our method achieves more than 86% overall accuracy and much faster computational speed. Another interesting finding is that learning of the $Q$-matrix by RBMs is robust to different CDMs, including the DINA, ACDM and GDINA models. We provide theoretical guarantees under certain conditions and conduct simulation studies to support our findings. Besides, because of the unsupervised learning nature of RBMs, the traditional cross-validation (CV) procedure is not directly applicable. As a result, we also present a new CV procedure specifically to the $Q$-matrix learning setting.

The remaining parts of the paper are organized as follows. Section 1 gives reviews on CDMs and RBMs, and discussion of their relationships and why the learning of the $Q$-matrix by RBMs is achievable across different CDMs. Section 2 introduces our proposed estimation method and the new CV procedure. Section 3 consists of simulation studies on data generated from three typical CDMs. Section 4 demonstrates the performance of our proposed method through the data analysis on a TIMSS mathematics data set. Section 5 concludes with discussions and potential future
directions. All the proofs and additional simulation results can be found in the Supplementary Materials.

1. Estimation of Q-matrix Using RBMs

1.1. Review of CDMs

Many CDMs have been developed in recent decades, among which the deterministic input noisy output “And” gate model (DINA, Haertel, 1989; Junker and Sijtsma, 2001) is one of the most popular and simple models and serves as the foundation for many complex CDMs. Other popularly used CDMs include the noisy input deterministic “And” gate model (NIDA, Junker and Sijtsma, 2001), the reduced reparametrized unified model (R-RUM, Hartz, 2002), the general diagnostic model (GDM, von Davier, 2005), the deterministic input noisy “Or” gate (DINO, Templin and Henson, 2006), the log-linear CDM (LCDM, Henson et al., 2008), the additive CDM (ACDM, de la Torre, 2011) and the generalized DINA model (GDINA, de la Torre, 2011).

Consider a CDM with \( J \) items and \( K \) latent attributes. There are two types of variables for each subject: the observed responses for \( J \) items \( R = (R_1, ..., R_J) \) and the latent attribute pattern \( \alpha = (\alpha_1, ..., \alpha_K) \), which are both assumed to be binary. \( R_j \in \{0, 1\} \) denotes whether the examinee answers item \( j \) correctly and \( \alpha_k \in \{1, 0\} \) denotes possession or non-possession of the attribute \( k \). The \( Q \)-matrix, \( Q = (q_{j,k}) \in \{0, 1\}^{J \times K} \), specifies the dependence structure between the items and the latent attributes; \( q_{j,k} \in \{0, 1\} \) denotes whether a correct response to item \( j \) requires the latent attribute \( k \). If we denote the \( j \)-th row of the \( Q \)-matrix to be \( q_j \), then \( q_j \) reflects the full attribute requirements of item \( j \). For a latent attribute pattern \( \alpha \), we say \( \alpha \) possesses all the required attributes of item \( j \) if \( \alpha \preceq q_j \), where \( \alpha \preceq q_j \) means \( \alpha_k \preceq q_{j,k} \) for all \( k = 1, ..., K \). Different CDMs model the item response functions \( P(R_j = 1 \mid \alpha) \) differently with the item parameters constrained by the \( Q \)-matrix and specific cognitive diagnostic assumptions. Below we introduce three popular CDMs that will be considered in later discussions.

Example 1. (DINA model) Let \( R_{i,j} \in \{0, 1\} \) denote whether the subject \( i \) answers the item \( j \) correctly. Under the DINA model (Haertel, 1989; Junker and Sijtsma, 2001), for the \( j \)-th item and the \( i \)-th subject with the latent attribute pattern \( \alpha_i = (\alpha_{i,1}, ..., \alpha_{i,K}) \), the ideal response variable is defined as \( \xi_{i,j} = \prod_{k=q_{j,k}=1}^{K} \alpha_{i,k} = \prod_{k=1}^{K} \alpha_{i,k}^{q_{j,k}} \). The ideal response \( \xi_{i,j} = 1 \) only if \( \alpha_i \preceq q_j \), that is, the subject \( i \) needs to possess all the latent attributes required by the item \( j \) to have a positive ideal response. The uncertainty is further incorporated by two parameters: the slipping parameter \( s_j \) and the guessing parameter \( g_j \). Specifically, \( s_j = P(R_{i,j} = 0 \mid \xi_{i,j} = 1) \) and \( g_j = P(R_{i,j} = 1 \mid \xi_{i,j} = 0) \). The slipping parameter and the guessing parameter further satisfy \( 1 - s > g \), which indicates that the capable subjects will have higher positive probability than the incapable ones. The DINA model is one of the most restrictive and interpretable CDMs for dichotomously scored test items. It is a parsimonious model that requires only two parameters for each item regardless of the number of attributes required for the item. It is appropriate when the tasks call for the conjunction of several equally important attributes, and lacking one required attribute for the item is the same as lacking all the required attributes.

Example 2. (ACDM) In the ACDM, mastering additional required attributes will increase the positive response probability for the items. Specifically, if we take the identity link function in the ACDM, then for the \( j \)-th item and the \( i \)-th subject with attribute pattern \( \alpha_i = (\alpha_{i,1}, ..., \alpha_{i,K}) \), we have

\[
P(R_{i,j} = 1 \mid \alpha_i) = \delta_{j,0} + \sum_{k=1}^{K} \delta_{j,k} \alpha_{i,k} q_{j,k},
\]
which implies that mastering the kth attribute increases the probability of success on the item j by \( \delta_{j,k} \) if the kth latent attribute is required by the item j. Since there are no interaction terms in (1), the contribution of each latent attribute is independent from one another. If the subject i lacks all the required attributes for the item j, the term \( \sum_{k=1}^{K} \delta_{j,k} \alpha_{i,k} q_{j,k} \) would be 0, and the intercept \( \delta_{j,0} \) is the probability of correctly answering the item j based on pure guessing. Furthermore, even if the ith subject has all the required latent attributes of the item j, \( \delta_{j,0} + \sum_{k=1}^{K} \delta_{j,k} \alpha_{i,k} q_{j,k} \) may not sum to 1. In that case, \( 1 - (\delta_{j,0} + \sum_{k=1}^{K} \delta_{j,k} \alpha_{i,k} q_{j,k}) \) would be the probability of making a careless mistake. The ACDM is more appropriate to use when the items call for independent latent attributes but with different contributions to correct response to the items.

Besides the identity link function, other link functions are also proposed. One commonly used link function is the logit link,

\[
P(R_{i,j} = 1 | \alpha_i) = \sigma \left( \delta_{j,0} + \sum_{k=1}^{K} \delta_{j,k} \alpha_{i,k} q_{j,k} \right).
\]

where \( \sigma(x) = (1 + \exp(-x))^{-1} \). Equation (2) is also equivalent to \( \logit(P(R_{i,j} = 1 | \alpha_i)) = \delta_{j,0} + \sum_{k=1}^{K} \delta_{j,k} \alpha_{i,k} q_{j,k} \), which is the log-odds of a positive response. The interpretation would then become that each required latent attribute contributes independently to the log-odds of correcting answering item j by \( \delta_{j,k} \) in an additive fashion.

Example 3. (GDINA model) Both the DINA and ACDM models are special cases of the more general GDINA model (de la Torre, 2011). In addition to the intercept and the main effects in the ACDMs, the GDINA model also allows interactions among the latent attributes. Equation (3) gives the item response function for the GDINA model with identity link.

\[
P(R_{i,j} = 1 | \alpha_i) = \delta_{j,0} + \sum_{k=1}^{K} \delta_{j,k} \alpha_{i,k} q_{j,k} + \sum_{k=1}^{K-1} \sum_{k'=k+1}^{K} \delta_{j,k,k'} \alpha_{i,k} \alpha_{i,k'} q_{j,k,q_{j,k'}} + \ldots + \delta_{j,12...K} \prod_{k=1}^{K} \alpha_{i,k} q_{j,k}.
\]

The parameters in Eq. (3) can be interpreted as follows: \( \delta_0 \) is the probability of a correct response when none of the required attributes is present; \( \delta_k \) is the change in the probability of a correct response when only mastering a single attribute \( \alpha_k \); \( \delta_{k,k'} \), a first-order interaction effect, is the change in the probability of a correct response due to the possessing of both \( \alpha_k \) and \( \alpha_k' \) in addition to the main effects of mastering the two individual attributes; and \( \delta_{12...K} \) represents the change in the probability of a correct response due to the mastery of all the required attributes in addition to the main effects and all the lower-order interaction effects. Similarly to the ACDM model, \( P(R_{i,j} = 1 | \alpha_i) \) is not required to be 1 even when the subject i possesses all the required attribute for the item j. In that case, \( 1 - P(R_{i,j} = 1 | \alpha_i) \) is the probability of making a careless mistake. Moreover, the intercept \( \delta_{j,0} \) and the main effects are typically nonnegative, but the interaction effects can take on any values. Therefore, the GDINA model is appropriate if the mixed effects of latent attributes on the probability of a correct response is of interest.

1.2. Review of Restricted Boltzmann Machines

RBM}s are generative models that can learn probabilistic distributions over a collection of inputs. RBMs were initially invented under the name Harmonium by Smolensky (1986) and
gained currency due to their fast learnability in the mid-2000. It has found vast applications in dimension reduction (Hinton and Salakhutdinov, 2006), classification (Larochelle and Bengio, 2008), collaborative filtering (Salakhutdinov et al., 2007) and many other fields.

RBM can also be viewed as a probabilistic bipartite graphical models, with observed (visible) units in one part of the graph and latent (hidden) units in the other part. Typically, all the hidden units and the visible units are binary. In this work, we denote the visible units by $\mathbf{R} = \{R_1, ..., R_J\} \in \{0, 1\}^J$ and hidden units by $\mathbf{\alpha} = \{\alpha_1, ..., \alpha_K\} \in \{0, 1\}^K$, respectively. One key feature of RBMs is that only interactions between hidden units and visible units are allowed. There are neither connections among the visible units, nor any connections among the hidden units, as shown in Fig. 1.

RBM is characterized by the energy functions with the joint probability distribution specified as

$$P(\mathbf{R}, \mathbf{\alpha}; \theta) = \frac{1}{Z(\theta)} \exp \{-E(\mathbf{R}, \mathbf{\alpha}; \theta)\},$$

where $E(\mathbf{R}, \mathbf{\alpha}; \theta)$ is known as the energy function and $Z(\theta)$ is the partition function,

$$Z(\theta) = \sum_{\mathbf{R} \in \{0, 1\}^J} \sum_{\mathbf{\alpha} \in \{0, 1\}^K} \exp \{-E(\mathbf{R}, \mathbf{\alpha}; \theta)\},$$

which has been proved to be intractable (Long and Servedio, 2010). In specific, the energy function is given by

$$E(\mathbf{R}, \mathbf{\alpha}; \theta) = -\mathbf{b}^T \mathbf{R} - \mathbf{c}^T \mathbf{\alpha} - \mathbf{R}^T \mathbf{W} \mathbf{\alpha}$$

$$= - \sum_{j=1}^{J} R_j b_j - \sum_{k=1}^{K} \alpha_k c_k - \sum_{j=1}^{J} \sum_{k=1}^{K} R_j w_{j,k} \alpha_k,$$

where $\theta = \{\mathbf{b}, \mathbf{c}, \mathbf{W}\}$ are the model parameters, $\mathbf{b} \in \mathbb{R}^J$ are visible biases, $\mathbf{c} \in \mathbb{R}^K$ are hidden biases, and $\mathbf{W} \in \mathbb{R}^{J \times K}$ is the weight matrix describing the interactions between the visible and the hidden units.

Since no “$\mathbf{R}$-$\mathbf{R}$” or “$\mathbf{\alpha}$-$\mathbf{\alpha}$” interactions are allowed, the hidden and visible units are conditionally independent given each other, and therefore the joint conditional probability mass functions

![Figure 1](image_url)
can be factored into a product. This can be easily seen from Eqs. (4) and (5). Specifically, we have

$$P(R | \alpha; \theta) = \prod_{j=1}^{J} P(R_j | \alpha; b, W),$$

(6)

$$P(R_j = 1 | \alpha; b, W) = \sigma(b_j + \sum_{k=1}^{K} w_{j,k} \alpha_k),$$

(7)

and

$$P(\alpha | R; \theta) = \prod_{k=1}^{K} P(\alpha_k | R; c, W),$$

(8)

$$P(\alpha_k = 1 | R; c, W) = \sigma(c_k + \sum_{j=1}^{J} w_{j,k} R_j),$$

(9)

where $\sigma(x) = 1/(1 + \exp(-x))$ is the logistic sigmoid function.

RBMs and CDMs are in fact closely related. The binary observed item responses and the latent attributes in CDMs can be viewed as counterparts to the visible units and the hidden units in RBMs, respectively. There is a direct connection between the two. If we fit an ACDM with the logit link, where the conditional probability mass function (2) of the observed responses is modeled as a sigmoid function of the latent attributes, then it takes exactly the same form as the conditional probability function (7) of a visible unit given the hidden units in RBMs. Moreover, in an ACDM, $q_{j,k} = 0$ indicates that there is no interaction between the item $j$ and the latent attribute $k$, while in the weight matrix of an RBM, $w_{j,k} = 0$ also implies no interaction between the $j$th visible unit and the $k$th hidden unit. Therefore, we would expect that $w_{j,k} = 0$ in an RBM whenever $q_{j,k} = 0$ in a CDM.

Using the previous example in Fig. 1 for illustration, on the left of (10) is the weight matrix $W$ of an RBM, where $w_{j,k} \neq 0$ indicates the presence of the interaction between the visible unit $R_j$ and the hidden unit $\alpha_k$. The corresponding $Q$-matrix in a CDM can be implied as shown on the right. As we illustrate previously, the nonzero entries in the $Q$-matrix of an ACDM can be exactly inferred from the nonzero entries in the weight matrix $W$ in an RBM. Interactions among the latent attributes are allowed in the DINA and GDINA models, which violates the assumptions of an RBM. However, the $Q$-matrix is still estimable in these models. We give detailed arguments in Sect. 1.3.

$$W = \begin{bmatrix} w_{11} & 0 & w_{13} & 0 \\ 0 & w_{22} & 0 & w_{24} \\ w_{31} & 0 & w_{33} & 0 \\ w_{41} & 0 & 0 & w_{44} \\ 0 & w_{52} & w_{53} & 0 \end{bmatrix} \quad \Rightarrow \quad Q = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

(10)

1.3. Robust Estimation of $Q$-matrix

In the previous section, we have discussed that RBMs can be used to learn the $Q$-matrix for the ACDM with logit link. A natural question to ask is whether we can generalize this result to other CDMs such as the DINA and GDINA models. In this section, we will illustrate that under certain conditions, robust estimation of the $Q$-matrix by RBMs is indeed achievable for common
CDMs. In particular, we will demonstrate that the $Q$-matrix can be estimated correctly under the DINA and GDINA settings.

We focus on the learning of a particular row of the $Q$-matrix. It is in fact a variable selection problem of the required latent variables for that particular item of interest. Conditional on $\alpha$, we have discussed that RBMs are equivalent to the ACDM with the logit link, while the latter exactly corresponds to the logistic regression with canonical link and additive main effects linear predictor. Therefore in essence, RBMs can also be treated as main effect models. Starting with the simplest case, we shall first study the model selection consistency with linear additive models when the true models are the DINA or the GDINA model. Since it is still an open and challenging problem to establish consistent variable selection under complex latent variable models, here we start with the ideal case by assuming $\{\alpha_1, \ldots, \alpha_K\}$ are independent, that is, all the latent variables are independent. Although this is a strong assumption and is rarely fully satisfied in real-world scenarios, it can be relaxed in practice which is discussed in Remark 2.

Before giving formal statements, we first introduce some notations. Without loss of generality, we focus on the analysis of the response to one single item. For a subject with $\alpha = \{\alpha_1, \ldots, \alpha_K\}$, the response to the considered item is denoted by $R$, where for clarity, we omit the item index in the notation. Let $K^*$ to be the number of required attributes for the item. Without loss of generality, we let the first $K^*$ attributes be the required attributes for this item, that is, the corresponding row in the $Q$-matrix is $q = (1, \ldots, 1, 0, \ldots, 0)$ with the first $K^*$ entries being 1 and all the remaining $K - K^*$ entries being 0. For the response $R$ generated from the DINA or the GDINA model, we denote $E^*[R | \alpha]$ as the regression mean function for the mis-specified linear regression model of $R$ on $\alpha_1, \ldots, \alpha_K$. We show in the following propositions that the mis-specified mean function $E^*[R | \alpha]$ can identify the required attributes from the non-required ones.

**Proposition 1.** (DINA model) Assume $\{\alpha_1, \alpha_2, \ldots, \alpha_K\}$ are independent with $\alpha_k \sim \text{Bernoulli}(p_k)$ where $p_k \in (0, 1)$, $k = 1, 2, \ldots, K$. If $R$ is generated from the DINA model, then the mis-specified linear additive model of $R$ regressed on $(\alpha_1, \alpha_2, \ldots, \alpha_K)$ has the mean function in the form of $E^*[R | \alpha] = \beta_0 + \beta_1 \alpha_1 + \beta_2 \alpha_2 + \ldots + \beta_K \alpha_K$ with $\beta_l \neq 0$ for $l = 1, 2, \ldots, K^*$ and $\beta_k = 0$ for $k = K^* + 1, \ldots, K$.

Proposition 1 states that under the independence condition and if the data is generated from the DINA model, the significant variables included in the true model can be selected correctly using a mis-specified linear model with additive main effects only.

**Proposition 2.** (GDINA model) Assume $\{\alpha_1, \alpha_2, \ldots, \alpha_K\}$ are independent with $\alpha_k \sim \text{Bernoulli}(p_k)$ where $p_k \in (0, 1)$, $k = 1, 2, \ldots, K$. If $R$ is generated from the GDINA model satisfying the monotonicity assumption (i.e., acquiring an additional required skill $\alpha_k$, $k = 1, 2, \ldots, K^*$, will always increase the probability of a correct response), then the mis-specified linear additive model has the corresponding mean function in the form of $E^*[R | \alpha] = \beta_0 + \beta_1 \alpha_1 + \beta_2 \alpha_2 + \ldots + \beta_K \alpha_K$ with $\beta_l \neq 0$ for $l = 1, 2, \ldots, K^*$ and $\beta_k = 0$ for $k = K^* + 1, \ldots, K$.

Similar to Proposition 1, Proposition 2 states that under suitable conditions, the significant variables included in the true GDINA model can be selected correctly using a mis-specified linear model with additive main effects only. The detailed proofs for all the propositions can be found in Section 2 of the Supplementary Materials.

Propositions 1 and 2 demonstrate that the model selection consistency can be achieved using a mis-specified linear main effect model. As we illustrated previously, the conditional probability of a visible unit on the hidden units in RBMs can be regarded as a main effect logistic regression model. Therefore, we next give some intuition on why the main effect logistic regression model will give a similar variable selection result to the linear models. Consider a main effect logistic regression model with the canonical link function, that is, $\text{logit}(P(R | \alpha)) = \beta_0 + \beta_1 \alpha_1 + \ldots + \beta_K \alpha_K$. Then for a mis-specified linear model, the mis-specified mean function $E^*[R | \alpha]$ can give a similar variable selection result to the linear models. Consider a mis-specified linear model with the canonical link function, that is, $\text{logit}(P(R | \alpha)) = \beta_0 + \beta_1 \alpha_1 + \ldots + \beta_K \alpha_K$. Then for a mis-specified linear model, the mis-specified mean function $E^*[R | \alpha]$ can give a similar variable selection result to the linear models.
\( \beta_K \alpha_K \). Let \( \mathcal{R} = (R_i, i = 1, \ldots, N) \) denote the response vector for all the \( N \) subjects, and let \( \mu = (\mu_i := P(R_i \mid \alpha_i), i = 1, \ldots, N) \) denote the response probabilities for the subjects. We use \( A = (\alpha_i)_{i=1}^N \in \{0, 1\}^{N \times K} \) to denote the latent attribute matrix for the \( N \) subjects and \( A^* \) to denote the \( N \times (K+1) \) matrix \([1; A]\) with the first column being an all-one vector. In linear models, we usually use the least square estimation to estimate the coefficients, while in logistic regression, the iteratively re-weighted least square (IRLS) method is used. Next we will give some intuition on why these two estimation methods will produce similar variable selection results.

Conditional on \( \alpha_i \)'s, in the \((t + 1)\)th step of IRLS, the updating rule for parameter \( \theta := (\beta_0, \beta_1, \ldots, \beta_K) \) is

\[
\theta^{(t+1)} = (A^* W^{(t)} A^*)^{-1} A^* W^{(t)} Z^{(t)},
\]

where \( Z^{(t)} = A^* \theta^{(t)} + (W^{(t)})^{-1} (\mathcal{R} - \mu^{(t)}) \) is the \( t \)th step working response and \( W^{(t)} = \text{diag}(\mu_1^{(t)}(1-\mu_1^{(t)}), \ldots, \mu_N^{(t)}(1-\mu_N^{(t)})) \) is a diagonal weight matrix with diagonal elements being the variance estimates for each \( R_i \). Since there is no closed form of IRLS estimator and there is randomness in the convergence process, it is very challenging to study the theoretical properties of the \( \theta \) estimated by IRLS. So we only consider a one-step update of IRLS starting from the ideal case of true parameter \( \theta_{\text{true}} \) for illustration. It is reasonable to study this ideal case because IRLS will converge close to the \( \theta_{\text{true}} \) given the correct model specification and a large sample size. If we start with the true parameters, that is, we let \( \theta^{(0)} = \theta_{\text{true}} \), then,

\[
\theta^{(1)} = (A^* W_{\text{true}} A^*)^{-1} A^* W_{\text{true}} Z_{\text{true}},
\]

where the working response, \( Z_{\text{true}} = A^* \theta_{\text{true}} + W_{\text{true}}^{-1} (\mathcal{R} - \mu_{\text{true}}) \) is just a linear transformation of observed response \( \mathcal{R} \). Note that this update takes the same form as the weighted least square estimation of regressing \( Z_{\text{true}} \) on \( A^* \). Hence, the variable selection result in the linear model would be similar to that of the logistic regression. Combining Propositions 1 and 2, we have justified that the learning of the \( Q \)-matrix by RBMs is achievable across the DINA, ACDM and GDINA models with both identity and logit links.

Remark 1. In practice, it is not uncommon that some of the \( 2^K \) latent attribute patterns do not exist in the collected observations, especially when \( K \) is large. How negatively will this impact on the model selection consistency? In the DINA model, we see from the proof of Proposition 1 (see Section 2 of the Supplementary Materials) that to ensure the variable selection consistency for each required attribute \( \alpha_k, k = 1, \ldots, K^* \), we need to observe data from subjects with \( \{\alpha \mid \alpha_k = 0, \alpha_i = 1, i = 1, \ldots, k-1, k+1, \ldots, K^*\} \) and \( \{\alpha \mid \alpha_i = 1, i = 1, \ldots, K^*\} \) in the GDINA model, from the proof of Proposition 2, we can see that to ensure the variable selection consistency for each \( \alpha_k, k = 1, \ldots, K^* \), we need to observe data from subjects with \( \{\alpha \mid \alpha_k = 0\} \) and \( \{\alpha \mid \alpha_k = 1\} \). Therefore, even though some of the latent patterns may not exist in our observed data, the selection consistency is still achievable as long as the required attribute patterns are present.

Remark 2. The independent assumption on the latent attributes \( \{\alpha_1, \ldots, \alpha_K\} \) can be relaxed to some extent in practice. To see this, consider the setting when \( \{\alpha_1, \ldots, \alpha_K\} \) are possibly dependent but the response \( R \) only directly depends on the first \( K^* \) attributes \( \{\alpha_1, \ldots, \alpha_{K^*}\} \). Given \( \alpha_1, \ldots, \alpha_{K^*} \), the response \( R \) is conditionally independent of \( \alpha_k \) for all \( k = K^*+1, \ldots, K \). When only \( \alpha_1, \ldots, \alpha_{K^*} \) are present in the linear regression model of \( R \) regression on \( \alpha \)'s, consider adding in one additional
α_k, for any k = K^* + 1, ..., K, into the regression model, then its coefficient can be expressed as

$$
\beta_k = \frac{Cov\left(R - \mathbb{E}^* [R \mid \alpha_1, ..., \alpha_{K^*}], \alpha_k - \mathbb{E}^* [\alpha_k \mid \alpha_1, ..., \alpha_{K^*}] \right)}{Var\left(R - \mathbb{E}^* [R \mid \alpha_1, ..., \alpha_{K^*}] \right)},
$$

(11)

where we denote $\mathbb{E}^* [A \mid B]$ as the regression mean function of $A$ on $B$. Since $R$ and $\alpha_k$ are conditionally independent given $\alpha_1, ..., \alpha_{K^*}$, the numerator of (11) is expected to be small. In real implementations, the shrinkage imposed by the $L_1$ penalty in our proposed method should be able to recover most of these 0's. This is indeed supported by our simulation results in Sect. 3, where we consider moderate to high correlation regimes among the latent attributes and our proposed method still achieves satisfactory estimation accuracy of the underlying $Q$-matrix. Note also that in the special case when $K^* = 1$, the covariance term in (11) can be shown exactly equal to zero, in which case $\beta_k$ can be removed easily in the variable selection process. For a more detailed discussion for the $K^* = 1$ case, please refer to Section 2 of the Supplementary Materials.

Remark 3. The rigorous consistency theory of using RBMs to learn the $Q$-matrix under a general CDM setting can be difficult to establish. In the literature, even when the true models are binary RBMs, consistency for training RBMs is an open and challenging problem. Due to the intractable partition function in the binary RBM, an approximate likelihood maximizing approach has to be employed, such as the popularly used contrastive divergence (CD) algorithm that will be further introduced in Sect. 2. Even though there are many works in literature studying the asymptotic properties of the CD algorithm (MacKay, 2001; Yuille, 2004; Carreira-Perpinán and Hinton, 2005; Bengio and Delalleau, 2009; Sutskever and Tieleman, 2010; Jiang et al., 2018), whether and why the CD algorithm provides an asymptotically consistent estimate for binary RBMs are still open questions. Therefore, establishing a consistency theorem using a mis-specified RBM model for the DINA or the GDINA model as in this work is even more challenging, which is left for future exploration. Nevertheless, the CD algorithm in practice has showed empirical success in training RBMs, and our simulation results in Sect. 3 also demonstrate its effectiveness in training RBMs to learn the $Q$-matrix in CDMs.

2. Proposed Estimation Method

In this section, we will introduce our proposed method in detail. As we have illustrated in Sect. 1, nonzero entries in the $Q$-matrix can be inferred from the corresponding nonzero entries in the weight matrix of RBMs. Therefore, we are interested in a sparse solution of the weight matrix $W$. It is well known that $L_1$ penalty has the property of producing sparse solutions (Rosasco, 2009). Hence, we propose the following $L_1$ penalized likelihood as our objective function,

$$
\min_{\theta} - \log \left\{ P(R; \theta) \right\} + \lambda \sum_{j=1}^{J} \sum_{k=1}^{K} |w_{j,k}|.
$$

(12)

where $\log \{ P(R; \theta) \}$ is the marginal log-likelihood of the observed responses $R$, $\theta = \{ b, c, W \}$ are the model parameters, and $\lambda$ is a nonnegative tuning parameter for the $L_1$ penalty.
Gradient descent algorithm is a standard numerical method to solve problem (12). The likelihood part, following the derivation by Schlueter (2014), can be shown that its gradient with respect to the parameters has the following decomposition:

\[
\frac{\partial}{\partial \theta} \log \left( P(\mathbf{R}; \theta) \right) = - \sum_{\alpha \in \{0,1\}^K} P(\alpha | \mathbf{R}; \theta) \frac{\partial}{\partial \theta} E(\mathbf{R}, \alpha; \theta) + \sum_{\mathbf{r} \in \{0,1\}^J} P(\mathbf{r}, \alpha; \theta) \frac{\partial}{\partial \theta} E(\mathbf{r}, \alpha; \theta) \quad (13)
\]

\[
= \mathbb{E}_{P(\alpha | \mathbf{R}; \theta)} \left[ - \frac{\partial}{\partial \theta} E(\mathbf{R}, \alpha; \theta) \right] - \mathbb{E}_{P(\mathbf{r}, \alpha; \theta)} \left[ - \frac{\partial}{\partial \theta} E(\mathbf{r}, \alpha; \theta) \right]. \quad (14)
\]

In deep learning literature, this is a well-known decomposition into the positive phase and the negative phase of learning, corresponding to the two expectations in (14), respectively. As the two expectations do not have closed forms and are not directly tractable, researchers propose to approximate the gradient by estimating these expectations through Monte Carlo sampling. In particular, the positive phase corresponds to sampling the hidden units given the visible units, while the negative phase corresponds to obtaining the joint hidden and visible samples from the current model.

The bipartite graph structure of RBMs gives the special property of its conditional distributions \( P(\alpha | \mathbf{R}) \) and \( P(\mathbf{R} | \alpha) \) being factorial and simple to compute and sample from, as shown in Sect. 1.2. Therefore, sampling for the positive phase is straightforward while obtaining samples from the model for negative phase is not since it requires the joint hidden and visible samples. A widely used algorithm to learn RBMs is known as the contrastive divergence (CD) algorithm, where the negative phase is approximated by drawing samples from a short alternating Gibbs Markov chain between visible units and hidden units starting from the observed training examples (Hinton, 2002). In this work, we use a CD-1 algorithm where Gibbs chains are run for 1 step to approximate the gradient of the log-likelihood part. Specifically, given the original data \( \mathbf{R}^{(0)} \), we first sample \( \alpha^{(0)} \) according to Eqs. (8) and (9) to approximate the positive phase. Then, given \( \alpha^{(0)} \), we sample \( \mathbf{R}^{(1)} \) based on Eqs. (6) and (7), and we use \( (\mathbf{R}^{(1)}, \alpha^{(0)}) \) to approximate the negative phase.

At \((t + 1)\)th iteration, based on the sampled data, the parameters’ updates take the same form as gradient descent if we do not consider \( L_1 \) penalty,

\[
w^{(t+1)}_{j,k} \leftarrow w^{(t)}_{j,k} + \gamma^{(t)} \left\{ \sum_{i=1}^{N} R^{(0)}_{i,j} P(\alpha_{ik} = 1 | \mathbf{R}_i^{(0)}; \theta^{(t)}) - \sum_{i=1}^{N} R^{(1)}_{i,j} P(\alpha_{ik} = 1 | \mathbf{R}_i^{(1)}; \theta^{(t)}) \right\}, \quad (15)
\]

\[
b^{(t+1)}_{j} \leftarrow b^{(t)}_{j} + \gamma^{(t)} \left\{ \sum_{i=1}^{N} \sum_{k=1}^{K} R^{(0)}_{i,j,k} - \sum_{i=1}^{N} R^{(1)}_{i,j,k} \right\} / N, \quad (16)
\]

\[
c^{(t+1)}_{k} \leftarrow c^{(t)}_{k} + \gamma^{(t)} \left\{ \sum_{i=1}^{N} P(\alpha_{ik} = 1 | \mathbf{R}_i^{(0)}; \theta^{(t)}) - \sum_{i=1}^{N} P(\alpha_{ik} = 1 | \mathbf{R}_i^{(1)}; \theta^{(t)}) \right\} / N, \quad (17)
\]

where \( \mathbf{R}_i^{(0)} = (R_{i1}^{(0)}, R_{i2}^{(0)}, \ldots, R_{iJ}^{(0)}) \), \( \mathbf{R}_i^{(1)} = (R_{i1}^{(1)}, R_{i2}^{(1)}, \ldots, R_{iJ}^{(1)}) \) and \( \gamma^{(t)} \) is the learning rate for the \( r \)th iteration. Here, we denote the updated weight matrix by \( \mathbf{W}' = (W'_{j,k})_{J \times K} \), since we also need to consider the gradient of the \( L_1 \) penalty term later, and thus Eq. (15) is an intermediate update for the weight matrix. Detailed derivations can be found in the notes written by Schlueter.
In this work, we use a linearly decreasing learning rate scheme, which is guaranteed to converge as shown in Collins et al. (2008).

For the $L_1$ penalty term, we adopt the implementation developed by Tsuruoka et al. (2009), which can achieve more stable sparsity structures. As pointed out by Tsuruoka et al. (2009), the traditional implementation of $L_1$ penalty in gradient descent algorithm does not always lead to sparse models because the approximate gradient used at each update is very noisy, which deviates the updates away from zero.

The main idea of the implementation is to keep track of the total penalty and the penalty that has been applied to each parameter, and then the $L_1$ penalty is applied based on the difference between these cumulative values. By doing so, it is argued that the effect of noisy gradient is smoothed away. To be more specific, at iteration $t$, let $u(t) := \lambda \sum_{l=1}^{t} \gamma(t)$ be the absolute value of the total $L_1$ penalty that each parameter could have received up to the point, where $\gamma(t)$ is the learning rate at step $l$. Let $c_{j,k}^{(t-1)} := \sum_{l=1}^{t-1} (w_{j,k}^{(l+1)} - w_{j,k}^{(l)})$ be the total $L_1$ penalty that $w_{j,k}$ has actually received up to step $t$, where $w_{j,k}^{(l)}$ is the intermediate update at step $l$ calculated by Eq. (15). Then at iteration $(t + 1)$, we update $w_{j,k}^{(t+1)}$ by

$$
\begin{align*}
    w_{j,k}^{(t+1)} &\leftarrow \max \left\{ 0, w_{j,k}^{(t+1)} - (u(t) + c_{j,k}^{(t-1)}) \right\} & \text{if } u_{j,k}^{(t+1)} > 0, \\
    w_{j,k}^{(t+1)} &\leftarrow \min \left\{ 0, w_{j,k}^{(t+1)} + (u(t) - c_{j,k}^{(t-1)}) \right\} & \text{if } u_{j,k}^{(t+1)} \leq 0.
\end{align*}
$$

Since the updates in Eqs. (15), (16) and (17) require summations over all the data samples, it would be computationally expensive when the sample size is large. To reduce computational burden, we implement a batch version of the CD-1 algorithm in practice, where we only use a small batch of the whole data set in each iteration. Specifically, we randomly partition the whole data set into $B$ batches, and iterating through all the batches is known as one epoch in machine learning literature. Here, we use $R = \{ R(1), R(2), \ldots, R(B) \}$ to denote the partitions, $N_B$ to denote the batch size, and $N_{epoch}$ to denote the number of epochs. The resulting algorithm is summarized in Algorithm 1.

In our proposed algorithm, there are two tuning parameters: $\lambda$ for the $L_1$ penalty and $\gamma_0$ for the learning rate. To get good estimates of our model, we need to select a suitable combination of hyper-parameters $\lambda$ and $\gamma_0$. A popularly used tuning procedure is cross-validation (CV). However, as RBMs are unsupervised learning models, we cannot rely on the so-called “test error” of the labels. Instead, since visible units are re-sampled at each iteration in the CD algorithm, we may use the reconstruction error of the visible units to assess the goodness of fit. Nevertheless, the visible reconstruction error will always increase as the penalty coefficient $\lambda$ increases, because larger penalty would introduce more bias. Therefore, the traditional CV procedures would not work here. To solve this problem, given values of $\lambda$ and $\gamma_0$, instead of directly using the $\hat{W}_{\lambda,\gamma_0}$ obtained from a penalized RBM to compute the reconstruction error, we propose to debias the nonzero entries in $\hat{W}_{\lambda,\gamma_0}$ by training an RBM with no penalty but fixing the zero positions the same as those in $\hat{W}_{\lambda,\gamma_0}$. The proposed CV procedure is summarized below.

1. Split the data into $M$ partitions. Each time we use one partition as the validation set and the remaining as the training set.
2. Apply the penalized CD Algorithm 1 to train the RBM on the training set with pre-specified $\lambda$ and $\gamma_0$, and obtain the estimates $\hat{W}_{\lambda,\gamma_0}$ and $\hat{Q}_{\lambda,\gamma_0}$.
3. Use the training set again to debias the nonzero entries of $\hat{W}_{\lambda,\gamma_0}$. Specifically, we use $\hat{W}_{\lambda,\gamma_0}$ as the initial value and set $\lambda = 0$ in Algorithm 1 to train an unpenalized RBM, and only update the nonzero entries of $\hat{W}_{\lambda,\gamma_0}$ while keeping the zero entries unchanged.
Algorithm 1: CD-1 algorithm with $L_1$ penalty

**Input:** Data $R = \{R^{(1)}_1, R^{(1)}_2, \ldots, R^{(1)}_B\}$, $\lambda$, $\gamma_0$, and $N_{epoch}$.

**Output:** Estimates $\hat{W}, \hat{b}, \hat{c}$.

Initialize $w^{(0)}_{j,k}, b^{(0)}_j, c^{(0)}_k, u^{(0)} = 0, c^{(0)}_{j,k} = c^{(1)}_{j,k} = 0$;

for $e = 0, \ldots, N_{epoch} - 1$ do

for $b = 0, \ldots, B - 1$ do

$t = e \times B + b$ (the number of iterations);

$\gamma^{(t)} = \frac{\gamma_0}{t + 1}$;

$R^{(0)}(t) \leftarrow R^{(0)}(b + 1)$;

Sample $\alpha^{(0)} \sim P(\alpha \mid R^{(0)}(0), c^{(0)}, W^{(t)})$;

Sample $R^{(1)}(t) \sim P(R \mid \alpha^{(0)}, b^{(t)}, W^{(t)})$;

$u^{(t)} \leftarrow u^{(t-1)} + \lambda \gamma^{(t)}$;

for $j = 1, \ldots, J, k = 1, \ldots, K$ do

$w^{(t+1)}_{j,k} \leftarrow w^{(t)}_{j,k} + \gamma^{(t)} \left\{ \sum_{i=1}^{N_B} R^{(0)}_{i,j} P(\alpha_{ik} = 1 \mid R^{(0)}_{i,j}) - \sum_{i=1}^{N_B} R^{(1)}_{i,j} P(\alpha_{ik} = 1 \mid R^{(1)}_{i,j}) \right\}$;

if $t \geq 2$ then

$c^{(t-1)}_{j,k} \leftarrow c^{(t-2)}_{j,k} + w^{(t)}_{j,k}$;

if $w^{(t+1)}_{j,k} > 0$ then

$w^{(t+1)}_{j,k} \leftarrow \max \left\{ 0, w^{(t+1)}_{j,k} - (u^{(t)} + c^{(t-1)}_{j,k}) \right\}$;

else

$w^{(t+1)}_{j,k} \leftarrow \min \left\{ 0, w^{(t+1)}_{j,k} + (u^{(t)} - c^{(t-1)}_{j,k}) \right\}$;


eend

end

for $j = 1, \ldots, J$ do

$b^{(t+1)}_j \leftarrow b^{(t)}_j + \gamma^{(t)} \left\{ \sum_{i=1}^{N_B} R^{(0)}_{i,j} - \sum_{i=1}^{N_B} R^{(1)}_{i,j} \right\} / N_B$;

end

for $k = 1, \ldots, K$ do

$c^{(t+1)}_k \leftarrow c^{(t)}_k + \gamma^{(t)} \left\{ \sum_{i=1}^{N_B} P(\alpha_{ik} = 1 \mid R^{(0)}_i) - \sum_{i=1}^{N_B} P(\alpha_{ik} = 1 \mid R^{(1)}_i) \right\} / N_B$;

end

end

Hidden bias $c$ and visible bias $b$ are updated at each step as usual. This step gives us the de-biased weight matrix $\hat{W}_{\lambda, \gamma_0}$.

4. Compute the reconstruction error on the validation set. In specific, at each iteration of the CD algorithm, we fix $W = \hat{W}_{\lambda, \gamma_0}$, and only update the hidden and visible biases. The reconstruction error is computed as the mean batch squared error between the latest sampled visible batches $\{R^{(1)}_1, \ldots, R^{(1)}_m\}$ and the observational batches $\{R^{(0)}_1, \ldots, R^{(0)}_m\}$ in the validation set.
5. For each combination of $\lambda$ and $\gamma_0$ in the candidate set, we repeat Steps 2-4 across all $M$ validation sets. The $\hat{Q}_{\lambda^*, \gamma_0^*}$ corresponding to the smallest mean batch squared error (see Sect. 3 for definition) is taken as the final estimate of the $Q$-matrix.

Another major difference from the traditional CV procedure is we select the $Q$-matrix corresponding to the smallest validation error instead of taking average of the validation errors and then training a new RBM with the best tuning parameters according to the smallest mean error. There are two advantages. On the one hand, the traditional way of averaging errors, though more stable, is very time-consuming in this problem. On the other hand, the gradient descent steps in the CD algorithm may only produce locally optimal results. To avoid being stuck in sub-optima, we run the CD algorithm $M$ times with different initializations and different training and validation sets for each combination of $\lambda$ and $\gamma_0$, and select the estimated $Q$-matrix corresponding to the smallest validation error. By doing so, the $Q$-matrix is expected to be more accurately estimated.

Remark 4. The computational cost of our proposed method only grows linearly in $K$ and this enables estimation of very large $Q$-matrices. As far as we know, the current methods in the literature have computational cost greater than $O(K)$, with the majority growing exponentially with $K$. For example, in Xu and Shang (2018), they proposed to learn the $Q$-matrix by estimating the coefficients in the LCDM plus a penalty term with the EM algorithm. In the E-step of the EM algorithm, $2^K$ posterior probabilities for each of the attribute patterns need to be updated. However, we also point out that there may be alternative approaches that are also computationally feasible. Thanks to one of the reviewers, who suggests it may also be feasible to use the traditional ACDM to learn large $Q$-matrices. Note that all of our arguments in Sect. 1.3 also apply to the ACDM model. With a high-order model that parameterizes the distribution of the binary vector of attributes, such as a probit model, the number of parameters that need to be learned can be reduced from $2^K$ to $O(K^2)$. Together with a stochastic gradient descent algorithm, this can also be a computationally feasible approach.

3. Simulation Studies

We conduct simulation studies on three popular CDMs, the DINA, ACDM and GDINA models, to study the performance of our proposed method in learning the $Q$-matrix under different CDM settings. In particular, we examine the scalability to the size of the $Q$-matrix and the estimation accuracy of the proposed algorithm.

We first introduce the metrics used to evaluate the performance of the proposed estimation method. To measure the convergence of the algorithm, we investigate the change in the mean batch errors against time. The mean batch error is the reconstruction error between the latest sampled visible batches $\{R_{(1)}^{(1)}, \ldots, R_{(B)}^{(1)}\}$ and the original observed batches $\{R_{(1)}^{(0)}, \ldots, R_{(B)}^{(0)}\}$, where $\{R_{(1)}^{(0)}, \ldots, R_{(B)}^{(0)}\}$ partitions the whole observed data set into $B$ batches. Given the batch-size $N_B$, the mean batch error is defined as

$$\frac{1}{BN_B} \sum_{b=1}^{B} \sum_{i=1}^{N_B} \sum_{j=1}^{J} \left(R_{(b),i,j}^{(1)} - R_{(b),i,j}^{(0)}\right)^2.$$

To evaluate the estimation accuracy, we report entry-wise overall percentage error (OE), out of true positives percentage error (OTP) and out of true negatives percentage error (OTN). Specifically,

$$\text{OE} := \frac{1}{JK} \sum_{j=1}^{J} \sum_{k=1}^{K} \mathbb{I}\{\hat{q}_{j,k} \neq q_{j,k}\}.$$
which is the percentage of wrongly estimated entries out of the total number of entries in the Q-matrix.

\[
\text{OTP} := \frac{\sum_{j=1}^{J} \sum_{k=1}^{K} \mathbb{1}\{\hat{q}_{j,k} = 0, q_{j,k} = 1\}}{\sum_{j=1}^{J} \sum_{k=1}^{K} \mathbb{1}\{q_{j,k} = 1\}},
\]

which is defined as the percentage of wrongly estimated entries out of all true positive entries (i.e., entries 1) in the Q-matrix.

\[
\text{OTN} := \frac{\sum_{j=1}^{J} \sum_{k=1}^{K} \mathbb{1}\{\hat{q}_{j,k} = 1, q_{j,k} = 0\}}{\sum_{j=1}^{J} \sum_{k=1}^{K} \mathbb{1}\{q_{j,k} = 0\}},
\]

which is defined as the percentage of wrongly estimated entries out of all true negatives (i.e., entries 0) in the Q-matrix. A challenge in computing these errors arises because the estimated Q-matrix can only be identified up to column permutations. To resolve this problem, we apply the Hungarian algorithm to match the columns of the estimated \(\hat{Q}\) to the true Q-matrix by jointly minimizing the total column-wise matching errors. Details of the Hungarian algorithm can be found in Kuhn (1955).

We consider different number of latent attributes \(K = 5, 10, 15, 20, 25\). To ensure the Q-matrix is identifiable so that it can be learned from the observational data, we specify it as follows:

\[
Q = \begin{bmatrix} I_K \\ Q_1 \\ Q_2 \end{bmatrix},
\]

where \(I_K\) is a \(K\)-dimensional identity matrix; \(Q_1 \in \{0, 1\}^{K \times K}\) with value 1 in the \((i, i)\)th entries for \(i = 1, ..., K\) and the \((i, i+1)\)th entries for \(i = 1, ..., K-1\), and values 0 for all the other entries; \(Q_2 \in \{0, 1\}^{K \times K}\) with value 1 in entries \((i, i)\) for \(i = 1, ..., K\), \((i, i-1)\) for \(i = 2, ..., K\) and \((i, i+1)\) for \(i = 1, ..., K-1\), and value 0 for all the remaining entries. The above construction sets the number of items to be \(J = 3K\). This Q-matrix satisfies the identifiability conditions in Gu and Xu (2019) and therefore is identifiable under the DINA setting in Simulation Study 3.1. Moreover, this construction also ensures the (generic) identifiability of the ACDM and GDINA models considered in Simulation Studies 3.2 and 3.3 (Xu, 2017; Gu and Xu, 2020a,b). A random design of the Q-matrix, in which its identifiability is not be guaranteed, is also considered in Section 1.1 of the Supplementary Materials.

In each simulation study, we consider two different sample sizes \(N = 2000\) or 10000. Both independent and dependent settings of latent attributes are explored. Denote the latent attribute matrix by \(A = (\alpha_i)_{i=1}^{N} \in \{0, 1\}^{N \times K}\), which depicts the latent attribute patterns of the \(N\) examinees. We use two steps to simulate the latent patterns (Chen et al., 2015). First, a Gaussian latent vector is generated for each subject \(z_i = (z_{i1}, ..., z_{iK}) \sim \mathcal{N}(0, \Sigma)\) for \(i = 1, ..., N\), where \(\Sigma = (1 - \rho)I_K + \rho I_K 1_K^\top\), \(1_K = (1, ..., 1)^\top\) and \(\rho\) is the correlation between any two different latent attributes. In practice, since some attributes may be harder to master than others, different thresholds are applied in the sampling of attribute profiles. In particular, for a given \(K\), we specify the thresholds ranging from \(-0.5\) to \(0.5\), with a step size of \(1/(K - 1)\), for each of attribute \(1, 2, ..., K\), respectively. Then, \(\alpha_{ik} = 1\) if \(z_{ik}\) is greater than its respective threshold and \(\alpha_{ik} = 0\) otherwise. For the independent setting, we set \(\rho = 0\), while for the dependent settings, we consider both a low correlation with \(\rho = 0.25\) and a high correlation with \(\rho = 0.75\). For
the tuning of hyper-parameters, we take the candidate sets as \( \lambda \in \{0.003, 0.004, \ldots, 0.015\} \) and \( \gamma_0 \in \{0.5, 1, \ldots, 5.5\} \), and perform 5-fold CV to select the best estimated \( Q \)-matrix. For each setting, 100 repetitions are simulated. The batch size and the number of epochs are fixed at 50 and 300, respectively.

### 3.1. Simulation Study 1. DINA Model

For the DINA test items, we consider two uncertainty levels, \( g_j = s_j = 0.1 \) or \( g_j = s_j = 0.2 \) for all \( j = 1, \ldots, J \). Figure 2 plots the mean batch errors against time for the independent case with \( K = 5 \) (the first row) and \( K = 25 \) (the second row) across different sample sizes and different noise levels. When \( K = 5 \), we can see that the CD-1 algorithm converge well after 6 seconds for all different sample sizes under different noise levels. This suggests that with a small number of latent attributes, the sample sizes and the uncertainty levels do not affect the convergence speed a lot. Focusing on the second row of Fig. 2, we note that although the size of the \( Q \)-matrix increases from 75 (\( K = 5 \)) to 1875 (\( K = 25 \)), the convergence time only increases by around 10 seconds, and the CD-1 algorithm converges well after just 15 seconds even when \( K = 25 \). This indicates that the proposed method is scalable with the size of the \( Q \)-matrix. Dependent settings have similar convergence rates and hence the results are omitted.

Figures 3 and 4 plot different estimation errors against the sizes of the \( Q \)-matrix for independent and dependent settings, respectively. For the independent case, in Fig. 3, we can see that the OE stays below 16% across all the settings. There is a decreasing trend in the OE as the \( Q \)-matrix size increases due to the increasing sparsity of the true underlying \( Q \)-matrix. Our proposed method performs significantly better than the baseline method predicting all the entries of the \( Q \)-matrix to be 0 (which would produce OE of 36% for \( K = 5 \)). Furthermore, we note that increasing uncertainty level will deteriorate the OTP, making the estimation of positive entries harder. Increasing the sample size \( N \) would in general help improve the estimation accuracy. For the dependent case, in Fig. 4, we observe that the results in the low correlation setting are very similar to that of the independent setting. This suggests that our proposed method is robust when moderate correlations among latent attributes exist. On the other hand, when the correlations among the attributes are high, we see increments in all the three error metrics, OE, OTP and OTN. The correlations among the attributes would compound the difficulty in estimation of the \( Q \)-matrix. However, all the OE’s still stay well below 20%. Hence, our proposed method can still achieve effective learning of the \( Q \)-matrix when the correlations among the attributes are high.

### 3.2. Simulation Study 2. ACDM Model

We conduct similar analysis using data generated from the ACDM to examine the convergence speed and estimation accuracy of our proposed method. Define \( K_j^* \) to be the number of required attributes for the item \( j \). Without loss of generality, we let the first \( K_j^* \) attributes be the required attributes for item \( j \), i.e., the corresponding row in the \( Q \)-matrix is \( q_{j} = (1, \ldots, 1, 0, \ldots, 0) \) with the first \( K_j^* \) entries being 1 and all the remaining \( K - K_j^* \) entries being 0. For an ACDM with the identity link function (1), we have \( P(R_j = 1 \mid \mathbf{l}_K) = \delta_{j,0} + \sum_{k=1}^{K_j^*} \delta_{j,k} := p_j \), the highest success probability achievable for the most capable subjects. Similar to the DINA setting, two different uncertainty levels are considered: case 1. \( \delta_{j,0} = 0.1 \), \( p_j = 0.9 \) for all \( j = 1, \ldots, J \) and case 2. \( \delta_{j,0} = 0.2 \), \( p_j = 0.8 \) for all \( j = 1, \ldots, J \). For \( k = 1, \ldots, K_j^* \), \( \delta_{j,k} \) is set to be \( (p_j - \delta_{j,0})/K_j^* \), that is, the contribution of each required attribute to the success probability is equal.

Figure 5 shows the convergence speed of our proposed method under the independent setting. We observe similar patterns as in the DINA case: uncertainty levels and sample sizes do not have significant impacts on the convergence speed. Our proposed algorithm is scalable with the size of the \( Q \)-matrix in the ACDM setting. Figures 6 and 7 plot different estimation metrics against the
Figure 2. Plots of mean batch errors against time for the DINA data.
Figure 3. Plots of different performance metrics against the size of the Q-matrix for the DINA data (independent case).
FIGURE 4.
Plots of different performance metrics against the size of the $Q$-matrix for the DINA data (dependent case with $g = s = 0.1$). Rows 1 and 2 correspond to correlation settings 0.25 and 0.75, respectively.
size of the $Q$-matrix for independent and dependent settings, respectively. From Fig. 6, we can see that the results are very similar to those observed in the DINA model setting, which demonstrates that our proposed methods are effective in the ACDM data. Furthermore, for the dependent setting in Fig. 7, we observe that when the correlation is of 0.25, the estimation accuracy remains similar to that in the independent settings. When the correlation is of 0.75, unlike in the DINA setting, the OE, OTP and OTN only increase very slightly. In particular, the OE stays well below 16.5% to that in the independent settings. When the correlation is of 0.75, unlike in the DINA setting, we observe that when the correlation is of 0.25, the estimation accuracy remains similar to the independent setting when the correlation is of 0.25. When the correlations are increased to 0.75, all the three error metrics only increase very slightly. This observation is similar to the ACDM setting. The OE's remain well below 16.5% for all $K = 5, 10, \ldots, 25$. This suggests that when the true data generating model is the ACDM, our proposed method is robust when the correlations among the attributes are high.

### 3.3. Simulation Study 3. GDINA Model

Let the highest success probability achievable for the most capable subjects be $P(R_j = 1 | \mathbf{1}_K) := p_j$ from Eq. (3). Similar to the ACDM setting, we consider two uncertainty levels: case 1. $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$ and case 2. $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. Using the $Q$-matrix specified at the beginning of this section, for each item $j$, we may have $K_j = 1, 2$ or 3. When $K_j = 1$, we set $\delta_{j,k} = p_j - \delta_{j,0}$. When $K_j = 2$, we let $\delta_{j,k} = \delta_{j,k'} = (p_j - \delta_{j,0})/3$ and when $K_j = 3$ we set $\delta_{j,k} = \delta_{j,k'} = \delta_{j,k''} = (p_j - \delta_{j,0})/7$. As such, the main effects and the interaction terms are all assumed to have the same contributions to the probability of a positive response. Both independent and dependent settings are considered.

Convergence rates under independent setting are summarized in Fig. 8. Similar patterns to the DINA and the ACDM settings can be observed, indicating that our algorithm is scalable to the size of the $Q$-matrix in the GDINA model. As before, dependent settings have similar convergence patterns, and hence the results are not presented here. Behaviors of different estimation metrics over the size of the $Q$-matrix for both the independent and dependent settings are summarized in Figs. 9 and 10, respectively.

For the independent setting in Fig. 9, slightly better estimation accuracy can be observed than in the DINA and the ACDM settings. This suggests that our proposed methods is effective in the learning the $Q$-matrix from data generated using the GDINA model. One thing to emphasize is that our method is competitive among the existing algorithms in the literature. For example, comparing to a similar simulation study in Xu and Shang (2018) for $K = 5$ independent attributes and $N = 2000$, our overall estimation accuracy of around 87% is significantly better than theirs, whose overall accuracy is 71.2%. Moreover, our method also has much smaller computational cost than their method. For the dependent setting in Fig. 10, we observe that the estimation accuracy remains similar to the independent setting when the correlation is of 0.25. When the correlations are increased to 0.75, all the three error metrics only increase very slightly. This observation is similar to the ACDM setting. The OE's remain well below 16.5% for all $K = 5, 10, \ldots, 25$. This suggests that when the true data generating model is the GDINA model, the proposed method is fairly robust to high attribute correlations.

### 3.4. Attribute Classifications

As discussed in Sect. 1.3, the marginal distributions of the attributes are mis-specified in RBMs, in which a conditional independent structure is assumed. However, in practice, the latent attributes are often highly correlated and the conditional independence assumption may not hold. This mis-specification in latent attribute distributions is expected to bring in additional errors in the estimated $Q$-matrix. In order to understand the practical implications of the mis-specification in the estimated $Q$-matrix, we compare the commonly used attribute classification accuracy (ACC) rate obtained using the estimated $Q$-matrix ($\hat{Q}$) and the true $Q$-matrix ($Q$). In particular, when there are $N$ examinees, the ACC of the $k$'th attribute is defined as
Figure 5. Plots of mean batch errors against the time for the ACDM data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 
Figure 6.
Plots of different performance metrics against the size of the $Q$-matrix for the ACDM data (independent case). Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 
Figure 7. Plots of different performance metrics against the size of the $Q$-matrix for the ACDM data (dependent case with $\delta_{0,j} = 0$, $\rho_j = 0.9$ for all $j = 1, \ldots, J$). Rows 1 and 2 correspond to correlation settings $0.25$ and $0.75$, respectively.
Figure 8.
Plots of mean batch errors against the size of the $Q$-matrix for the GDINA data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 

**FIGURE 8.**
Plots of mean batch errors against the size of the $Q$-matrix for the GDINA data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 

**FIGURE 8.**
Plots of mean batch errors against the size of the $Q$-matrix for the GDINA data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 

**FIGURE 8.**
Plots of mean batch errors against the size of the $Q$-matrix for the GDINA data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 

**FIGURE 8.**
Plots of mean batch errors against the size of the $Q$-matrix for the GDINA data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 

**FIGURE 8.**
Plots of mean batch errors against the size of the $Q$-matrix for the GDINA data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 

**FIGURE 8.**
Plots of mean batch errors against the size of the $Q$-matrix for the GDINA data. Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$.
Plots of different performance metrics against the size of the $Q$-matrix for the GDINA data (independent case). Case 1 represents the setting when $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$. Case 2 represents the setting with higher uncertainty levels when $\delta_{j,0} = 0.2$, $p_j = 0.8$ for all $j = 1, \ldots, J$. 
Figure 10.
Plots of different performance metrics against the size of the $Q$-matrix for the GDINA data (dependent case with $\delta_{j,0} = 0.1$, $p_j = 0.9$ for all $j = 1, \ldots, J$). Rows 1 and 2 correspond to correlation settings 0.25 and 0.75, respectively.
Table 1. Average ACC rates out of 100 repetitions for \( K = 5, 10 \) attributes, respectively, obtained using the true CDMs.

| Model   | \( N = 2000 \) | \( N = 10000 \) |
|---------|----------------|-----------------|
|         | \( \rho = 0.25 \) | \( \rho = 0.75 \) | \( \rho = 0.25 \) | \( \rho = 0.75 \) |
|         | \( \hat{Q} \) | \( Q \) | \( \hat{Q} \) | \( Q \) | \( \hat{Q} \) | \( Q \) | \( \hat{Q} \) | \( Q \) |
| \( K = 5 \) | DINA | 0.806 | 0.944 | 0.888 | 0.956 | 0.830 | 0.945 | 0.890 | 0.957 |
|         | ACDM | 0.812 | 0.926 | 0.911 | 0.946 | 0.921 | 0.928 | 0.915 | 0.948 |
|         | GDINA | 0.918 | 0.928 | 0.935 | 0.949 | 0.928 | 0.929 | 0.947 | 0.950 |
| \( K = 10 \) | DINA | 0.801 | 0.939 | 0.898 | 0.954 | 0.811 | 0.940 | 0.894 | 0.956 |
|         | ACDM | 0.815 | 0.922 | 0.913 | 0.946 | 0.910 | 0.925 | 0.906 | 0.950 |
|         | GDINA | 0.885 | 0.924 | 0.939 | 0.949 | 0.926 | 0.926 | 0.899 | 0.951 |

\( \hat{Q} \) and \( Q \) denote the estimated \( Q \)-matrix from the proposed method and the true \( Q \)-matrix, respectively.

\[
ACC(k) := \frac{1}{N} \sum_{i=1}^{N} |\hat{\alpha}_{ik} - \alpha_{ik}|
\]

where \( \hat{\alpha}_{ik} \) and \( \alpha_{ik} \) represent the estimated and the true attribute values, respectively.

The simulation setups remain the same as the dependent settings in Sect. 3. All the DINA data, the ACDM data and the GDINA data are considered. Attribute classifications are performed using the estimated \( \hat{Q} \) and the true \( Q \) under the corresponding true underlying CDMs. The results are summarized in Table 1.

Not surprisingly, we observe that the ACC rates obtained using \( \hat{Q} \) are worse than that using \( Q \) in all settings across all models. The errors in \( \hat{Q} \) stem from two sources, the mis-specification error in the latent attributes’ marginal distribution and the sample estimation error. On the other hand, we also note that the ACC rates obtained using \( \hat{Q} \) do not deteriorate too much from using the true \( Q \) when sample size is large, especially under the ACDM and GDINA models. This suggests that the \( Q \)-matrix estimation accuracy in the ACDM and GDINA models may be less prone to the mis-specification in the latent attributes’ marginal distribution. Furthermore, the ACC rates drop as the number of attributes increases in the model. This reflects the increasing difficulty in attribute classifications as the number of attributes increments. Surprisingly, the ACC rates are generally higher when the correlation among attributes is higher. This may be because the higher dependency among the attributes results in fewer numbers of possible attribute patterns, making the estimation relatively easier. Not so surprisingly, we also observe that increasing sample size can in general help improve ACC rates.

We also conduct simulation studies to explore the potential of using the proposed method to perform latent attribute classifications directly. The performance of the proposed method in attribute classifications is satisfactory. For more details on the additional simulation results, please refer to the Supplementary Materials.

4. Real Data Analysis

We apply our proposed method to a TIMSS data set. TIMSS provides data on the mathematics and science curricular achievement of the fourth and the eighth-grade students across countries such as the USA. The data set contains 23 mathematical items from TIMSS 2003 items and is packed in the CDM package in R (Robitzsch et al., 2020). Both a binary scored examinees’ response matrix and an associated expert constructed \( Q \)-matrix are included in the data set. In
Table 2.
Clusters of items according to the underlying skill attributes.

| Skill attributes                                                                 | Items         |
|----------------------------------------------------------------------------------|--------------|
| 1. Understand concepts of a ratio and a unit rate and use language appropriately   | 1, 7, 20     |
| 2. Use ratio and rate reasoning to solve real-world and mathematical problems     | 3, 11, 15, 19, 22 |
| 3. Compute fluently with multi-digit numbers and find common factors and multiples | 12, 18       |
| 4. Apply and extend previous understandings of numbers to the system of rational numbers | 4, 17, 23 |
| 5. Apply and extend previous understandings of arithmetic to algebraic expressions | 8, 13, 16, 21 |
| 6. Reason about and solve one-variable equations and inequalities                 | 2, 5, 6, 10, 14 |
| 7. Recognize and represent proportional relationships between quantities          | 3, 6         |
| 8. Use proportional relationships to solve multi-step ratio and percent problems   | 11           |
| 9. Apply and extend previous understandings of operations with fractions to add, subtract, multiply and divide rational numbers | 4, 8, 18, 23 |
| 10. Solve real-life and mathematical problems using numerical and algebraic expressions and equations | 5 |
| 11. Compare two fractions with different numerators and different denominators; Understand a fraction $a/b$ with as $a > 1$ a sum of fractions $1/b$ | 1, 9, 18 |
| 12. Solve multi-step word problems posed with whole numbers and having whole number answers using the four operations, including problems in which remainders must be interpreted. Represent these problems using equations with a letter standing for the unknown quantity; generate a number or shape pattern that follows a given rule. Identify apparent features of the pattern that were not explicit in the rule itself | 5, 15 |
| 13. Use equivalent fraction as a strategy to add and subtract fractions           | 1, 12, 18    |

particular, the binary response matrix consists of 757 observations, and it is therefore of dimension 757 by 23. The $Q$-matrix on the other hand specifies how the 23 items are related to 13 binary mathematical skill attributes, as summarized in Table 2.

Note that the provided $Q$-matrix may not fully represent the ground truth because the construction of the $Q$-matrix by experts is almost always subjective. In this case, the provided $Q$-matrix was constructed from the consensus of two experts. When they are not able to reach an agreement for any item through discussion, a third expert would step in to resolve the conflict. The percentage of two experts’ overall agreement for the constructed $Q$-matrix is only 88.89%, according to Su et al. (2013). We denote this expert constructed $Q$-matrix as $Q^0$ and its $(i, j)$th entry as $q^0_{ij}$. A heat plot of $Q^0$ is summarized in Fig. 11. To demonstrate the practical implications of our proposed method, we start with this expert constructed $Q^0$ and explore further whether our proposed method can improve on the quality of the $Q$-matrix to better represent the ground truth.

We initialize the weight matrix with $Q^0$ in our proposed method. The estimated $Q$-matrix is denoted as $\hat{Q}$ and its $(i, j)$th entry as $\hat{q}_{ij}$. If we treat the expert constructed $Q^0$ as the truth for evaluation purpose, then the entry-wise proportional “error” rate, the out of true positives “error” rate and out of true negatives “error” rate of $\hat{Q}$ are 0.126, 0.053 and 0.139, respectively. The low
Heat plot of the expert constructed $Q^0$. The white/black blocks correspond to $q^0_{ij} = 0/1$, respectively.

Heat plot to compare between the estimated $\hat{Q}$ and the expert constructed $Q^0$. The white blocks represent entries $(i, j)$ when both $\hat{q}_{ij} = q^0_{ij} = 0$. The black blocks represent entries $(i, j)$ when both $\hat{q}_{ij} = q^0_{ij} = 1$. The red blocks represent entries $(i, j)$ when $\hat{q}_{ij} = 0$ and $q^0_{ij} = 1$. The blue blocks represent entries $(i, j)$ when $\hat{q}_{ij} = 1$ and $q^0_{ij} = 0$.

"error" rates suggest that $Q^0$ and $\hat{Q}$ are similar and our proposed method can indeed recover the main latent structure, especially the positive entries, in the expert constructed $Q^0$.

Figure 12 presents the heat plot of the comparison between the estimated $\hat{Q}$ and the expert constructed $Q^0$. In particular, white and black entries represent the cases when $\hat{q}_{ij} = q^0_{ij} = 0$ and when $\hat{q}_{ij} = q^0_{ij} = 1$, respectively. While blue and red entries represent the cases when $\hat{q}_{ij} = 1, q^0_{ij} = 0$ and when $\hat{q}_{ij} = 0, q^0_{ij} = 1$, respectively. We see that the majority of the positive entries in $Q^0$ are picked up by $\hat{Q}$, and only 4 of them are predicted to be 0 in $\hat{Q}$, as represented by the red blocks in Fig. 12. This suggests that the proposed method can estimate the $Q$-matrix with high sensitivity. Some of these false negatives do make sense. For example, item 5 describes three figures arranged in matchsticks with some patterns and asks for the total number of matchsticks that would be used to construct figure 10 if the pattern continues. It is a pattern recognition problem and does not seem to be closely related to attribute 6, “reason about and solve one-variable equations and inequalities.” However, we acknowledge that this data-driven approach can sometimes make mistakes. For example, the other three false negatives predicted may not make much sense. Take item 10 for example, which reads “inequality equivalent to
It clearly requires the knowledge of attribute 6, which is not successfully identified by the proposed method. On the other hand, the white regions, representing the agreed entry 0’s, occupy the majority of the plot. This suggests that the specificity is controlled. Moreover, we see some blue blocks scattering in Fig. 12, which represent the entries that are 0 in $Q^0$ but are predicted to be 1 in $\hat{Q}$. Some of these blocks capture information that is neglected by the expert when constructing the $Q$-matrix. Take item 22 for example, whose description is “At a play, $\frac{3}{25}$ of the people in the audience were children. What percent of audience is this?” In the expert constructed $Q$-matrix, this item only requires mastering attribute 2. However, in our estimated $\hat{Q}$, this item is further related to attribute 4, “understanding of rational numbers,” 7, “recognizing proportional relationships” and 9, “applying operations with fractions.” Nevertheless, we also want to point out that the proposed method may over-select, resulting in redundant attributes being selected. For example, item 8 reads “If $x = 3$, what is the value of $-3x$.” The proposed method predicts that it is related to attribute 2, “Use ratio and rate reasoning to solve real-world and mathematical problems,” but in fact item 8 does not seem to be related to attribute 2. Therefore, careful examination of the predicted entries is still needed, but it can potentially help to improve the quality of the $Q$-matrix.

We further compare the goodness of fit of $Q^0$ and $\hat{Q}$ across different CDMs, including the DINA, ACDM and GDINA models using both AIC and BIC as criteria. We note that out of the three models tested, the ACDM gives the smallest values of both AIC and BIC. Moreover, using $\hat{Q}$ gives much smaller AIC (19348.71) than using $Q^0$ (19568.17) under the ACDM, which suggests the estimated $\hat{Q}$ fits better under the ACDM than the expert constructed $Q^0$ in terms of AIC. On the other hand, using $\hat{Q}$ achieves a BIC value of 20313.98, slightly worse than a BIC value of 20286.44 obtained by using $Q^0$. However, the two values are comparable in size and the improvement is not significant. Nonetheless, since we do not know what the true underlying model and the $Q$-matrix are, consultation to the domain experts is still needed to make assertive conclusions about which of $\hat{Q}$ and $Q^0$ is better.

5. Discussions

In conclusion, our proposed method using RBMs with $L_1$ penalty can achieve both fast and accurate learning of the large $Q$-matrices in different types of CDMs. This is shown by both the theoretical proofs developed in Sect. 1.3 and the simulation studies carried out in Sect. 3. The real data analysis on TIMSS data set further suggests that our method can also work well in real-world scenarios, and thus it would provide a powerful tool in large-scale exploratory cognitive diagnosis assessments.

We discuss some potential use cases of our proposed method. One potential use case is to provide a reasonably accurate $Q$-matrix for cognitive diagnoses such as latent attribute classifications, when no $Q$-matrix or only an inaccurately specified $Q$-matrix is available. Depending on the accuracy requirements, the estimated $Q$-matrix can either be used directly in CDMs to perform latent attribute classifications or can serve as a starting point for domain experts for further refinement before use. Another potential use case is to provide a $Q$-matrix estimate for test item categorizations and enabling efficient design for future assessments. Similarly, whether the estimated $Q$-matrix can be used directly depends on the accuracy requirements in different real settings. To add reliability and confidence for direct usage, goodness-of-fit measures such as AIC or BIC can always be evaluated and compared between the estimated $Q$-matrix and the potentially inaccurate specified $Q$-matrix if it is available, as a first step. If the goodness of fit of the estimated $Q$-matrix is bad, then either the model used is not appropriate or the estimated $Q$-matrix is inaccurate. In these cases, consultation to domain experts is still necessary. Nevertheless, our proposed method may help reduce the burden placed on the experts. Based on the estimated
\(Q\)-matrix, if one finds out that additional items with specific \(q\)-vectors need to be included in the test, then it is likely such an item is indeed missing from the original test design. In this scenario, we recommend to include the additional item into the test design to keep safe. Furthermore, in the case when the accuracy requirement is exceptionally high, our proposed method can still help. In this scenario, we recommend to set the penalty term to be 0 and apply CD Algorithm 1 to train the original RBM on the whole data set to obtain \(\hat{W}\). Then for each item \(j\), experts can rank \(\{|\hat{w}_{jk}|: k = 1, \ldots, K\}\) in a descending order first and pay more attention to those \(\hat{w}_{jk}\) with large absolute values as those correspond to the \(q_{jk}\) that are most likely to be 1’s.

Note that by initializing the RBM parameters \(W\), \(b\) and \(c\) randomly, the proposed estimation method assumes no prior knowledge of the \(Q\)-matrix. In practice, we may have partial knowledge of the \(Q\)-matrix, using which we could potentially obtain a better initialization of the parameters. For example, we may have a pre-specified \(Q\)-matrix design with possible mis-specifications in some entries; in such cases, we can initialize the weight matrix \(W\) and the visible bias vector \(b\) based on our prior knowledge of the \(Q\)-matrix. Note that \(w_{j,k}\) in \(W\) correspond to \(\delta_{j,k} q_{j,k}\) in the ACDM. From the perspective of initialization, we find what affects the learning accuracy most significantly are the signs of the initial values. So, to keep things simple, we can initialize \(W\) with the partially available \(Q\)-matrix directly. For the visible biases, if the underlying model is believed to be the DINA model, by considering \(\alpha = 0\), we can derive \(b_j = \log(g_j/(1 - g_j))\). Under the ACDM or the GDINA model, we can obtain \(b_j = \log(\delta_{j,0}/(1 - \delta_{j,0}))\) using a similar argument. Though we do not know \(g_j\) or \(\delta_{j,0}\) in reality, very likely these values are between 0 and 0.5, in which case \(b_j < 0\). It is therefore reasonable to initialize each \(b_j\) from a Uniform\((-5, 0)\) distribution. This would help improve the estimation accuracy.

Some limitations of our method include it does not take into account the interactions between the latent attributes due to the assumptions imposed on RBMs. In many real-world scenarios, it is not uncommon that the latent attributes interact with one another and have joint effects on the distribution of the observed responses. One potential way to solve this problem is to apply deep Boltzmann machines (DBMs) to model the distribution of the responses. Since DBMs allow interactions between the latent attributes, it will capture the interactions between the latent attributes and take that into account. Moreover, this paper focuses more on the estimation part while inference on the estimated \(Q\)-matrix is not discussed. It would be interesting to pin down the asymptotic distributional form of this \(Q\)-matrix estimator to facilitate inferences such as hypothesis testing and constructing confidence intervals.

Acknowledgments

The authors are grateful to the Editor-in-Chief Professor Matthias von Davier, an Associate Editor, and three referees for their valuable comments and suggestions. This research is partially supported by NSF CAREER SES-1846747, DMS-1712717 and SES-1659328.

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

References

Bengio, Y., & Delalleau, O. (2009). Justifying and generalizing contrastive divergence. Neural Computation, 21(6), 1601–1621.

Carreira-Perpinan, M. A., & Hinton, G. E. (2005). On contrastive divergence learning. In: Aistats, vol. 10, pp. 33–40. Citeseer.

Chen, Y., Culpepper, S. A., Chen, Y., & Douglas, J. (2018). Bayesian estimation of the DINA Q matrix. Psychometrika, 83(1), 89–108.

Chen, Y., Liu, J., Xu, G., & Ying, Z. (2015). Statistical analysis of Q-matrix based diagnostic classification models. Journal of the American Statistical Association, 110(510), 850–866.
Chiu, C. Y. (2013). Statistical refinement of the Q-matrix in cognitive diagnosis. *Applied Psychological Measurement*, 37(8), 598–618.

Choi, K., Lee, Y. S., & Park, Y. S. (2015). What CDM can tell about what students have learned: An analysis of TIMSS eighth grade mathematics. *Eurasia Journal of Mathematics, Science and Technology Education*, 11, 1563–1577.

Chung, M., & Johnson, M. S. (2018). An MCMC algorithm for estimating the Q-matrix in a Bayesian framework. arXiv preprint arXiv:1802.02286.

Collins, M., Globerson, A., Koo, T. K., Carreras, X., & Bartlett, P. L. (2008). Exponentiated gradient algorithms for conditional random fields and max-margin markov networks. *Journal of Machine Learning Research*, 9, 1775–1822.

Culpepper, S. (2019). Estimating the cognitive diagnosis Q matrix with expert knowledge: Application to the fraction subtraction dataset. *Psychometrika*, 84(2), 333–357.

de la Torre, J. (2011). The generalized DINA model framework. *Psychometrika*, 76(2), 179–199.

de la Torre, J., & Chiu, C. Y. (2016). A general method of empirical Q-matrix validation. *Psychometrika*, 81(2), 253–73.

de la Torre, J. (2008). An empirically based method of Q-matrix validation for the DINA model: Development and applications. *Journal of Educational Measurement*, 45(4), 343–362.

Hartz, S. (2002). A Bayesian framework for the unified model for assessing cognitive abilities: Blending theory with practicality. *Unpublished doctoral dissertation*.

Henson, R. A., Templin, J. L., & Willse, J. T. (2008). Defining a family of cognitive diagnosis models using log-linear models with latent variables. *Psychometrika*, 74(2), 191.

Hinton, G. E. (2002). Training products of experts by minimizing contrastive divergence. *Neural Computation*, 14(8), 1771–1800.

Hinton, G. E., & Salakhutdinov, R. R. (2006). Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 504–507.

Jiang, B., Wu, T.-Y., Jin, Y., Wong, W.-H., et al. (2018). Convergence of contrastive divergence algorithm in exponential family. *The Annals of Statistics*, 46(6A), 3067–3098.

Kuhn, H. W. (1955). The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 21(1–2), 83–97.

Larochelle, H., & Bengio, Y. (2008). Classification using discriminative restricted Boltzmann machines. In: Proceedings of the 25th international conference on machine learning. ICML ’08, pages 536–543, New York, NY, USA. ACM.

Lee, Y. S., Park, Y. S., & Taylan, D. (2011). A cognitive diagnostic modeling of attribute mastery in Massachusetts, Minnesota, and the U.S. national sample using the TIMSS 2007. *International Journal of Testing*, 11, 144–177.

Liu, J., Xu, G., & Ying, Z. (2012). Data-driven learning of Q-matrix. *Applied Psychological Measurement*, 36(7), 548–564.

Long, P. M., & Servedio, R. A. (2010). Restricted Boltzmann machines are hard to approximately evaluate or simulate. In: Proceedings of the 27th international conference on machine learning. ICML ’10, page 703–710, Madison, WI, USA. Omnipress.

MacKay, D. (2001). Failures of the one-step learning algorithm. In *Available electronicaly at* http://www.inference.phy.cam.ac.uk/mackay/abstracts/gbm.html.

Robitzsch, A., Kiefer, T., George, A. C., Umlue, A., & Robitzsch, M. A. (2020). *Handbook of diagnostic classification models*. New York: Springer.

Rosasco, L. (2009). Sparsity based regularization. *MIT class notes*.

Salakhutdinov, R., Mnih, A., & Hinton, G. (2007). Restricted Boltzmann machines for collaborative filtering. In: Proceedings of the 24th international conference on machine learning. ICML ’07, pages 791–798, New York, NY, USA. ACM.

Schluter, J. (2014). Restricted Boltzmann machine derivations. *Notes*.

Smolensky, P. (1986). *Information processing in dynamical systems: Foundations of harmony theory*. Colorado: Colorado University at Boulder Department of Computer Science.

Su, Y.-L., Choi, K., Lee, W., Choi, T., & McAninch, M. (2013). Hierarchical cognitive diagnostic analysis for timss 2003 mathematics. *Centre for Advanced Studies in Measurement and Assessment*, 35, 1–71.

Sutskever, I., & Tieleman, T. (2010). On the convergence properties of contrastive divergence. In: Proceedings of the thirteenth international conference on artificial intelligence and statistics, pages 789–795.

Templin, J., & Henson, R. (2006). Measurement of psychological disorders using cognitive diagnosis models. *Psychological Methods*, 11, 287–305.
Tsuruoka, Y., Tsujii, J., & Ananiadou, S. (2009). Stochastic gradient descent training for L1-regularized log-linear models with cumulative penalty. In: Proceedings of the joint conference of the 47th annual meeting of the ACL and the 4th international joint conference on natural language processing of the AFNLP: Vol. 1, pp. 477–485. Association for Computational Linguistics.

von Davier, M. (2005). A general diagnostic model applied to language testing data (ETS research report RR-05-16). Princeton: Educational Testing Service.

von Davier, M. (2008). A general diagnostic model applied to language testing data. *British Journal of Mathematical and Statistical Psychology, 61*(2), 287–307.

Wu, Z., Deloria-Knoll, M., & Zeger, S. L. (2016). Nested partially latent class models for dependent binary data; Estimating disease etiology. *Biostatistics, 18*(2), 200–213.

Xu, G. (2017). Identifiability of restricted latent class models with binary responses. *Annals of Statistics, 45*(2), 675–707.

Xu, G., & Shang, Z. (2018). Identifying latent structures in restricted latent class models. *Journal of the American Statistical Association, 113*(523), 1284–1295.

Yuille, A. L. (2004). The convergence of contrastive divergences. *Advances in Neural Information Processing Systems, 17*, 1593–1600.

*Manuscript Received: 28 MAY 2020*

*Final Version Received: 5 OCT 2021*

*Published Online Date: 28 JAN 2022*