ANALYSIS OF COMPUTATIONAL APPROACHES TO COGNITIVE DIAGNOSIS

A Thesis submitted in partial fulfillment of the requirements for the degree
Master of Science

Department of Computer Science
Western Kentucky University
Bowling Green, Kentucky

By
Andrew J. Toussaint

May 2025

ANALYSIS OF COMPUTATIONAL APPROACHES TO COGNITIVE DIAGNOSIS

Date Recommended 2/11/2025

Huanjing Wang Digitally signed by Huanjing Wang Date: 2025.02.11 14:29:20 -06'00'

Chair

Zhonghang Xia Digitally signed by Zhonghang Xia Date: 2025.02.11 14:55:38 -06'00'

Committee Member

Guangming Xing Digitally signed by Guangming Xing Date: 2025.02.11 15:55:24 -06'00'

Committee Member

Committee Member

Executive Director for Graduate Studies

ABSTRACT

Access to good education is crucial to the well-being of individuals as well as communities. Recent technological advancements in the field of computer science show promise of generating precise descriptions of student cognitive states regarding specified knowledge concepts through a process called cognitive diagnosis. This can facilitate the creation of more targeted lesson plans and more personalized educational software. Experiments were conducted to evaluate the performance of four computerized cognitive diagnosis models. The models include three existing models: Item Response Theory, Neural Cognitive Diagnosis , Knowledge Association Neural Cognitive Diagnosis, and a proposed model, Concept Agnostic Knowledge Evaluation, which was used to quantify the value of labeling exercises with knowledge concepts. These models were trained and evaluated using four separate data sets: the 2009 ASSISTments data set, the NeurIPS 2020 dataset, the Junyi 2015 data set, and the ASSISTments 2012 dataset. Each model-data set combination was evaluated under five distinct conditions: "basic" and "sampled" to determine the effect of data sampling, as well as "undersampleEven", "undersampleCorrect", and "undersampleIncorrect" to determine the effect of manipulating the class ratios. The Accuracy, Area under the ROC curve, mean average error, and root mean squared error of each experiment was recorded. Results reaffirmed the superiority of neural network models over psychometric models. Concept labeling was found to greatly improve interpretability with comparable performance results. Data sampling was shown to be an effective means of improving neural network training time. It was shown that model performance may degrade as user knowledge improves. Future research may investigate ways to adapt to changing user knowledge states, the development of automatic question labeling, or question generation from labels to further facilitate the collection of data on the cognitive diagnosis task.

DEDICATION

Dedicated to my late grandmother Rosemary Taylor

ACKNOWLEDGEMENTS

I would like to thank Dr. Huanjing Wang for her guidance in completing this thesis as well as committee members Dr. Guangming Xing and Dr. Zhonghang Xia for their expert input and academic wisdom throughout the process of completing my degree. I could not have reached this level of understanding without them or all of my previous teachers and mentors.

I am grateful to my parents who have continued to provide support as they always have. I will never be able to repay the gifts that they have given me so I strive to pay them forward instead. I would also like to thank all my peers, friends, and family who have offered their help and encouragement along the way. Especially my fiancée Cynthia who has been a source of endless support and graciously tolerated the division of my attention for my studies.

Contents

List of Tables

List of Figures

1 Introduction

Education is essential to individual and societal well-being. The clear importance of education is reflected in the magnitude of resources that have been applied to providing educational access to everyone. Following the institution of the No Child Left Behind Act in 2001, federal spending on education in the United States increased by 64% [4]. Commercial software for education has also increased in popularity. In 2017 the market for PreK-12 learning software was projected to exceed $8 billion, and the global edTech market to exceed $252 billion by 2020 [10]. This trend was only reinforced by the COVID-19 pandemic and the mass implementation of various distance learning methods. These investments, intended to increase the quality of time spent on education, have not translated into direct results. There has been no significant increase in test scores in the United States [4], and as the economy demands a greater degree of specialization, the amount of required education before beginning a long-term career has increased. This is reflected in an increase in the amount of time young people devote to education in their daily life [14] as well as an increased number of people pursuing advanced degrees despite rapidly increasing costs [22]. However, the nature of education has not responded to these shifts. While software solutions are implemented in the classroom, they often simply recreate the existing paradigm of a lecture or a textbook in a digital format. This is contrary to evidence that a single linear approach to lessons is ineffective at teaching students of diverse backgrounds and skill levels. For students who do not have the time or support at home to continue studying this creates a widening disadvantage [10].

While teachers will always be an invaluable resource, they will also always be outnumbered by students. Therefore, it is necessary to equip them with the tools to cater to student's individual education needs and to empower students to be able to make effective use of the personal time that they choose to dedicate towards learning. One dimension of these tools should be the ability to provide personalized experiences to students. To this end, this study investigates computational models for cognitive diagnosis. Cognitive diagnosis is the task of assessing student proficiency on various knowledge concepts. The concepts may be as fine grained or broad as the administrator desires. This is generally achieved by providing students with an assessment and analyzing their responses to various questions to determine their proficiency in certain areas. Recent developments in machine learning technology have led to efficient methods for computers to complete this task. The goal of this study is to conduct a thorough evaluation of these models to understand their performance under various conditions and the quality of data and training time that would be necessary to implement them in practice. This is an important step toward novel educational software applications. The analysis provided by a computerized cognitive diagnosis model can provide teachers with an instant, in-depth profile of each student's capabilities and provide students with personalized exercises that will help them improve in the areas they need the most assistance in.

Four specific cognitive diagnosis models were analyzed: Item Response Theory (IRT), Neural Cognitive Diagnosis (NCDM), Knowledge Association-Neural Cognitive Diagnosis (KaNCD), and Concept Agnostic Knowledge Evaluation (CAKE) which is a proposed reduction of NCDM to investigate the value of knowledge concept labeling. Section 2 will present the formal definition of cognitive diagnosis, review the machine learning and neural network technology that makes these models possible, explain the four models used in detail. Section 3 will explain what experiments were conducted to evaluate the models and Section 4 will present the results of these experiments. Section 5 discusses how to interpret these results and Section 6 discusses what conclusions can be drawn.

2 Existing Research

2.1 Cognitive Diagnosis

The cognitive diagnosis problem has many different forms. In this case, the following definition is used [24]: Suppose there are a number of users $\{U_1, U_2, ..., U_I\}$ which are sometimes called students or examinees, and a number of test items $\{Q_1, Q_2, ..., Q_j\}$ which are sometimes called questions or tasks. An "item" consists of stimulus and a prescriptive form of responding to the stimulus intended to yield a response from the user from which some psychological construct such as proficiency may be inferred [20]. Each item is related to some combination of a set of "knowledge concepts" $\{K_1, K_2, ..., K_n\}$ where the inclusion of a knowledge concept K_n in the set related to test item Q_j signifies that for a user to correctly respond to Q_j they must possess sufficient mastery over the concept K_n. The relationship between Q and K is typically captured in what is referred to as a "Q-matrix" which contains one row for each item in Q and one column for each concept in K. A value of "1" in cell n, j indicates that K_n is a concept relevant to Q_j while a value of "0" indicates the absence of a relationship. Additionally, a set of responses is collected for each user R = $\{R_i, i = 1, ..., I\}$ where R_i is the vector of responses generated by user U_i and r_{ij} is the value of the response of user U_i to item Q_j. In some versions of the problem each response may contain a unique value or one of several different classes of response values [27]. In this case, the content of a response is processed and classified as either correct or incorrect in the context of the corresponding item. A response value of "1" indicates a correct response

and a value of "0" indicates an incorrect response.

Given the described scenario, the cognitive diagnosis task consists of generating the proficiency of each user on the specified knowledge concepts K given the list of users, U, items Q, responses R, and the Q-matrix. A user's proficiency is typically expressed as θ which may take the form of a scalar or multidimensional value depending on the model used.

Traditionally, the following assumptions are also held to be true regarding the cognitive diagnosis problem:

1. Monotonicity: The probability of a correct response to any item is monotonically increasing at any dimension of the student's knowledge proficiency [25]

2. Consistent User State: The state of a user's proficiency as measured by cognitive diagnosis does not change between item responses [21]

3. Independent Item Responses: A user's ability to respond to one item is not impacted by their interaction with any other item[21]

These assumptions allow the problem of cognitive diagnosis to be addressed on a formal mathematical basis. These assumptions are not strictly true for all circumstances, however many of the models discussed in Section 2.2 and Section 2.3.2 rely on these assumptions and achieve positive results on the cognitive diagnosis task.

Since there is no way to establish a ground truth regarding a human user's proficiency directly. The historic method of evaluating the performance of a model on the cognitive diagnosis problem is to apply the results of the model to the secondary task of predicting user performance on a new item by reversing the process: Given the existing user proficiency, the model is made to determine the probability of a correct response to an item including specified knowledge concepts. If the model is successful in this regard, then it is assumed that the model has correctly mapped the relation between knowledge concept proficiency and item responses.

The most straightforward model for judging a user's proficiency would be to attribute the highest proficiency to the user with the greatest number of correct responses. This approach, termed classical test theory [21], is ubiquitous in modern education. A more complex model may consider the different knowledge concepts associated with each item to provide a proficiency value that accounts for breadth of proficiency as well as depth or even to provide a detailed assessment of proficiency on individual concepts. It should be noted that this form of cognitive diagnosis is inappropriate for formal high-stakes assessments such as entrance exams or standardized testing. Any deviation from a system that strictly rewards the magnitude of correct responses can lead to overly complex test taking strategies where a user may find it in their best interest to intentionally respond incorrectly to an item to manipulate their score.

2.2 Psychometric Modeling

Before the widespread use of machine computation in scientific research, psychometric modeling was the basis of performing the cognitive diagnosis task. Psychometric models use statistical formulations to produce expressions that describe the relationship between proficiency and item response performance. The first psychometric models were popularized by Fredrick Lord and Georg Rasch in the 1950s and 1960s [21]. The original item response theory (IRT) and multidimensional item response theory (MIRT) models proposed by Lord and Rasch are discussed in Sections 2.2.1 and 2.2.2 respectively. Over time, new mathematical techniques were incorporated to create more complex models with different formulations such as Deterministic input noisy "and" Gate (Section 2.2.3) and the higher order version (Section 2.2.4). This selection of models is by no means exhaustive, but serves as a representative group of models from which a general idea of psychometric modeling can be understood.

2.2.1 Item Response Theory

Item response theory (IRT), originally developed by Frederic Lord [15], is perhaps the simplest model that can be used to complete the cognitive diagnosis task. Several proposed versions of IRT exist which all rely on the same foundational principles as presented in Reckase (2006) [21]. It is assumed that the relation between locations in multidimensional cognitive space and the probabilities of a correct item response can be modeled by a continuous function. Each user is represented by a single scalar proficiency value and, to make this value interpretable, the items included in an assessment are grouped around some common theme or subject. Thus, the proficiency value can be said to describe a user's proficiency on the general subject of the items. The original formulation of IRT is shown in Equation 2.1.

$$P(u_{ij} = 1|A_j, B_i) = \frac{A_j B_i}{1 + A_j B_i} \tag{2.1}$$

u_{ij} represents the possible score of the j^{th} user who possesses proficiency A_j on the i^{th} item of difficulty B_i. From the provided formulation, maximum likelihood estimation is applied to determine the values of A for each user. This model is relatively simple and ensures the monotonicity assumption, however, it suffers from the fact that for a given B value, the lower half of the probability distribution for a correct response is mapped to A values from (0,B) while the upper half is mapped from (B, ∞). The original model was improved upon to generate the Rasch model described in Equation 2.2. Ψ is the cumulative logistic density function, A is replaced by θ as the measure of user proficiency and problem difficulty is described by b rather than B.

$$P * u_{ij} = 1|\theta_j, b_i) = \frac{e^{\theta_j - b_i}}{1 + e^{\theta_j - b_i}} = \Psi(\theta_j - b_i) \tag{2.2}$$

This formulation of IRT includes several advantages over the original model. θ and b scale from $(-\infty, \infty)$ with higher values of θ representing proficient users and higher values of b representing more difficult questions. The slope of the response curve increases as θ is

closer to b which makes a question of b_i difficulty ideal for differentiating students who have values of θ close to b_i. All users with the same number of correct responses will have the same maximum likelihood estimation of θ.

Subsequent models include the two-parameter logistic model shown in Equation 2.3 and the three-parameter logistic model shown in Equation 2.4.

$$P(u_{ij} = 1|\theta_j, b_i) = \frac{e^{a_i(\theta_j - b_i)}}{1 + e^{a_i(\theta_j - b_i)}} \tag{2.3}$$

The two-parameter logistic model adds to the Rasch model by introducing a new variable a to act as a "weight" on each item to reflect that some items may be more indicative of proficiency irrespective of their difficulty.

$$P(u_{ij} = 1|\theta_j, b_i) = c_i + (1 - c_i) * \frac{e^{a_i(\theta_j - b_i)}}{1 + e^{a_i(\theta_j - b_i)}} \tag{2.4}$$

The three-parameter logistic model includes all terms in the two parameter model and the additional term c which represents the user's probability of correctly guessing the answer to a question without truly being proficient in the requisite knowledge. Empirical estimates of the guess parameter often show it as less than the probability of a truly random guess suggesting that those who selected a wrong answer were not randomly guessing. The three-parameter logistic model is more accurate as guessing is a known phenomenon, however the inclusion of this term makes each individual item less discriminating between users.

The National Center for Education Statistics (NCES) presents a slightly different formulation of the three-parameter logistic model [19] (Equation 2.5). The constant of ± 1.7 is applied so that the rate of change of item discrimination is essentially the same as that produced by the more complex normal ogive model shown in Equation 2.6. All general references to IRT outside of this section are specifically referring to the three-parameter logistic model as defined by NCES.

$$P(x_j = 1|\theta_k, a_j, b, j, c_j) = c_j + \frac{(1 - c_j)}{1 + exp[-1.7a_j(\theta_k - b_j)]} = P_j(\theta_k) \tag{2.5}$$

$$P(U_{ij} = 1|\theta_j, a_i, b_i, c_i) = c_i + (1 - c_i) \int_{-\infty}^{z} \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} dz, z = a_i * (\theta_j - b_i) \tag{2.6}$$

2.2.2 Multidimensional Item Response Theory

The multidimensional item response theory (MIRT) model was created as an iteration on the foundations of IRT. Reckase provides a detailed description [21]: The MIRT model addressed the shortcomings of IRT in being limited to a single scalar value to represent the user's proficiency, and therefore being limited to evaluation in a single subject that the set of items must be formed around. MIRT overcomes this limitation by describing the user's proficiency (θ) as a vector of m different qualities. These qualities can be thought of as the "knowledge concepts" described in the general definition of the cognitive diagnosis problem. The individual values within a user's proficiency vector relate to their proficiency on a given concept. Likewise, the description of an item's difficulty must be extended to a vector of length m to represent the difficulty of the question with respect to each knowledge concept, essentially the level of proficiency required in each area to generate a correct response. This requirement generates an additional task of determining the value of this difficulty vector for each item.

The evaluation process of MIRT is similar to IRT. A formula is defined to describe how user proficiency and item difficulty relate to response correctness and the expectation maximization algorithm is applied to optimize the proficiency values. The ways in which the various knowledge concepts combine to create a response can be separated into two different categories: compensatory, and partially compensatory. The compensatory model is shown in Equation 2.7. The formulation for the compensatory model is very similar to

the three-parameter logistic form of IRT, but with the use of vectors in place of some scalar values.

$$P(U_{ij} = 1|\theta_j, a_i, c_i, d_i) = c_i + (1 - c_i)\frac{e^{a_i\theta_j^{'}+d_i}}{1 + e^{a_i\theta_j^{'}+d_i}} \tag{2.7}$$

In a compensatory model, a high proficiency in one of the required knowledge concepts can almost completely make up for a low proficiency in another required knowledge concept when responding to an item. Alternatively, the partially compensatory model does not allow for high proficiency in one knowledge concept to significantly make up for low proficiency in another. Under the partially compensatory model, a user's proficiency in each knowledge concept must be comparable to the question's difficulty level in that concept for the user to have a favorable chance of responding correctly. The formulation for the partially compensatory model is given in Equation 2.8.

$$P(U_{ij} = 1|\theta_j, a_i, b_i, c_i) = c_i + (1 - c_i)(\Pi_{l=1}^{m}\frac{e^{1.7a_{il}(\theta_{jl}-b_{il})}}{1 + e^{1.7a_{il}(\theta_{jl}-b_{il})}}) \tag{2.8}$$

Whether a compensatory or partially compensatory characterization will result in a more effective model is dependent on the nature of the item set and knowledge concept set. All general references to MIRT outside of this section refer specifically to the partially compensatory model.

2.2.3 Deterministic Input Noisy "and" Gate

The Deterministic Input Noisy "and" Gate (DINA) model differs from IRT and MIRT in formulation while drawing inspiration from these older models. De La Torre provides a description of the model [8]: The DINA model assumes that each latent knowledge concept is completely independent of the other existing knowledge concepts. The way that knowledge concepts are combined is different from any form of MIRT. Each concept is assumed to be a binary value. Each user either possess the necessary knowledge concept

or does not posses it. The binary skill vector for a given user is determined based on the input data through the expectation maximization algorithm. A formulation determining an individual user's response is given in Equation 2.9.

$$\nu_{ij} = \Pi_{k=1}^{K} \alpha_{ik}^{q_{ik}} \tag{2.9}$$

$\nu_i j$ represents the i^{th} user's response to the j^{th} item, "1" for a correct response and "0" for an incorrect response. $\alpha_i k$ is the i^{th} user's proficiency on the k^{th} knowledge concept, "1" for being proficient and "0" for not having proficiency, and $q_j k$ is the relevance of the k^{th} knowledge concept to the j^{th} item, "1" for relevant and "0" for irrelevant.

2.2.4 Higher Order DINA

The higher-order DINA model, HO-DINA, is an extension of the DINA model that was designed for situations that include a large number of knowledge concepts [9]. The HO-DINA model includes a higher order "aptitude" parameter associated with each user in addition to the knowledge concept parameters introduced in the DINA model. The success of a user responding to an item remains independent among knowledge concept proficiencies when considering a single latent aptitude. In contrast to DINA, HO-DINA uses the Monte Carlo Markov Chain approach to generate the probabilities of each knowledge concept being possessed by a user. The probabilities are coerced into a binary decision with a cut-off value of 0.5.

2.3 Neural Network Modeling

The recent rise of machine learning models has led to many advances in the fields of data science and statistical analysis. The cognitive diagnosis problem is no exception with a variety of neural network based models being applied to the problem. In Section 2.3.1

a brief overview is given of the concepts of deep learning and neural networks. Then, Section 2.3.2 describes how this technology is specifically applied to the problem of cognitive diagnosis and provides implementation details for a few specific models.

2.3.1 Deep Learning and Neural Networks

The concept of machines that perform the intellectual work apparently exclusive to humans can be seen as early as the Greek myth of the automaton [18]. This fantasy remained in the human consciousness until it inspired Alan Turing to theorize a practical implementation of such a machine, laying the groundwork for the field of computer science [12]. Developments in mathematics, computation, and other technologies continued to evolve the field to the current "deep learning revolution" including the creation of popular large language models such as ChatGPT and the integration of artificial intelligence into smartphones. Initially, investigations into deep learning models found them to be unreliable and computationally expensive. However, in 2012 a deep neural network was used to significantly advance the capability of computers to perform image recognition[17]. Based on this foundation, research into neural network models has exploded into a wide range of applications. The study of deep learning models is a rich sub-field of computer science which would require an entire textbook to give a thorough overview. The modeling techniques discussed here have been expanded and modified in many ways beyond just the problem of cognitive diagnosis. The goal of this section is to familiarize the reader with the most foundational concepts of the field so that their specific application to the cognitive diagnosis problem can be better understood.

Storing information in a computer naturally lends itself to a hierarchical approach. Simple concepts are stored and combined to form more complex concepts. The application of a hierarchical approach to storing abstract concepts was originally discarded because it was too taxing to provide a description of an abstract concept that was sufficiently complete for a computer to be able to use it. "Deep learning" refers to the use of models that address

this problem by allowing a computer to build its own representation [2]. Using the example of image recognition, rather than providing a complete description of what arrangements of pixels do and do not constitute an image of a dog, a model can be provided with many examples of images that are appropriately labeled and construct for itself a means of differentiating those that are images of dogs from those that aren't. The process of building this representation is facilitated by a neural network.

This is an intriguing idea but how is it practically implemented? The abstract representation can exist within data as a series of rigidly defined mathematical functions with changing parameters referred to as a neural network. In general, a neural network consists of an input layer, several hidden layers, and an output layer. The input layer is a function designed to receive as an input a numerical representation of the input of the problem. Inputs are typically coerced into a set of vectors containing numerical values for computational efficiency. Continuing to use the example of image recognition, the input layer could be arranged to receive a set of vectors describing the color values at each point of the image. The input layer will have a node corresponding to each input value. The output layer is designed to provide the desired output. If the task of the network is to classify images into the categories of "dog" or "not a dog" the output layer may include a single node which outputs a value from 0-1 to represent the model's confidence that the image is a dog. The model must also include an arbitrary number of hidden layers which serve as the connection between the input and output layer. Each hidden layer contains a number of nodes which each receive the output of the previous layer as their input and pass their output to the next layer. Each node includes a mathematical function which calculates the output from the input using weight, W, and bias, b, parameters. The weight and bias parameters are procedurally adjusted as the model encounters more examples to form the model's abstract understanding of the task it is designed for. Typically, an activation function is also included after the weight and bias are applied to restrict the possible output values. An example of the calculation performed by a single node is shown in Equation 2.10 with

the input vector represented by x and the output vector represented by y. ϕ represents the application of the sigmoid function, a common activation function that "squeezes" the output between the values of 0 and 1.

$$y = \phi(W \times x + b) \tag{2.10}$$

The model can learn a representation of the information being passed to it by incrementally adjusting the values of the weight and bias variables at each node based on the feedback it receives from each example it is trained on. This incremental change is done according to the gradient descent method to move the weight and bias at each node toward their ideal values for completing the task at hand. The number of layers, nodes per layer, and connections between layers can all be freely adjusted to optimize performance. In theory, this process can be used to accurately approximate any function and therefore any single logical task[2].

In practice, there are several pitfalls that prevent neural networks from being able to achieve perfect performance. Over-fitting is a problem where the model becomes to specialized in the data set that it is trained on, and becomes unable to effectively perform its task for data outside of the training set. If a complete set of all relevant examples with labels already exists a neural network is pointless because producing the desired output can be achieved by simply searching within this set of examples. Conversely, a neural network cannot create a function without having a set of training data to work with. An effective model is able to generalize the features present in the training set to make accurate predictions on external data. To achieve this, it is necessary to have a set of training data that is representative of the set of all examples that the model could be applied to. Additionally, the number of nodes and layers included in the model must be sufficient to generalize the features present in the data set, without extrapolating nonexistent patterns. This leads to the "No free lunch theorem" which states that no single model will provide optimal performance on all problems [29]. Instead, a model must be custom built to the

specified problem and trained on an appropriate data set.

2.3.2 Neural Networks in Cognitive Diagnosis

The purpose of a neural network model is to be able to determine a complex function that maps some input to a desired output. This can be directly applied to the cognitive diagnosis problem. The inputs are the responses provided by various users to items and the knowledge concepts associated with each item. The output is the value describing whether the response is correct or incorrect. By training a purposefully built neural network on these response logs, the network can develop an internal representation of the user's cognitive state which can be extracted to understand the user's proficiency values.

Many different models have been proposed for this purpose. The following sections provide brief discussion of NCDM and KaNCD, the models used in this investigation, as well as other notable models.

NeuralCDM

The Neural Cognitive Diagnosis Model (NCDM) was originally proposed by Wang et al. [25]. This model was among the first attempts to apply deep learning and neural network technology to the problem of academic cognitive diagnosis [24]. The model was intended as a general model that could be refined with extensions and modifications. NCDM can generalize both the MIRT and IRT functions as well as a broader range of functions outside of what those approaches would be able to cover. The success of the model supports the claim that neural networks are appropriate for modeling the complex non-linear process of students interacting with exercises. The NCDM model aims to produce an interpretable diagnostic report of user proficiency on specified knowledge concepts based on a set of student responses to items and a Q-matrix describing an associated between exercises and knowledge concepts. The required inputs for the model are a response log for users including user IDs, item IDs, and correct/incorrect response values as well as a Q-matrix

mapping the item IDs to associated knowledge concepts. The model works as follows: User proficiency (h^s) is determined by multiplying a user one-hot vector (x^s) with a trainable matrix A and applying the sigmoid function.

$$h^s = \phi(x^s \times A) \tag{2.11}$$

Knowledge relevancy (Q_e) is determined by multiplying an item one-hot vector (x^e) with the provided Q-matrix (Q).

$$Q_e = x^e \times Q \tag{2.12}$$

Knowledge difficulty (h^{diff}) is determined by multiplying the exercise one-hot vector with a trainable matrix B. The values within the matrix B are restricted to be positive.

$$h^{diff} = \phi(x^e \times B), B \in \mathbb{R}^{M \times K} \tag{2.13}$$

Knowledge discrimination (h^{disc}) is determined by multiplying the exercise one-hot vector with a trainable matrix D. The values within the matrix D are restricted to be positive.

$$h^{disc} = \phi(x^e \times D), D \in \mathbb{R}^{M \times 1} \tag{2.14}$$

Once the above values are determined, their interaction is based on the principles of MIRT defined in Section 2.2.2. The difficulty is subtracted from the user proficiency to determine if the user's abilities meet or exceed the difficulty of the item. This is then multiplied element-wise by the knowledge relevancy to only consider the user's abilities in relevant concepts. Finally, this result is multiplied by the discrimination vector to augment the values associated with each knowledge concept proportional to their relevancy to the specific item being answered. The result of this is the input vector to the neural network portion of the model, x.

$$x = Q_e \circ (h^s - h^{diff}) \times h^{disc} \tag{2.15}$$

The resulting vector x is then passed through two fully connected layers of a neural network and a final output layer. The layers contain 512, 256, and 1 nodes respectively. Each node includes trainable weight and bias values and uses the sigmoid function as an activation function. The monotonicity assumption is enforced by restricting the weight values of each layer ($W1, W2, W3$) to non-negative values.

$$\begin{aligned} f_1 &= \phi(W_1 \times x^T + b_1) \\ f_2 &= \phi(W_2 \times f_1 + b_2) \\ y &= \phi(W_3 \times f_2 + b_3) \end{aligned} \tag{2.16}$$

The loss function used to train NCDM is cross entropy loss between the output y and the true label of a response record.

To use the model, as with all neural networks, a set of response records must be used to train the weights, biases, and embedding matrices of the neural network. After the training is complete, the model can be used to predict the performance of users from the training set on any items defined in relation to the knowledge concepts used in the training set. The goal of cognitive diagnosis is to produce an evaluation of the user's proficiency on each concept. This can be achieved through this model by extracting the student proficiency vector which is the trained embedding used to relate a user to each defined knowledge concept. In practice, the model is implemented in Python using the PyTorch library [7]. Implementation details can be found in the EduCDM GitHub repository [3].

The original presentation of NCDM also includes three extensions: Knowledge Extraction from text content (CNCD-Q), Factor extraction from text content (CNCD-F), and knowledge association (KaNCD). The CNCD-Q and CNCD-F models were not included as part of this study due to the inconsistency in text content availability in the available data sets. The KaNCD extension is discussed next.

Knowledge Association NeuralCDM

Knowledge Association NeuralCDM (KaNCD) is an extension of NCDM as previously described [25]. The KaNCD model is designed to address the poorer performance of NCDM in the case that response logs covering every knowledge concept are not available for each student. The general approach of the KaNCD model is to assign latent knowledge traits to each explicitly defined knowledge concept and use a user's proficiency in these latent traits to predict their performance on untested concepts. The adjusted theory is implemented as follows: Rather than learning the matrix A directly as described in Equation 2.11, each value in A is calculated by the interaction of a latent vector that represents a user l_i^s and a latent vector representing a knowledge concept l_j^k. The latent vector values are learned through training. The number of dimensions chosen for the latent vectors is the number of presumed latent concepts that are foundational to each of the explicitly defined knowledge concepts. In practice, this number is an arbitrary hyper-parameter that may be adjusted as needed. The proficiency of a given user on a given knowledge concept is calculated as a weighted sum of the filtered latent traits (a_1).

$$a_1 = l_i^s \circ l_j^k \tag{2.17}$$

This proficiency is used to calculate an individual element in the A matrix using trainable parameters W_{a2} and b_{a2}.

$$A_{i,j} = W_{a2} \times a_1 + b_{a2} \tag{2.18}$$

In a similar way, the same process is applied to the difficulty matrix B using another latent vector describing an item (l_i^e), and the same latent vector describing a knowledge concept (l_j^k) to calculate a single element of B.

$$b_1 = l_i^e \circ l_j^k \tag{2.19}$$

$$B_{i,j} = W_{b2} \times b_1 + b_{b2} \tag{2.20}$$

Once these latent vectors have been used to calculate A and B the resulting matrices are applied to Equations 2.11 and 2.13 respectively. From that point forward, the process proceeds the same as NCDM. Although the mathematical representations of these two models are identical up to a point, the implementation provided is different [3]. These differences may have subtle impacts on model performance when testing the models which could not be understood by analyzing the described formulas alone.

Other Models

It is impractical to give a complete account of all neural network based models for cognitive diagnosis. A few more notable models that include elements unique from NCDM and KaNCD are described:

- *DCD*: Disentanglement based Cognitive Diagnosis (DCD) seeks to address the challenges of limited exercise labels. The model is designed to perform in situations where the Q-matrix is sparsely labeled, and some items may not include any knowledge concept labels. The model relies on the concepts of group-based disentanglement and limited-labeled alignment to complete the cognitive diagnosis task [6].

- *LDM-ID and LDM-HMI*: The general form of the learning diagnosis model (LDM) is to use an ensemble of psychometric models to produce several initial learning diagnoses, then pass this result through a network that models the representations of users and their relationship to items and knowledge concepts. From this, deep and shallow learning features are extracted and then pooled to predict performance. Two implementations are presented: LDM-ID which incorporates IRT and DINA, and LDM-HMI which incorporates Ho-DINA, MIRT, and IRT [26].

- *DKT-STDRL*: Deep Knowledge Tracing Based on Spatial and Temporal Deep Representation Learning for Learning Performance Prediction (DKT-STDRL) is a pro-

posed model that relies on understanding the sequence in which users respond to items as well as their response values to create a model of learning for performance prediction. The proposed model uses a convolutional neural network to extract the spatial feature information of users' exercise sequences. Then, the spatial features relate to the original users' exercise features. A BiLSTM is used to extract temporal features and generate prediction information [16].

3 Experiments

Experiments were conducted to quantify model speed and performance of cognitive diagnosis models under different scenarios. Section 3.1 gives an overview of the existing data sets that were used to train the models. The evaluated models include IRT, NCDM, KaNCD, and a modified version of NCDM , CAKE, discussed in Section 3.2. Section 3.3 explains what experiments were performed on models and Section 3.4 gives a summary of the technology used and implementation details.

3.1 Existing Data Sets

Four existing data sets were used in training the cognitive diagnosis models: The 2009 ASSISTments data set (ASSIST09) [1], the 2012 ASSISTments data set (ASSIST12) [11], the data set provided as part of the NeurIPS 2020 Educational Challenge (NEUR20) [28], and a data set from the online tutoring platform Junyi (JUNYI15) [5]. For conciseness and clarity the data sets will be referred to by the accompanying abbreviations. Each data set included, at a minimum, the information necessary to construct a response log and Q-matrix as required by the cognitive diagnosis task. Table 3.1 shows the number of users, items, responses, and knowledge concepts included in each data set [24]. The following sections provide a brief description of the circumstances under which each data set was collected.

Data set	Users	Items	Responses	Knowledge Concepts
ASSIST09	4,163	171,751	346,860	146
NEUR20	118,971	27,613	15,867,850	388
JUNYI15	247,606	722	25,925,922	41
ASSIST12	46,674	179,999	6,123,270	265

Table 3.1: Data Set statistics

3.1.1 ASSIST09

The ASSIST09 data set was collected by the ASSISTments system [11]. ASSISTments is a system designed to integrate assistance and assessment and provide additional data to teachers regarding student performance. Data was collected from two middle schools in Massachusetts from 2004-2006. Knowledge concepts are identified and related to each item in a list so that each item may relate to multiple knowledge concepts [1]. The relevant columns are "user_id", "problem_id", and "correct" which identify the unique user, item, and response in each log entry as well as the "list_skill_ids" column which includes the relationship between the identified knowledge concepts and the problem.

3.1.2 NEUR20

The NeurIPS 2020 Education Challenge featured the task of predicting student performance given a set of response logs from the U.S. based E-learning platform "Eedi" [28]. The data set was released for research use after the competition. Each question could be answered by selecting one of four multiple choice answers. Students could encounter the same problem multiple times and therefore generate multiple response logs in which case the log of their most recent response was included in the data set. Each item could be related to multiple knowledge concepts. The relevant columns are "UserId", "QuestionId", and "IsCorrect" which identify the unique users, items, and their responses respectively. Answer metadata was provided as a Q-matrix in a separate file.

3.1.3 JUNYI15

The JUNYI15 data set was collected from the Chinese online learning platform Junyi Academy [5]. The data set includes response logs from October 2012 through January 2015. The complete data set was made available for use through the educational data hosting platform DataShop [13]. Though the data set does not explicitly identify knowledge concepts, each item is given one of 41 different "topic" values which is similar to a knowledge concept and can be used to construct a Q-matrix [24]. The limitation of this approach is that each item has exactly one knowledge concept value which provides a less detailed description of the item compared to other data sets using multiple values. The relevant columns are "user_id", "exercise", and "correct" which identify the unique users, items, and their responses respectively.

3.1.4 ASSIST12

The ASSIST12 data set was another set of data collected from the ASSISTments system [11] like that discussed in Section 3.1.1. The data was collected during the 2012 school year with a focus on predicting student affect. Notably, each item in the data set is only associated with a single "skill" rather than multiple skills as was the case for the 2009 data set. As a result, each item in the data set is limited to a single knowledge concept. The relevant columns are "user_id", "problem_id", and "correct" which identify the unique user, item, and response in each log entry as well as the "skill_id" column which includes the identified knowledge concept for each item.

3.2 Novel Model: Concept Agnostic Knowledge Evaluation

To better understand the effectiveness of labeled knowledge concepts in cognitive diagnosis models a proposed neural network model, Concept Agnostic Knowledge Evaluation, (CAKE) was created. The CAKE model follows the same internal structure as NCDM described in Section 2.3.2 with removal of any references to the Q-matrix. Equation 2.15 is changed to not include the Q-matrix as shown in Equation 3.1.

$$x = (h^s - h^{diff}) \times h^{disc} \tag{3.1}$$

Since this change affects the dimensions of the input, x, appropriate dimension changes were made to the trainable elements in each layer to accommodate this core change and allow for proper matrix multiplication. The result is that the proficiency vector associated with each user uses all its values to determine the user's capability of responding to a given item rather than being restricted to the rows specified by the Q-matrix. The CAKE model should be viewed as a reduced form of NCDM. It is similar to a reduced model introduced alongside NCDM, NeuralCDM_Q [25] which estimated knowledge-relevancy vectors during unsupervised training without a Q-matrix. A more thorough description of NeuralCDM_Q or specific implementation details were not provided.

Rather than learning specific knowledge concepts that were expert labeled for each item, the CAKE model constructs its own representation of each item using latent knowledge concepts. This leads to a major loss of interpretability as the latent concepts no longer correspond directly to the proficiency concepts that cognitive diagnosis is designed to measure. The purpose of implementing this model was to determine the effectiveness of using a Q-matrix and whether the explicit labeling for interpretability caused a measurable decrease in performance. Future research may address ways to recover interpretability

from the CAKE model.

3.3 Experiment Design

Experiments were conducted to evaluate the performance of the cognitive diagnosis models. Each model was trained and evaluated separately on each of the four data sets described in Section 3.1. Training was performed using 5-fold cross validation. This process was repeated for each of five conditions to test model performance in different scenarios. The conditions include "basic", "sampled", "undersampleEven", "undersampleCorrect", and "undersampleIncorret". The four models, four data sets, and five conditions result in 80 model performance evaluations. The goal of these experiments was to compare overall performance between the models, characterize the relationships between number of training tuples, model performance, and training time, and to determine the effect of imbalanced response classes between the training an testing set on model performance.

3.3.1 Data Formatting

The data from each data set needs to be processed from its raw form into a uniform format so that all data sets could be applied to all models. This process was broadly the same for all data sets. First, the relevant columns described in Section 3.1 were extracted and formatted into a response log including user ids, item ids, and a value representing whether the response was incorrect. Separately, a Q-matrix for each data set was constructed by creating a matrix of zeros with a row for each item, a column for each knowledge concept, and then filling in a value of 1 for each knowledge concept associated with each item according to the data set.

Once the initial formatting was complete, the data sets were refined to make them usable within the models. Any tuples with essential columns that were left blank or "NA" in the original data set were dropped. Any users that answered less than 15 problems and any

problems that were answered by less than 15 users were dropped. This was done so that under the sampled data condition, every user could have at least one response within each of the 5 folds of the data set tested. To accommodate matrix size limitations within the models themselves, the number of user ids and item ids in each data set were truncated to 5,000 and 10,000 respectively. These thresholds were chosen as they approximately match the size of the smallest data set ASSIST09 which was able to facilitate model training without exceeding the allowable embedding matrix size. To make use of the most amount of data available, users and items that had the most responses were selected for use in the training process. The data set statistics shown in Table 3.1 were collected regarding the raw data sets before the refining process. Table 3.2 shows the same statistics as Table 3.1 after the data sets had been cleaned and truncated.

data set	Users	Items	Responses	Knowledge Concepts
ASSIST09	4,332	6,671	346,276	146
NEUR20	4,940	10,000	2,563,722	388
JUNYI15	4,993	697	577,183	40
ASSIST12	4,958	10,000	1,090,842	179

Table 3.2: Data set statistics after cleaning

3.3.2 Comparing Model Performance

The first goal of experimentation was to compare the performance of the four cognitive diagnosis models being evaluated. To achieve this, each model was trained and then evaluated on each data set using 5-fold cross validation. For a given data set, the data was split into five distinct subsets with users represented equally throughout the five subsets. A single fold consisted of 4 of these subsets being used to train the model and the 5th subset being divided in half to create a testing and a validation set used to determine the model's performance at each epoch and overall respectively. This process was repeated a total of five times, with each data subset being used as the testing/validation subset once. The final performance metric of each model was calculated by averaging the performance on each

of the five folds. To determine a baseline performance for each model the entire refined data set was used. This constitutes the "basic" condition.

Performance was evaluated based on four metrics: accuracy (ACC), area under the ROC curve (AUC), mean absolute error (MAE), and root mean squared error (RMSE). A description of each metric and the purpose of its inclusion is given below:

- *ACC*: Accuracy is a direct measure of the proportion of user responses that the model was able to correctly classify. This metric is the most practical evaluation of how the model would perform if given the task of anticipating user performance. One drawback of this metric is that user performance is binary, but model output is a continuous value between 0 and 1. To translate this into a binary decision an arbitrary cut-off point must be determined. Intuitively, 0.5 was chosen as the cut-off point for calculating this metric but this cutoff could be treated as a hyper-parameter to be adjusted in practice.

- *AUC*: Area under the ROC curve is a more robust metric compared to accuracy. While accuracy is bound to a single cut-off point, AUC considers model performance across all possible cut-off points. For this reason, AUC is considered a more representative evaluation metric of model performance than accuracy.

- *MAE*: The mean absolute error considers how far off a model's prediction is from the correct result of 1 or 0. This is one way to evaluate model stability.

- *RMSE*: Similar to MAE, root mean squared error is another way to evaluate model stability [26]. The difference between the metrics is that RMSE imposes a greater performance penalty for the presence of outlier predictions than MAE. Including both metrics is useful because comparing a models MAE to its RMSE can characterize the presence of outliers within the model's predictions.

Additionally, the training time for each fold was measured and an average train time per fold was computed for each scenario.

3.3.3 Effect of Tuple Count on Training Time

Another goal of experimentation was to determine the effect of reducing the number of training tuples on the performance and training time of models. The basic condition was used as a baseline of model performance. For comparison, another scenario was created by sampling approximately 40% of the tuples from each data set before performing the 5-fold split, training, and evaluating the models. This is referred to as the sampled condition. Tuples were sampled from each user ID to ensure that each user had the same proportion of representation in the sampled set that they had in the full data set.

3.3.4 Performance on Different Response Ratios

Three additional conditions were created to test the model's performance when the ratios of correct and incorrect responses were different between the training and test sets. As a baseline, the "undersampleEven" condition was created by removing any excess responses from each users response category that had the greater number of entries to create smaller data sets with an even proportion of correct and incorrect responses. The "undersampleCorrect" condition was created by undersampling the data to create a training set where the ratio of correct to incorrect tuples was 6:4 and the ratio in the testing/evaluation set was 4:6 to test the scenario that the model was trained on a higher ratio of correct responses than it encountered in practice. Inversely, the "undersampleIncorrect" condition generated a data set with a correct to incorrect ratio of 4:6 in the training data and 6:4 in the testing data to investigate the opposite scenario. To maintain these ratios, a true 5-fold cross validation approach could not be used on any of the undersampled conditions. Instead, 5 random splits that included 80% of the data in the training set, 10% in the test set, and 10% in the evaluation set while maintaining the correct ratios were used as an approximate alternative.

3.4 Experiment Implementation

The code to format the data, split the data into 5-fold sets, and train and test the models was implemented in Python. As much as possible, the original implementations for the models provided by the authors were used [3]. A jupyter notebook was used to run the program to facilitate a flexible environment for exploring the data in different ways. The hardware used for experiments was a 3.60 GHz AMD Ryzen 5 2600X Six-Core Processor. All code is publicly available on my GitHub [23]. Sources for the download of the data sets used are cited in Section 3.1.

The hyper-parameters applied to each model during evaluation were maintained at their original or default values as implemented in the EduCDM repository provided by the original authors [3]. Table 3.3 shows the hyperparameters used with each model.

Model	Learning Rate	Epochs	Latent Dimensions
IRT	0.001	2	N/A
NCDM	0.002	3	N/A
KaNCD	0.002	3	20
CAKE	0.002	3	(matches number of knowledge concepts)

Table 3.3: Hyperparameters used for each model

4 Results

4.1 Model Performance

Table 4.1 shows the performance of each model on the tested data sets as well as the standard deviation of the model's performance in terms of accuracy, area under the ROC curve (AUC), mean absolute error (MAE), and root mean squared error (RMSE) as determined by 5-fold cross validation under the basic condition. For example, the average Accuracy of the IRT model was 0.633 with a standard deviation of 0.003 when trained and evaluated on the ASSIST09 data set.

Since all responses are classified as either correct or incorrect, the task of determining a user's response value is a binary classification problem. For every dataset, more responses were correct than incorrect (see Figure 4.4). For a binary classification problem with unbalanced class ratios, the most appropriate metric for judging performance is AUC [26]. AUC includes information about the models performance across all cutoff points while all other metrics rely on the arbitrary cutoff point of 0.5. The other metrics are included to provide more insights into the specifics of model performance. Comparing RMSE and MAE can determine the frequency of large outliers. Accuracy shows the model's actual performance on the task when using the intuitive cutoff point of 0.5 for a binary classification between 0 and 1.

Neural network models significantly out-perform IRT on all metrics on all data sets. This result was expected based on existing research [25] and supports the claim that a

ASSIST09				
Model	Accuracy	AUC	MAE	RMSE
IRT	0.633±0.003	0.629±0.003	0.410±0.001	0.491±0.001
NCDM	0.721±0.003	0.768±0.002	0.351±0.007	0.434±0.002
KaNCD	0.727±0.002	0.782±0.001	**0.338**±0.005	0.430±0.002
CAKE	**0.731**±0.002	**0.782**±0.001	0.355±0.004	**0.425**±0.001
NEUR20				
Model	Accuracy	AUC	MAE	RMSE
IRT	0.613±0.002	0.596±0.019	0.414±0.002	0.518±0.007
NCDM	0.731±0.002	0.795±0.001	**0.341**±0.004	0.423±0.001
KaNCD	**0.738**±0.002	**0.803**±0.002	0.344±0.013	**0.420**±0.002
CAKE	0.731±0.001	0.793±0.002	0.360±0.006	0.423±0.001
JUNYI15				
Model	Accuracy	AUC	MAE	RMSE
IRT	0.773±0.004	0.683±0.007	0.290±0.002	0.406±0.003
NCDM	0.818±0.004	0.812±0.003	0.235±0.005	0.363±0.003
KaNCD	0.831±0.003	0.830±0.003	**0.235**±0.009	0.351±0.002
CAKE	**0.836**±0.002	**0.838**±0.003	0.236±0.005	**0.345**±0.002
ASSIST12				
Model	Accuracy	AUC	MAE	RMSE
IRT	0.647±0.002	0.557±0.002	0.394±0.001	0.495±0.001
NCDM	0.736±0.004	0.734±0.002	0.347±0.010	0.424±0.002
KaNCD	0.737±0.005	**0.749**±0.002	**0.346**±0.010	0.420±0.001
CAKE	**0.741**±0.001	0.747±0.001	0.355±0.005	**0.419**±0.000

Table 4.1: Performance of models when trained under the basic condition (Average ± Standard Deviation)

neural model is appropriate for the task of student cognitive diagnosis.

On the ASSIST09 data set CAKE performs the best in terms of Accuracy (0.731), AUC (0.782), and RMSE(0.425). KaNCD has the lowest MAE on the ASSIST09 data set at 0.338 which is followed by NCDM at 0.351 and then CAKE at 0.355. The cause of KaNCD having a better MAE score than CAKE while having a worse RMSE score could be that KaNCD produces relatively larger outliers than CAKE, but few enough outliers overall to still perform well. The size of the outliers will greatly impact the RMSE while MAE will be increased less.

On the NEUR20 data set KaNCD performs the best in terms of Accuracy (0.738), AUC (0.803), and RMSE (0.420). In terms of MAE, NCDM performed best (0.341) followed by KaNCD (0.344) and CAKE (0.360). Notably, this is the only metric for any data set where NCDM was the best performing. Similarly to ASSIST09, the discrepancy between MAE and the other metrics could be explained by the magnitude of outliers generated by NCDM.

On the JUNYI15 data set CAKE performs the best in terms of Accuracy (0.836), AUC (0.838), and RMSE (0.345). KaNCD has the lowest MAE at 0.235 which is followed by NCDM at 0.235 and then CAKE at 0.236. The model rankings on the JUNYI15 dataset are the same as those for ASSIST09.

On the ASSIST12 data set KaNCD performs the best in terms of AUC (0.749) and MAE (0.346) while CAKE performs the best in terms of Accuracy (0.741) and RMSE (0.419). The discrepancy between the best performing MAE and RMSE is likely due to the magnitude of outliers each model generates as discussed when analyzing performance on the ASSIST09 data set. When comparing the accuracy and AUC metrics, the AUC is a stronger indicator of model performance in general as it includes an analysis of the models ability to classify at every possible cutoff point, while accuracy only considers the fixed cutoff point of 0.5. For this reason, it can be said that KaNCD performed better on ASSIST12 overall.

The key difference between the CAKE and KaNCD models is that KaNCD makes use of the expert labeled Q matrix while CAKE does not use the Q matrix. One explanation for the differing performance on each data set could be the quality of labeling available in the Q matrix associated with that data set. The JUNYI15 data set was not constructed with the idea of a Q matrix in mind. Each exercise was labeled with a "topic" which was appropriated as the knowledge concept for the purpose of training the model. Because of this, each item was limited to a single knowledge concept, and this concept may not have been as precise of a representation of the item's content as the intentionally labeled Q-matrices used in the NEUR20 and ASSIST12 data sets. The discrepancy between performance on the ASSIST09 data set was much less significant between KaNCD and CAKE which may indicate that the Q-matrix used with this data set is still relatively informative.

The improvement of KaNCD and CAKE over NCDM in all cases except for the MAE metric when testing the NEUR20 data set supports the theory that knowledge concepts can be described as a combination of more foundational latent knowledge concepts.

4.2 Training Speed

Figure 4.1 compares the time required to train a single "fold" of each model under the basic condition with the time required to train a fold of the model under the sampled condition. While a basic run incorporates all available tuples into the training and evaluation process for the model being trained, a sampled run samples approximately 40% of the available data by sampling 40% of the responses from each user and only uses these tuples when training and evaluating the model. For each data set the actual ratio between the size of the sampled set and the size of the basic set was 0.40 ± 0.001.

The smaller number of tuples does not greatly decrease the time taken to train the IRT model. The largest improvement is observed on the NEUR20 data set where the train

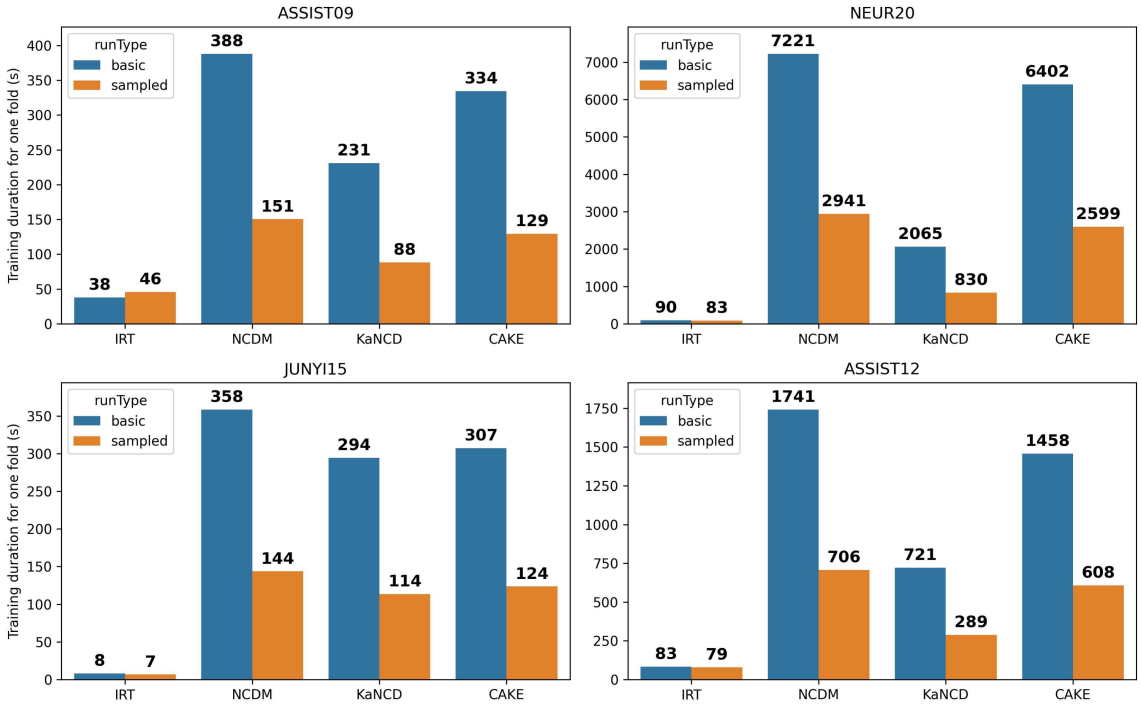

Figure 4.1: Time taken to train a single fold of a model for the basic and sampled data sets

time per fold is reduced from 90 seconds to 83 seconds and, in the case of the ASSIST09 data set, the average train time increases from 38 to 46 seconds. This suggests that the train time of this purely psychometric model is not directly correlated to the number of tuples used to train it. One explanation for the increase in train time is that with fewer examples to train on the model takes a greater number of iterations to reach stable values. The neural models showed more noticeable improvement. On average across all data sets, NCDM trained 40% faster on the sampled run, KaNCD trained 39% faster, and CAKE trained 40% faster. The increase in training speed was consistent across models and data sets with the greatest training time improvement occurring for KaNCD on the ASSIST09 data set (38%) and the least occurring for CAKE on the ASSIST12 data set (42%). On every data set each neural model was able to be trained in about 40% of the time when using 40% of the available tuples. This number correlates to the rate at which tuples were sampled which indicates that, for tuple counts within the range of the test cases, there is a linear relationship between the number of tuples used in training/evaluation and the time

to complete training. Each data set has a different number of tuples, since 5 conditions

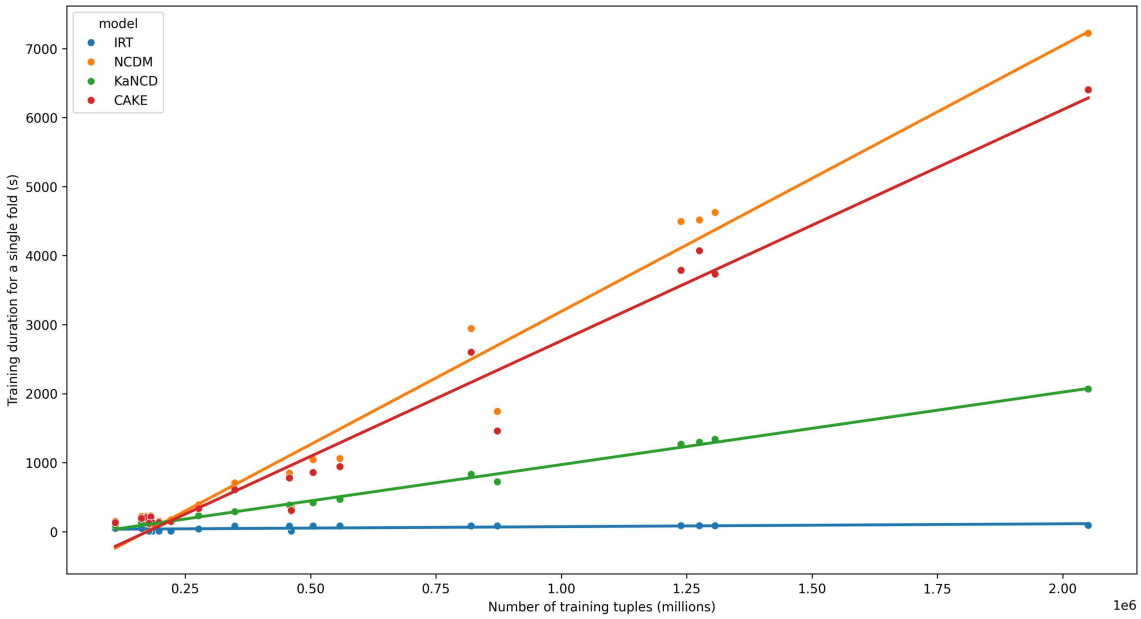

Figure 4.2: Training time vs. number of tuples, basic and sampled runs

were performed on each data set, training times could be observed for each model under 20 different tuple counts. This data was used to develop linear trend lines for each model to demonstrate the relationship between the number of tuples and the training time as shown in Figure 4.2. The trend line for IRT has a slope close to 0 which reflects the small or inconsistent change in training time relative to tuple count. Among the neural network models KaNCD has the smallest slope followed by CAKE and then NCDM which is only slightly steeper than CAKE. CAKE follows the same general structure as NCDM but does not reference a Q-matrix which may explain its slightly improved training time. While still being a neural network model, KaNCD has an implementation that diverges more from the other two neural network models. Based on the data collected, this structure is more efficient at processing training examples for tuple counts within 2 million compared to the other neural network models. It should be noted that the linear trend may not continue for tuple counts beyond the 2 million range observed. Significantly larger tuple counts could result in a higher order growth function.

Assuming that data is not biased, a greater amount of data for models to train on generally results in more performant models. Conversely, reducing the volume of data available to train the models by sampling will reduce the performance [2]. Fig 4.3 shows the change in each performance metric from the basic condition to the sampled condition averaged across all data sets. To achieve visual consistency the y-axis is inverted for the RMSE and MAE metrics. Therefore, a bar extending above the x-axis indicates the sampled run resulted in improved performance over the basic run, while a bar extending below the x-axis indicates the sampled run decreased performance compared to the basic run.

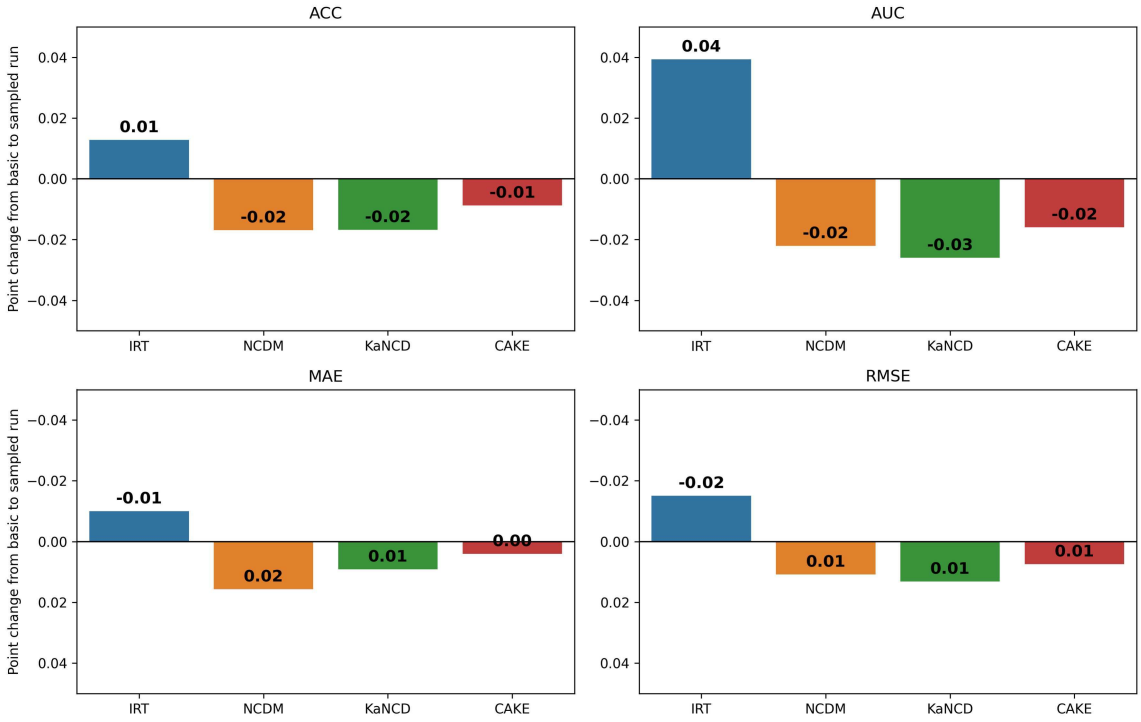

Figure 4.3: Difference in performance metrics between the basic and sampled runs

Notably, the IRT model improves in every performance metric when using a smaller number of samples. The accuracy improves by 1 percentage point, AUC by 4 percentage points, MAE by 1 percentage point, and RMSE by 2 percentage points. This makes the sampled run an optimal way to train IRT in terms of both data collection overhead, and performance. Conversely, the neural network based models consistently perform worse with fewer tuples to train on. NCDM reduces in accuracy, AUC, and MAE by 2 percentage

points and RMSE by 1 percentage point. KaNCD shows similar changes, reducing by 2 percentage points of accuracy, 3 percentage points of AUC, and 1 percentage point of MAE and RMSE respectively. Among the neural network models CAKE responds best to the change, losing only a 1 percentage point in accuracy, 2 percentage points in AUC, less than 1 percentage point in MAE, and 1 percentage point in RMSE. The change in performance among all the neural models is not insignificant but it is not so great as to render the models ineffective. Considering the magnitude of the change in training time, it is certainly possible there are use cases where it would be acceptable to sacrifice few percentage points of performance to dramatically increase training speed. Further research is necessary to precisely define the boundaries of the tradeoff between performance and training tuple count for neural network based models.

4.3 Response Class Ratios

Figure 4.4 shows the portions of recorded responses in each data set that were classified as correct and incorrect. These ratios were preserved in both the training data and the evaluation data for each individual user when training the models under the basic condition. Three other conditions were performed to investigate the effect of undersampling to manipulate the ratio of correct responses in training and evaluation data sets. Each undersampling procedure included less tuples than the entire data set, but more than the sampled condition discussed in Section 4.2.

4.3.1 Equal Ratio

The equal ratio condition undersampled the responses of each user to ensure an even number of correct and incorrect responses for each user. The result was an even number of correct and incorrect responses from each user in the data sets used to train and evaluate the models. The change in performance to each model by equalizing the portions of correct and

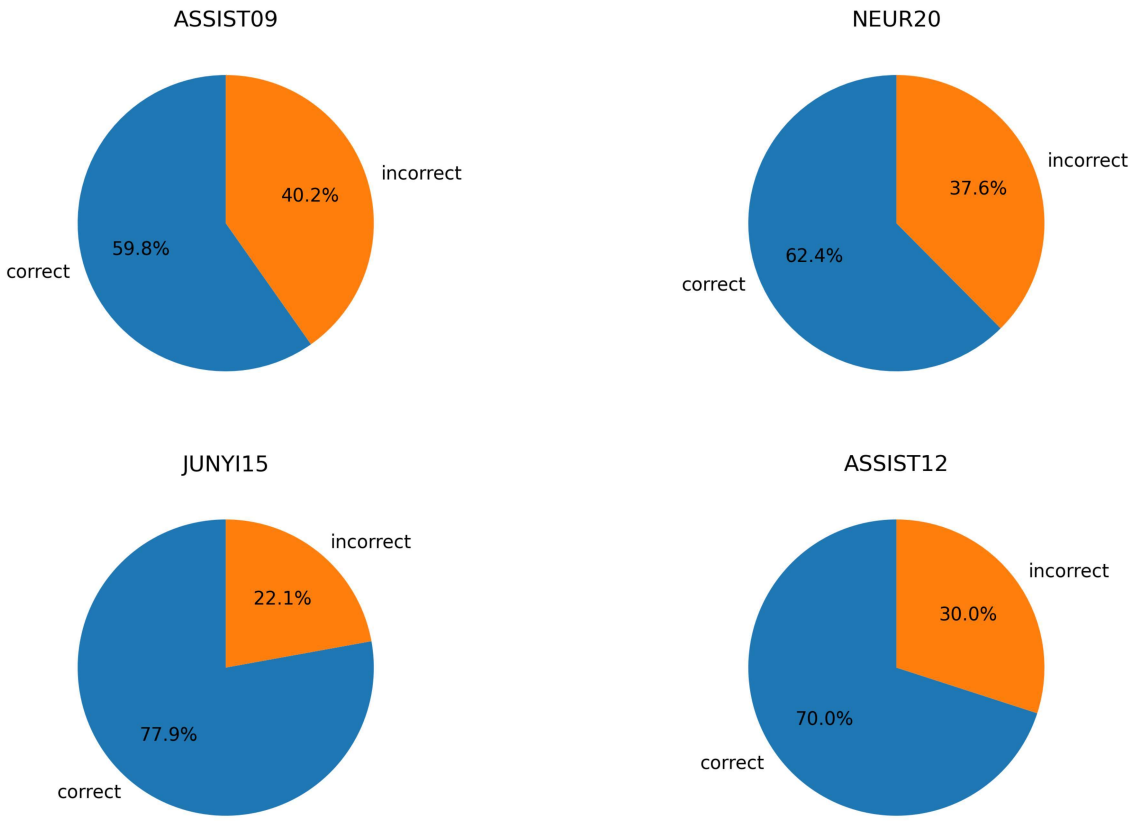

Figure 4.4: Ratio of correct vs. incorrect item responses present in each data set

incorrect responses is shown in Figure 4.5. For conciseness an average across all data sets for each model is shown. Ultimately, this type of sampling resulted in worse performance for every model according to every metric.

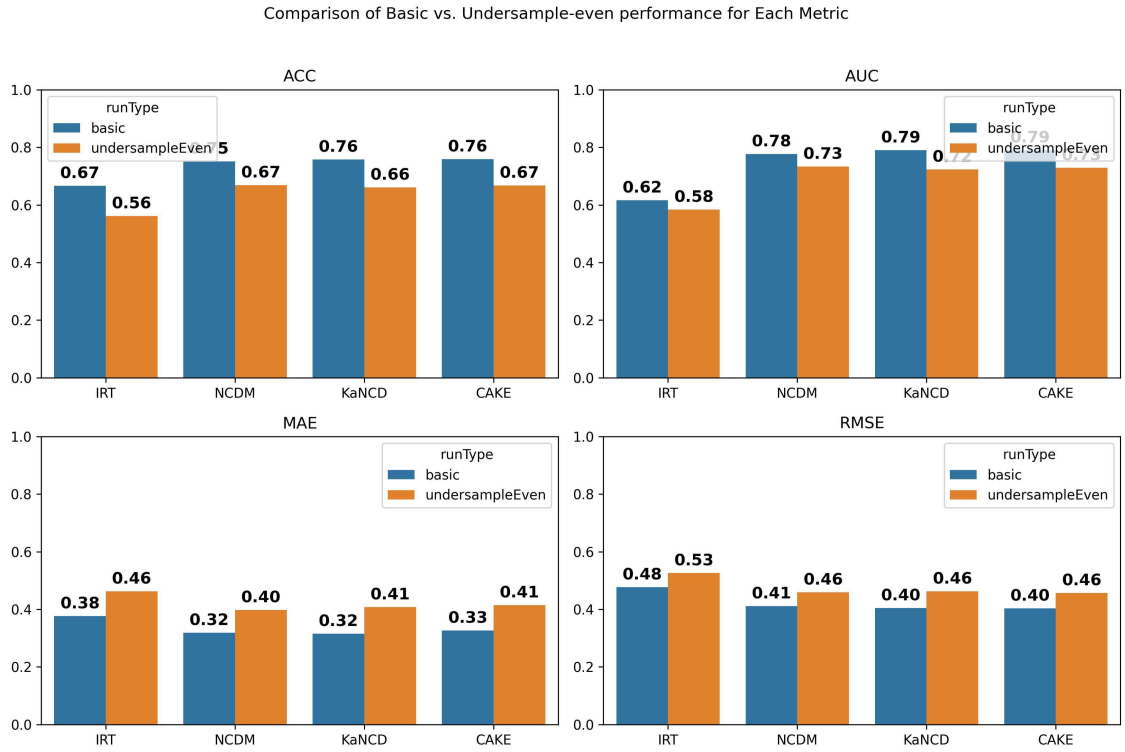

Figure 4.5: Difference in performance metrics between the basic and undersampled-equal runs

4.3.2 Correct Responses Favored in Training

The undersample-correct condition undersampled the responses of each user to attempt to create a training set that includes 60% correct responses and 40% incorrect responses as well as test and validation sets that each have the inverse ratio (40% correct and 60% incorrect). These ratios are applied on a per-user basis so in practice the desired percentage could not always be met exactly. The final ratios produced for each data set are shown in Figure 4.6.

The average change in each model's performance on the modified data set is shown in Figure 4.7. In all cases model performance was significantly worse.

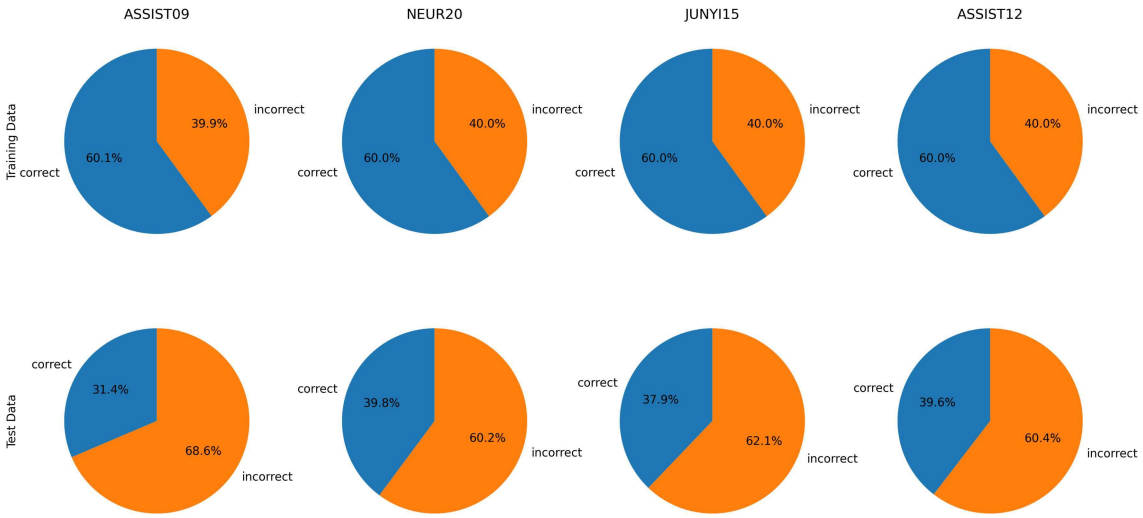

Figure 4.6: Ratio of correct vs. incorrect item responses present in each data set after splitting for the undersample-correct condition

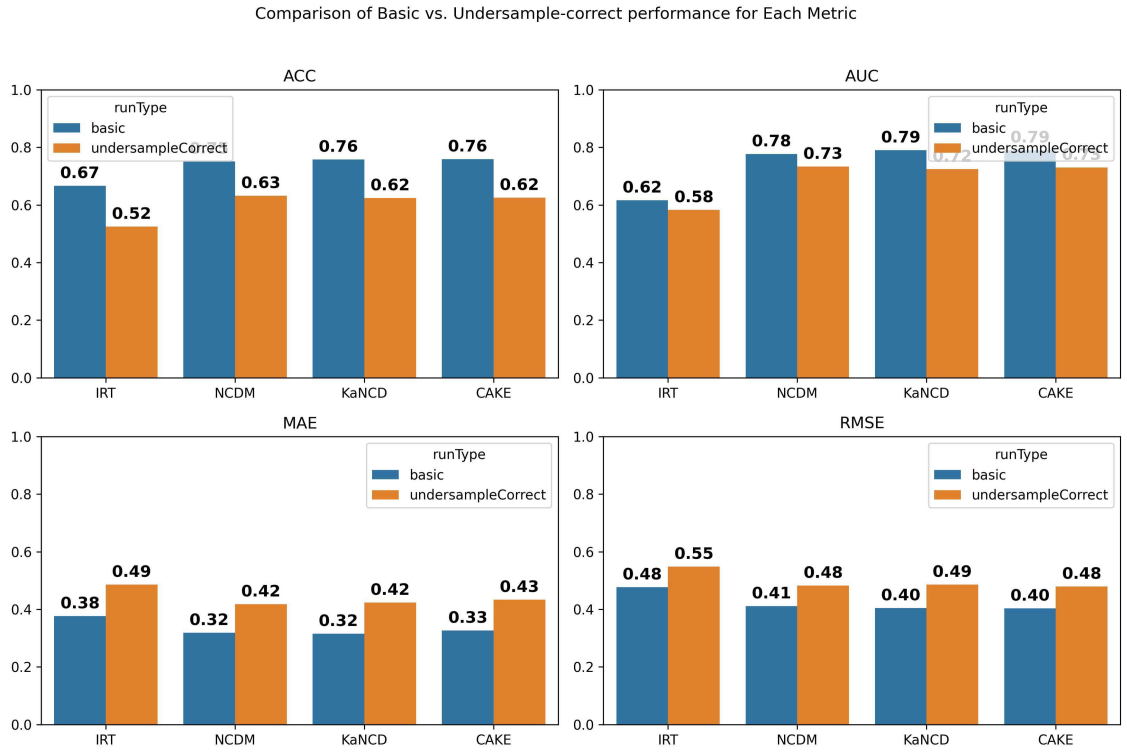

Figure 4.7: Difference in performance metrics between the basic and undersampled-correct runs

4.3.3 Incorrect Responses Favored in Training

The undersample-incorrect condition tested the opposite scenario of the undersampled-correct condition by creating training data that includes 60% incorrect responses and 40% correct responses as well as test and validation sets that each have the inverse ratio. Figure 4.8 shows the final ratios produced for each data set.

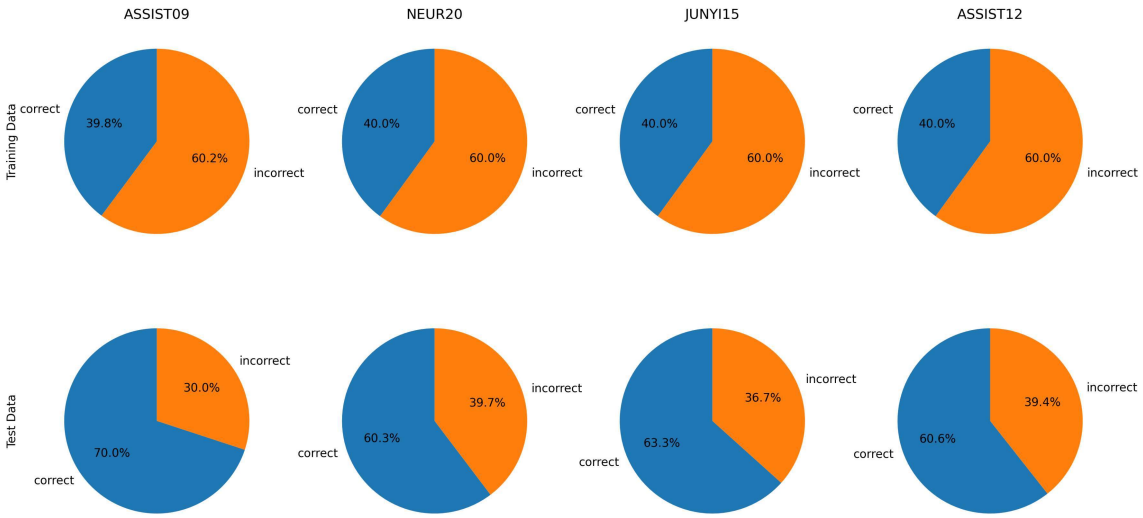

Figure 4.8: Ratio of correct vs. incorrect item responses present in each data set after splitting for the undersample-incorrect condition

The average change in each model's performance on the modified data set is shown in Figure 4.9. In all cases model performance was significantly worse.

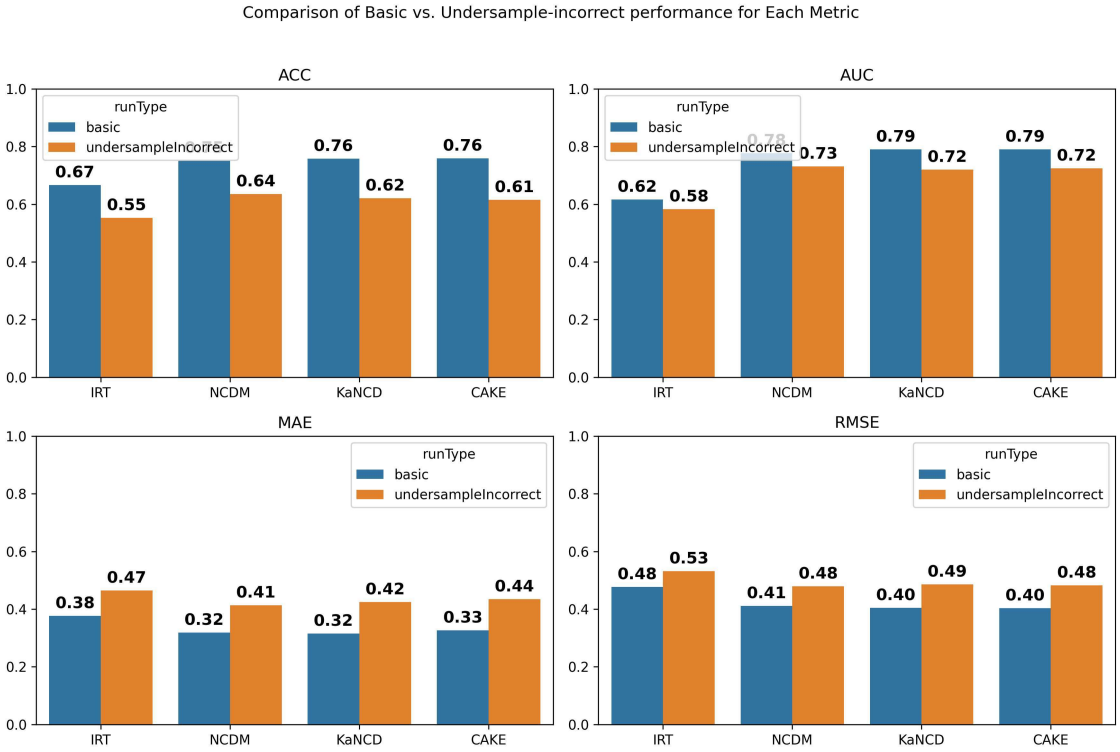

Figure 4.9: Difference in performance metrics between the basic and undersampled-incorrect runs

5 Discussion

The IRT model presents the worst performance across all measured metrics, supporting the idea that neural network models are more appropriate for the cognitive diagnosis task and surpass solely psychometric models in performance. Among the neural network models, KaNCD generally performed the best on the NEUR20 and ASSIST12 data sets having the best accuracy (0.738), AUC (0.803), and RMSE (0.420) on NEUR20 and the best AUC (0.749) and MAE (0.346) on ASSIST12. For the other data sets, CAKE was the best performing model with the best accuracy (0.731), AUC (0.782) and RMSE (0.425) for ASSIST09 and the best accuracy (0.836), AUC (0.838), and RMSE (0.345) for JUNYI15. This suggests that a well labeled Q-matrix is able to add information to a latent-concept based model that improves the model's performance at response classification. A Q-matrix also improves the model's interpretability by providing insights to the specific conceptual strengths and weaknesses of users. For these reasons the Q-matrix is an essential piece of the cognitive diagnosis system.

Despite being a well performing model in other metrics, CAKE produced the second worst MAE for all data sets, performing better than only IRT. CAKE performed more favorably in terms of RMSE, which suggests that CAKE generated outliers of a lower magnitude than the other models. Performance on the RMSE metric is significantly impacted by large outliers, while MAE is less so. If KaNCD occasionally generated a large outlier, but had lower error on the average prediction, it would achieve a better MAE score than CAKE while achieving a worse RMSE. From this we can conclude that, on the data

sets CAKE performs better on, CAKE is less precise in classifying an item response as correct/incorrect but generates few outliers and is better able to classify the item correctly with respect to a cutoff point of 0.5.

On the basic condition the average training time for a single fold rounded to the second was 55 seconds for IRT, 2,427 seconds for NCDM, 828 seconds for KaNCD, and 2,125 seconds for CAKE. While the different conditions altered the run time of the models, the order of performance remained the same for all conditions. The representative psychometric model, IRT, was able to be trained much faster than any of the neural network models. The fastest neural network model to train was KaNCD, followed by CAKE and finally NCDM. Random sampling of data sets was shown to be an effective means of reducing the training time of neural network based models, but not psychometric models. For neural models the relation between the number of tuples in the training set and training time is approximately linear. The observed trend shows that relative order of model training speeds will hold as tuple numbers increase. However, while the neural network based models generally performed slightly worse under the sampled condition, IRT improved in performance. Further research is necessary to determine the optimal tuple count and sampling rate for model performance.

All three undersampling conditions resulted in significantly worse model performance across all models and data sets. There is no apparent benefit to undersampling the data to enforce some specific ratio of correct or incorrect answers. Ideally, the data provided to the model for analysis will have the same ratio of correct/incorrect answers as when the model was trained. This suggests that the model may degrade in performance as users become more proficient. Further research into the use of models that involve more temporal factors and recurrent neural network technology could address this concern [16].

It is important to understand these results in the context of this study. One limitation of these experiments was the use of a single type of specific hardware. GPUs are able to achieve faster performance for training neural networks than CPUs and would likely be the

tool of choice for any large-scale implementation of these models. However, this study only included CPU hardware so the results on a different hardware type may be different. To apply the data sets to the current implementation also required a reduction of the number of unique user and item IDs from the whole dataset. This is another limitation that prevents scalability. This study only includes a few different models for comparison, however there are many other existing models such as those discussed in Section 2.3.2 which are worthy of their own investigation.

6 Conclusions

The performance of the existing IRT, NCDM, and KaNCD Cognitive Diagnosis models was investigated. The novel modification of NCDM, CAKE was also investigated to quantify the impact of labeling knowledge concepts for use in training a neural network based model. Models were separately trained on the 2009 ASSISTments, 2020 NeurIPS, 2015 Junyi, and 2012 ASSISTments data sets to gain a broader understanding of their performance. Each model-data set pair was trained under five different conditions: The "basic" condition which used the entire viable portion of the given data set, the "sampled" condition which utilized approximately 40% of the viable portion of the given data set, the "undersampledEven" condition which selectively sampled tuples to enforce an even number of correct and incorrect samples in the given data set, the "undersampleCorrect" condition which included a greater number of correct responses in the training data compared to the testing data, and the "undersampleIncorrect" condition which included a greater number of incorrect responses in the training data compared to the testing data.

Neural models KaNCD, NCDM, and CAKE were found to outperform the purely psychometric model IRT in all scenarios. However, IRT has a much shorter training time. Among the Neural models KaNCD performed the best when working with data sets that included a well labeled Q-matrix to describe the knowledge concepts associated with each question. KaNCD was also the neural network model that required the least training time per tuple of data. In situations where the Q-matrix was not well described CAKE performed the best. This supports the necessity of a well labeled Q-matrix for the cognitive

diagnosis task, as there is currently no clear method for extracting latent knowledge factors from the CAKE model. In general, NCDM was outclassed by either CAKE or KaNCD in each data set tested.

For neural network models a positive linear trend was discovered between the number tuples in the training data set and the training time. While training time could be reduced by approximately 40% by sampling 40% of the available data, this also resulted in a performance penalty of as much as 4%. Further research may characterize this relationship more precisely and establish limits on the appropriate number of training tuples for these models.

Investigating different undersampling ratios of the data resulted in no improvement in model performance. The scenario that resulted in the best performance was the basic condition where both the train and test data set have the same ratio of correct and incorrect responses as the whole data set. Incidentally, for all data sets this included a greater number of correct responses compared to incorrect responses.

This study is a step toward the integration of cognitive diagnosis models in education software. By evaluating the performance of the models under various scenarios, future developers can be assured that the models are robust and will produce valuable data regarding their students. The importance of creating well labeled exercises was also highlighted. By investigating the training speed of the models and their performance under different correctness ratios, application designers are provided with the necessary data to make informed decisions so that future applications can be responsive and personalized to student needs.

The task of creating a well labeled Q-matrix is laborious and requires manual expert input. Future work could investigate ways to ease this task by automatic question labeling or question generation from labels. Alternatively, a method could be developed to extract question labels from embeddings created by the CAKE model. Another research topic worthy of consideration is developing models that can adapt to student growth in the long

term. All tests were run on data sets collected over relatively short time periods and it was assumed that the latent aptitude of users remained constant. In the long term this should not be true as students are expected to improve in knowledge. Future models may be able to use temporal factors to remain effective when analyzing users as they experience knowledge growth.

7 Works Cited

[1] Assistments. 2009-2010 assistments data, 2009. Accessed: 2024-10-15.

[2] Yoshua Bengio, Ian Goodfellow, and Aaron Courville. *Deep learning*, volume 1. MIT press Cambridge, MA, USA, 2017.

[3] bigdata ustc. Educdm. `https://github.com/bigdata-ustc/EduCDM`, 2021.

[4] David Bolden and Peter Tymms. Standards in education: Reforms, stagnation and the need to rethink. *Oxford Review of Education*, 46(6):717–733, 2020.

[5] Haw-Shiuan Chang, Hwai-Jung Hsu, Kuan-Ta Chen, et al. Modeling exercise relationships in e-learning: A unified approach. In *EDM*, pages 532–535, 2015.

[6] Xiangzhi Chen, Le Wu, Fei Liu, Lei Chen, Kun Zhang, Richang Hong, and Meng Wang. Disentangling cognitive diagnosis with limited exercise labels. *Advances in Neural Information Processing Systems*, 36, 2024.

[7] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS workshop*, 2011.

[8] Jimmy De La Torre. Dina model and parameter estimation: A didactic. *Journal of educational and behavioral statistics*, 34(1):115–130, 2009.

[9] Jimmy De La Torre and Jeffrey A Douglas. Higher-order latent trait models for cognitive diagnosis. *Psychometrika*, 69(3):333–353, 2004.

[10] Maya Escueta, Vincent Quan, Andre Joshua Nickow, and Philip Oreopoulos. Education technology: An evidence-based review. 2017.

[11] Mingyu Feng, Neil Heffernan, and Kenneth Koedinger. Addressing the assessment challenge with an online system that tutors as it assesses. *User modeling and user-adapted interaction*, 19:243–266, 2009.

[12] Andrew Hodges. *Alan Turing: The Enigma*. Burnett Books, London, 1983.

[13] Kenneth R Koedinger, Ryan SJd Baker, Kyle Cunningham, Alida Skogsholm, Brett Leber, and John Stamper. A data repository for the edm community: The pslc datashop. *Handbook of educational data mining*, 43:43–56, 2010.

[14] Gretchen Livingston. The way us teens spend their time is changing, but differences between boys and girls persist. 2019.

[15] Frederic M Lord. A theory of test scores (psychometric monograph no. 7). *Iowa City, IA: Psychometric Society*, 35, 1952.

[16] Liting Lyu, Zhifeng Wang, Haihong Yun, Zexue Yang, and Ya Li. Deep knowledge tracing based on spatial and temporal representation learning for learning performance prediction. *Applied Sciences*, 12(14):7188, 2022.

[17] Kevin P Murphy. *Probabilistic machine learning: an introduction*. MIT press, 2022.

[18] Augustus Taber Murray. *Homer: The Iliad; with an English Translation*. William Heinemann, 1963.

[19] National Center for Education Statistics. Item response theory (irt) models: Three-parameter logistic (3pl). `https://nces.ed.gov/nationsreportcard/tdw/analysis/scaling_models_3pl.aspx`, 2023. Accessed: 2024-10-03.

[20] Osterlind. Toward a uniform definition of a test item. *Educational research quarterly*, 14(4):2–5, 1990.

[21] Mark D Reckase. 18 multidimensional item response theory. *Handbook of statistics*, 26:607–642, 2006.

[22] Camille L Ryan and Kurt Bauman. Educational attainment in the united states: 2015. us census bureau. *Population Characteristics Report (P20-578). Washington, DC*, 2016.

[23] Andrew Toussaint. Masters thesis. `https://github.com/ajtoussaint/Masters_Thesis`, 2024.

[24] Fei Wang, Weibo Gao, Qi Liu, Jiatong Li, Guanhao Zhao, Zheng Zhang, Zhenya Huang, Mengxiao Zhu, Shijin Wang, Wei Tong, et al. A survey of models for cognitive diagnosis: New developments and future directions. *arXiv preprint arXiv:2407.05458*, 2024.

[25] Fei Wang, Qi Liu, Enhong Chen, Zhenya Huang, Yu Yin, Shijin Wang, and Yu Su. Neuralcd: a general framework for cognitive diagnosis. *IEEE Transactions on Knowledge and Data Engineering*, 35(8):8312–8327, 2022.

[26] Zhifeng Wang, Wenxing Yan, Chunyan Zeng, Yuan Tian, and Shi Dong. A unified interpretable intelligent learning diagnosis framework for learning performance prediction in intelligent tutoring systems. *International Journal of Intelligent Systems*, 2023(1):4468025, 2023.

[27] Zichao Wang, Angus Lamb, Evgeny Saveliev, Pashmina Cameron, Jordan Zaykov, Jose Miguel Hernandez-Lobato, Richard E Turner, Richard G Baraniuk, Craig Barton, Simon Peyton Jones, et al. Results and insights from diagnostic questions: The neurips 2020 education challenge. In *NeurIPS 2020 Competition and Demonstration Track*, pages 191–205. PMLR, 2021.

[28] Zichao Wang, Angus Lamb, Evgeny Saveliev, Pashmina Cameron, Yordan Zaykov, José Miguel Hernández-Lobato, Richard E Turner, Richard G Baraniuk, Craig Barton,

Simon Peyton Jones, Simon Woodhead, and Cheng Zhang. Diagnostic questions: The neurips 2020 education challenge. *arXiv preprint arXiv:2007.12061*, 2020.

[29] D.H. Wolpert and W.G. Macready. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, 1997.