Exploring the Applications of Neural Networks in the Adaptive Learning Environment

Baladitya Swaika, Rahul Khatry

Abstract**—**Computer Adaptive Tests (CATs) are one of the most efficient ways for testing the cognitive abilities of students. CATs are based on Item Response Theory (IRT) which is based on item selection and ability estimation using statistical methods of maximum information selection/selection from posterior and maximum-likelihood (ML)/maximum a posteriori (MAP) estimators respectively. This study aims at combining both Classical and Bayesian approaches to IRT to create a dataset which is then fed to a Neural Network which automates the process of ability estimation and then comparing it to traditional CAT models designed using IRT. This study uses python as the base coding language, pymc for statistical modelling of the IRT and scikit-learn for Neural Network implementations. On creation of the model and on comparison, it is found that the Neural Network based model has a 7-10% less accuracy than the IRT model for score estimations. Although performing less accurately compared to the IRT model, the Neural Network model can be beneficially used in back-ends for reducing time complexity as the IRT model would have to re-calculate the ability every-time it gets a request whereas the prediction from a Neural Network could be done in a single step for an existing trained Regressor. This study also proposes a new kind of framework whereby the Neural Network model could be used to incorporate feature sets, other than the normal IRT feature set and use a Neural Network's capacity of learning unknown functions to give rise to better CAT models. Categorical features like test type, etc. could be learnt and incorporated in IRT functions with the help of techniques like logistic regression and can be used to learn functions and expressed as models which may not be trivial to be expressed via equations. This kind of a framework, when implemented would be highly advantageous in psychometrics and cognitive assessments. This study gives a brief overview as to how Neural Networks can be used in adaptive testing, not only by reducing time-complexity but also by being able to incorporate newer and better datasets which would eventually lead to higher quality testing.

Keywords**—** Computer Adaptive Tests, Item Response Theory, Machine Learning, Neural Networks.

I. Introduction

LEARNING is a complex cognitive phenomenon and modelling human learning could be a challenging task. Several techniques have been proposed and used to model human learning and more are being formulated since each has their limitations and none can model the full complexity of an educational environment [1]. Multiple choice items are perhaps the most widely applied tool in testing. This is particularly true in the case of the testing of the cognitive abilities or achievements of a group of students [2].

Dichotomous items are items which can have two possible results, generally, Yes/No or True/False. In this study all working has been done with multiple choice items which can easily be classified as dichotomous items since although having multiple choices for each item, attempting such an item is always going to have an outcome of either Correct/Incorrect. In the current study, the *Rasch Dichotomous Curve* has been used due to this consideration.

"*In psychometrics, item response theory (IRT), also known as latent trait theory, strong true score theory, or modern mental test theory, is a paradigm for the design, analysis, and scoring of tests, questionnaires, and similar instruments measuring abilities, attitudes, or other variables*." IRT differs from traditional testing methods in the sense that it does not assume that "each item is equally difficult". The absence of this assumption has lead statisticians to make models such that tests can be adaptive, i.e., the next item to be administered in the test depends on the response given to the current item being administered. IRT treats the difficulty of each item (the ICCs) as information to be incorporated in scaling items. ICC stands for item characteristic curve.

The general design principles for IRT based CAT are:

1. A calibrated item bank with appropriate parameter values
2. Initial ABILITY estimation
3. Start TEST
4. Select ITEM based on current ABILITY
5. Interim ABILITY estimation based on ITEM responses
6. Stop based on the stopping rule
7. Final ABILITY estimation

These design principles have been dealt with in greater detail by Veldkamp and Matteucci [3].

The IRT Equation:

$$p_i(\theta) = c_i + \frac{(1 - c_i)}{(1 + e^{(-a_i \times (\theta - b_i))})} \qquad (1)$$

For 2 Parameter Logistic models, $c_i = 0$.

For 1 Parameter Logistic models, $a_i = 1$, $c_i = 0$.

Equation (1) describes the Item Response Function of a particular item based on the values of the parameters. Fig. 1 shows sample Item Response Functions of an item in the different models of IRT.

The different parameters of the Item Response Function have been described below:

Ability(*θ*): It represents the estimated score of a subject from the sample.

B. Swaika, is a 3rd Year B.Tech Student in the Department of Computer Science Engineering at Techno India University, Kolkata, WB 700091 IN. (e-mail: bswaika96@gmail.com)

R. Khatry, Data Scientist Automotive Telematics and Autonomy, TRL Ltd. Wokingham, Berkshire, RG403GA, UK (e-mail: rkhatry@trl.co.uk)

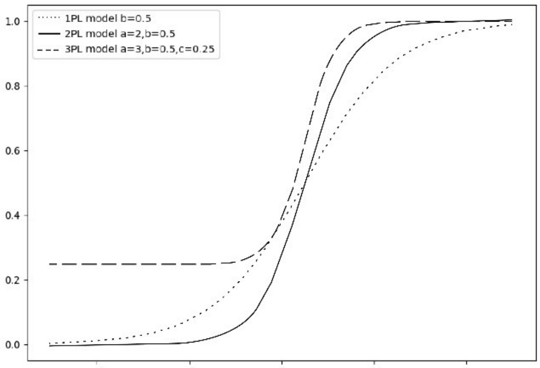


Fig. 1 Item Response Function for a Calibrated Item

Difficulty(*b*): It represents the difficulty of an item and shifts the Item Response Function towards left or right.

Discrimination(*a*): It represents how well an item discriminates between subjects of the sample having extreme score estimates and also determines slope of the Item Response Function.

Guessing Probability(*c*): It represents how guessing affects the score estimate of a sample and shifts the whole Item Response Function up based on that intercept.

IRT has always been implemented in either of the two ways:

1. Classical Methods
2. Bayes' Methods

Classical methods have generally used "maximum-information" selection and "ML" estimators, whereas Bayesian methods have generally used "posterior-based" selection and "MAP" estimators.

In this study, a mixed approach has been used by combining "maximum-information" selection and "MAP" estimator. Since a Rasch model has been used for dichotomous items, this model is equivalent to the 1 Parameter Logistic model, i.e., just the difficulty parameter has been used.

Neural Networks is a machine learning technique loosely inspired by biological neural networks in the brain which helps animal beings to recognize, classify or predict things. These are systems which learn to do tasks by being trained on examples without the task-specific programming. They are highly useful in scenarios where the algorithm for a task is generally not known or is highly complex to program. With the advent of large datasets and compute power, more and more progress has been made in this area and they have been able to solve problems with increasing complexity. Neural Networks are comprised of layers which are constituted of neuron-like nodes. The nodes are interconnected and are activated by an activation function (generally the sigmoid function). The connections are dealt with by the propagation function which is generally a weighted sum of the inputs to a particular layer. The Multi-Layer Perceptron Regressor is a feed-forward neural network where the connections between the units don't form a cycle [4].

This study aims at exploring the applications of a Feedforward Neural Networks in the adaptive learning environment, by carrying out performance and time related comparisons between a Neural Network based testing model and an IRT-based testing model, and looking into how the Neural Network based model can be used to create much complex adaptive tests.

II. Literature Review

Measuring cognitive leaning is a challenging task and various statistics are devised as an attempt to derive understanding and obtain applications in the real world. As all other techniques IRT has its limitations some of the major ones being the time taken because of the computational complexity of calculating latent variables [5] and the challenges of having to model the parameters manually [6].

One of the approaches available for tackling these challenges is the application of machine learning for modelling and making predictions. Machine learning has found application in almost every field today and it continues to grow. There have been many attempts to use machine learning in adaptive learning setting and has shown promising results [6]. Kuo. et. al. [7] used it to obtain the difficulty of questions by including the concept of knowledge maps and neural networks and dividing concepts into concept hierarchy and concept schema. Martinez et. al. [8] had a different approach of trying to find the difficulty of questions by giving the question to different machine learning classifiers and applying the concepts of IRT. Lalor et. al. [9] attempted to produce an evaluation scale using IRT for measuring the ability of machine learning algorithms for different tasks in a GS (Gold Standard) test set for recognizing textual entailment.

Piech et. al. [6] used Recurrent Neural Networks having LSTM units for tracing the knowledge of students and to model student learning. They tried to implement the concept of Bayesian knowledge tracing (BKT) with recurrent neural networks in deep knowledge tracing (DKT) and showed that DKT obtained better accuracy than BKT (best results obtained) and BKT* (best reported in literature). Wilson et. al. [10] demonstrated that IRT based approach worked as accurate or better in terms of predicting responses than the RNN approach provided by Piech et. al. [6] over many different kinds of datasets. Wilson et. al. [11] wrote a detailed comparison of the two approaches mainly focusing on accuracy of the models against several datasets.

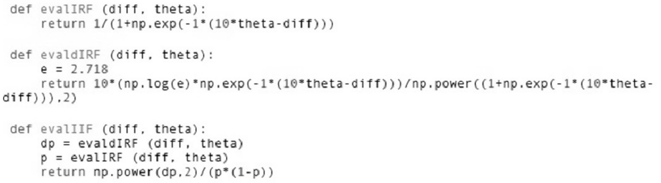
As previously mentioned calculation of latent parameters involved in probabilistic programming also makes the process computationally expensive and unsuitable for real time applications. Chui-Hsing et. al. [5] applied deterministic moment matching in order to estimate the parameters in IRT in real time for a system that can function in an online manner.

The present manuscript tries to devise a feedforward neural network model to mimic the Rasch model function used in IRT and attempt to have a generic technique which could be used for more and more complicated setting in a similar way and would be suitable for online settings.

III. Methodology and Findings

A mixed approach to IRT has been used by combining the "maximum information" selection and "MAP" estimator. For this pymc2.3.6, a statistical package for python, has been used. For this study, abilities of students ranging from 0.000 to 1.000 maintaining the same precision and difficulties of 500 items linearly spaced between 1 to 10 for 20 intervals, and a test length of 10 items, has been fixed.

For initial ability estimation, an average ability for all students, i.e., θ = 0.500 has been started with. The Item Information function to fetch the information given by all questions has been used and then the item that yields maximum information for a specific θ of a student has been chosen. This enabled to get the most suitable question for a student with a particular ability. Fig. 2 shows the code for the Item Information function.



```
def evalIRF (diff, theta):
    return 1/(1+np.exp(-1*(10*theta-diff)))

def evaldIRF (diff, theta):
    e = 2.718
    return 10*(np.log(e)*np.exp(-1*(10*theta-diff)))/np.power((1+np.exp(-1*(10*theta-
diff))),2)

def evalIIF (diff, theta):
    dp = evaldIRF (diff, theta)
    p = evalIRF (diff, theta)
    return np.power(dp,2)/(p*(1-p))
```

Fig. 2 Code for Item Information Function

Then a map has been created using the Item Information function, for all abilities ranging from 0.000 to 1.000 maintaining the precision, so that the maximum information could be stored for all abilities, thus making it easier and more efficient to fetch the most suitable question for any given ability. Fig. 3 shows the mapping between difficulties and abilities based on the Item Information Function. After creating the map, it is found out that the map represents a stair function because the higher the ability of a student the more difficult a question that student should be able to solve. If there would be the same number of difficulties as the abilities the representation would be a linear function, but here it has been assumed that the scale of difficulty is discretized and is to be set by the testing authorities.

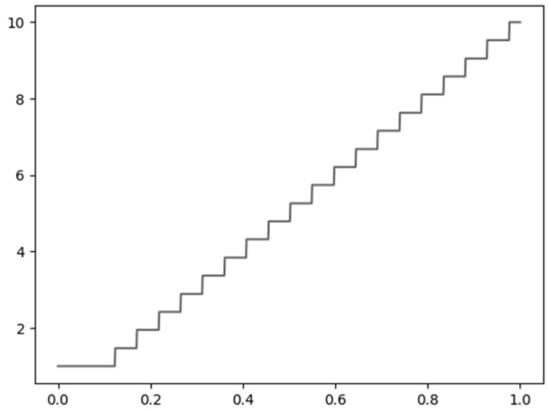
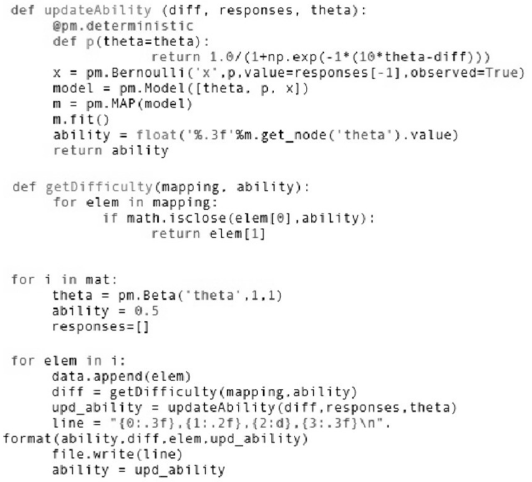


Fig. 3 Mapping between difficulties and abilities based on the Item Information Function

Since, the test length has been kept fixed to 10 items, it is clear that all students can solve the test in either of 2^{10} ways since this study adheres to a dichotomous model. So, the dataset has been statically generated by making the computer answer the test in all the 1024 ways possible. It is important to note that here a Bayesian approach for ability estimation, i.e., the "MAP" estimator by assuming a beta distribution (α=1, β=1) as a prior for each of the 1024 times has been used. Fig. 4 shows the related code.



```
def updateAbility (diff, responses, theta):
    @pm.deterministic
    def p(theta=theta):
            return 1.0/(1+np.exp(-1*(10*theta-diff)))
    x = pm.Bernoulli('x',p,value=responses[-1],observed=True)
    model = pm.Model([theta, p, x])
    m = pm.MAP(model)
    m.fit()
    ability = float('%.3f'%m.get_node('theta').value)
    return ability

def getDifficulty(mapping, ability):
    for elem in mapping:
        if math.isclose(elem[0],ability):
            return elem[1]


for i in mat:
    theta = pm.Beta('theta',1,1)
    ability = 0.5
    responses=[]

for elem in i:
    data.append(elem)
    diff = getDifficulty(mapping,ability)
    upd_ability = updateAbility(diff,responses,theta)
    line = "{0:.3f},{1:.2f},{2:d},{3:.3f}\n".
format(ability,diff,elem,upd_ability)
    file.write(line)
    ability = upd_ability
```

Fig. 4 Code for static generation of dataset

It has been found out that the changes in abilities are almost linear except for a few outliers because initial MAP estimation from a moderate level, i.e., θ=0.5 throws θ to the extremes of 0.0 or 1.0 on the initial item being administered incorrectly or correctly, respectively. Fig. 5 shows how the present ability of a student relates to the difficulty of the item being administered and the updated ability of the student based on the response for the item.

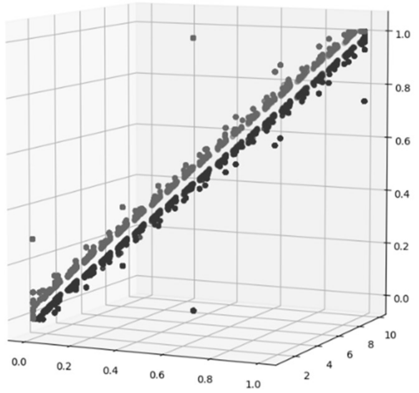


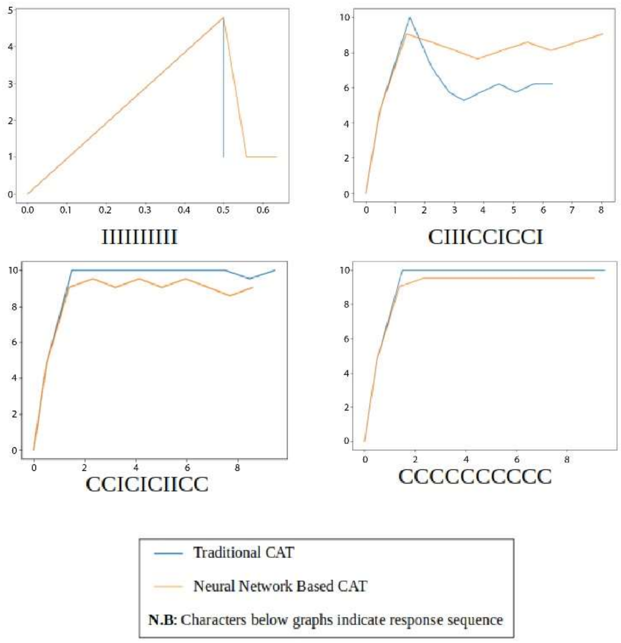
Fig. 5 Ability v/s Difficulty v/s Updated Ability. Light Gray Points: Correct Responses. Dark Gray Points: Incorrect Responses

The dataset that has been created above consists of 10240 data points. The dataset has three features, namely the present_ability, the difficulty, and the response, and the updated_ability column has been kept as the label for the model. This dataset, is then fed to a Neural Network for training the Regressor. For the Neural Network, scikit-learn, a machine learning package for python has been used. Scikit-learn provides MLPRegeressors, or Multi-Layer Perceptron Regressors for regression using Neural Networks. The optimized configuration parameters based on the limited dataset were finalized to 5 hidden layers, tolerance=0.000001, and ReLU (Rectified Linear Units) function as the activation function. After training the Regressor, it is found out that the Regressor is almost 96% accurate.

A comparison has then been made between the statistics-based CAT, i.e., the mixed approach CAT using "maximum-information" item selection and "MAP" estimator by extracting data from the dataset and the Neural Network based CAT by making the Regressor predict for the same set of values.

IV. Result Analysis

It has already been seen that the statically generated data fits into our model as the graphs for the map and the dataset can be successfully explained with the help of statistics and intuition. A few interesting things from the comparison between the two types of CATs have been found out. Fig. 7 shows the performance comparison between the traditional CAT model and the Neural Network based CAT model.


IIIIIIIII

CIIICCICCI

CCICICIICC

CCCCCCCCCC

Fig. 7 Scores of Traditional CAT v/s Scores of Neural Network based CAT

It can be clearly seen that the Neural Network based model does poorly in comparison to the traditional CAT model. The Neural Network model has an error rate of about 7-10% for learning the IRT model, based on differences of predicted scores, which could improve by training on larger dataset (as on a production system) and implementing a suitably deep network. It can be seen that the general trend of the trained model and its prediction of learner's ability always follows the traditional CAT trend with differences in the absolute value depending on the correctness (incorrectness) pattern.

To compare time-complexities between the two approaches, an analysis of the code has been made. It is of no surprise that the prediction time-complexity for the MLP Regressor is O(1), which implies that for 'n' questions, the total time complexity would be O(n) whereas on careful investigation of the traditional IRT approach for a web-based application that needed to update the ability after every answer gave us a total time complexity of $O(n^2)$ as every time the previous answers had to be taken into consideration since, as of now storing distributions directly into the database was not an option. For example, after the first question has been administered the ability estimation takes O(1) to update the ability as it is the only question that has been answered. For the second question it takes O(2) to update the ability as the first response needs to update the ability from the prior beta distribution and then the second response can update the distribution on top of that. This is mainly because of the fact that distribution is not stored in a database. Similarly, for the third question it takes O(3) to update the ability and so on. Thus, for a total of 5 questions, the total time complexity would be O(1)+O(2)+O(3)+O(4)+O(5). It can then naturally lead to the conclusion that the total time complexity for updating ability of a student who has been administered with 'n' questions is $O(n^2)$. After carrying out measurements on complexity evaluation that the web-based IRT system, takes almost 6-7 times more time to run through a set of ten questions and calculate and update the latent variables than the Neural Network based system. Although, this shows that the IRT based approach is a lot slower than the Neural Network based approach, it is important to note that on stand-alone applications, where questions and responses can be both stored and administered locally, the IRT based approach would also lead to a time complexity of O(n).

Another important point to note is that though the MAP estimator has the advantage that it can take in several responses and calculate the MAP value of the ability, such applications do not arise in a production system where questions are evaluated and ability updated one after another. It should also be noted that the MAP calculation involves the running of as optimization function which is a slower step than the forward step of a neural network which can be greatly parallelized.

V. Discussion

As it has already been observed, that the Neural Network based testing model can be effectively used in web-based applications for time-efficiency, not having to compromise a lot on the performance of the test. On crucial examination of the scores estimated by both the models, it has been observed (Fig 7) that our IRT based CAT model makes huge jumps to extreme difficulties after administration of the first attempt, i.e., if a test-taker gives a correct response to the first question, his ability is immediately estimated to be 1.0 whereas on giving an incorrect response, his ability is immediately estimated as 0.0. This kind of an estimation might not be ideal at all times, and often the second question or item might actually get wasted and not contribute enough to the estimation process. This, however, is not the case with the Neural-Network based model which doesn't jump to extremes and thus, can be used for modelling ideal tests.

The main upside, however, to using a Neural Network based testing model is the ability of a Neural Network to act as a function approximator. This ability enables the testing model to not be limited to distributions under IRT. It might be used for more general distributions. Thus, other functions not related to IRT could be approximated from the self-learning property of Neural Networks. It could also be used to run experiments in an adaptive learning setting and incorporate other parameters in such functions to incorporate expressions whose formulation are not exactly understood but could be constructed from data.

Different functions which could be possible to learn are:

1. Type of question, Difficulty level, proficiency → probability of answering correctly
2. Difficulty level, Proficiency, time allotted for a question → probability of answering correctly
3. Question difficulty, days after taught, proficiency → probability of answering correctly
4. Type of question, probability of answering correctly, proficiency → difficulty of question
5. Length of question, time of day, proficiency → probability of answering correctly

Another way to look at this line of reasoning is that the IRT equation, (1) computes the probability of correct response and hence is equivalent to the equation below

$$P(\theta) = F\ (c_i, a_i, \theta, b_i)$$

Where a_i, b_i, c_i are item parameters and θ is the person's ability. In this study attempt has been made to replace F with M where M is a neural network base model and hence the equation would become

$$P(\theta) = M\ (c_i, a_i, \theta, b_i)$$

This concept can be extended in case of larger datasets (as in production systems) where we could obtain the probability of answering correctly using observed responses with some confidence interval and used as labels for training a linear regression model or logistic regression model. Hence the extended equations could become.

$$P(\theta) = M\ (X_i)$$

Where X_i could be a parameter set (listed above) for which data is being collected.

Though the parameters cannot be directly derived but any of the columns could be considered as a label as long as it is sensible to do so and could be justified in a causal framework. Of course, other statistical techniques should be used to actually prove and understand that there is a correlation since the dilemma of correlation versus causation would still apply. If it could be justified that such a correlation exists, then the functions could be learnt and used in the systems to formulate better testing methods.

Piech. et. al. [6] discuss the concept of the application of RNNs to knowledge tracing to provide incorporation of other features as inputs (such as time taken), explore other educational impacts (such as hint generation, dropout prediction), and validate hypotheses posed in education literature (such as spaced repetition, modeling how students forget).

VI. Conclusion

This study successfully compares Neural Network based testing model to an IRT based testing model. It has been seen that the performance drop of 7-10% hardly compares to the time complexity reduction from $O(n^2)$ to $O(n)$ in some web-based scenarios. Moreover, further studies can be aligned more with exploring the role of Neural Networks as function approximators to model complex learning environments. Further investigations can be made as to which characteristic would be a better feature while modelling such complex adaptive systems. The power of Neural Networks can seemingly put a huge impact on the adaptive learning scenario.

Acknowledgment

This work has been done as a part of developing the product for *ziksuka.com* an education startup working on creating an adaptive learning platform.

References

[1] F. M. Lord, M. R. Novick, "Statistical theories of mental test scores, Reading", in *MA: Addison-Wesley*, 1968.

[2] R. Adams, and M. Whu, "Modelling a Dichotomously Scored Multiple Choice Test with the Rasch Model", in *ConQuest.* 2010.

[3] B. Veldkamp, and M. Matteucci, "Bayesian Computerized Adaptive Testing. Essay: Evaluation and Public Policies in Education", in 2013.

[4] R. Pimprika, S. Ramachandran, and K. SenthilKumar, "Use of Machine Learning Algorithms and Twitter Sentiment Analysis for Stock Market Prediction. International Journal of Pure and Applied Mathematics", in 2017.

[5] R. C. Weng and D. S. Coad, "Real-time Bayesian parameter estimation for item response models", in 2017.

[6] C. Piech, J. Spencer, J. Huangz, S. Ganguli, "Deep Knowledge Tracing", in June 2015.

[7] R. Kuo, W. P. Lien, M. Chang, and J. S. Heh, "Analyzing Problem's Difficulty based on Neural Networks and Knowledge Map", in *Educational Technology & Society*, 7 (2), 2004, 42-50.

[8] F. M. Plumed and R. B. C. Prudecio and A. M. Uso and J. H. Orallo, "Making Sense of Item Response Theory in Machine Learning".

[9] J. P. Lalor, H. Wu, T. Munkhdalai, Hong Yu, "An Analysis of Ability in Deep Neural Networks".

[10] K. H. Wilson, Y. Karklin, B. Han, C. Ekanadham, "Back to the basics: Bayesian extensions of IRT outperform neural networks for proficiency estimation", in *arXiv:1604.02336v2* [cs.AI], May 2016.

[11] K. H. Wilson, X. Xiong, M. Khajah, R. V. Lindsey, S. Zhao, Y. Karklin, E. G. VanInwegen, B. Han, C. Ekanadham, J. E. Beck, N. Heffernan, M. C. Mozer, "Estimating student proficiency: Deep learning is not the panacea", in *NIPS*, 2016.