Predicting Item Characteristic Curve (ICC) Using a Softmax Classifier

Dmitry I. Belov

Abstract The objective of item difficulty modeling (IDM) is to predict the statistical parameters of an item (e.g., difficulty) based on features extracted directly from the item (e.g., number of words). This paper utilizes neural networks (NNs) to predict a discrete item characteristic curve (ICC). The presented approach exploits one-to-one mapping from monotonically non-decreasing discrete ICCs to probability mass functions (PMFs). An NN was trained using soft labels for each item (by mapping ICCs to PMFs), with a softmax output layer representing PMF and the Kullback-Leibler divergence representing a loss function. Results of a cross-validation of the NN on 1742 retired logical reasoning items from the Law School Admission Test are presented and discussed.

Keywords Item difficulty modeling · Item response modeling · Item characteristic curve · Neural networks · Machine learning · Natural language processing · Semantic similarity

1 Introduction

The primary task of item difficulty modeling (or, perhaps more appropriately, item response modeling) is to predict the statistical properties of an item, such as difficulty, based on features extracted directly from the item. An example of such a feature might be the number of words in the item. Item difficulty modeling (IDM) adopts various techniques from data mining, machine learning, and natural language processing. For a review of IDM and its applications see, for example, Sheehan and Mislevy (1990), Huff (2006), or Ferrara et al. (2021).

Due to the recent massive migration of high-stakes testing programs from in-person testing to online testing, the following two issues became much harder to

D. I. Belov (✉)
Law School Admission Council, Newtown, PA, USA
e-mail: dbelov@lsac.org

M. Wiberg et al. (eds.), *Quantitative Psychology*, Springer Proceedings
in Mathematics & Statistics 393, https://doi.org/10.1007/978-3-031-04572-1_13

address without IDM. First, online test proctoring cannot protect against existing technology used to steal test content. In the world of online testing, item preknowledge may happen due to (a) using the same test section over different time slots within the same administration due to a limited number of live proctors; and (b) pretesting new items. Second, a larger number of test sections is needed to tackle the problem set forth in (a). However, developing new items without controlling their statistical parameters may unbalance the pool and limit the assembly of more test sections, thus decreasing item pool usability.

A recent meta-analysis by Ferrara et al. (2021), compiled from over 100 IDM-related studies, demonstrated the following. Only about 10% of the studies reported the coefficient of determination D over 0.5. Most of the research dealt with reading comprehension (RC) items (commonly associated with a long text). The most popular item question concerned the main idea of the passage. All methods predicted only item difficulty. The majority of prediction models utilized linear regression or a regression tree. Features defined by item writers (i.e., nonautomatic features) were often the best predictors of item difficulty.

The current paper goes beyond the typical IDM research described above and instead applies neural networks (NNs) to predict item characteristic curves (ICCs) for logical reasoning (LR) items (which are associated with a shorter amount of text compared to RC items) from the Law School Admission Test (LSAT). For a given item, its ICC maps the examinee's latent trait (ability) to the probability that the item will be answered correctly (Lord, 1980). The ICC is bounded between 0 and 1, is monotonically non-decreasing, and is commonly assumed to take the shape of a logistic function.

This paper considers discrete ICCs defined on the set of ability levels $\{-3, -2, -1, 0, 1, 2, 3\}$ (a coarse grid was chosen just for the sake of illustration; a finer grid is easily supported). There are at least three advantages of predicting discrete ICCs. First, one can avoid the noise produced by the item parameter estimation procedure, while fitting empirical ICCs with an IRT model, by dealing directly with empirical ICCs (see Fig. 1). Second, ICCs provide unification when the item pool has a mixture of models (e.g., part of the item pool modeled by the two-parameter logistic model [2PLM] and the other part by the 3PLM; Lord, 1980): all parts can be represented by ICCs computed using corresponding models. Third, once discrete ICCs are predicted, it is easy to simulate responses from any targeted population of examinees and then calibrate IRT models (1PLM, 2PLM, or 3PLM), thus providing continuous ICCs.

This paper is organized as follows. First, the construction, training, and validation of a neural network (NN) to predict ICCs are described. Second, the data from retired LR items from the LSAT and features extracted from each item are depicted. Third, the results of applying the developed NN to the data are presented. Finally, the results are critically reviewed; followed by a discussion about further research, design changes, and practical applications.

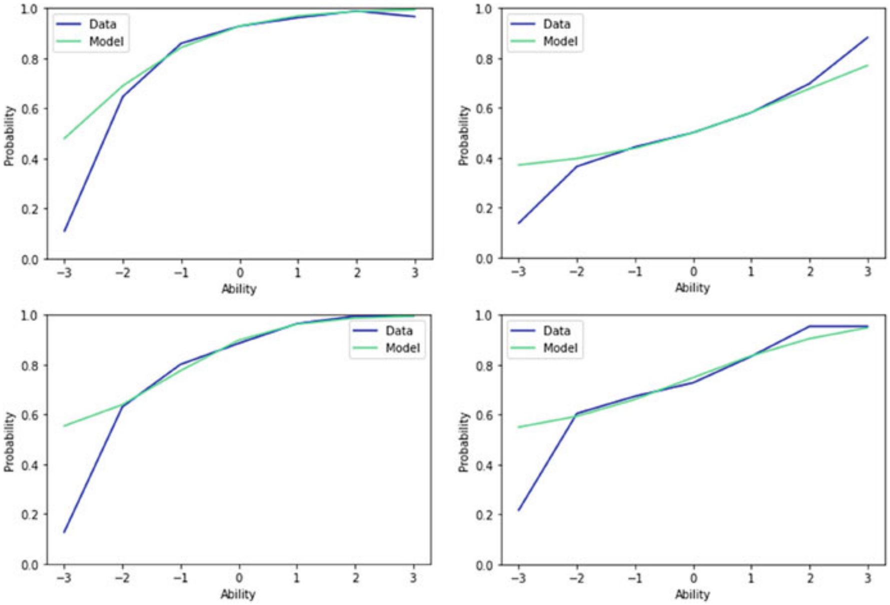

Fig. 1 When empirical ICCs are fitted by item response theory (IRT) models, there is always a possibility of misfit. Here are four real cases showing a large misfit between empirical ICCs (Data) and ICCs produced by a fitted three-parameter logistic (3PL) model (Model)

2 Method

A neural network (NN) can be considered a vector function with a vector argument. In this study, the NN maps the vector of features extracted from an item to its ICC. Parameters of this function can be estimated using a “training” sample, where for each argument there is a predetermined output of the function called a label, by minimizing a loss function. In this process, called *supervised learning*, it is crucial that the training sample be representative of the general data. The loss function measures a discrepancy between the output of NN applied to given arguments and their labels from the training sample. A typical NN has a network structure with layers of interconnected nodes (Fig. 2) inspired by mathematical modeling of a biological brain. Each connection has a weight. Each node has an activation function that maps the node’s input to the node’s output. The node’s input can be defined as a scalar product of the vector of outputs of the nodes connected to this node and the vector of weights of the corresponding edges plus an intercept. The weights and intercepts are estimated by the supervised learning. Neural networks were successfully applied for image recognition and recently were extended to other fields (Skansi, 2018). For more information about neural networks and machine learning terminology used in this paper, the reader is referred to Goodfellow et al. (2016) or Skansi (2018).

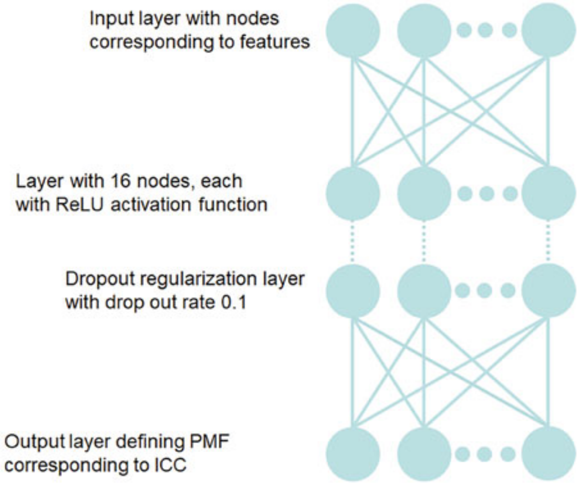
Input layer with nodes corresponding to features
Layer with 16 nodes, each with ReLU activation function
Dropout regularization layer with drop out rate 0.1
Output layer defining PMF corresponding to ICC

Fig. 2 Structure of the NN

Before describing the NN, a specific transformation is built based on the assumption that the ICC is discrete and monotonically non-decreasing. This one-to-one transformation maps the discrete ICC to the probability mass function (PMF), where $n = 7$ is the number of ability levels (−3, −2, −1, 0, 1, 2, 3) indexed as (1, 2, 3, 4, 5, 6, 7). Direct mapping is used to create labels, and inverse mapping is used to predict ICCs; they are defined by the following two equations, respectively:

$$\begin{aligned} &\mathrm{PMF}\,[1] = \mathrm{ICC}\,[1] \\ &\mathrm{PMF}\,[i] = \mathrm{ICC}\,[i] - \mathrm{ICC}\,[i-1]\,, \quad i = 2, 3, \ldots, n \\ &\mathrm{PMF}\,[n+1] = 1 - \mathrm{ICC}\,[n] \end{aligned} \tag{1}$$

$$\begin{aligned} &\mathrm{ICC}\,[1] = \mathrm{PMF}\,[1] \\ &\mathrm{ICC}\,[i] = \mathrm{PMF}\,[i] + \mathrm{PMF}\,[i-1]\,, \quad i = 2, 3, \ldots, n \end{aligned} \tag{2}$$

There are numerous degrees of freedom in terms of the number of layers, the types of layers, the number of nodes in each layer, the types of activation functions, and the types of regularizations, all affecting properties of the corresponding NN. These so-called hyperparameters are usually identified via a cross-validation study, which is discussed later in this section. The result of that study is the following NN (Fig. 2):

1. Input layer with nodes corresponding to features extracted directly from an item (see the next section about the actual features used in this study), where the number of nodes corresponds to the number of features used for predicting the ICC.
2. Layer with 16 nodes and ReLU activation function $f(x) = \max(0,x)$ for each node.

3. Dropout regularization layer with 0.1 rate that functions as follows: during the training of the NN, the output of a node from the previous layer (Fig. 2) is dropped with probability 0.1. This layer prevents an overfitting of the NN to the training data.
4. Output layer with softmax activation function $f(z_i) = \exp(z_i)/S$, $S = (\exp(z_1) + \exp(z_2) + \cdots + \exp(z_8))$, $i = 1,2,\ldots,8$, where z_i is input of node i of the output layer. This layer allows the NN to perform a soft classification, where each class gets a positive probability of being assigned such that the sum of probabilities is 1. This is in contrast to a conventional classification, where the class assignment probabilities are from {0, 1}. Thus, the output of the NN is the PMF.

A common choice of loss function for the soft classification is cross entropy, since (usually in practice) one class is intended to be selected. However, in this study, the PMF (corresponding to the ICC) should be matched as closely as possible; therefore, Kullback-Leibler divergence (Kullback & Leibler, 1951) was chosen as a loss function. This technique also works as an additional regularization to ensure that predicted ICCs are always monotonically non-decreasing.

In order to train the NN, a stochastic gradient descent (SGD, Goodfellow et al., 2016) minimized the loss function under the following parameters: learning rate 0.01; number of epochs 30; number of samples per gradient update 100. The parameters of SGD, the number of layers in the NN, the number of hidden layers (i.e., layers between the first layer and the last layer), the number of nodes in hidden layers, and the drop rate of 0.1 for the regularization layer were chosen during multiple empirical trials in order to achieve a stable output of a cross-validation described next. The NN has only one hidden layer with 16 nodes (Fig. 2); any increase either in the number of hidden layers or in the number of nodes degraded the results.

To validate the NN, the k-fold cross-validation method was used (Goodfellow et al., 2016), where labeled data is divided into k non-overlapping samples. Then, in each iteration (out of k iterations total), the $k - 1$ samples are used to train the NN and 1 sample is used to test the NN on predicting ICCs. This way, each data point is used once to train the NN and once to test it. In this study, $k = 10$ in order to comply with studies reviewed by Ferrara et al. (2021). The output of the validation includes error (E), residual (R), and coefficient of determination D computed using errors on true and average (estimated on testing samples) ICCs.

With a true ICC (given the ICC of an item) and its prediction by the NN, denoted as ICC and ICC*, respectively, the error is computed as follows:

$$E = \left(\left(\mathrm{ICC}\,[1] - \mathrm{ICC}^*\,[1]\right)^2 + \cdots + \left(\mathrm{ICC}\,[7] - \mathrm{ICC}^*\,[7]\right)^2\right)/7 \tag{3}$$

And the residual is computed as follows:

$$R = \left(\left(\mathrm{ICC}\,[1] - \mathrm{ICC}^*\,[1]\right) + \cdots + \left(\mathrm{ICC}\,[7] - \mathrm{ICC}^*\,[7]\right)\right)/7 \tag{4}$$

Additional output includes outliers with E over 0.15 and data points with R within a certain range.

3 Data

A total of 1742 retired logical reasoning (LR) items from the Law School Admission Test were used in this study to build a prediction model. Each item has the following structure:

1. Passage
2. Question
3. Five answer options (A, B, C, D, E)

Detailed information about each LR item can be described as follows:

1. Text of the passage
2. Text of the question
3. Text of each answer option (A, B, C, D, E)
4. Correct option index (the key)
5. Item type
6. Item subtype defining the type of question
7. Item property 1
8. Item property 2
9. Item property 3
10. Item property 4
11. Item rank {1, 2, 3, 4} (an estimation of item difficulty by item writer)
12. Item pretest position (item position, defined by item writer, in unscored section for pretesting new items)
13. ICC computed from corresponding 3PLM (response matrices were not available for computing empirical ICCs)

The above information was used to compute multiple features for each item. An additional 2009 LR items without item rank and item pretest position were used to compute some numerical features for the above 1742 LR items. As a rule of thumb, an acceptable performance of an NN is observed when there are around 5000 labeled data points per category (Goodfellow et al., 2016). Therefore, the studied data is too small to expect superior results; in fact, because there are nine different question types, the data is partitioned into even smaller pieces.

The rest of this section describes features extracted directly from an item. For each item, six categorical features were provided by the item writer:

1. Item type (categorical feature from {1, 2})
2. Item subtype (categorical feature from {1, 2, 3, 4, 5, 6, 7, 8, 9})
3. Item property 1 (categorical feature from {1, 2, 3})
4. Item property 2 (categorical feature from {1, 2, 3, 4, 5})

5. Item property 3 (categorical feature from {1, 2, 3})
6. Item property 4 (categorical feature from {1, 2})

Each categorical feature was represented as a one-hot code vector (Goodfellow et al., 2016); for example, if the item type (see above) was 1 then it was represented as vector $(1, 0)$, and if the item type was 2 then it was represented as vector $(0, 1)$. Using this representation allows neural networks to apply the divide-and-conquer strategy similarly to CARTs (Breiman et al., 1984). There were 12 groups of numerical features:

1. Item rank, denoted as *itemRank.*
2. Item pretest position, denoted as *itemPosition.*
3. Text features for passage: *PTF.nSentences* (number of sentences), *PTF.nWords* (number of words), *PTF.nNouns* (number of nouns), *PTF.nNSynsets* (number of synsets (Fellbaum, 1998) of the nouns), *PTF.nVerbs* (number of verbs), *PTF.nVSynsets* (number of synsets of the verbs), *PTF.nAdjs* (number of adjectives), *PTF.nASynsets* (number of synsets of the adjectives), *PTF.readability* (Dale–Chall readability index; Dale & Chall, 1948).
4. Text features for question: *QTF.nWords* (number of words), *QTF.nNouns* (number of nouns), *QTF.nNSynsets* (number of synsets of the nouns), *QTF.nVerbs* (number of verbs), *QTF.nVSynsets* (number of synsets of the verbs), *QTF.nAdjs* (number of adjectives), *QTF.nASynsets* (number of synsets of the adjectives), *QTF.readability* (Dale–Chall readability index).
5. Text features for options: *OTF.nSentences* (number of sentences), *OTF.nWords* (number of words), *OTF.nNouns* (number of nouns), *OTF.nNSynsets* (number of synsets of the nouns), *OTF.nVerbs* (number of verbs), *OTF.nVSynsets* (number of synsets of the verbs), *OTF.nAdjs* (number of adjectives), *OTF.nASynsets* (number of synsets of the adjectives), *OTF.readability* (Dale–Chall readability index).
6. Semantic similarity between passage and correct option (answer) denoted as *spa*. Semantic similarity between two texts is computed as a scalar product between two embeddings corresponding to two texts; for information about embeddings see Goodfellow et al. (2016).
7. Semantic similarity between passage with correct option (answer) and other options (distractors), denoted as *spad.*
8. Mean, variance, minimum value, and maximum value of semantic similarity between answer and distractors, denoted as: *sadMean*, *sadVar*, *sadMin*, *sadMax.*
9. Mean, variance, minimum value, and maximum value of semantic similarity between all unique pairs of options, denoted as: *sooMean*, *sooVar*, *sooMin*, *sooMax*
10. Mean, variance, minimum value, and maximum value of semantic similarity between all unique pairs of sentences in the passage, denoted as: *sppMean*, *sppVar*, *sppMin*, *sppMax.*
11. The additional 2009 LR items (called *atlas items*) without item rank and item pretest position were used to compute this group of features. The atlas

items were partitioned into nonintersecting classes based on their type and subtype. An item, used for constructing the NN, was associated with a class corresponding to the type and subtype of the item. Each element of the associated class had the passage and difficulty of some atlas item. For each element in the class, the semantic similarity between the element's passage and the item's passage multiplied by the element's difficulty was sampled. Finally, the mean, variance, minimum value, and maximum value were estimated from the sample and denoted as: *sppbMean*, *sppbVar*, *sppbMin*, *sppbMax*.

12. Similarly to the previous group, the following features were computed for the question: *sqqbMean*, *sqqbVar*, *sqqbMin*, *sqqbMax*.

Each numerical feature from above was normalized by subtracting its mean and then dividing by its standard deviation; such normalization substantially improves the convergence of SGD (Goodfellow et al., 2016).

Table 1 shows features that correlate with at least |0.1| with *a*, *b*, or *c* parameters of 3PLM. One may observe some interesting patterns in Table 1. The highest correlations are observed for *itemRank* and *itemPosition*. Most correlations are with item difficulty except for *sppbMean* and *sqqbMean* (which perhaps relate to their estimation procedure). Most text features in Table 1 are for options, only two for

Table 1 Features that correlate with at least |0.1| with *a*, *b*, or *c* parameters of 3PLM

Feature (see description of each feature above)	Correlation with *a*	Correlation with *b*	Correlation with *c*
itemRank	**0.10**	**0.43**	0.07
itemPosition	**0.12**	**0.43**	0.08
sppbMean	**0.12**	**0.15**	**0.17**
sqqbMean	**0.14**	**0.17**	**0.21**
sqqbVar	0.00	**0.12**	0.02
sqqbMax	0.02	**0.12**	0.03
sppMean	0.06	**0.11**	−0.02
sadMean	0.07	**0.10**	0.03
sadMin	0.06	**0.11**	0.03
sooMean	0.07	**0.11**	0.01
sooMin	0.07	**0.11**	0.03
QTF.nVerbs	**0.10**	0.08	0.07
QTF.nVSynsets	**0.13**	**0.10**	0.07
OTF.nSentences	0.06	**0.15**	**0.11**
OTF.nWords	0.05	**0.20**	**0.12**
OTF.nNouns	0.01	**0.14**	0.07
OTF.nNSynsets	0.01	**0.13**	0.03
OTF.nVerbs	0.04	**0.18**	**0.14**
OTF.nVSynsets	0.02	**0.15**	**0.11**
OTF.nAdjs	0.03	**0.12**	0.03
OTF.readability	−0.07	**−0.12**	−0.04

question, and none for the passage, which is unexpected. Overall, text features have higher absolute correlations than features based on semantic similarity. Feature *sppMean* has a positive correlation with item difficulty, which means that more difficult items may have more closely related sentences in their passages (this is supported by real data).

4 Results

Table 2 shows the results of 10-fold cross-validation of the NN for different subsets of features. One can observe that the best results were achieved with features provided only by item writers (see fifth column in Table 2). Thus, the use of automatically generated features did not improve the results, although some of them weakly correlate with a, b, or c parameters of 3PLM (Table 1).

The cross-validation of the NN constructed from categorical features and numerical features *itemRank*, *itemPosition* provided additional results as follows. Graphical representation of error and residual computed for each ability level separately is illustrated in Figs. 3 and 4, where distributions of error and residual are characterized by box plots. One can see that the largest errors and residuals happened for ability level 0. Figure 5 shows a random sample of nine pairs of true and predicted ICCs, shown as blue and green curves, respectively, where the residual fell within one standard deviation from its mean; overall, 67% of predicted ICCs satisfied that range. One can observe that true and predicted ICCs are different in terms of variability (see Fig. 5), and that the variability of true ICCs is higher than the variability of predicted ICCs (Fig. 6; this finding is compatible with low values of D in Table 2).

5 Discussion

This paper describes the NN approach to predicting ICCs using features extracted directly from an item. A total of 1742 retired LR items from the LSAT were used to build, train, and validate the NN.

Multiple features extracted directly from an item were used in the input layer of the NN (see Fig. 2). A cross-validation study using different subsets of the features demonstrated (see Table 2) that using features provided by item writers (categorical features and numerical features *itemRank*, *itemPosition*) produced the best predictions whereas automatically generated features did not improve the predictions. Even more, just using two features (*itemRank* and *itemPosition*) produced the second best results. This indicates that the data sample is too small for the categorical features to play any role in prediction. That may also explain why automatically generated features were useless, since the number of items in each

Table 2 Results of 10-fold cross-validation of the NN for different subsets of features

Measure	Categorical and numerical features	Categorical and numerical features from Table 1	Categorical and numerical features from Table 1 without *itemRank*, *itemPosition*	Categorical and *itemRank*, *itemPosition*	Only *itemRank* and *itemPosition*
E mean	0.022	0.021	0.024	0.020	0.021
E st.dev.	0.025	0.023	0.024	0.022	0.022
R mean	−0.003	0.000	0.000	−0.001	0.000
R st.dev.	0.120	0.116	0.125	0.115	0.116
R skewness	−0.188	−0.205	−0.150	−0.194	−0.191
D	0.105	0.153	−0.007	0.178	0.160
Number of outliers	8	2	1	1	0
Min *E* on the outliers	0.151	0.190	0.153	0.161	–
Max *E* on the outliers	0.239	0.192	0.153	0.161	–

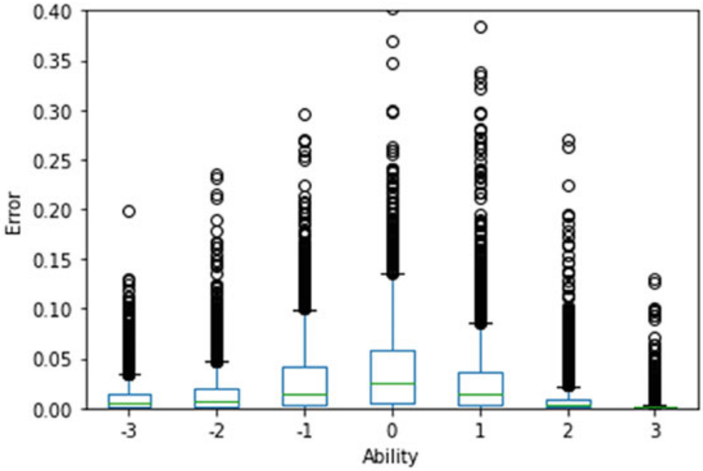

Fig. 3 Box plots of error for each ability level

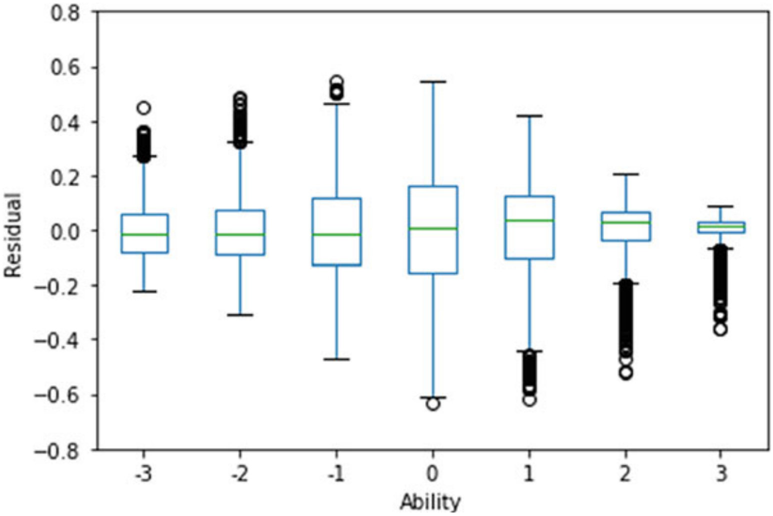

Fig. 4 Box plots of residual for each ability level

category was even smaller. The best subset of features provided a low *D*, which was expected as the data was too small.

This study is in line with 90% of the studies (Ferrara et al., 2021) reporting a low coefficient of determination *D*. This paper confirms that the best predictors are features provided by item writers (see Table 2). As expected, in contrast to reading comprehension items, LR items have a weak correlation between text features and item difficulty (see Table 1).

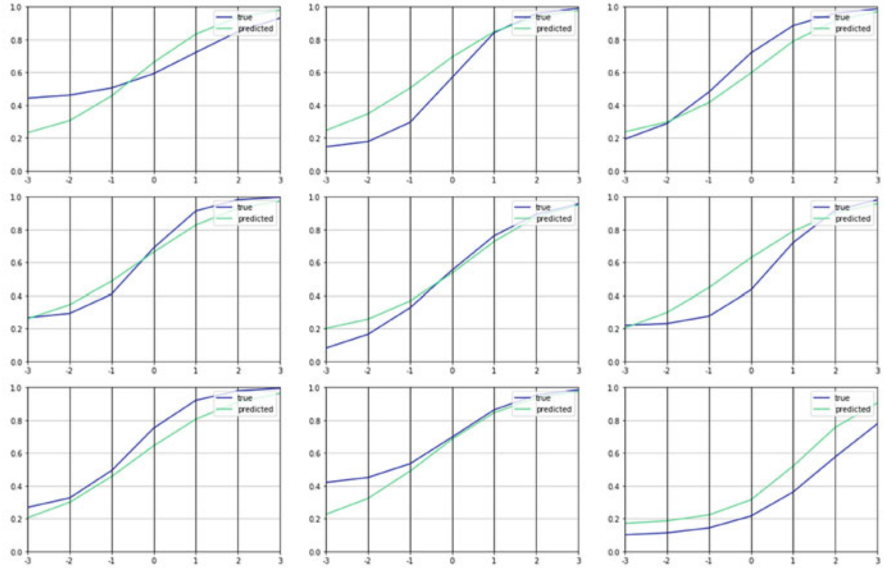

Fig. 5 A random sample of nine pairs of true and predicted ICCs (blue and green curves respectively), where the residual was within one standard deviation from its mean. Overall, 67% of predicted ICCs fell within that range

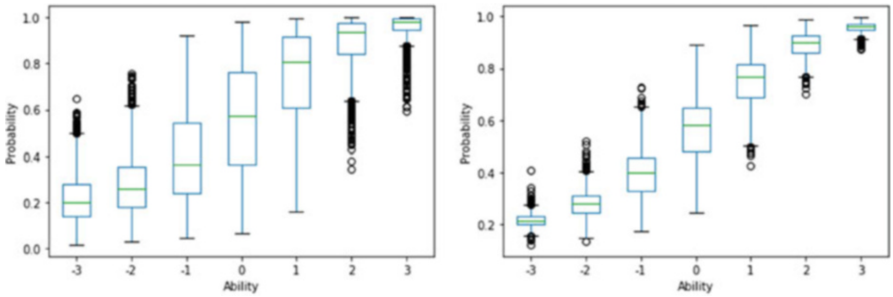

Fig. 6 Distribution of true ICCs (left box plot) and predicted ICCs (right box plot)

Online test proctoring cannot protect against existing technology used to steal test content. Therefore, one has to accept an error generated by a predictive model if the error is symmetrically distributed about zero. A possible application scenario would be as follows: (1) develop a model predicting ICC; (2) use predicted ICCs to simulate a real administration of pretested sections; (3) based on simulated responses, calibrate items (e.g., calibrate 3PLMs); (4) assemble a test using these items as already pretested; (5) administer the test to a real population; (6) use real responses for recalibrating items and updating the model. In this scenario, new items are not pretested (i.e., not administered to a real subpopulation of examinees). Therefore, a regular scaling based on anchor items pretested in the past is no longer possible. This can be addressed by using an additional, previously administered section as an anchor section. Modern methods of detecting item preknowledge (Belov, 2016, 2020; Drasgow et al., 1996; Karabatsos, 2003; McLeod et al., 2003; Tendeiro & Meijer, 2012; van Krimpen-Stoop & Meijer, 2001) can be applied to filter out examinees with possible preknowledge of the anchor section in order to estimate scaling coefficients without bias.

If presented approach were applied to predict empirical ICCs, then the assumption of monotonically non-decreasing ICCs could be violated by some empirical ICCs (Fig. 1). In this case, the NN could be modified as follows: The output layer with linear activation function could have seven nodes corresponding to ability levels (-3, -2, -1, 0, 1, 2, 3), and the loss function could be the mean squared error.

The approach could be easily adapted to predict parameters of 3PLM directly. The only modification would be that the output layer with linear activation function would have three nodes (for a, b, and c, respectively) and the loss function would be the mean squared error.

Future research will be directed toward procuring a larger data sample, engineering new features, minimizing E, and maximizing D, while keeping R symmetrically distributed about zero. The latter is crucial in order for items with predicted statistical parameters to be included on a test. Larger data may allow the use of embeddings (Goodfellow et al., 2016) for the passage, question, and options directly (instead of computing features as semantic similarities between various parts of the item, as was done in this study); that way a deeper NN could “figure out” more useful features. Text features used in this study can be extended with Coh-Metrix (Ferrara et al., 2021). Another method to generate new features is described in the final two groups of numerical features in the Data section above. For a given item, the method could be generalized as follows. From the atlas items (items without item rank and item pretest position), form a class using a certain criterion; for example, select items with multiple negations in their passages. Each element of the class has the passage and difficulty of some atlas item. For each element in the class, the semantic similarity between the element’s passage and the item’s passage, multiplied by the element’s difficulty, is sampled. Then the mean, variance, minimum value, and maximum value estimated on the sample could be used as the new features.

References

Belov, D. I. (2016). Comparing the performance of eight item preknowledge detection statistics. *Applied Psychological Measurement, 40*, 83–97.

Belov, D. I. (2020). Monte Carlo detection of examinees with item preknowledge. *Behaviormetrika, 48*, 23–50.

Breiman, L., Friedman, J. H., Olshen, R., & Stone, C. J. (1984). *Classification and regression trees*. Wadsworth International Group.

Dale, E., & Chall, J. (1948). A formula for predicting readability. *Educational Research Bulletin, 27*, 11–20.

Drasgow, F., Levine, M. V., & Zickar, M. J. (1996). Optimal identification of mismeasured individuals. *Applied Measurement in Education, 9*, 47–64.

Fellbaum, C. (Ed.). (1998). *WordNet: An electronic lexical database*. MIT Press.

Ferrara, S., Steedle, J., & Frantz, R. (2021). *Response demands of reading comprehension items: A review of item difficulty modeling studies* [Manuscript submitted for publication]

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT Press.

Huff, K. (2006). *Using item difficulty modeling to inform descriptive score report*. Annual Meeting of the National Council on Measurement in Education, San Francisco, CA, United States

Karabatsos, G. (2003). Comparing the aberrant response detection performance of thirty-six person-fit statistics. *Applied Measurement in Education, 16*, 277–298.

Kullback, S., & Leibler, R. A. (1951). On information and sufficiency. *Annals of Mathematical Statistics, 22*, 79–86.

Lord, F. M. (1980). *Applications of item response theory to practical testing problems*. Lawrence Erlbaum.

McLeod, L. D., Lewis, C., & Thissen, D. (2003). A Bayesian method for the detection of item preknowledge in computerized adaptive testing. *Applied Psychological Measurement, 27*, 121–137.

Sheehan, K., & Mislevy, R. J. (1990). Integrating cognitive and psychometric models to measure document literacy. *Journal of Educational Measurement, 27*(3), 255–272.

Skansi, S. (2018). *Introduction to deep learning*. Springer.

Tendeiro, J. N., & Meijer, R. R. (2012). A CUSUM to detect person misfit: A discussion and some alternatives for existing procedures. *Applied Psychological Measurement, 36*, 420–442.

van Krimpen-Stoop, E. M. L. A., & Meijer, R. R. (2001). CUSUM-based person-fit statistics for adaptive testing. *Journal of Educational and Behavioral Statistics, 26*, 199–217.