Research on the Difficulty Prediction for Questions Based on Learners' Response

Jiaqi Long[a], Hui Zhao[b], Jie Pu[c], Yifan Liu[d], Binghan Ju[e], Suojuan Zhang[*]

School of Command and Control Engineering, Army Engineering University of PLA, Nanjing 210007, Jiangsu, China

[a]2908555057@qq.com, [b]xlbhcf@foxmail.com, [c]pj85@qq.com, [d]1346691903@qq.com, [e]jubinghan206@126.com, [*]Corresponding author: suojuazhang@aeu.edu.cn

Abstract. The question difficulty assessment is an important research direction in educational data mining. The traditional difficulty assessment is often completed manually, which is time-consuming and subjective. Existing difficulty prediction models are usually limited to the final score or only for statistical analysis of question text, without combining learners' responses and question context or detailed information. They often cannot effectively reflect the difference between the cognition of the question and the difficulty of the question, unable to meet the requirements of the instructional practice. Therefore, this paper aims to propose a question difficulty prediction model based on learners' responses and combine natural language processing technology to realize the automatic prediction of question difficulty. Specifically, the paper first based on the BERT training model, extracts the question information embedded vector, combined with convolutional neural network and long and short-term memory network, the fusion of learners' response (including score, response time, submit time, etc.), establish the correlation between the question text information and the question difficulty, construct the difficulty prediction of questions model based on learners' response, and achieve accurate question difficulty prediction.

Keywords: difficulty prediction; deep learning; learners' response; intelligent education

1 Introduction

In the Education Informationization 2.0 Action Plan, the Ministry of Education proposes to promote intelligent education[1] vigorously. In the China Education Modernization 2035, the State Council of China proposes to build an integrated intelligent education service platform to achieve universal learning and personalized teaching[2]. To achieve personalized learning in intelligent education, it is necessary to accurately measure learners' knowledge levels and provide targeted questions to enhance their weaknesses[3]. Appropriate difficulty questions to be recommended is a critical issue. As an important research direction in intelligent education, the difficulty prediction of

Y. Kuang et al. (eds.), *Proceedings of the 2024 5th International Conference on Education, Knowledge and Information Management (ICEKIM 2024)*, Atlantis Highlights in Computer Sciences 22,
https://doi.org/10.2991/978-94-6463-502-7_46

questions involves fields such as educational measurement, psychology, computer science, etc. Questions at appropriate difficulty can accurately evaluate learners' knowledge status, stimulate their learning potential, and help them bridge the zone of proximal development[3, 4].

The difficulty prediction of questions is a complex task that needs to consider many factors, such as the concept knowledge of the questions, difficulty coefficient, question type, and differences among learners. Traditional difficulty prediction methods usually rely on manual experience or machine learning models based on feature engineering, which have limited accuracy and require a lot of time and human resources[5]. Moreover, the predicted results are subject to human subjective factors[6, 7]. Difficult prediction models often limit themselves to final score submissions or only conduct statistical analysis on texts without considering learners' response and contextual information of question texts. These models may not fully capture the differences in question difficulty between the questioner and the respondent effectively. Therefore, it is important to explore how to extract more information from question texts to achieve more accurate difficulty predictions.

At the same time, this paper carries out difficulty prediction research for programming questions. The difficulty of programming questions involves multiple factors, such as background knowledge, language difficulty, problem-solving approaches, etc. The weight of these factors may be different, too. In addition, learners' response can also reflect question difficulty, such as correct rate, time taken, number of submissions, etc. Another challenge is integrating question and performer characteristics to honestly and objectively reflect question difficulty.

Therefore, this paper aims to address these challenges by using deep neural networks, integrating learners' responses, and combining absolute difficulty and relative difficulty of questions to achieve more accurate, efficient difficulty prediction.

2 Related Work

Regarding the problem of predicting the difficulty of questions, scholars within domestic and international contexts conduct a lot of research and exploration and put forward different methods and ideas. Wang[8] separated nine problem attributes from the questions by manual coding. The results showed that the Support Vector Machine (SVM) model based on question properties predicted more accuracy. Wu et al.[9] proposed a method for estimating the difficulty of reading comprehension questions based on the SVM， with classification accuracy rate and mean squared error as evaluation indicators. Ma[10] used SOLO's approach of defining learners' learning outcomes by five levels to predict the difficulty of questions from their content structure. With the continuous development of machine learning and deep learning, the linear regression method in machine learning is utilized to difficulty prediction model[11]. Tong[7] used the question text and answer record to realize the data-driven difficulty prediction model for mathematical questions. Song et al.[12] proposed a deep neural network model to establish the correlation between question text information and actual difficulty by combining question text information with examinee response records, so as to

solve problems such as the prediction of question item difficulty parameters. Xie et al.[13] have developed an Attention Deep Belief Neural Network (ADBN) framework that integrates an attention mechanism. This framework employs a neural network architecture derived from Deep Belief Networks (DBN) to facilitate the encoding of question semantics. Furthermore, the ADBN framework undergoes training to effectively estimate the difficulty level of novel items utilizing the attention mechanism.

3 The BDCL Model

In this section, we propose a BERT-based Difficulty Prediction Model with CNN and LSTM based on Response Data (BDCL model for short). In more details, we expand the question text to both the question and answer based on the characteristics of programming questions that combine natural language and code language. Secondly, we fuse the absolute difficulty of the question and the relative difficulty reflecting the learners' response. By doing so, we can capture the difficulty characteristics of the question from a more comprehensive perspective and achieve adaptive question difficulty prediction.

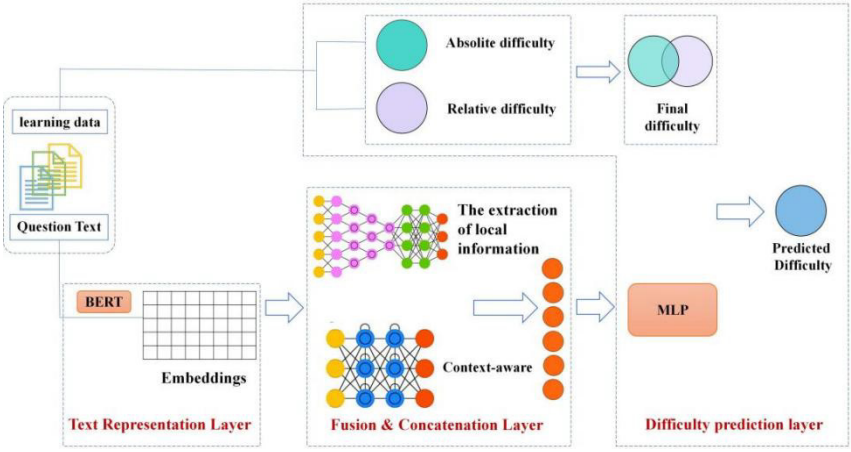

Fig. 1. The framework of BDCL model

The BDCL model is composed of a text representation layer, a fusion & concatenation layer, and a difficulty prediction layer (Fig. 1.). Firstly, the BERT pre-training model[14] is used in the text representation layer to extract the embedded vectors of question texts. Then, in the fusion & concatenation layer, local feature extraction layers based on CNN and LSTM, as well as context-aware layers are respectively introduced to enrich the feature information contained in the embedded vectors, then feature vectors are fused. In the difficulty prediction layer, the absolute difficulty and relative difficulty are combined to obtain the actual difficulty based on learner's response and difficulty labels of the questions. Finally, the difficulty of predicting the output problem is output by a fully connected neural network (MLP).

3.1 Text Representation Layer

The question text contains a wealth of information, including relevant knowledge, question requirements, sample output, standard answers, etc., from which a large number of associations and details can be extracted to evaluate the difficulty of the question. Therefore, integrating question texts can improve the accuracy and reliability of the model when predicting question difficulty[15].

Firstly, we utilize the BERT pre-trained model to extract the embedding vector of the question text, and the embedding vector containing the question information can be input into the hidden layer for further processing. The input to the model is defined as a collection of question texts:

$$\text{Question Set} = \{X_1, X_2, \ldots, X_{qn}\} \tag{1}$$

where qn denotes the number of questions, and X_n is the text content of a question. For simplicity, X_n is directly represented by X in this section. The input to the BERT model is a sequence X, X is an item containing n words, and after BERT processing, matrix $H \in \mathbb{R}^{n \times d}$ is generated, where each row represents the vector representation of the corresponding word.

To reduce the amount of training, our work directly uses the Chinese-pre-training model Chinese-BERT-wwm based on whole word mask (Whole Word Masking) technology released by HFL (Harbin Institute of Technology-iFLYTEK Joint Laboratory)[16, 17]. After processing by the L-layer transformer encoder, the final output vector sequence formula Eq.(2) can be obtained.

$$H^{(L)} = \{H_i^{(L)}\}_{i=1}^{n} = \text{BERT}(X) \tag{2}$$

Where, $H_i^{(L)}$ is the vector representation obtained at the i-th position after processing by the L-layer encoder. Finally, the sequence X can be represented a high-quality vector representation $\{H_i^{(L)}\}_{i=1}^{n}$ that can be used for the input features of downstream tasks.

3.2 The Fusion & Concatenation Layer

In the BDCL model, CNN extracts local features of the text sequence output by the BERT model. At the same time, RNN is used to context-aware the text sequence output by the BERT model further to enhance the model's understanding of the lengthy text. Finally, the output results of CNN and RNN are fused and concatenated to obtain the question text's local features and context-aware representation vector.

Local Feature Extraction

The BDCL model uses CNN to extract the local features of question text to enhance the expression ability of the model further. The core of CNN is convolution operation, ReLU activation function and batch normalization[18]. In our work, we consider the relationships between several adjacent words or characters within the text, ensuring that

the convolution kernel is set to encompass these relationships. Due to the need to concatenate with LSTM output, the parameters padding and stride, which are used to control the output size after the convolution operation, are set to 1 in this model, so that the size of the output after each convolution operation is unchanged. Specifically, assuming that the size of the convolution kernel is (K, H, C), where K represents the length of the convolution kernel, H represents the word vector dimension of BERT model output, and C represents the number of output channels, then the convolution operation can be represented as:

$$y_{i,j}^{c} = \sum_{r=0}^{K-1} X_{transposed,i,h_{j+r}} w_{h,c,r} \tag{3}$$

Where $X_{transposed,i,h_{j+r}}$ is the 3-dimensional tensor obtained from the text sequence output by the BERT model, $y_{i,j}^{c}$ denotes the output of the convolution kernel in the *i-th* sample, the *j-th* position, and the *c-th* channel, and $w_{h,c,r}$ represents the value of the convolution core in the *h-th* line, the *c-th* channel, and the *r-th* position.

The BDCL model adopts the standard ReLU function to learn and express the input information better. i.e., $\max(0, x)$, as expressed in the following equation:

$$\hat{y}_{i,j}^{c} = max\left(0, y_{i,j}^{c}\right) \tag{4}$$

Where $\hat{y}_{i,j}^{c}$ represents the output after ReLU activation.

Finally, to further enhance the expression ability of the model and suppress the overfitting phenomenon, the BDCL model adopt batch normalization technology to process the output obtained by convolution operation. Here, we simply define the output of the CNN layer for the next processing:

$$X = CNN\left(H^{(L)}\right) \tag{5}$$

Context Awareness

Our model utilizes RNN to contextualize the output text sequence of the BERT model, further enhancing the model's understanding of long text. Specifically, the BDCL model employs short-term and long-term memory networks (LSTM). LSTM is a variant of RNN with better long-term memory and forgetting capabilities compared to traditional RNN structures, effectively handling sequence data such as text.

The BDCL model improves its performance by utilizing bidirectional LSTM. During the forward propagation of the model, the word vector outputted by BERT is first transformed through an LSTM. Specifically, the word vector outputted by BERT is first transposed and processed through a layer of LSTM. Assuming that the dimension of the word vector output by BERT is (H,L), where H is the dimension of each word vector, and L is the sequence length, then the input of the LSTM layer is the tensor $sequence\_length \times batch\_size \times input\_size$, i.e. $L \times N \times H$, where N is $batch\_size$. In LSTM, each time step will output the hidden and cell states that are also the next step, i.e. (h_t, c_t), and then continue as input in the next step. Specifically, for the *t-th* time step, the LSTM is calculated as follows:

$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{6}$$

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{7}$$

$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{8}$$

$$c_t = f_t * c_{t-1} + i_t * \tilde{c}_t \tag{9}$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{10}$$

$$h_t = o_t * \tanh(c_t) \tag{11}$$

Here, x_t represents the input of LSTM at the *t-th* time step, h_t represents the output of LSTM at the *t-th* time step, c_t represents the cell state of LSTM at the *t-th* time step, and f_t , i_t and o_t represent the weights of the forget gate, input gate and output gate of LSTM at the *t-th* time step, respectively. W_f , W_i , W_c , W_o , U_f , U_i , U_c , U_o and, b_f , b_i , b_c , b_o denote the trainable weights and bias terms, respectively.

Finally, the output of the LSTM layer after transposition is simply defined as:

$$Y = \mathrm{LSTM}\left(H^{(L)}\right) \tag{12}$$

Fusion Splice

In the forward propagation of the model, after the output of BERT is processed by CNN and LSTM, and the outputs of CNN and RNN are transposed respectively, and the following two tensors are obtained (the two dimensions of the two are respectively (batch_size, seq_len, cnn_hidden_size) and (batch_size, seq_len, rnn_hidden_size)):

$$\begin{cases} X = [x_1, x_2, \cdots, x_n], x_i \in \mathbb{R}^{h_c} \\ Y = [y_1, y_2, \cdots, y_n], y_i \in \mathbb{R}^{h_r} \end{cases} \tag{13}$$

Where n represents the length of the sequence, and x_i and y_i represent the output vector of the CNN and RNN at the *i-th* position in the sequence.

Next, we stitch them together along the last dimension (i.e., the dimension cnn_hidden_size and rnn_hidden_size they are in) to get a new tensor:

$$Z = \mathrm{Cat}(X, Y), z_i \in \mathbb{R}^{(h_c + h_r)} \tag{14}$$

Where Cat represents the splicing operation ($Z \in \mathbb{R}^{(batch_{size}, seq_{len}, h_c + h_r)}$), z_i represents the output vector at the *i-th* position of the splicing result.

In this way, a new tensor is obtained, which connects the outputs of the CNN and LSTM for subsequent operation.

Fully Connection

After the processing of the fully connected layer, the final prediction result is expressed as follow:

$$y = \tanh(W \cdot \mathrm{Flatten}(Z) + b) \tag{15}$$

Where Flatten represents the leveling operation, flattening a multi-dimensional tensor into a one-dimensional vector. $W \in R^{1 \times (cnn\_hidden\_size + 2 \times rnn\_hidden\_size)}$ is the

weight matrix, $b \in R$ is the bias vector, represents the matrix multiplication, and tanh represents the nonlinear activation function.

Finally, the prediction results are scaled and shifted so that the prediction results are output in the target interval. Specifically, the prediction y is multiplied by the extreme difference α and the mean β is added to obtain the final output $P = y \times \alpha + \beta$.

3.3 Difficulty Prediction Layer

The difficulty prediction layer is the output of the BDCL model. When predicting the difficulty of the question, it only needs to input the text of the question to be predicted into the model, and load the trained parameters and weights. The output value obtained by the model is the difficulty of the predicted question. In the questions in the field of programming, if the collected question text and the amount of answering data are sufficient and the length of the question text is appropriate or appropriate after deletion, the value of the difficulty prediction layer can be considered to represent the difficulty value of the question.

Absolute Difficulty

In this section, we combine the question score and use this formula to expand the range of absolute difficulty:

$$Da_i = D_i\left(1 + \log\left(\frac{S_i \div D_i}{avg(\sum_1^i (S_j \div D_j))}\right)\right) \tag{16}$$

In the Eq.(16), S_i represents the expert-set score and D_i represents the original difficulty provided in the datasets. This formula can evaluate the ratio of score to difficulty on an average level, and using logarithms can ensure that the absolute difference in difficulty will not cause distortion. For example, if the difficulty of question A is 3 and its score is 1000, the difficulty of question B is also 3 and its score is 500, and the difficulty of question C is 1 and its score is 100. Assuming that the average value of the ratio of scores to difficulties for all questions is 300, then the final absolute difficulty of question A is 3.137, the final absolute difficulty of question B is 2.846, and the final absolute difficulty of question C is 0.523.

Relative Difficulty

Most literature references related to the field of difficulty prediction for examination questions currently only use the score rate[3, 5, 6] simply. The existing literature on difficulty prediction for examination questions does not effectively utilize effective attributes such as the number of attempts and time spent on the question. Considering the characteristics of programming learning, this paper introduces the number of attempts and time spent on the question to define relative difficulty based on the score rate. The relative difficulty is calculated with reference to the average score rate, average number of attempts, and average time spent on the question. The main factor in determining relative difficulty is the score rate, with weights of 0.2 and 0.1 assigned to the number

of attempts and time spent on the question, respectively. The final relative difficulty formula is as follow:

$$\mathrm{Dr_i} = \mathrm{R_i}(1 + \tilde{\mathrm{t}})(1 + \tilde{\alpha}) \tag{17}$$

Where $\mathrm{R_i}$ is the score rate, $\tilde{t}$ denotes the answer time and $\tilde{\alpha}$ denotes the number of submissions.

Actual Difficulty Calculation

Consider the absolute difficulty and relative difficulty of the question as a representation of the actual difficulty, the specific formula is as follows:

$$\mathrm{Df_i} = \mathrm{Da_i} + \mathrm{Dr_i} \tag{18}$$

The BDCL model can use the question difficulty as the label, and the mean variance loss function (MSE) as the loss function:

$$\mathrm{loss_{MSE}} = \frac{1}{\mathrm{n}}\sum_{\mathrm{i=1}}^{\mathrm{n}}(\mathrm{p_i} - \mathrm{Df_i})^2 \tag{19}$$

Where n represents the number of sample data, $\mathrm{p_i}$ and $\mathrm{Df_i}$ represent the predicted and true values for each sample, respectively.

4 The Experiment

4.1 Experimental Setup

The deep neural network method used in this article is to predict the difficulty of questions by combining pre trained BERT models, CNN, and LSTM. The input is the question text, and the output is the prediction difficulty. In the BDCL model, MSELoss is selected as the loss function to quantify the difference between the predicted and true values of the model; At the same time, AdamW is used as the optimizer and a learning rate decay strategy is adopted. Specifically, an initial learning rate of 3e-5 is selected, according to Google's recommended learning rate [19], and the learning rate is reduced by 0.1 after every 10 epochs. In terms of datasets partitioning, this article chooses to divide the datasets into training and validation sets in a ratio of 72%:18%:10%. This ratio can effectively balance training accuracy and validation accuracy through experiments.

4.2 Datasets

This paper uses the private datasets of online programming platform, covering programming questions in different languages, such as C, C++, python, java, SQL, etc. Students can program online, submit the answers and obtain the compiler output in real time for correction. In this paper, the student online code evaluation records for 2018 and 2019 are collected. The interaction records, which include the student ID, question ID, final score, number of submissions, start time of the answer, and end time of the

answer, can be utilized to assess the relative difficulty. And the information of the questions, including the question text, answer, score, and difficulty, can be extracted and utilized to assess the absolute difficulty of the questions.

In this paper, from the 4416 questions of the original datasets, the questions with more than 100 answers are selected for model training and evaluation. A total of 2283 valid questions are screened out, and 2485432 answering records are recorded (Table 1).

Table 1. Number of original answer records

	original datasets	processed datasets
Number of questions	4416	2283
Number of students	46743	46710
Number of knowledge points	51	51
Number of answers recorded	2532524	2485432

4.3 Processing of Question Text

After statistics, the average character length of the original text is 1865, and the median is 1600. This article processes the content of the text, sets up a stop word table, removes all duplicate text in the questions, and although the BERT model can accept punctuation input and understand its semantics, longer text length is obviously more significant than punctuation marks. After sorting from small to large, the results shown in the Fig. 2. are obtained.

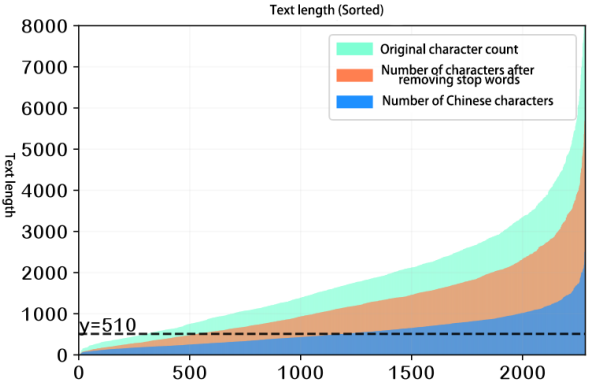

Fig. 2. The text length comparison after sorting

After text processing, the average character length is 1296 with a median of 1121. Because the longer text back segments are mostly code examples, mainly in English, which can be entered according to the whole word Token. Although the text length of Chinese characters has been effectively reduced, according to the actual verification, the effect is not as good as the text due to the loss of some important key words. Accordingly, it is believed that the text processing after removing the stopped words has achieved the effect and can be used for model input.

4.4 Comparative Analysis of the Experimental Results

In this paper, we employ three indicators, such as pearson correlation coefficient, spearman's rank correlation coefficient and determination coefficient. Our model is run on the RTX2080Ti server for 6 hours for 300 epochs. Fig. 3. shows the performance of the BDCL model during 300 epochs. To more intuitively observe the trend of the performance curve, we utilize the Savitzky-Golay smoothing method to eliminate noise influence (Fig. 4).

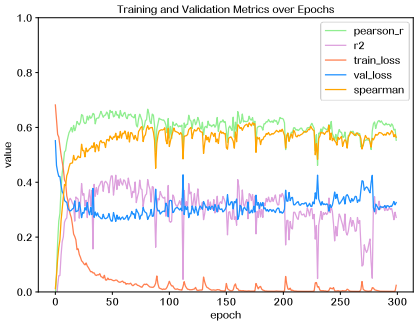

Fig. 3. Performance of the training round model

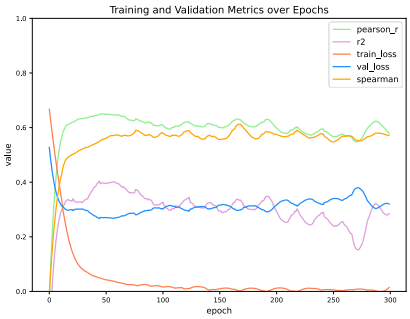

Fig. 4. Performance after smoothing

Next, we compare the BDCL model with a simplified version of the BDCL model - BERT and BERTCNN. Each model is trained for 60 epochs and their performance is shown in Fig. 5.:

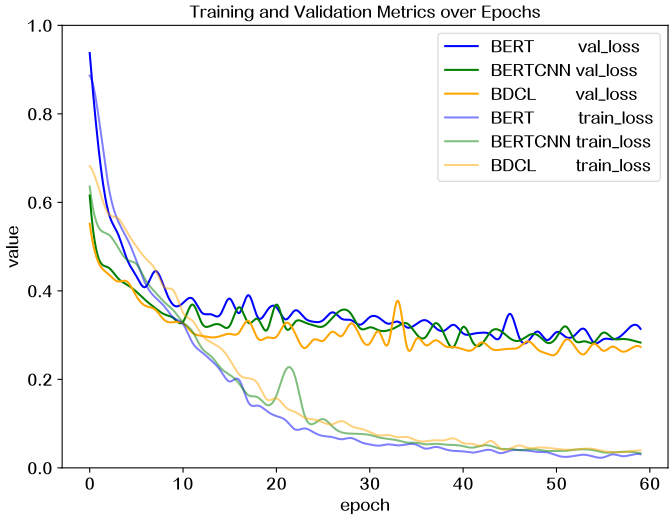

Fig. 5. Performance comparison of BERT, BERTCNN, and BDCL models

As can be seen from the figure, the convergence rate of BDCL model is significantly slower than the original BERT and BERTCNN models in terms of training set loss. However, in terms of validation set loss, the performance of BDCL model is better than

BERT model from the beginning of training. After all three models converge, the effect of BDCL is better than the other two. After 60 epochs, the specific parameters of each model are shown in Table 2:

Table 2. Comparison of the performance parameters of the three models

	train_loss	val_loss	pearson_r	r^2	spearman
BERT	0.029636	0.293649	0.619125	0.344172	0.528640
BERTCNN	0.041917	0.266075	0.656242	0.405755	0.575367
BCDL	0.043485	0.257141	0.658452	0.425707	0.578868

The experimental results of the three deep neural network models are significantly better than the first two traditional support vector machine and machine learning models, indicating that the deep neural network has a stronger ability to model this task.

According to the Pearson and Spearman correlation coefficient, it can be analyzed that BDCL performs better than the other two models in both linear and nonlinear regression. In addition, the determination coefficient of BDCL is also relatively high, indicating that the BDCL model can better explain the changes in the data, that is, with stronger generalization ability. The reason is that the BDCL model adds CNN and LSTM layers to the BERT model to capture the details and context sequence relations in the question text, because the number of parameters is larger, so the convergence rate is slower, and because it captures the hidden information in the text.

Accordingly, it can be considered that the BDCL model has stronger generalization ability and accuracy than the previous difficulty prediction model, which can be used for question difficulty prediction.

5 Conclusions

This paper proposes the BDCL question difficulty prediction model, which achieves better results than previous statistical based and traditional machine learning methods in online programming datasets, and can be used to support questions generation, automatic set papers and personalized recommendation of questions. The main work is as follows:

1. We propose a BERT-based Difficulty Prediction Model with CNN and LSTM based on Response Data (BDCL).

2. By the analysis of the program questions, we balance the weight and integrate the absolute difficulty and relative difficulty.

3. Comparison on the programming datasets, Pearson's correlation coefficient, Spearman's correlation coefficient and judgment coefficient are improved by more than 4% compared with BERT model, while Pearson's correlation coefficient, Spearman's correlation coefficient and determination coefficient increased by 7% - 10% compared with TD-IDF, basically meeting the prediction requirements.

In the future work, we will combine the difficulty prediction of questions with intelligent cognitive diagnosis, and accurately recommend questions according to the weak knowledge points of students, so as to truly realize teaching students in accordance with their aptitude.

Acknowledgment

This research was partially supported by National Natural Science Foundation of China(No.62207031).

References

1. (2018) Notice of the Ministry of Education on the Issuance of action Plan of Education Informatization 2.0 [EB/OL]. http://www.moe.gov.cn/srcsite/A16/s3342/201804/t20180425_334188.html.
2. (2019) The CPC Central Committee and The State Council issued China's Education Modernization 2035 [EB/OL]. https://www.gov.cn/zhengce/2019-02/23/content_5367987.htm.
3. Dongdai Zhou, Xiaoxiao Dong, et al. (2022) Research on the construction of automatic difficulty prediction model with multiple influencing factors. J. MODERN DISTANCE EDUCATION, 2022(04):32-41. https://doi.org/10.13927/j.cnki.yuan.20220706.006.
4. Tianyu Zhu, Zhenya Huang, Enhong Chen, et al. (2017) A personalized questions based on cognitive diagnosis. J. Chinese Journal of Computers, 40(01):176-191. https://kns.cnki.net/kcms/detail/11.1826.tp.20160510.1458.006.html
5. Yuni Susanti, Takenobu Tokunaga, et al. (2017) Controlling item difficulty for automatic vocabulary question generation J. Research and practice in technology enhanced learning, vol. 12(1): 1–16. https://doi.org/10.1186/s41039-017-0065-5
6. Jia Xu, Tingting Wei, Ge Yu. (2022) Review of topic difficulty assessment methods[J]. Journal of Frontiers of Computer Science and Technology, 16(04): 734-759. https://kns.cnki.net/kcms/detail/11.5602.TP.20211130.1040.002.html
7. Wei Tong, Fei Wang, Qi Liu, et al. (2019) Data-driven difficulty prediction of mathematical questions[J]. Journal of Computer Research and Development, 56(5): 13. https://kns.cnki.net/kcms2/article/abstract?v=Y_ITemen1J7YV3bjELrYhiO4R4PeKLZ0_jLzqGOYJ0x7lQjIfYiVxYjGU_zzNsq3N0h3mBjZtlmh7wuAU-lBeI9vJjhCYW5Nqy6cZiHWW8SgFot0BXf6_UBenqxlromOSembI_W18dUlHRMXxXwnPQ==&uniplatform=NZKPT&language=CHS
8. Qingmei Wang. (2018) Research on the difficulty prediction of questions based on subject properties and text D. Jiangxi University of Finance and Economics, https://kns.cnki.net/kcms2/article/abstract?v=Y_ITemen1J5zFszoLhuSq7Y2n0z3rv6dIPTLYDJqK6OfDbu--89lRZDTRjjssOaYkoojhb1XfjkzOv97gTtXIgsRUiMQLFlhujGEwyXY5BeHiH_FJUveXqhgc71Xbm9moYRXivIkIj3BKU6xDg1tWA==&uniplatform=NZKPT&language=CHS
9. Shenglei Wu, Jie Ren. (2022) Study on the difficulty of reading comprehension questions based on support vector machine J. Examination Research, 18(5): 68-77. https://kns.cnki.net/kcms2/article/abstract?v=Y_ITemen1J5UCYkLP_O_5VxCWZOsiBA9hJ4qv3CY38KGd3twHHlWWtNyIi3bNMqQK54dKuzpvLdBPTvKfrncF9VrIsKgojzj-mEKVwQm7r3nsl9Rwg81viVqogyKphvFDVIlAeJCoko8=&uniplatform=NZKPT&language=CHS
10. Li Ma. (2021) Design of question difficulty based on SOLO classification theory J. Foreign Language Teaching in Primary and secondary schools (Middle school section), 44(04): 33-37. https://kns.cnki.net/kcms2/article/abstract?v=Y_ITemen1J5-9vwp-

kv0OdlGb0aHBYOQBIqQ_723WUki9ZQvhEp_G285UGH-VaGIfNyJ5_UqOUTS6jIUBLjmeHxncFMwEnaXLwYc2ANTgey9OJBUy4SBRgU-6b5Dq78U8tGm1QjtCw4Q=&uniplatform=NZKPT&language=CHS
11. Fang Liu, Weiqun Wang, Xing Wu. (2022) Construction of Item Difficulty Model of Senior High School Academic Achievement Test J. Chemistry Education (In Both Chinese and English), 43(21): 43-47. https://doi.org/10.13884/j.1003-3807hxjy.2022020113
12. Huiyuan Song, Xingjian Xu, Fanjun Meng. (2022) Research on the difficulty prediction of questions based on topic correlation knowledge J. Journal of Inner Mongolia Normal University (Chinese edition of Natural Sciences), 51(03):305-311. https://kns.cnki.net/kcms2/article/abstract?v=Y_ITemen1J6K8Zvr6UEc-UvwCxFX2pov-OdMek2KzEK5Q0JzklFXnFIACPPJCLnhpisj4dpmgkRevRMr113eqS5sI8AtUNC3d-23ue022h-Y-3d60e7ME5sXGW-LLDw-kAS3frzSede4=&uniplatform=NZKPT&language=CHS
13. Ying Xie, Rongbin Xu. (2018) Group volume difficulty prediction based on a deep confidence neural network J. Journal of Shaoguan University, 39(9):5. https://kns.cnki.net/kcms2/article/abstract?v=Y_ITemen1J7vHH1EwNgQaGSaWMh-jOf0zY524TbgkENhJAX5ocBjgl0Ay0skwxZdOmklO5usvaX4U9WXj4fhgrZuLcQa2LK-MJIeS9Ppq0vAYcwWWLZpwiC_HxAqPte3tH7bWjKutD6A=&uniplatform=NZKPT&language=CHS
14. J. Devlin, M. Chang, K. Lee, and K. Toutanova. (2018) BERT: Pre-training of deep bidirectional transformers for language understanding. ArXiv preprint. https://doi.org/10.18653/v1/N19-1423
15. He Jun, Peng Li, Sun Boa, et al. (2021) Automatically Predict Question Difficulty for Reading Comprehension Exercises In: 2021 IEEE 33rd International Conference on Tools with Artificial Intelligence (ICTAI), Washington, DC, USA, 2021:1398-1402. https://doi.org/10.1109/ICTAI52525.2021.00222
16. Yiming Cui, Wanxiang Che, Ting, Liu Bing Qin, Ziqing Yang. (2019) Pre-Training with Whole Word Masking for Chinese BERT J. ArXiv abs/1906.08101 https://doi.org/10.1109/TASLP.2021.3124365
17. ymcui.(2023) Chinese-BERT-wwm[EB/OL]. https://github.com/ymcui/Chinese-BERT-wwm.
18. Hongxue Xu, Anqi Wang, Yingkui Du, Wanyou Sun. (2019) Basic model of Deep Learning and its application J. Journal of Changchun Normal University, 39(12):47-54+93. https://kns.cnki.net/kcms2/article/abstract?v=Y_ITemen1J54NESZhzxNiR4dsj9sriB9PXpXxhXeBlv_U50Um-LXtdD0m54dnB2KjSL44_I8NdQYuB6cYTOgiLrb0V4cwd42Ud-cRBfMoQWo1hdMOA93WEnpBqJ_8JGiICcxa4DcVvw2w=&uniplatform=NZKPT&language=CHS
19. google-research. (2020) BERT[EB/OL]. https://github.com/google-research/BERT.