Difficulty-Focused Contrastive Learning for Knowledge Tracing with a Large Language Model-Based Difficulty Prediction

Unggi Lee[1], Sungjun Yoon[2], Joon Seo Yun[2], Kyoungsoo Park[2], YoungHoon Jung[2]
Damji Stratton[3] and Hyeoncheol Kim[4†]
Enuma, Inc.[1], i-Scream Edu[2], The University of Missouri System[3], Korea University[4]
unggi@enuma.com, {yseong555, jsyun0610, kspark0818, yhjung}@i-screamedu.co.kr
dhsdfn@umsystem.edu, harrykim@korea.ac.kr

Abstract

This paper presents novel techniques for enhancing the performance of knowledge tracing (KT) models by focusing on the crucial factor of question and concept difficulty level. Despite the acknowledged significance of difficulty, previous KT research has yet to exploit its potential for model optimization and has struggled to predict difficulty from unseen data. To address these problems, we propose a difficulty-centered contrastive learning method for KT models and a Large Language Model (LLM)-based framework for difficulty prediction. These innovative methods seek to improve the performance of KT models and provide accurate difficulty estimates for unseen data. Our ablation study demonstrates the efficacy of these techniques by demonstrating enhanced KT model performance. Nonetheless, the complex relationship between language and difficulty merits further investigation.

Keywords: Knowledge tracing, large language model, contrastive learning

1. Introduction

Knowledge tracing (KT) is a field of research that aims to predict student learning progress by analyzing their past interactions with question items within an educational context (Abdelrahman et al., 2023; Corbett and Anderson, 1994). Difficulty estimation plays a crucial role in understanding dynamic student learning progress (Minn et al., 2018). Accordingly, developing embeddings adapting item response theory (IRT) models such as the Rasch model has been used to calculate item difficulty (Ghosh et al., 2020). Other studies adapted classical test theory (CTT) to estimate difficulty (Lee et al., 2022b).

In addition, contrastive learning has emerged as an effective framework in various research areas, including computer vision, representation learning, as well as KT (Chen et al., 2020a; Wang and Isola, 2020; Lee et al., 2022b). Contrastive learning learns representations by comparing positive and negative samples (Wang and Isola, 2020; Le-Khac et al., 2020). While there is some previous research on contrastive learning applied to KT, few studies have focused on incorporating the difficulty information to improve model performance.

Moreover, the textual features of questions in educational contexts contain valuable information about the required skills, question difficulty, and student interaction with questions. According to Abdelrahman, Wang, and Nunes (2023), Deep learning KT models have utilized these textual characteristics to acquire an understanding of question patterns and monitor the levels of knowledge in students. However, the potential role of natural language in KT is not yet fully understood.

The current study aims to address these gaps by proposing a new model, called Difficulty-Focused Contrastive Learning for Knowledge Tracing with a Large Language Model (DCL4KT+ LLM). The model utilizes CTT to calculate concept difficulty and question difficulty and incorporates the contrastive learning framework to enhance the performance of the model. Furthermore, it leverages the textual features of questions to improve the accuracy of knowledge tracing. The architecture of the proposed model consists of embedding layers, encoder blocks based on the MonaCoBERT model, and a contrastive learning framework. In this study, we tested DCL4KT + LLM using the benchmark datasets and compared its performance using AUC and RMSE. Additionally, an ablation study was conducted to examine the effect of difficulty-focused contrastive learning and difficulty prediction using LLM, and the effect of the data augmentation.

2. Background

2.1. Difficulty in Knowledge Tracing

The difficulty has a significant impact on student learning practices.(Minn et al., 2018). Previous research in education has explored methods to calculate difficulty in questions or concepts. CTT and IRT are popular methods to calculate difficulty. IRT depicts the relationship between an individual's response to an item and their level on the scale's underlying construct. (Edelen and Reeve, 2007). Attentive knowledge tracing (AKT) (Ghosh et al., 2020) used the Rasch embedding strategy to represent difficulties of question and concept, inspired by the Rasch model of item response theory. Other studies have adapted CTT as it

LREC-COLING 2024, pages 4891–4900

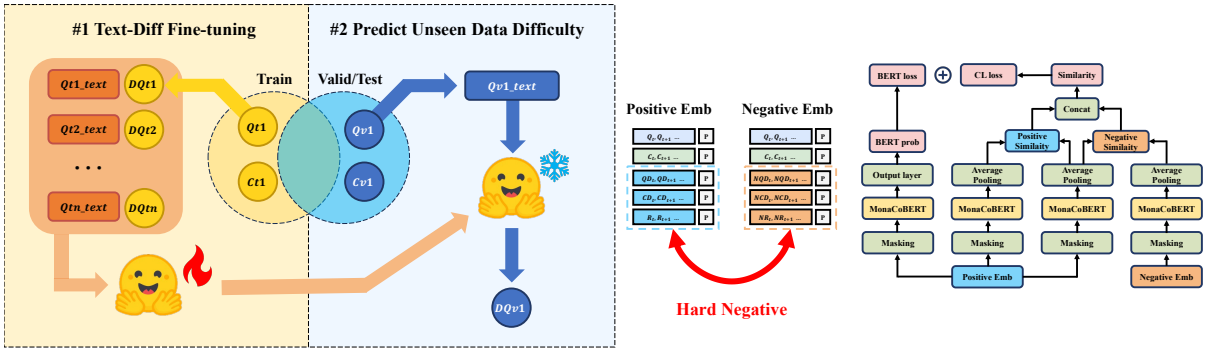

Figure 1: Architectures of DCL4KT+LLM. *Left*: LLM-based difficulty prediction framework in KT. *Right*: Whole architecture of DCL4KT+LLM

is much simpler and thus easier to interpret the results. (Petrillo et al., 2015). In fact, various KT models use CTT to calculate difficulty; bidirectional encoder representation of knowledge tracing (BEKT) (Tiana et al., 2021), monotonic attention-based ConvBERT for knowledge tracing (MonaCoBERT) (Lee et al., 2022a), and contrastive learning for knowledge tracing (CL4KT) model (Lee et al., 2022b). Meanwhile, the Graph Neural Network (GNN)-based model in KT used a representation of difficulty by using the relationship of questions (concepts) and students' responses (Song et al., 2022; Luo et al., 2022).

In this research, we used CTT to calculate difficulty and included a concept called 'hard negative difficulty' to empower the performance of our model.

2.2. Contrastive Learning in Knowledge Tracing

Recently, contrastive learning-based models have achieved better performance in a lot of research areas, such as computer vision, natural language processing, and recommendation systems. Contrastive learning is a method that learns the representation by comparing the positive samples with the negative samples (Wang and Isola, 2020; Le-Khac et al., 2020). Momentum Contrast for Unsupervised Visual Representation Learning (MOCO) (He et al., 2020) proposed a dynamic dictionary using a queue and moving-averaged encoder, which improved the performance in unsupervised visual representation tasks. SimCLR (Chen et al., 2020a,b) achieved better performance by using data augmentation for contrastive learning. Yet, there have been few KT studies that have utilized the contrastive learning framework. CL4KT (Lee et al., 2022b) employed contrastive learning in KT by using reversed answer data as negative samples and suggested several data augmentation techniques. In addition, there have been attempts to combine contrastive learning and GNN (Song et al., 2022; Wu and Ling, 2023; Dai et al., 2022). Nonetheless, there are only a few contrastive learning-based KT studies that have explored the role of difficulty in enhancing model performance.

2.3. Knowledge Tracing with Natural Language Dataset

The text of a question can contain a great wealth of information, such as the skills required by the question, the difficulty of the question, and the relationships between questions. Several deep learning KT models have leveraged the textual features of question texts to learn question representations and track students' knowledge states (Abdelrahman et al., 2023).

Relation-aware self-attention for knowledge tracing (RKT) and hierarchical graph knowledge tracing (HGKT) also extract features from the textual information of questions to learn question representations in their models. Exercise-enhanced recurrent neural network (EERNN) and exercise-aware knowledge tracing (EKT) extended from EERNN proposed a framework that considers both exercising records and the texts of exercises for predicting student performance (Su et al., 2018; Liu et al., 2019). Adaptable knowledge tracing (AdaptKT), which transfers knowledge from the source domain to the target one, and In Exercise Hierarchical Feature Enhanced Knowledge Tracing utilize Bert has been proposed (Cheng et al., 2022; Tong et al., 2020). QuesNet is also an unsupervised learning method that leverages a large corpus of unlabelled questions (Yin et al., 2019).

Yet, previous research has not considered much about the latent representation of the textual features in the questions and concepts. Text-aware KT models are motivated by leveraging the textual features of questions and concepts to enhance performance in tackling KT tasks.

3. Methodology

3.1. Problem Statement

By analyzing the sequence of interaction data collected from a learning management system (LMS)

or intelligent tutoring system (ITS), KT attempts to predict the likelihood of a student answering accurately. Student interactions can be represented as $x_1, ..., x_t$. Each interaction in KT consists of three components: the query id, the related educational concept, and the student's response. $x_t = (c_t, q_t, r_t)$ describes the t-th interaction. c_t represents the educational concept associated with the t-th inquiry in this equation. The q_t variable represents the question's identifier. r_t represents the student's response to the t-th query, where r_t in $0, 1$, where 0 denotes an incorrect response and 1 denotes a correct response. Difficulties can be divided into two; concept difficulties cd_t and question difficulties qd_t. The difficulty was set to an integer value ranging from 0 to 100. Based on classical test theory (CTT), The formula for calculating difficulty is the number of students who got the question (concept) correct divided by the total number of questions (concepts).

3.2. Proposed Model Architecture

3.2.1. Embedding Layers

DCL4KT uses a positive embedding layer block and a negative embedding layer. The positive embedding layer $E_{positive}$ consists of element-wise embedding layers; questions E_q, concepts E_c, question difficulties E_{qd}, concept difficulties E_{cd} and students' response E_r. In addition, position embedding E_p also contained the positive embedding block. The formulation of the positive embedding layer is below.

$$E_{positive} = E_q + E_c + E_{qd} + E_{cd} + E_r + E_p \quad (1)$$

The negative embedding layer $E_{negative}$ consisted of element-wise embedding layers; questions E_q, concepts E_c, hard negative question difficulties E_{nqd}, hard negative concept difficulties E_{ncd} and hard negative students' response E_{nr}. The formulation of the negative embedding layer is

$$E_{negative} = E_q + E_c + E_{nqd} + E_{ncd} + E_{nr} + E_p \quad (2)$$

The position embedding E_p also contained the negative embedding block. The details of the negative embedding are shown in section 3.3.2.

3.2.2. Encoder Architecture

In this research, we used MonaCoBERT (Lee et al., 2022a) as an encoder block. MonaCoBERT is a transformer-based model which changes the attention module by combining with span-based dynamic convolution (SDC) and monotonic attention (MA), which model can represent students' response sequence locally and globally while representing the students' forgetness. DCL4KT uses four MonaCoBERT encoder modules where each module used four transformer layers Tr. Three encoder modules are used for the contrastive learning framework. One encoder calculates binary cross entropy (BCE) loss, and three encoders calculate contrastive learning loss.

3.3. Contrastive Learning Framework

3.3.1. Loss Function

The loss function of DCL4KT is calculated by summing the BCE loss $\mathcal{L}_{bce}$ and contrastive loss $\mathcal{L}_{cl}$ (Lee et al., 2022b). The ratio between BCE loss and contrastive loss is controlled by the hyperparameter λ_c which ranges from $[0, 1]$. The whole loss function is formulated as

$$\mathcal{L} = (1 - \lambda_c) \times \mathcal{L}_{bce} + \lambda_c \times \mathcal{L}_{cl}, \quad (3)$$

BCE loss $\mathcal{L}_{bce}$ is a binary cross entropy loss between prediction $\hat{r}_t$ and real students' response r_t, defined as

$$\mathcal{L}_{bce} = \sum_t -\left(r_t \log \hat{r}_t + (1 - r_t) \log\left(1 - \hat{r}_t\right)\right) \quad (4)$$

Contrastive loss $\mathcal{L}_{cl}$ is a concatenate of concept similarity sim_c and question similarity sim_q,

$$\mathcal{L}_{cl} = concat(sim_c, sim_q) \quad (5)$$

When positive concept pair c^+_{t1}, c^+_{t2} is passed through the encoder layer tr, the result is $cz^+_{t1}, cz^+_{t2} = tr(c^+_{t1}, c^+_{t2})$. When positive and negative concept pair c^+_{t1}, c^-_{t2} is passed through the encoder layer tr, the result is $cz^+_{t1}, cz^-_{t2} = tr(c^+_{t1}, c^-_{t2})$. Thus, the concept similarity is defined as

$$sim_c = -\log \frac{e^{\text{sim}\left(cz^+_{t1}, cz^+_{t2}\right)}}{e^{\text{sim}\left(cz^+_{t1}, cz^+_{t2}\right)} + \sum \text{sim}\left(cz^+_{t1}, cz^-_{t2}\right)} \quad (6)$$

When positive question pair q^+_{t1}, q^+_{t2} is passed through the encoder layer tr, the result is $qz^+_{t1}, qz^+_{t2} = tr(q^+_{t1}, q^+_{t2})$. When positive and negative concept pair q^+_{t1}, q^-_{t2} is passed through the encoder layer tr, the result is $qz^+_{t1}, qz^-_{t2} = tr(q^+_{t1}, q^-_{t2})$, such that

$$sim_q = -\log \frac{e^{\text{sim}\left(qz^+_{t1}, qz^+_{t2}\right)}}{e^{\text{sim}\left(qz^+_{t1}, qz^+_{t2}\right)} + \sum \text{sim}\left(qz^+_{t1}, qz^-_{t2}\right)} \quad (7)$$

3.3.2. Embedding with Hard Negative

The novel implementation of the contrastive learning framework in CL4KT (Lee et al., 2022b) included negative embedding for student responses. Our research expands hard negative embedding to question and concept difficulty. The positive and

negative embeddings are depicted on the right Figure 1.
The positive embedding is a composition of question components E_q, concepts E_c, question difficulty E_{qd}, conceptual difficulty E_{cd}, and student responses E_r. Negative embedding, on the other hand, integrates element-wise combinations of question components E_q, concepts E_c, hard negative of question E_{nqd}, concept difficulty E_{ncd}, and hard negative student responses E_{nr}.
To provide additional clarity, the hard negatives are derived in a particular manner. For example, if a student's answer is correct, i.e. 1, the corresponding hard negative becomes 0. In contrast, if a student's response is incorrect i.e. 0, the hard negative is marked as 1. Similarly, concerning difficulty, if the difficulty of a query or a concept is rated at 0.75, then its hard negative equivalent would be 0.25. If the difficulty rating is 0.25, the corresponding negative value is 0.75. This can be summarized as

$$E_{nqd} = 1 - E_{qd}, E_{ncd} = 1 - E_{cd}, E_{nr} = 1 - E_r, \quad (8)$$

where $E_{qd}, E_{cd}, E_r, E_{nqd}, E_{ncd}, E_{nr}$ is $[0, 1]$.

3.4. LLM-based Difficulty Prediction Framework

In KT, when we calculate difficulty from questions and concepts, it is not possible to calculate the difficulty of the dataset, which is contained in the validation and test dataset but not in the training dataset, due to data splitting. Previous KT Model with difficulty used human-selected hyper-parameters (Lee et al., 2022a) or used representations of Question and Concept (Ghosh et al., 2020; Lee et al., 2022b). However, this approach is not stable in realistic educational Intelligent Tutoring Systems (ITS) or online learning platforms, which consistently add new questions or concepts to the e-learning system. This need led to the creation of a new approach to predict the difficulty of unseen questions or concepts which are not contained in the training dataset.
We present an LLM-based difficulty prediction framework to calculate difficulty, which is contained in the validation and test datasets but not in the training dataset, using the text of questions and concepts. The left side of Figure 1 shows the LLM-based difficulty prediction framework.
We define the notation to formulate the LLM-based difficulty prediction framework.

- D: a dataset of students' responses in knowledge tracing.
- D_{train}, D_{valid}, D_{test}: subsets of D representing the training, validation, and test sets, respectively.
- $d(q, c)$: difficulty score for a question q and a concept c.
- B_{pr}: a pre-trained BERT model.
- B_{ft}: a fine-tuned BERT model.

First, the dataset D consists

$$D = (q_i, c_i, r_i)_{i=1}^{N}, \quad (9)$$

where q_i is the i-th question, c_i is the ith concept, and r_i is the i-th response. And we split the dataset into training, validation, and testing sets as

$$D_{train}, D_{valid}, D_{test} = Split(D, ratio) \quad (10)$$

where $Split$ is a function that divides the dataset based on a specific ratio.
Then, we calculate the difficulty $CalDiff$ scores from the training set, not the validation and test set as

$$d(q_i, c_i) = CalDiff(D_{train}) \quad (11)$$

Using the $d(q_i, c_i)$, we execute fine-tune, FT, the pre-trained BERT model, which is trained by text corpus. In this research, we used pre-trained KoBERT [1] downloaded from Huggingface [2], because our dataset contained Korean text, not English. The formulation is below.

$$B_{ft} = FT(B_{pr}, d(q_i, c_i)), \quad (12)$$

where FT is a function that updates the model parameters using the training dataset and the calculated difficulties.
The fine-tuned BERT model is used to predict the difficulties of questions and concepts in the validation/test sets,

$$\hat{d}(q_j, c_j) = B_{ft}(q_j, c_j) \quad (13)$$

$$\forall (q_j, c_j) \in D_{valid} \cup D_{test}, \quad \forall (q_j, c_j) \notin D_{train} \quad (14)$$

where $\hat{d}(q, c)$ is the predicted difficulty of questions and concepts in the validation/test sets which are not contained in the training sets.

3.4.1. Data Augmentation

Referencing the previous research in KT and NLP, we developed and applied eleven data augmentation strategies for DCL4KT. To control the probability of application, we set the probability hyper-parameter to each augmentation strategy. If the hyper-parameter γ_{crop} is 0.2, then the probability of application crop is 20%. All of the data augmentation strategies are applied to the training session, not the validating or testing session.

[1] https://huggingface.co/beomi/kobert
[2] https://huggingface.co/

- **Token cutoff, span cutoff (Shen et al., 2020)**: The token cutoff is a simple augmentation technique that removes random data portions from an input sentence to produce limited perspectives. As a variant of the cutoff procedure, span cutoff removes a continuous segment of text.

- **Concept and question mask (Lee et al., 2022b)**: This method masks the concept or question randomly. The probability is the same as the original BERT. Note that the MonaCoBERT encoder already uses the students' correctness mask.

- **Crop (Lee et al., 2022b)**: A method which crops the parts of the sequence.

- **Summarize**: Maintains the order of the sequence and extracts some elements in the sequence.

- **Reverse**: Reverses the order of elements in the sequence.

- **permute (Lee et al., 2022b; Yang et al., 2019)**: Permutes the order of elements in the sequence randomly.

- **Segment permute**: Makes segments, then permutes those segments.

- **Replace higher and lower difficulty (Lee et al., 2022b)**: Replaces questions or concepts up to the difficulty.

- **Concatenate sequence**: Concatenates two sequences to make new sequences.

3.5. Experiment Setting

3.5.1. Datasets

- **ASSISTment09**: The ASSISTment datasets were collected from the ASSISTment intelligent tutoring system (ITS), predominantly from middle schools in the U.S., with participants randomly assigned (Heffernan and Heffernan, 2014). We used ASSISTments09 and ignored ASSISTments15 which does not contain question information[3].

- **Algebra05, 06**: The algebra datasets, provided by the KDD Cup 2010 Educational Data Mining Challenge, were collected from Cognitive Tutor. This ITS, developed by Carnegie Learning, focuses on middle school students (Ritter et al., 2007)[4].

- **EdNet**: Ednet dataset is provided by an edtech company in South Korea named Santa with a primary focus on the English test TOEIC presented by ETS. It consists of a total of 131,441,538 interactions, accumulated from 784,309 students since 2017, primarily targeting adult learners who need to certify their English competency (Choi et al., 2020)[5]. We extracted 5,000 interaction data from the original dataset.

- **Homerun20**: The open-source data in KT were not contained full of text about questions and concepts. Because of that, we used the homerun20 dataset, which is not published. This dataset is owned by i-Scream Edu which is an edTech company in South Korea. We used 351,425 responses from 201 elementary school students who used i-Scream Homerun[6] math education service in 2020. We erased all personal identity information (PII) before using this dataset. We used de-identified user id, questions and concepts id, the text of questions and concepts, and timestamp.

3.5.2. Evaluation Metrics and Validation

We employed AUC and RMSE as performance metrics. In addition, we utilized a five-fold cross-validation in our evaluation.

3.5.3. Baseline Models

We compared DCL4KT and DCL4KT-A to the baseline models, such as DKT (Piech et al., 2015), DKVMN (Zhang et al., 2017), SAKT (Pandey and Karypis, 2019), and the latest models, such as AKT (Ghosh et al., 2020), CL4KT (Lee et al., 2022b) and MonaCoBERT (Lee et al., 2022a).

3.5.4. Large Language Models

While developing DCL4KT, we used KoBERT. In ablation studies, we used three LLMs; KoBERT, KoElectra and KoBigbird. These models are trained by Korean corpus. We fine-tune these models using Huggingface transformers to predict the difficulties of unseen data.

3.5.5. Hyperparameters for Experiments

To compare each model, we used the same parameters for the model training.

- **Batch size**: The batch size was 512. Owing to a limitation of resources, we also used a gradient accumulation.

- **Early stop**: The early stop parameter was 10. For example, if the validation score was not successively increased during the ten iterations, the training session was stopped.

[3]https://sites.google.com/site/assistmentsdata/home
[4]https://pslcdatashop.web.cmu.edu/KDDCup
[5]https://github.com/riiid/ednet
[6]https://www.home-learn.co.kr/main/Index.do

Dataset	Metrics	DKT	DKVMN	AKT	CL4KT	MCB-C	DCL4KT	DCL4KT-A
ASSISTments09	AUC	0.7285	0.7271	0.7449	0.7600	0.8059	0.8111	**0.8153**
	RMSE	0.4328	0.4348	0.4413	0.4337	0.4063	0.4068	**0.4034**
Algebra05	AUC	0.8088	0.8146	0.7673	0.7871	0.8201	0.8288	**0.8295**
	RMSE	0.3703	0.3687	0.3918	0.3824	**0.3584**	0.3657	0.3644
Algebra06	AUC	0.7939	0.7961	0.7505	0.7789	0.8064	0.8258	**0.8278**
	RMSE	0.3666	0.3661	0.3986	0.3863	0.3672	0.3522	**0.3504**
EdNet	AUC	0.6609	0.6602	0.6687	0.6651	0.7336	0.7392	**0.7403**
	RMSE	0.4598	0.4597	0.4783	0.4750	0.4516	0.4505	**0.4500**
Homerun20	AUC	0.7619	0.7543	0.5903	0.6014	0.7659	0.7766	**0.7808**
	RMSE	0.4092	0.4212	0.4745	0.4631	0.4880	0.4042	**0.4014**

Table 1: Overall performance of KT models based on four benchmark datasets and one custom dataset. The best performance is denoted in bold, and the second is underlined. DCL4KT-A indicates DCL4KT that used augmentation strategies. We can see that DCL4KT-A achieved the best results, and DCL4KT was second for most of the benchmark datasets.

- **Training, validation, test ratio**: The training ratio was 80% of the entire dataset, and the test ratio was 20%. The valid ratio was 10% of the training ratio.
- **Learning rate and optimizer**: The learning rate was 0.001, and Adam was used as the optimizer.
- **embedding size**: The embedding size was 512.
- **Contrastive learning ratio**: We used a contrastive learning ratio as 0.1.
- **Augmentation setting**: For augmentation, we set the probability option to control the application of augmentation. Mask-prob is 0.2, crop-prob is 0.1, summarize-prob is 0.2, reverse-prob is 0.1, permute-prob is 0.1, segment-permute-prob is 0.1, replace-higher-diff-prob is 0.1, replace-lower-diff-prob is 0.1, concat-seq-prob 0.1. Also, we used cut-off, not span-cut-off, and the cut-off-prob is 0.03.
- **Others**: We used eight attention heads. The max sequence length was 100, and the encoder number was 4. Models used for comparison, such as AKT[7] and CL4KT[8], used the default settings.

4. Result and Discussion

4.1. Overall Performance

We estimated the overall performance of KT models based on four benchmark datasets and one custom dataset. Table 1 show the performance of each model. Except for the algebra05 (RMSE), DCL4KT-A achieved the highest performance in all of the benchmark datasets. DCL4KT-A is the version where the augmentation strategies are applied to DCL4KT (The hyper-parameter setting of DCL4KT-A can see the **Ablation Studies - Effect of Data Augmentation**). The performance of DCL4KT also followed DCL4KT-A.

4.2. Ablation Studies

4.2.1. Effect of Difficulty-focused Contrastive Learning

To investigate the effect of difficulty-focused contrastive learning, we compared the performance of two cases; 1) non-difficulty-focused contrastive learning (*Non-Diff-CL*), 2) difficulty-focused contrastive learning (*Diff-CL*). For the *Non-Diff-CL* case, the difficulty level of 0.75 is applied to all unseen data in both positive and negative embeddings. Meanwhile, for the *Diff-CL* case, the difficulty level is at 0.75 for positive embedding and 0.25 for negative embedding. As a result, *Diff-CL* achieved higher performance on all of the benchmark datasets. The result is summarized in Table 2.

4.2.2. Difficulty Prediction using LLM

Using the RMSE metric, we compared two hyper-parameters and three LLMs to determine whether LLMs are capable of predicting difficulty. The two hyper-parameters we selected were 0.75 and 0.25, respectively, representing the average difficulty. Using 0.75 as the hyper-parameter in DCL4KT, the model performed optimally based on our provided data. In contrast, the performance was at its lowest when the hyper-parameter was set to 0.25. Con-

[7] https://github.com/arghosh/AKT
[8] https://github.com/UpstageAI/cl4kt

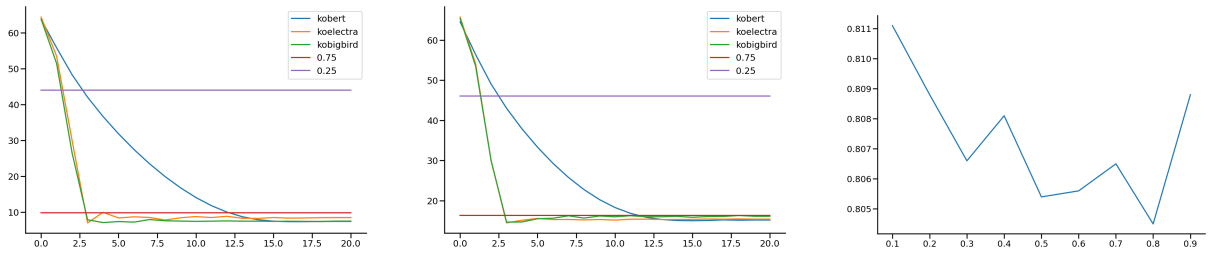

Figure 2: *Left*: Concept difficulty prediction. *Center*: Question difficulty prediction between LLMs. The *x-axis* is training step and *y-axis* means RMSE score. The RMSE score of LLMs are lower than hyper-parameter 0.75. That means LLMs can predict difficulty by using text data of questions and concepts. *Right*: Relationship between contrastive learning ratio (*x-axis*) and model's AUC score (*y-axis*).

sequently, if the LLM prediction score is near the hyper-parameter value of 0.75, the LLM can effectively replace the use of hyper-parameters and heuristics. KoBERT, KoElectra, and KoBigbird were each trained on a Korean corpus to align with our Korean text dataset, and subsequently evaluated for use in our experiment.

As a result, the left plot in Figure 2 shows the concept difficulty prediction. The hyperparameter 0.75 scores 9.8455 and 0.25 scores 44.0410. Meanwhile, KoBERT scores 7.4140, Koelectra scores 8.4801, and KoBigbird scores 7.4923. The center plot in Figure 2 shows the question difficulty prediction. The hyperparameter 0.75 scores 16.3373 and 0.25 scores 44.0410. The LLMs also score better than the hyperparameter 0.75. KoBERT scores 15.1846, Koelectra scores 15.4034, and KoBigbird scores 16.1202. These indicate that LLMs can predict difficulty and our proposed LLM-based difficulty prediction framework works effectively on real data. Moreover, we can assume the corpus of problems or concepts has information related to the difficulties, and difficulty can be represented by the corpus.

4.2.3. Contrastive Learning Loss Ratio

We experiment how the contrastive learning framework affects the performance of the model (AUC). We used DCL4KT, a model to which augmentation strategies have not been applied, and the ASSISTments09 dataset for comparison. The right plot in Figure 2 *right* shows the relationship between contrastive learning loss ratio (*x-axis*) and the model's performance (*y-axis*). When the contrastive learning loss ratio is 0.1, the performance is best (0.8111). Meanwhile the contrastive learning loss ratio is 0.8, the performance is worst (0.8045).

4.2.4. Effect of Data Augmentation

We estimated AUC score of eleven augmentation strategies, shown in (Figure 3), on the ASSISTments09 dataset. Each augmentation strategy is applied independently. The baseline is non-augmented DCL4KT (0.8111). Some of the augmentation strategies are higher than baseline; *cut-off*, *span cutoff*, *replace higher difficulty*. However, when we estimate performance of mixed augmentation, the probabilities are higher (0.8153) than performance of each augmentation independently. Our hyperparameter settings are as follows: *mask-prob* is 0.2, *crop-prob* is 0.2, *summarize-prob* is 0.2, *reverse-prob* is 0.2, *permute-prob* is 0.3, *segment-permute-prob* is 0.2, *replace-higher-diff-prob* is 0.3, *replace-lower-diff-prob* is 0.2, *concat-seq-prob* is 0.2. We used *cutoff* instead of *span-cutoff*, *cutoff-prob* is 0.03. Note that this setting is not optimized, and therefore, there is room to increase the performance of the model.

Dataset	Metric	*Non-Diff-CL*	*Diff-CL*
ASSISTments09	AUC	0.8080	0.8111
	RMSE	0.4070	0.4068
Algebra05	AUC	0.8223	0.8288
	RMSE	0.3721	0.3657
Algebra06	AUC	0.8254	0.8258
	RMSE	0.3525	0.3522
EdNet	AUC	0.7357	0.7392
	RMSE	0.4598	0.4505

Table 2: Comparing performance of *Non-Diff-CL* and *Diff-CL*. *Non-Diff-CL* is applied difficuty as 0.75 to all of the unseen data. Meanwhile, *Non-Diff-CL* is applied up to positive embedding (0.75) and negative embedding (0.25). The performance of *Diff-CL* is better than the *Non-Diff-CL*.

4.3. Relationship between language and difficulty

In the section titled **Estimating Difficulty with LLM**, we demonstrated the predictive potential of LLM based on text data. This indicates that the language of text data contains inherent information about its difficulty. To delve deeper into the relationship between language and difficulty, we examined

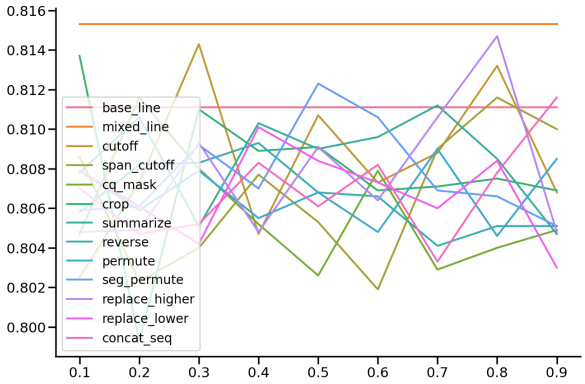

Figure 3: Comparing data augmentation strategies. The *x-axis* is data augment probabilities and *y-axis* means AUC score. The baseline is non-augmented DCL4KT.

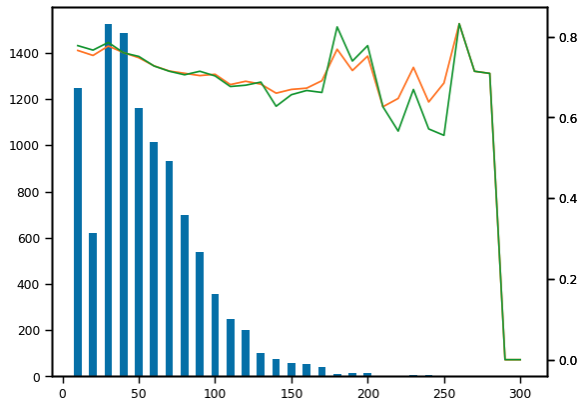

Figure 4: Relationship between character length and difficulty. *x-axis* is the character count of questions. *y-axis left* and *blue histogram* mean the number of character length in the dataset. *y-axis right* mean difficulty. *orange line* is mean of correctness, *green line* is median of correctness. When the character count is less than 120, we can see that the students' correctness decreases as the character length increases.

the relationship between variables derived from text data and difficulty.

Figure 4 displays the correlation between character count and difficulty. The *x-axis* represents the character length of queries, whereas the *y-axis* left and *blue histogram* represent the character count within the dataset. The phrase *y-axis right* refers to adversity. The *orange line* represents the mean level of correctness, while the *green line* represents the median level of correctness. Due to the relatively small number of queries exceeding 120 characters, we only considered instances where the character count was less than 120. The graph depicts a decline in students' accuracy as character length increases, which holds for both the mean and median correctness.

The character count extracted from text data can be regarded as one of the hidden variables influencing the text's difficulty level. Nonetheless, a more comprehensive examination with additional data from various disciplines must confirm the above findings.

5. Conclusion

The significance of difficulty level on student learning habits and the efficacy of the KT model is noteworthy. However, previous KT research has yet to exploit difficulty to improve performance fully and has also struggled to calculate difficulty in unseen data. In response to these obstacles, we have developed a difficulty-centered contrastive learning technique for KT models and a Large Language Model (LLM)-based difficulty prediction framework. These novel techniques can optimize the performance of the KT model and estimate the difficulty level of unknown data. Our ablation investigation confirmed the efficacy of these new techniques for improving the KT model. Nonetheless, the relationship between language and difficulty requires additional study. In subsequent research, we intend to identify the linguistic characteristics that possibly indicate difficulty level.

6. Acknowledgement

We thank i-Scream Edu for their initial support and confidence in our project, which was vital in making this research possible. Their early contribution laid the groundwork for our exploration and has brought our ideas to fruition, marking the beginning of a significant journey in our field.

We also extend our sincere appreciation to Enuma, Inc. for their generous backing and foresight, ensuring the continuation and expansion of our research. Their support has been a cornerstone, providing us with the necessary resources and motivation to delve deeper into our studies and pursue new avenues of inquiry.

7. References

Ghodai Abdelrahman, Qing Wang, and Bernardo Nunes. 2023. Knowledge tracing: A survey. *ACM Computing Surveys*, 55(11):1–37.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020a. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR.

Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey E Hinton. 2020b. Big self-supervised models are strong semi-supervised learners. *Advances in neural information processing systems*, 33:22243–22255.

Song Cheng, Qi Liu, Enhong Chen, Kai Zhang, Zhenya Huang, Yu Yin, Xiaoqing Huang, and Yu Su. 2022. Adaptkt: A domain adaptable method for knowledge tracing. In *Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining*, pages 123–131.

Youngduck Choi, Youngnam Lee, Dongmin Shin, Junghyun Cho, Seoyon Park, Seewoo Lee, Jineon Baek, Chan Bae, Byungsoo Kim, and Jaewe Heo. 2020. Ednet: A large-scale hierarchical dataset in education. In *International Conference on Artificial Intelligence in Education*, pages 69–73. Springer.

Albert T Corbett and John R Anderson. 1994. Knowledge tracing: Modeling the acquisition of procedural knowledge. *User modeling and user-adapted interaction*, 4(4):253–278.

Huan Dai, Yue Yun, Yupei Zhang, Wenxin Zhang, and Xuequn Shang. 2022. Contrastive deep knowledge tracing. In *Artificial Intelligence in Education. Posters and Late Breaking Results, Workshops and Tutorials, Industry and Innovation Tracks, Practitioners' and Doctoral Consortium: 23rd International Conference, AIED 2022, Durham, UK, July 27–31, 2022, Proceedings, Part II*, pages 289–292. Springer.

Maria Orlando Edelen and Bryce B Reeve. 2007. Applying item response theory (irt) modeling to questionnaire development, evaluation, and refinement. *Quality of life research*, 16:5–18.

Aritra Ghosh, Neil Heffernan, and Andrew S Lan. 2020. Context-aware attentive knowledge tracing. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2330–2339.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. 2020. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738.

Neil T Heffernan and Cristina Lindquist Heffernan. 2014. The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, 24:470–497.

Phuc H Le-Khac, Graham Healy, and Alan F Smeaton. 2020. Contrastive representation learning: A framework and review. *IEEE Access*, 8:193907–193934.

Unggi Lee, Yonghyun Park, Yujin Kim, Seongyune Choi, and Hyeoncheol Kim. 2022a. Monacobert: Monotonic attention based convbert for knowledge tracing. *arXiv preprint arXiv:2208.12615*.

Wonsung Lee, Jaeyoon Chun, Youngmin Lee, Kyoungsoo Park, and Sungrae Park. 2022b. Contrastive learning for knowledge tracing. In *Proceedings of the ACM Web Conference 2022*, pages 2330–2338.

Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Hui Xiong, Yu Su, and Guoping Hu. 2019. Ekt: Exercise-aware knowledge tracing for student performance prediction. *IEEE Transactions on Knowledge and Data Engineering*, 33(1):100–115.

Rui Luo, Fei Liu, Wenhao Liang, Yuhong Zhang, Chenyang Bu, and Xuegang Hu. 2022. Dagkt: Difficulty and attempts boosted graph-based knowledge tracing. *arXiv preprint arXiv:2210.15470*.

Sein Minn, Feida Zhu, and Michel C Desmarais. 2018. Improving knowledge tracing model by integrating problem difficulty. In *2018 IEEE International conference on data mining workshops (ICDMW)*, pages 1505–1506. IEEE.

Shalini Pandey and George Karypis. 2019. A self-attentive model for knowledge tracing. *arXiv preprint arXiv:1907.06837*.

Jennifer Petrillo, Stefan J Cano, Lori D McLeod, and Cheryl D Coon. 2015. Using classical test theory, item response theory, and rasch measurement theory to evaluate patient-reported outcome measures: a comparison of worked examples. *Value in Health*, 18(1):25–34.

Chris Piech, Jonathan Bassen, Jonathan Huang, Surya Ganguli, Mehran Sahami, Leonidas J Guibas, and Jascha Sohl-Dickstein. 2015. Deep knowledge tracing. *Advances in neural information processing systems*, 28.

Steven Ritter, John R Anderson, Kenneth R Koedinger, and Albert Corbett. 2007. Cognitive tutor: Applied research in mathematics education. *Psychonomic bulletin & review*, 14:249–255.

Dinghan Shen, Mingzhi Zheng, Yelong Shen, Yanru Qu, and Weizhu Chen. 2020. A simple but tough-to-beat data augmentation approach for natural language understanding and generation. *arXiv preprint arXiv:2009.13818*.

Xiangyu Song, Jianxin Li, Qi Lei, Wei Zhao, Yunliang Chen, and Ajmal Mian. 2022. Bi-clkt: Bi-graph contrastive learning based knowledge tracing. *Knowledge-Based Systems*, 241:108274.

Yu Su, Qingwen Liu, Qi Liu, Zhenya Huang, Yu Yin, Enhong Chen, Chris Ding, Si Wei, and Guoping Hu. 2018. Exercise-enhanced sequential modeling for student performance prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Zejie Tiana, Guangcong Zhengc, Brendan Flanaganb, Jiazhi Mic, and Hiroaki Ogatab. 2021. Bekt: Deep knowledge tracing with bidirectional encoder representations from transformers. *Proceedings of the 29th International Conference on Computers in Education*.

Hanshuang Tong, Yun Zhou, and Zhen Wang. 2020. Exercise hierarchical feature enhanced knowledge tracing. In *Artificial Intelligence in Education: 21st International Conference, AIED 2020, Ifrane, Morocco, July 6–10, 2020, Proceedings, Part II 21*, pages 324–328. Springer.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *International Conference on Machine Learning*, pages 9929–9939. PMLR.

Tangjie Wu and Qiang Ling. 2023. Self-supervised heterogeneous hypergraph network for knowledge tracing. *Information Sciences*, 624:200–216.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Yu Yin, Qi Liu, Zhenya Huang, Enhong Chen, Wei Tong, Shijin Wang, and Yu Su. 2019. Quesnet: A unified representation for heterogeneous test questions. In *Proceedings of the 25th acm sigkdd international conference on knowledge discovery & data mining*, pages 1328–1336.

Jiani Zhang, Xingjian Shi, Irwin King, and Dit-Yan Yeung. 2017. Dynamic key-value memory networks for knowledge tracing. In *Proceedings of the 26th international conference on World Wide Web*, pages 765–774.