Domain Adaption for Knowledge Tracing

Song Cheng\(^1\), Qi Liu\(^{1\ast}\), Enhong Chen\(^1\)

\(^1\)Anhui Province Key Laboratory of Big Data Analysis and Application, School of Computer Science and Technology, University of Science and Technology of China, chsong@mail.ustc.edu.cn; \{qiliuq, cheneh\}@ustc.edu.cn;

ABSTRACT

With the rapid development of online education system, knowledge tracing which aims at predicting students’ knowledge state is becoming a critical and fundamental task in personalized education. Traditionally, existing methods are domain-specified. However, there are a larger number of domains (e.g., subjects, schools) in the real world and the lacking of data in some domains, how to utilize the knowledge and information in other domains to help train a knowledge tracing model for target domains is increasingly important. We refer to this problem as domain adaptation for knowledge tracing (DAKT) which contains two aspects: (1) how to achieve great knowledge tracing performance in each domain, (2) how to transfer good performed knowledge tracing model between domains. To this end, in this paper, we propose a novel adaptable framework, namely adaptable knowledge tracing (AKT) to address the DAKT problem. Specifically, for the first aspect, we incorporate the educational characteristics (e.g., slip, guess, question texts) based on the deep knowledge tracing (DKT) to obtain a good performed knowledge tracing model. For the second aspect, we propose and adopt three domain adaptation processes. First, we pre-train an auto-encoder to select useful source instances for target model training. Second, we minimize the domain-specific knowledge state distribution discrepancy under maximum mean discrepancy (MMD) measurement to achieve domain adaptation. Third, we adopt fine-tuning to deal with the problem that the output dimension of source and target domain are different, to make the model suitable for target domains. Extensive experimental results on two private datasets and seven public datasets clearly prove the effectiveness of AKT for great knowledge tracing performance and its superior transferable ability.

KEYWORDS

Domain Adaptation, Knowledge tracing, Deep learning

1 INTRODUCTION

In recent decades, a large number of computer-aided education systems (CAE) developed rapidly, such as massive open online courses [1] and intelligent tutoring systems [4]. They can give students an open access to the world-class instruction and a reduction in the growing cost of learning, which attracts lots of relevant users worldwide to join in these platforms. With the abundant learning records data of students collected from these platforms (e.g., Khan Academy, Edx), many researchers enrol to research and monitor students’ learning process [7, 19], to recommend adaptive learning materials and arrange personalized plans for them. Actually, it is helpful for students to learn the knowledge concepts which are not mastered, and to consolidate the knowledge concepts which are already mastered. To achieve this goal, the researchers introduce the knowledge tracing (KT) problem which tracking and estimating the knowledge state of the student, i.e., how much the student masters the knowledge concepts, to predict the score of him on the next question.

In the literature, there are many efforts in KT problem. Existing methods can be divided into two categories: traditional and deep approaches. The representative works of conventional knowledge tracing methods such as bayesian knowledge tracing (BKT) [9] and performance factors analysis (PFA) [29] have been widely applied in many application scenarios. As for the deep approaches, deep knowledge tracing (DKT) [30] is the first deep learning based method, which outperforms all of the conventional methods because of its nonlinear input-to-state and state-to-state transitions. Therefore, many variants (e.g., DKT+forgetting [26], DKT-tree [39]) of DKT have been proposed to improve KT performance.

Though great success has been achieved by BKT, DKT, etc., there still exist some issues that limit the application of them. First, as shown in Figure 1, in the real world, there are many different subjects, schools and even grades that can be seen as different domains. However, existing KT models are domain (e.g., subject, schools, grades) specified methods, and the specific (e.g., math) model cannot be applied to other domains (e.g., physics) directly. Therefore, it is labor and resource-intensive to build so many models for all of the application scenarios. Second, the another issue is that there are some domains (e.g., some schools) usually lack abundant data for training models. Generally, how to utilize the knowledge and
Concepts

| Question Texts | Concepts |
|----------------|----------|
| If function $f(x) = x^2 - 2x + 2$ and $x \in [0,3]$, what is the range of $f(x)$? | Function |
| What is the y-intercept of the graph of equation $y = 2x \cdot (4x - 4) - 10$? | Function |
| If function $f(x) = 2x - 2$ and $x \in [-1,1]$, what is the range of $f(x)$? | Function |

Figure 2: Example of the question texts, slip and guess cases.

information in other domains to help train a KT model for target domains is increasingly important. We note to this problem as domain adaptation for knowledge tracing (DAKT) which contains two aspects: (1) how to achieve great knowledge tracing performance in each domain? (2) how to transfer good performed knowledge tracing model between domains?

In this paper, to solve DAKT problem introduced above, we propose an adaptable knowledge tracing (AKT) framework. Specifically, on one hand, for achieving great knowledge tracing performance in each domain, following previous works [8, 40], we model some educational characteristics (e.g., slip, guess, question texts) which are important to model student learning process and ignored by existing methods. For example, as shown in Figure 2, it is slipping that the student gets a wrong answer because he mistakes the minus sign for a plus sign. It is guessing that he chooses a correct option by rotating the pencil to get a random recommendation (e.g., C). It is better to embed question texts for representation than knowledge concepts since question texts are totally different though the knowledge concepts are the same. On the other hand, for achieving good information transferring between domains, we propose and adopt three domain adaptation processes. First, with reconstruction error, we pre-train an auto-encoder to select useful source instances for target model training. Second, we minimize the domain-specific knowledge state distribution discrepancy which is caused by the different distributions of student score $r_i$ and question texts $q_i$ under maximum mean discrepancy (MMD) [3] to achieve domain adaptation. Third, we adopt fine-tuning [18, 42] to deal with the problem that the output dimensions of source and target domains are different to make the model suitable for target domain. Extensive experimental results on two private datasets and seven public datasets clearly demonstrate the effectiveness of AKT for great knowledge tracing results and its superior transferable ability.

The main contributions in this work are summarized as follows:

1. To our best knowledge, it is the first time that we propose the domain adaptation for knowledge tracing (DAKT) problem for the wide application of knowledge tracing.
2. We model the educational characteristics (e.g., slipping, guessing, question texts) to achieve good knowledge tracing performance for the first aspect of DAKT.
3. We propose and adopt instance selection via pre-training, domain discrepancy minimizing and fine-tuning to achieve great domain adaptation for the second aspect of DAKT.

2 PRELIMINARIES

In this section, we introduce the knowledge tracing task and domain adaptation techniques, which are the cornerstone of our adaptable knowledge tracing framework.

2.1 Knowledge Tracing

Knowledge tracing (KT) is a fundamental task in online education platforms, which aims at tracing the knowledge state of students with their historical interaction performances. It is usually formulated as a supervised time sequence learning problem.

In an intelligent education system, suppose there are $|M|$ students and $|Q|$ questions. We record the interaction process of a certain student as $I = \{s_1, s_2, ..., s_T\}, s_t = (q_t, r_t)$, where $q_t \in Q$ represents the question practiced by the student at interaction step $t$, and $r_t$ denotes the corresponding score. Generally, if the student answers the question $q_t$ rightly, $r_t$ equals to 1, otherwise $r_t$ equals to 0. Also, each question $q$ contains its corresponding knowledge concept $k$. Without loss of generality, in this paper, we represent each student-$A$ interaction sequence as $I = \{(k_1, s_1), (k_2, s_2), ..., (k_T, s_T)\}$ for knowledge tracing task. More formally, knowledge tracing can be formulated as follows:

Definition 2.1 (Knowledge Tracing). Given the question interaction sequence $I$ of each student and the materials of each question from interaction step 1 to $T$, the goal is two-fold: (1) tracking the change of his knowledge states and estimating how much he masters knowledge concepts from step 1 to $T$; (2) using his current knowledge state of him to predict the score $r_{T+1}$ on the next candidate question $q_{T+1}$.

2.2 Domain Adaptation

Domain adaptation (DA) is a particular case of transfer learning (TL) which utilizes labeled data in one or more relevant source domains to execute new tasks in target domains. Given a labeled source domain $D_S = \{(x_{iS}, y_{iS})\}_{i=1}^{ns}$ and a target domain $D_T = D_{T1} \cup D_{Tu}$, where $D_{T1} = \{(x_{i}, y_{i})\}_{i=1}^{nT1}$ is the limited labeled part and $D_{Tu} = \{x_i\}_{i=1}^{nTu}$ is the abundant unlabeled part of target domain, we formally introduce the definition of domain adaptation as follows:

Definition 2.2 (Domain Adaptation). Assume the feature space, label space and conditional probability distributions of source and target domains are the same, i.e., $X_S = X_T, Y_S = Y_T$ and $P_S(y_{iS}|x_{iS}) = Q_T(y_{iT}|x_{iT})$. Meanwhile, the margin distributions of source and target domain are different, that is $P_S(x_S) \neq P_T(x_T)$. The goal of DA is to leverage $D_S$ with the help of $D_T$ to learn a task function $f : x_T \mapsto y_T$ for $D_T$.

3 AKT: MODELING AND TRANSFERRING

In this section, we introduce AKT modeling and the transferring process which aim at solving DAKT problem in details. First, we give a formal definition of DAKT problem. Then, we describe the AKT architecture which introduces the educational characteristics
for boosting KT task. Afterwards, we describe the transferring process of AKT, i.e., instance selection via pre-training, domain discrepancy minimizing and fine-tuning.

3.1 DAKT Problem Definition

Knowledge tracing is not the same as conventional machine learning tasks, since the input of KT task is the tuple \(z_t = (q_t, r_t)\), where \(z_t\) not only contains question \(q_t\) but also students’ score \(r_t\). Meanwhile, \(r_{t+1}\) is served as the label to be predicted next. This educational characteristic of KT makes it challenging to address domain adaptation for knowledge tracing (DAKT) problem. With different subjects, schools and even grades as domains, we give a formal definition of DAKT problem as follows:

**Definition 3.1 (DAKT Problem).** Given the student’ interaction sequence \(I_S\) in source domain and limited labeled part \(I_T\) of interaction sequence in target domain, the goal of DAKT is two-fold: (1) learning a good knowledge tracing model \(M\) by leveraging \(I_S\) in source domain \(D_S\); (2) transferring \(M\) to target domain \(D_T\) with the help of \(I_T\).

In the following sections, we will address DAKT problem from two aspects: (1) how to introduce the educational characteristics (e.g., slip, guess, questions texts) for achieving better knowledge tracing; (2) how to conduct transferring processes to transfer good performed knowledge tracing model between domains for achieving domain adaptation.

3.2 AKT Modeling

As shown in Figure 3 (c), the backbone of AKT model follows the architecture proposed by the original work of deep knowledge tracing (DKT) [30]. In the next, we will introduce the architecture of AKT framework.

3.2.1 Question Texts Integrating. Since the question texts may be totally different though the knowledge concepts tested by them are the same, therefore, it will be better to represent the question by understanding its texts semantic than representing with knowledge concepts which is popular in previous works [26, 30, 39, 45]. For integrating semantic of question texts, we adopt an unsupervised trained auto-encoder as shown in Figure 3 (a) and Figure 3 (b) to understand them, which is needed for transferring process described in next sections. The formal definition of the auto-encoder is as follows:

\[
\begin{align*}
\text{encoder:} & \quad q_i = \pi_e(x_i), \\
\text{decoder:} & \quad \hat{x}_i = \pi_d(q_i),
\end{align*}
\]

(1)

where \(x_i = (w_1, w_2, ..., w_L)\) is the question text embedding sequence, \(L\) is the length of the question text, and \(\hat{x}_i = (\hat{w}_1, \hat{w}_2, ..., \hat{w}_L)\) is the reconstructed result of \(x_i\).

Specifically, for the encoder \(\pi_e\), its output \(q_i\) can be considered as a higher-level semantic and robust representation for \(i\)-th question. Note that the input of the encoder is the question text which is a word sequence. In this case, we choose Bi-LSTM shown in Figure 3 (a) as the encoder \(\pi_e\) because it can make use of the most of contextual content information of question text from both forward and backward directions [16]. Given \(i\)-th question text embedding sequence \(x_i\), at each step \(t\), the forward hidden state \(\hat{h}^{enc}_t\) and backward hidden state \(\overrightarrow{h}^{enc}_t\) are updated with current word \(w_t\), and previous hidden state \(\hat{h}^{enc}_{t-1}\) for forward direction or \(\overrightarrow{h}^{enc}_{t-1}\) for backward direction in a recurrent formula as:

\[
\begin{align*}
\hat{h}^{enc}_t &= \text{LSTM}(\hat{h}^{enc}_{t-1}, w_t; \theta^{enc}_{\text{LSTM}}), \\
\overrightarrow{h}^{enc}_t &= \text{LSTM}(\overrightarrow{h}^{enc}_{t+1}, w_t; \theta^{enc}_{\text{LSTM}}),
\end{align*}
\]

(2)

where recurrent formula follows Hochreiter et al. [15].

As hidden state at each direction only contains one-side context, it is beneficial to concatenate the hidden state of both directions into one vector to capture the linguistic information at each step. Therefore, we model the semantic representation at \(t\)-th step as:

\[
\eta_t = \text{concatenate}(\hat{h}^{enc}_t, \overrightarrow{h}^{enc}_t),
\]

(3)
and then combine all representations at all time steps into one single vector $q_t$ with max-pooling, more formally:

$$q_{ij} = \max(\eta_{ij}, \eta_{2j}, ..., \eta_{Lj})$$  \hspace{1cm} (4)

As for the decoder $d_t$, since the input $x_t$ that we aim to reconstruct is a word sequence, and LSTMs have yielded state-of-the-art results for many problems, such as image captioning [37], natural language processing [35], and handwriting recognition [13]. Therefore, for the sake of simplicity, we adopt a single layer LSTM neural network as the decoder structure similar to sequence autoencoders [35]. Comparatively, the input at each $t$-th time of the decoder $d_t$ is different from the general LSTM model, which is, $d_t$ leverages the $(t-1)$-th step output $\hat{w}_{t-1}$ as the $t$th step input. More formally, the definition of the $t$-th step hidden state can be formulated as:

$$h^\text{dec}_t = \text{LSTM}(h^\text{dec}_{t-1}; \theta^\text{LSTM}),$$

$$\hat{w}_t = \text{sigmoid}(W^\text{dec} h^\text{dec}_t + b^\text{dec}),$$

where $\theta^\text{LSTM}$, $W^\text{dec}$ and $b^\text{dec}$ are the parameters of the decoder.

Without the loss of generality, we assume $r_t$ in each interaction $s_t = (q_t, r_t)$ is either 0 or 1. Methodology-wise, to be consistent with DKT when integrating semantic understanding of question texts [34], we first give a zero vector $0 = (0, 0, ..., 0)^T$ which has the same dimension $d_q$ as $q_t$, then we concatenate $q_t$ and $0$ to obtain the interaction representation $s_t$ at each time step $t$ as:

$$s_t = \begin{cases} (q_t \oplus 0), & r_t = 1, \\ (0 \oplus q_t), & r_t = 0, \end{cases}$$

where $\oplus$ is the operation of concatenating two vectors.

### 3.2.2 Slipping and Guessing Modeling.

To more precisely predict student’s knowledge state and give strong interpretability for it, we introduce the slipping factor $s_t$ and the guessing factor $g_t$ factors of question $q_t$ to KT process. Different from conventional random sampling approach [43], we propose to exploit the slipping and guessing factors from the semantic representation of the question texts. Specifically, as shown in Figure 2, following previous work [8, 40], we propose two single layer neural networks $S$ and $G$ as shown in Figure 3 (c) to model the slipping and guessing factors as follows:

$$s_t = S(q_t),$$

$$g_t = G(q_t),$$

(7)

where $s_t$ and $g_t$ have the same dimension of $d_h$.

### 3.2.3 Knowledge State Tracing.

With the interaction sequence $I = \{s_1, s_2, ..., s_N\}$ of a student where $N$ is the max sequence length. An LSTM architecture as described in DKT is then utilized to model the student learning process, and the hidden state $h_t \in \mathbb{R}^{d_h}$ at $t$-th step is updated as following formulas:

$$i_t = \sigma(W^s_i \cdot s_t + W^h_i \cdot h_{t-1} + b_i),$$

$$f_t = \sigma(W^s_f \cdot s_t + W^h_f \cdot h_{t-1} + b_f),$$

$$o_t = \sigma(W^s_o \cdot s_t + W^h_o \cdot h_{t-1} + b_o),$$

$$c_t = f_t c_{t-1} + i_t \tanh(W^s_c \cdot s_t + W^h_c \cdot h_{t-1} + b_c),$$

$$h_t = o_t \tanh(c_t),$$

where $i_t, f_t, c_t, o_t$ are the input gate, forget gate, memory cell and output gate of LSTM respectively, $W^s_i, W^h_i \in \mathbb{R}^{d_h \times d_q}$, $W^h_o \in \mathbb{R}^{d_h \times d_h}$ and $b_i, b_f, b_o, b_c$ are the learned parameters.

Particularly, the matrix $W^s_c$ in Eq. 8 can be divided into a positive one $W^s_c^+ \in \mathbb{R}^{d_h \times d_q}$ and a negative one $W^s_c^- \in \mathbb{R}^{d_h \times d_q}$ [34], which can separately capture the influences of question $q_t$ with both right (e.g., $r_t = 1$) and wrong (e.g., $r_t = 0$) response. Therefore, it can naturally predict student knowledge states by integrating both question texts and responses.

After obtaining the representation $h_t$ of the interaction $s_t = (q_t, r_t)$, and the corresponding slipping $s_t$ and guessing $g_t$ factors of the question $q_t$, we simulate the student knowledge states $\kappa_t \in \mathbb{R}^{d_h}$ as follows:

$$\kappa_t = (1 - s_t) \otimes h_t + g_t \otimes (1 - h_t),$$

(9)

where $\otimes$ is the element-wise multiply operation. $(1 - s_t) \otimes h_t$ means the amount of knowledge state changing of a student when he masters the question and answers it successfully (i.e., without carelessness), while $g_t \otimes (1 - h_t)$ represents the amount of knowledge state changing of the student when he guesses a right response without mastering [43]. That is, these are the two ways for a student to give a correct response.

Following DKT, we adopt a linear prediction layer to predict the probabilities $y_t \in \mathbb{R}^Q$ of answering correctly for all questions, with the simulated knowledge state $\kappa_t$ of the student. Comparatively, we add an adaptation layer before the prediction layer to achieve domain adaptation which is described in details in next sections. More formally, we define the probabilities $y_t$ as follows:

$$a_t = \text{adaptation}(\kappa_t; \Theta_{\text{adapt}}),$$

$$y_t = \text{sigmoid}(W_{\text{out}} \cdot a_t + b_{\text{out}}),$$

(10)

where $\Theta_{\text{adapt}}$ represents the parameters of adaptation layer, $a_t$ is the output of adaptation layer at $t$-th step, $W_{\text{out}}$ and $b_{\text{out}}$ are the parameters of the prediction layer.

### 3.3 Domain Adaptation

To achieve domain adaptation of KT to solve DAKT problem for wide application of it, we need to conduct some transferring process from source domain to target domain. Therefore, We propose to adopt three approaches from different aspects to address it greatly. Specifically, we first select the questions from source domain that are similar and useful for target domain, there should be a pair of encoder $\pi_s$ and decoder $\pi_d$ models whose reconstruction errors on source and target domain data are both small. Therefore, following previous work [36], we propose an auto-encoder [2] as shown in Figure 3 (a) and Figure 3 (b) to select questions from source domain via minimizing reconstruction errors on the selected source domain questions and all target domain questions.
The reconstruction error of the auto-encoder can be defined as:

$$\mathcal{R}(\hat{x}_i, x_i) = \frac{1}{L} \sum_{l=1}^{L} \|\hat{x}^l_i - x^l_i\|^2.$$  

(11)

To select the useful questions from source domain for target domain, we propose to pre-train the pair of $(\pi_s, \pi_t)$ to minimize the reconstruction errors on both selected questions in source domain and all the questions in target domain simultaneously. The formal definition of the objective function is as follows:

$$J_0(\pi_s, \pi_t, u_S) = \frac{1}{n_S} \sum_{i=1}^{n_S} u^S_i \mathcal{R}(\hat{x}_i^S, x_i^S) + \frac{1}{n_T} \sum_{i=1}^{n_T} \mathcal{R}(\hat{x}_i^T, x_i^T) + \Gamma(u_S).$$  

(12)

where $x_i^S$ and $x_i^T$ are the questions in source and target domains, $\hat{x}_i^S$ and $\hat{x}_i^T$ are the reconstructions of $x_i^S$ and $x_i^T$, $u_S = (u_1^S, u_2^S, ..., u_{n_S}^S)$, and $u_i^S \in \{0, 1\}$ is the selection indicators (e.g., 1 for selected, 0 for unselected) for the $i$-th question in source domain. The $\Gamma(u_S)$ in the objective function is a regulation term on $u_S$ to avoid the case that all values in $u_S$ is zero, it is defined as follows:

$$\Gamma(u_S) = -\frac{\lambda}{n_S} \sum_{i=1}^{n_S} u_i^S,$$  

(13)

where $\lambda \in [0, 1]$ controls the importance of the regularization term.

It is obvious that more useful questions are selected, more robust semantic understanding can be learned by the pre-trained auto-encoder architecture.

To select useful questions in source domain, we update the parameters $\Theta_{auto}$ of auto-encoder and $u_S$ alternately. When $u_S$ is fixed, $\Theta_{auto}$ can be updated with back-propagation algorithm, comparatively, when $\Theta_{auto}$ is fixed, the solution of $u_S$ can be obtained as follows:

$$u_i^S = \begin{cases} 1, & \mathcal{R}(\hat{x}_i^S, x_i^S) < \lambda, \\ 0, & \text{otherwise}. \end{cases}$$  

(14)

The reasonable explanation of this alternate training strategy is two folds: (1) fixing $\Theta_{auto}$ to update $u_S$, the useless questions in the source domain are removed; (2) updating $\Theta_{auto}$ with $u_S$ fixed, the auto-encoder is only trained on the selected questions.

3.3.2 Domain Discrepancy Minimizing. After obtaining the selected useful and similar questions in source domain, we can perform transfer learning across different domains. However, a model trained only on source domain usually leads to negative transfer, which will reduce the performance of the framework on target domain. This scenario is caused by the discrepancy of the probability distribution between the source and target domains. Specifically, the margin distribution $\mathcal{P}_S(x_S)$ and $\mathcal{P}_T(x_T)$ of the question texts of source and target domains are different, moreover, the distribution of the score $r$ which reflects the latent trait of the student is not the same too. That is, the student’s hidden knowledge states that learned with $(q_t, r_t)$ are mismatched between two domains. As shown in figure 3 (c), our intuition in this paper is to add an adaptation layer before output layer, to learn a sharing representation that minimizes the discrepancy between source knowledge state distribution $\mathcal{P}(x_S)$ and target knowledge state distribution $\mathcal{P}(x_T)$.

We consider adopting the widely applied distribution distance metric, maximum mean discrepancy (MMD) [3]. It is an effective criterion that compares distributions without their density functions. More formally, with two probability distributions $P$ and $Q$ on common feature space $X$, MMD is defined as follows:

$$\text{MMD}(\omega, P, Q) = \sup_{f \in \mathcal{F}} \{\mathbb{E}_P[f(x)] - \mathbb{E}_Q[f(x)]\},$$  

(15)

where $\omega$ is a set of functions $f : X \mapsto \mathbb{R}$, and $\mathbb{E}[-]$ denotes the mean of the samples. Thus, based on the data $x_S = \{x_1^S, x_2^S, ..., x_n^S\}$ and $x_T = \{x_1^T, x_2^T, ..., x_n^T\}$ of source and target domains, MMD can be rewritten as follows:

$$\text{MMD}(x_S, x_T) = \left\| \frac{1}{2n_S} \sum_{i=1}^{n_S} \phi(x_i^S) - \frac{1}{2n_T} \sum_{j=1}^{n_T} \phi(x_j^T) \right\|_{\mathcal{H}},$$  

(16)

where $\phi(x) : X \mapsto \mathcal{H}$ is referred as the feature space map, which maps the variable that in original feature space $X$ to the reproducing kernel hilbert space (RKHS) $\mathcal{H}$ [3].

Moreover, our goal is not only to minimize the discrepancy between domains, but also to learn a robust knowledge tracing model. Therefore, the general form of the objective function can be expressed as:

$$J_{\text{AKT}} = J_{\text{KT}} + \gamma \text{MMD}^2(x_S, x_T),$$  

(17)

where $\gamma \in [0, 1]$ is the regularization parameter that controls the importance of MMD, the $J_{\text{KT}}$ denotes the loss of the knowledge tracing task [30]. More formally, it is defined as follows:

$$J_{\text{KT}} = \frac{1}{n} \sum_{i=1}^{n_T} \mathcal{L}(y^{(i)}T, \delta(\hat{x}_i^T), r_i^T),$$  

(18)

where $n$ is the number of students, $T$ is the cross-entropy loss, $\delta(\cdot)$ is the one-hot encoding function which indicates $x_i$ is the $i$-th question.

Similarly, we separate $J_{\text{KT}}$ and $\text{MMD}^2(\cdot, \cdot)$, and minimize them alternately in two steps: (1) we first minimize $J_{\text{KT}}$ with mini-batch stochastic gradient descent via back-propagation algorithm; (2) then $\text{MMD}^2(\cdot, \cdot)$ is minimized with full-batched gradient descent.

3.3.3 Fine-tuning. After training, AKT should be able to minimize the discrepancy between different domains and transfer knowledge from source domain to target domain. However, there is an issue which needs to be addressed, otherwise, we cannot apply our model trained on source data to target domain. Specifically, the number of the questions in the domains are different from each other, for example, there are totally $Q_S$ questions in source domain and $Q_T$ questions in target domain, so the model trained on source data would have a $Q_S$ dimension output layer, which mismatches the number $Q_T$ of target domain.

As we can see, the amounts of the questions of different domains are not the same. To transfer our AKT framework from source domain to target domain, only the output layer is required to be replaced and retrained for the specific target domain. It is termed as fine-tuning, which leads to effective knowledge transferring between different domains.

In summary, AKT has the following advantages. First, it provides a simple but effective way to capture the slipping and guessing factors of the questions, and leverage the semantic understanding of the texts to represent the question better for knowledge tracing performance improving. Second, it can achieve domain adaptation between source domain and target domain via pre-training, discrepancy minimizing and fine-tuning techniques.
4 EXPERIMENTS

In this section, extensive experiments are conducted from two aspects to demonstrate the effectiveness of our AKT framework. First, the comparative experiments are conducted on two private datasets and seven public well-established datasets in KT task. The datasets used in this paper include two private datasets, to demonstrate its superior transferable ability.

4.1 Experimental Setup

4.1.1 Datasets. The datasets used in this paper include two private datasets and seven public well-established datasets in KT task. The basic statistical information for all the datasets are summarized in Table 1. The detail descriptions are as follows:

- **zx.math.** This is a math subject dataset which is private and composed of the mathematical data collected from a number of senior high school in China. It is really a huge-scale dataset, we randomly selected 329,533 interactions to from 5,000 student with 60,752 questions.

- **zx.physics.** This is a physics subject dataset which is private the same as **zx.math**, and collected from a number of senior high school in China. it is a huge-scale dataset too. we randomly selected 397,263 interactions to from 5,000 student with 75,141 questions.

- **Synthetic.** This dataset\(^1\) is simulated by piech et al. [30], which simulated 2000 virtual students answering 50 questions both in training and testing dataset, which are drawn from five virtual concepts. In this dataset only, all students answer the same sequence of 50 questions.

- **Static2011.** This dataset\(^7\) is from a college-level engineering statics course with 189,927 interactions from 333 student and 1,223 questions. In our experiments, we have adopted the processed data provided by Zhang et al. [45].

- **Kddcup2010.** This dataset\(^8\) is provided as the development dataset in KDD Cup 2010 competition, which comes from the Cognitive Algebra Tutor 2005-2006 Algebra system. After processing this dataset as Xiong et al. [38], it contains 607,026 interactions from 574 student with 436 questions.

- **junyi.** This dataset\(^4\) is collected from Junyi Academy\(^5\), which is an online education website. It is too large to effectively conduct experiments, so we randomly select 50,000 students with 7,868,952 interactions on 701 questions. It is still the largest dataset with a large number of interactions.

- **ASSIST2009.** This dataset\(^6\) is collected from ASSISTments\(^9\) online tutoring system\(^7\), which has been widely applied in KT tasks. Before conducting the experiments, we remove the duplicated records in the original dataset. The pruned dataset contains 325,673 interactions for 4,151 students with 110 questions.

- **ASSIST2015.** This dataset\(^6\) is collected from the same platform as ASSISTments2009, it contains 683,801 effective interactions from 19,840 students on 100 questions after processed.

- **ASSIST2017.** This dataset\(^6\) is also collected from the same platform as ASSISTments2009, it is provided by the 2017 ASSISTments data mining competition, the average number of interactions per student is extremely high as it contains 686 students with 942,816 interactions on 102 questions.

4.1.2 AKT Setup. For embedding, we incorporate Word2vec [25] on the whole corpus to get an initial word to vector mapping with size 100. The hidden state of the Bi-LSTM encoder is set to 50, to keep the output size of Bi-LSTM the same as the input size of the LSTM decoder. We adopt different regularization term importance via set the value of $\lambda$ as 0.0, 0.25, 0.5, 0.75, 1.0. We set the output size of slipping and guessing network to 100 which is the same as the hidden size of LSTM in AKT framework. To determine what dimension that our adaptation layer should have, we set the width size 100. The hidden state of the Bi-LSTM encoder is set to 50, to keep the output size of Bi-LSTM the same as the input size of the LSTM decoder. We adopt different regularization term importance via set the value of $\lambda$ as 0.0, 0.25, 0.5, 0.75, 1.0.

Table 1: The statistics of the dataset.

| Dataset          | # Questions | # Students | # Interactions |
|------------------|-------------|------------|----------------|
| zx.math          | 60,752      | 5,000      | 329,533        |
| zx.physics       | 75,141      | 5,000      | 397,263        |
| Synthetic        | 50          | 4,000      | 200,000        |
| Static2011       | 1,223       | 333        | 189,297        |
| Kddcup2010       | 436         | 574        | 607,026        |
| Junyi            | 701         | 50,000     | 7,868,952      |
| ASSISTments2009  | 110         | 4,151      | 325,673        |
| ASSISTments2015  | 100         | 19,840     | 683,801        |
| ASSISTments2017  | 102         | 686        | 942,816        |

\(^1\)https://github.com/chrispiech/DeepKnowledgeTracing/tree/master/data/synthetic
\(^2\)https://paledata.data.humdata.org/DatasetInfo?datasetid=507
\(^3\)https://paledata.data.humdata.org/KDDCup/downloads.jsp
\(^4\)https://datashop.web.cmu.edu/DatasetInfo?datasetid=1198
\(^5\)https://www.junyiacademy.org/
\(^6\)https://sites.google.com/site/assistmentsdata/home/2015-assistments-skill-builder-data
\(^7\)https://sites.google.com/site/assistmentsdata/home/2015-assistments-skill-builder-data
\(^8\)https://sites.google.com/view/assistmentsdatamining/dataset?authuser=0
\(^9\)https://www.assistments.org/
### Table 2: Effectiveness of the slipping and guessing factors, the comparison of AUC of AKT-tx-tr with DKT on two private and seven public datasets. "s. dim" represents the hidden knowledge state dimension.

| Model     | s. dim | zx.math | zx.physics | ASSIST2009 | ASSIST2015 | ASSIST2017 | Synthetic | Static2011 | Kddcup2010 | Junyi |
|-----------|--------|---------|------------|------------|------------|------------|-----------|------------|------------|-------|
| DKT       | 10     | 0.7047  | 0.6984     | 0.8349     | 0.7155     | 0.6534     | 0.8037    | 0.8186     | 0.7725     | 0.8783 |
|           | 50     | 0.7248  | 0.7021     | 0.8369     | 0.7314     | 0.7029     | 0.8251    | 0.8292     | 0.7989     | 0.8952 |
|           | 100    | 0.7255  | 0.6989     | 0.8398     | 0.7303     | 0.6806     | 0.8171    | 0.8193     | 0.7724     | 0.8874 |
|           | 200    | 0.7253  | 0.6993     | 0.8397     | 0.7301     | 0.6861     | 0.8232    | 0.8217     | 0.7732     | 0.8901 |
| AKT-tx-tr | 10     | 0.7542  | 0.7416     | 0.8357     | 0.7075     | 0.6534     | 0.7995    | 0.8429     | 0.7784     | 0.8864 |
|           | 50     | 0.7718  | 0.7533     | 0.8440     | 0.7282     | 0.7181     | 0.8125    | 0.8606     | 0.8185     | 0.9041 |
|           | 100    | 0.7623  | 0.7422     | 0.8461     | 0.7245     | 0.7056     | 0.8090    | 0.8488     | 0.8069     | 0.8957 |
|           | 200    | 0.7621  | 0.7401     | 0.8447     | 0.7251     | 0.7097     | 0.8179    | 0.8532     | 0.8086     | 0.8955 |

### Table 3: Effectiveness of the slipping and guessing factors, the comparison of F1-score of AKT-tx-tr with DKT on two private and seven public datasets. "s. dim" represents the hidden knowledge state dimension.

| Model     | s. dim | zx.math | zx.physics | ASSIST2009 | ASSIST2015 | ASSIST2017 | Synthetic | Static2011 | Kddcup2010 | Junyi |
|-----------|--------|---------|------------|------------|------------|------------|-----------|------------|------------|-------|
| DKT       | 10     | 0.7054  | 0.7025     | 0.8545     | 0.8506     | 0.3767     | 0.7878    | 0.8513     | 0.9031     | 0.8297 |
|           | 50     | 0.7285  | 0.7163     | 0.8551     | 0.8521     | 0.4687     | 0.7953    | 0.8661     | 0.9086     | 0.8405 |
|           | 100    | 0.7338  | 0.7022     | 0.8563     | 0.8520     | 0.4163     | 0.7947    | 0.8548     | 0.9032     | 0.8341 |
|           | 200    | 0.7314  | 0.7013     | 0.8564     | 0.8518     | 0.4216     | 0.7990    | 0.8591     | 0.9039     | 0.8388 |
| AKT-tx-tr | 10     | 0.7793  | 0.7539     | 0.8543     | 0.8497     | 0.3424     | 0.7872    | 0.8600     | 0.9059     | 0.8402 |
|           | 50     | 0.7837  | 0.7635     | 0.8580     | 0.8511     | 0.4887     | 0.7872    | 0.8740     | 0.9117     | 0.8503 |
|           | 100    | 0.7751  | 0.7541     | 0.8576     | 0.8505     | 0.4427     | 0.7885    | 0.8708     | 0.9101     | 0.8421 |
|           | 200    | 0.7736  | 0.7524     | 0.8583     | 0.8506     | 0.4636     | 0.7937    | 0.8724     | 0.9106     | 0.8484 |

Before any training step, we set the learning rate and batch size as $1 \times 10^{-5}$ and $64$ respectively. The parameters are randomly initialized with uniform distribution in the range between $-\sqrt{6}/(\text{nin} \times \text{nout})$ and $\sqrt{6}/(\text{nin} \times \text{nout})$ which follows Orr et. al. [27], where $\text{nin}$ and $\text{nout}$ are the numbers of input and output size. Then at the training process, parameters are updated by Adam optimization algorithm [20].

#### 4.1.3 Evaluation.

For the non-simulated data, we evaluate our results using 5-fold cross validation and in all cases, hyper-parameters are learned on training data. To evaluate the performance of the KT methods, the precise knowledge state can predict students’ performance accurately. Thus, we use the area under the receiver operating characteristic (ROC) curve which is termed as AUC [21], and the F1 measure [12] which is the weighted harmonic mean of precision and recall. For both AUC and F1, the value 0.5 means the prediction results are guessed randomly, and the larger, the better.

Note, we tune the parameters AKT and other models used in the next sections to achieve the best performance for the purpose of a fair comparison, and implement it with PyTorch on a Linux server with a Tesla K20m GPU.

#### 4.2 Educational Characteristics Modeling

4.2.1 Effectiveness of Slipping and Guessing.

To investigate the effectiveness of our framework for modeling the slipping and guessing factors, we conduct extensive experiments on AKT-tx-tr under AUC and F1 evaluation metrics, where "-" is a minus sign, which means AKT abandons the question texts and transferring process. For comparison, we adopt DKT as the baseline since it is similar to the backbone of AKT-tx-tr, that is both of them are based on LSTM neural network except the slipping and guessing modeling parts.

The experimental results which clearly investigate the differentiation of the performance between AKT-tx-tr and DKT are shown in Table 2 and Table 3. Specifically, we compare DKT with AKT-tx-tr under different hidden layer dimensions (e.g., 10, 50, 100, 200), which also can be termed as state dimension (s.dim). There are several observations. First, on the two private datasets (e.g., zx.math, zx.physics) which contain question texts, AKT-tx-tr performs better than DKT on two measurements, especially on 50 dimension knowledge state. Second, to further validate the effectiveness of the slipping and guessing modeling parts, we repeat the same experiments on the other seven public KT datasets described in section 4.1.1. Obviously, we can find that although the performance of AKT-tx-tr is slightly lower (e.g., 0.4% on AUC, 0.5% on F1) than DKT on ASSISTments2015 and Synthetic, it gets better performance than DKT on five public KT datasets, especially on Static2011, it surpasses DKT by a large margin (e.g., 3%) on AUC metric. Third, there is an interesting discovery that the AUC of both DKT and AKT-tx-tr on dataset ASSISTments2017 is more than 0.7, but both of them have bad F1-score (e.g., no more than 0.5), that is DKT and AKT-tx-tr does not perform well, but a suitable threshold can be selected to obtain great AUC. Therefore, whether AUC is a reasonable measurement to evaluate knowledge tracing models is worth exploiting. In general, all the evidences observed above indicate...
that our framework can modeling slipping and guessing factors well for boosting knowledge tracing performance.

4.2.2 Effectiveness of Question Texts. To demonstrate the effectiveness of our framework of understanding the semantic of question texts for better question representation than one-hot to improve the KT performance. We conduct extensive experiments on AKT-tx where “-” is a minus sign too with two private datasets (e.g., zx.math, zx.physics) since public datasets contain no question texts. The statistic information of the question texts of the two datasets are shown in Figure 4. For comparison, we adopt following three approaches as baselines:

- BKT [9] is a Bayesian inference-based approach, in which the knowledge state is represented as a binary variable, it indicates the mastering of a single concept.
- DKT [30] is an important KT model, which applies a recurrent neural network to model the student learning process for estimating students’ mastery of concepts.
- DKVMN [45] which utilizes a key-value memory to extend the memory-augmented neural network is known as the state-of-the-art approach for KT.

The comparison results on each dataset on five different models including AKT are shown in Figure 5. It is obvious that our adaptable method is able to boost the performance on each dataset comparing with baselines. Moreover, there are some observations. First, without understanding the question texts, the results of the variant AKT-tx (has no texts) is similar with DKT except the slipping and guessing factors modeling parts. Second, AKT surpasses these comparison model by a large margin (e.g., 3.2% on AUC, 3.24% on F1), that because the rich question texts help train more parameters than one-hot representation. In general, the total observations prove that AKT can capture the semantic understanding of question texts effectively for boosting knowledge tracing performance.

In summary, our framework is helpful for the boosting knowledge tracing performance by modeling educational characteristics (e.g., slipping, guessing, question texts) for solving DAKT problem.

4.3 Transferring Results

In transferring experiments, for each target domain, we randomly select a small amount of data as the limited labeled part $I_{Tl}$, and the rest of it are used for validating.

4.3.1 Subjects Transferring. To validate that our AKT framework can achieve domain adaptation from source domain to target domain perfectly for DAKT problem, we adopt two subjects (e.g., math, physics) as two different domains by using datasets zx.math and zx.physics shown in Table 1 and term them as M and P respectively. Therefore, there are $P_s^2 = 2$ transferring tasks $M \rightarrow P$ and $P \rightarrow M$ that need to be conducted, where $\rightarrow$ means transferring from the left to the right. Since there no existing methods for DAKT problem, we compare AKT with its variant AKT-tr where “-tr” means training AKT on source domain and applied to target domain directly without doing any transferring process. Moreover, to determine what dimension our learned adaptation layer should have, we try different widths from 64 to 512, stepping by a power of two each time. The experimental results are shown in Figure 6, we can find that both AUC and F1 of AKT-tr are approximate 0.5, these performances of it just like random prediction, that because no information is transferred from source domain to target domain, even causing negative transfer [28]. However, both AUC and F1 of the adaptation method AKT are good on $M \rightarrow P$ and $P \rightarrow M$ tasks, especially, AKT performs best with 256 dimensions of adaptation layer. Totally, AKT is successful for addressing DAKT problem between subjects.

![Figure 5: Question texts effectiveness validation.](image)

![Figure 6: Evaluation of AKT for DAKT problem.](image)

| Schools | A | B | C | D |
|---------|---|---|---|---|
| # Questions | 708 | 694 | 611 | 764 |
| # Students | 66 | 58 | 62 | 51 |
| # Interactions | 4,673 | 4,066 | 3,892 | 4,398 |
4.3.2 Schools Transferring. To demonstrate the applicability of AKT for DAKT problem more completely. We further adopt four schools only selected from zy.math since zy.physics contains no information of school, and term them as A, B, C and D, the statistic of them are shown in Table 4. Therefore, there are $P_4 = 12$ transferring tasks between four schools that need to be conducted. The same as subjects transferring, we term $\rightarrow$ as transferring from the left to the right, adopt AKT-tr as the comparison baseline and validate the effects of different adaptation layer dimension. The experimental results are shown in Table 5 and Table 6, they are AUC and F1-score of the models for the 12 transferring tasks. We can find that the performance of AKT-tr is better than random prediction (e.g., 0.5), that because the questions in the four schools are all selected from the same subject dataset zy.math, they are similar to some extent. Second, it is obvious that AKT obtains better transferring results than AKT-tr on both AUC and F1-score, that proves the effectiveness of AKT for solving DAKT problems between schools.

4.3.3 Sensitivity Analysis. As presented in section 3.3, there are two critical regularization factors $\lambda$ and $\gamma$, the well selected values of them can help the framework performs better, therefore, we conduct some empirical experiments to evaluate the sensitivity of them by setting different values (e.g., 0.0, 0.25, 0.5, 0.75, 1.0) of them on $M \rightarrow P$ and $P \rightarrow M$ tasks. The experimental results are shown in Figure 7. We can find that an appropriate $\lambda$ is of great significance for the performance of AKT, such as $\lambda = 0.0$ means no source domain data are used, so the corresponding transferring results is not good, and the larger the $\lambda$ is, the more source domain data will be selected. The best value of $\lambda$ is around 0.5 where both AUC and F1 of AKT are good. Meanwhile, it is obvious that $\gamma$ also plays an important role in AKT, since it balances the training loss and domain discrepancy. $\gamma = 0.0$ means ignoring the domain discrepancy, that is the adaptation layer is abandoned rather than doing domain adaptation and the larger, the more important. Totally, the performance of AKT is sensitive to $\lambda$ and $\gamma$ to some extent, a good and patient selection of them will get ideal feedback.

### Table 5: The AUC of transferring information between four different schools which are sign as A, B, C and D.

| Model | $\phi$.dim | A $\rightarrow$ B | B $\rightarrow$ A | A $\rightarrow$ C | C $\rightarrow$ A | A $\rightarrow$ D | D $\rightarrow$ A | B $\rightarrow$ C | C $\rightarrow$ B | B $\rightarrow$ D | D $\rightarrow$ B | C $\rightarrow$ D | D $\rightarrow$ C |
|-------|-------------|------------------|------------------|-----------------|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| AKT-tr| 64          | 0.6576           | 0.6497           | 0.6482          | 0.6394          | 0.6511         | 0.6368         | 0.6401         | 0.6438         | 0.6381         | 0.6402         | 0.6432         | 0.6477         |
|       | 128         | 0.6704           | 0.6637           | 0.6599          | 0.6413          | 0.6549         | 0.6537         | 0.6526         | 0.6518         | 0.6402         | 0.6507         | 0.6501         | 0.6588         |
|       | 256         | 0.6924           | 0.6831           | 0.6816          | 0.6682          | 0.6670         | 0.6595         | 0.6645         | 0.6629         | 0.6523         | 0.6572         | 0.6547         | 0.6526         |
|       | 512         | 0.6816           | 0.6715           | 0.6689          | 0.6527          | 0.6602         | 0.6498         | 0.6489         | 0.6507         | 0.6536         | 0.6474         | 0.6409         | 0.6501         |

### Table 6: The F1-score of transferring information between four different schools which are sign as A, B, C and D.

| Model | $\phi$.dim | A $\rightarrow$ B | B $\rightarrow$ A | A $\rightarrow$ C | C $\rightarrow$ A | A $\rightarrow$ D | D $\rightarrow$ A | B $\rightarrow$ C | C $\rightarrow$ B | B $\rightarrow$ D | D $\rightarrow$ B | C $\rightarrow$ D | D $\rightarrow$ C |
|-------|-------------|------------------|------------------|-----------------|-----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|----------------|
| AKT   | 64          | 0.7169           | 0.7086           | 0.7185          | 0.7073          | 0.7115         | 0.7009         | 0.6989         | 0.7070         | 0.7101         | 0.7178         | 0.7016         | 0.7151         |
|       | 128         | 0.7232           | 0.7187           | 0.7224          | 0.7234          | 0.7247         | 0.7149         | 0.7120         | 0.7236         | 0.7215         | 0.7286         | 0.7203         | 0.7250         |
|       | 256         | 0.7338           | 0.7242           | 0.7305          | 0.7212          | 0.7160         | 0.7205         | 0.7208         | 0.7117         | 0.7232         | 0.7351         | 0.7162         | 0.7208         |
|       | 512         | 0.7291           | 0.7163           | 0.7269          | 0.7036          | 0.7093         | 0.7154         | 0.7092         | 0.7106         | 0.7197         | 0.7263         | 0.7172         | 0.7113         |
4.4 Discussion
From the above experiments, it is obvious that AKT can perfectly address the DAKT problem. Moreover, we can find that the transferring results between different schools are better than the results between different subjects, which because the subjects for different schools are the same, the question texts of them are more similar. On the contrary, the question texts of different subjects are absolutely different. Therefore, transferring during different schools is easier than different subjects. Also, AKT can be applied not only between different subjects and schools but grades, even different subjects between different schools. However, there is still some room for improvement. First, AKT is sensitive to the hyper-parameters λ and γ, how to model more reasonably, so that no need to manually set the λ and γ parameters is a problem worth studying. Second, the modeling process of slipping and guessing can be well designed in the future for more interpretable.

5 RELATED WORK
Generally, our related works can be summarized as three categories, i.e. knowledge tracing, slipping and guessing factors of questions and domain adaptation techniques.

5.1 Knowledge Tracing
Conventional knowledge tracing models such as BKT [9] and PFA [29] are the representative traditional approaches, they have been widely applied into many computer-aided education systems[32, 33]. The BKT model has been proposed during 1990s. It is a two-stage dynamic Bayesian network in which student performance is the observed variable and student knowledge is the latent data, and it assumes a single knowledge is tested per question. Specifically, BKT adopts HMM to trace the knowledge states for each knowledge concept, and the knowledge state is represented by the hidden state, which is a binary variable that indicates whether or not the student masters the knowledge concept. Many variants of BKT have been proposed to solve these problems, such as [14, 17]. The PFA is a reconfiguration of LFA [5] and offers higher sensitivity to student performance rather than student ability. It is proposed as an alternative to BKT, which relaxes the static knowledge assumption, models multiple knowledge concepts simultaneously with its basic structure and has the ability to handle the multiple knowledge concepts problems, but it cannot deal with the inherent dependency, e.g., “addition” is the prerequisite of “multiplication”, and cannot provide a deep insight into student’s latent knowledge state [41].

Since deep learning approaches outperform the traditional models in many areas (e.g., natural language processing), some researchers attempt to leverage deep learning techniques to predict knowledge state more precisely. The deep knowledge tracing (DKT) was the first attempt, to the best of our knowledge, to utilize recurrent neural networks (e.g., RNN and LSTM) to model student’s learning process for predicting her performance [30]. It outperforms conventional methods in most of the application scenes because of its nonlinear input-to-state and state-to-state transitions. Following DKT, there are amounts of studies propose the variant of it, such as DKT+forgetting [26] aggregates extra question information (e.g., forgetting) and DKT-tree [39] introduces the learning time to DKT.

5.2 Slipping and Guessing Factors
For more precisely simulating student learning process, it is necessary to completely model the situation, i.e. slipping and guessing that students may encounter when answering questions, which are strongly interpretable for student knowledge state. There are two mediating parameters (e.g., slipping, guessing) in BKT [11], the slipping parameter acknowledges that even student who masters a knowledge concept can make an occasional mistake, and the guessing parameter represents the fact that a student may sometimes generate a correct response, in spite of not knowing the correct knowledge concept. The item response theory (IRT) also adopts a parameter c to represent the guessing factors of items, which mirrors the probability that a student gets a right answer when his latent trait is low[31]. Wu et al. introduces slipping and guessing factors into cognitive diagnosis model (CDM) to compute examinees’ problem mastery and generates examinees’ observable scores on problems [43], and propose the gaming factor which can be seen as guessing factors for student learning modeling, to obtain more precise and reasonable cognitive analysis[44]. Totally, there are only several studies that exploit the slipping and guessing factors of the question, and almost have not been studied in deep knowledge tracing area.

5.3 Domain Adaptation Techniques
Domain adaptation is a branch of transferring learning, which focuses on dealing with the problem that the feature and label space are consistent respectively while the feature distribution is inconsistent. It has been widely applied in amounts of directions such as text classification [24] and object detection [6], etc. Up to now, there are many transfer learning techniques which can be categorized as instance based [36], feature based [23], model based [22] and relationship based [10] approaches. As one of the simplest transfer learning techniques, fine-tuning has been adopted to address some real problems, i.e. Daniel et al. introduce fine-tuning to train a deep learning model for optical coherence tomography images classification in cell journal [18]. However, to the best of our knowledge, there is no attempt to propose the domain adaptation for knowledge tracing (DAKT) problem. Therefore, we design a general transferable framework for knowledge tracing, this is the first time to do some attempts in DAKT problem by leveraging transfer learning techniques in this study.

6 CONCLUSION
In this paper, we addressed the issues of existing KT methods and studied the domain adaptation for knowledge tracing (DAKT) problem. We proposed an adaptable knowledge tracing (AKT) framework which integrates slipping and guessing factors into framework for more reasonable knowledge state modeling results, leverages the semantic understanding of question texts to replace the one-hot representation for more precise knowledge tracing, and contains instance selection via pre-training, domain discrepancy minimizing and fine-tuning three transferring steps for DAKT problem. Extensive experimental results demonstrated that AKT was successful for DAKT problem and superior to comparison methods for better KT results. We hope this study builds a solid baseline for DAKT problem to promote more researches in this field.
