Multi-task Learning for Low-resource Second Language Acquisition Modeling

Yong Hu[a], Heyan Huang[a,*], Tian Lan[a], Xiaochi Wei[c], Yuxiang Nie[a], Jiarui Qi[a], Liner Yang[b], Xian-Ling Mao[a]

[a]*Department of Computer Science, Beijing Institute of Technology*
[b]*Beijing Language and Culture University*
[c]*Baidu Inc.*

Abstract

Second language acquisition (SLA) modeling is to predict whether second language learners could correctly answer the questions according to what they have learned. It is a fundamental building block of the personalized learning system and has attracted more and more attention recently. However, as far as we know, almost all existing methods cannot work well in low-resource scenarios because lacking of training data. Fortunately, there are some latent common patterns among different language-learning tasks, which gives us an opportunity to solve the low-resource SLA modeling problem. Inspired by this idea, in this paper, we propose a novel SLA modeling method, which learns the latent common patterns among different language-learning datasets by multi-task learning and are further applied to improving the prediction performance in low-resource scenarios. Extensive experiments show that the proposed method performs much better than the state-of-the-art baselines in the low-resource scenario. Meanwhile, it also obtains improvement slightly in the non-low-resource scenario.

Keywords: low-resource, second language acquisition modeling, multi-task learning

1. Introduction

Knowledge tracing (KT) is a task of modeling how much knowledge students have obtained over time so that we can accurately predict how students will perform on future exercises and arrange study plans dynamically according to their real-time situations [1, 2]. Particularly, second language acquisition (SLA) modeling is a kind of KT in the filed of language learning. With the increasing

*Corresponding author
Email addresses: huyong@bit.edu.cn (Yong Hu), hhy63@bit.edu.cn (Heyan Huang), lantiangmftby@gmail.com (Tian Lan), weixiaochi@baidu.com (Xiaochi Wei), jerrrynie@gmail.com (Yuxiang Nie), Rita2663269@gmail.com (Jiarui Qi), lineryang@gmail.com (Liner Yang), xlmao@bit.edu.cn (Xian-Ling Mao)

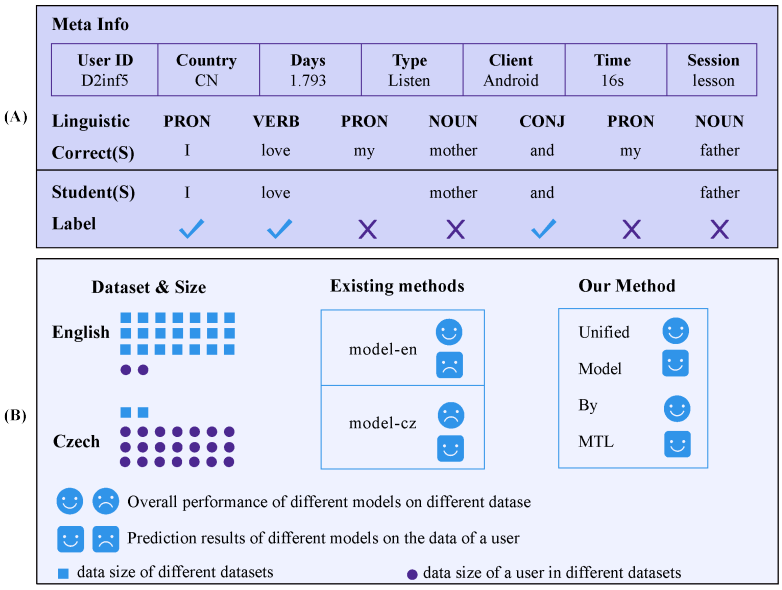

Figure 1: (A) Illustration of an example of SLA modeling task. (B) Illustration of two kinds of low-resource phenomenons and the comparison of our method and existing methods.

importance of language-learning activity in people's daily life [3], SLA modeling attracts more and more attention. For example, NAACL 2018 had held a public SLA modeling challenge.[1] Therefore, in this paper, we focus on SLA modeling.

SLA modeling is the learning process of a specific language, thus each SLA modeling task has a corresponding language, e.g., English, Spanish, and French. Meanwhile, each language is composed of many exercises, and an exercise is the smallest data unit. For an exercise, there are three possible types, i.e., *listen*, *Translation*, and *Reverse Tap*, and the answers to the exercises are all sentences regardless of the type of the exercise. In an exercise, a student will answer the given question and write its answer sentence. Then the student-provided sentence and the correct sentence will be compared word by word to evaluate the ability of the student. As shown in Fig. 1 (A), taking an English listening exercise as an example, the correct sentence is " *I love my mother and my father*", and the answer of the student is " *I love mader and fhader*"; It can be shown that there are three words that are correctly answered. Therefore, SLA modeling task is to predict whether students can answer each word correctly according to the exercise information (meta-information, correct sentence with corresponding linguistic information). Thus, it can be simply token into a word-level binary classification task.

In SLA modeling task, low-resource is a common phenomenon which affects

[1] http://sharedtask.duolingo.com/

the training process significantly. Specifically, this phenomena is mainly caused by two reasons: (1) For some specific language-learning datasets, e.g. Czech, the size of data may be very small becuse we cannot collect enough language-learning exercises; (2) For a user, he/she will encounter cold start scenario when starting to learn a new language. However, almost all existing methods for SLA modeling task train a model separately for each language-learning dataset and thus their performance largely depends on the size of training data. Thus, they can hardly work well in low-resource scenarios. Fig. 1 (B) illustrates an example. Suppose that we have two language: English and Czech, existing methods will train two separate models for these two languages: *model_en* and *model_cz*. These two models will perform poorly in two low-resource scenarios: (1) If the English dataset has a large amount of data, the *model_en* will perform well, but the small size of Czech dataset may significantly hinders the performance of *model_cz*; (2) Suppose that a user has a large number of exercises for learning Czech, but when he/she begins to learn English, the number of English exercises for him/her will be very small, even zero. Thus, *model_en* can hardly predict the answers of his/her English exercises well.

Intuitively, there are lots of common patterns among different language-learning tasks, such as the learning habits of users and grammar learning skills. If the latent common patterns across these language-learning tasks can be well learned, they can be used to solve the low-resource SLA modeling problem.

Inspired by this idea, in this paper, we propose a novel multi-task learning method for SLA modeling, which is a unified model to process several language-learning datasets simultaneously. Specifically, the proposed model learns shared features across all language-learning datasets jointly, which is the inner nature of the language-learning activity, and can be taken as important prior-knowledge to deal with small language-learning datasets. Moreover, the embedding information of a user is shared, so the learning habits and language talents of the user could be shared in the unified model for other low-resource language-learning tasks. Therefore, when a user begins to learn a new language, the unified model can work well even though there is no exercise data for this user.

The main contributions of this paper are three-fold. (1) As far as we know, this is the first work applying multi-task neural network to SLA modeling and we effectively solve the problem of insufficient training data in low-resource scenarios. (2) We deeply study the common patterns among different languages and reveal the inner nature of language learning. (3) Extensive experiments show that our method performs much better than the state-of-the-art baselines in low-resource scenarios, and it also obtains improvement slightly in the non-low-resource scenario. Additionally, we have publicly released our codes to facilitate follow-on researchers.[2]

[2]https://github.com/nghuyong/MTL-SLAM

2. Related Work

2.1. SLA Modeling

Existing methods for SLA modeling can be roughly divided into three categories: (a) logistic regression based methods, (b) tree ensemble methods, and (c) sequence modeling methods. (a) The logistic regression based methods [4, 5, 6] take the meta and context features provided by datasets and other manually constructed features as input and output the probability of answering each word correctly. These methods are simple but their performances are not very poor. (b) The tree ensemble methods (e.g., Gradient Boosting Decision Trees (GBDT)) [7, 8, 9] can powerfully capture non-linear relationships between features. Therefore, although the input and output of these methods are the same with (a), they are generally better than methods that belong to (a). (c) The sequence modeling methods (e.g., Recurrent Neural Networks (RNNs)) [10, 11, 12] use neural networks, especially RNNs so that they can capture users' performance over time. The performance of these methods are also very competitive.

However, methods above hardly can work well in low-resource scenarios because their performance largely depends on the size of training data.

2.2. Multi-Task Learning

Multi-task learning (MTL) has been widely used in various tasks, such as machine learning[13, 14, 15], natural language processing [16, 17, 18], speech recognition [19, 20, 21] and computer vision [22, 23, 24]. It effectively increases the sample size that we are using for training our model. Thus, it can improve generalization by leveraging the domain-specific information contained in related tasks, and enables the model to obtain a better sharing representation between each related task.

MTL is typically done with hard or soft parameter sharing of hidden layers and hard parameter sharing is the most commonly used approach to MTL in neural networks [25]. It is generally applied by sharing the hidden layers between all tasks, while keeping several task-specific output layers.

SLA modeling has different language-learning tasks, and each task has something in common, which gives us an opportunity to use MTL to improve the overall performance.

3. Model

3.1. Problem Definition

Suppose there are N second language-learning datasets $\{D^1, D^2, .., D^N\}$, and the k^{th} dataset D^k is composed of M^k exercises $\{e_1^k, e_2^k, ..., e_{M^k}^k\}$, where e_j^k is the j^{th} exercise in the k^{th} dataset.

There are two kinds of information in an exercise e_j^k, i.e., the meta information and the language related context information.

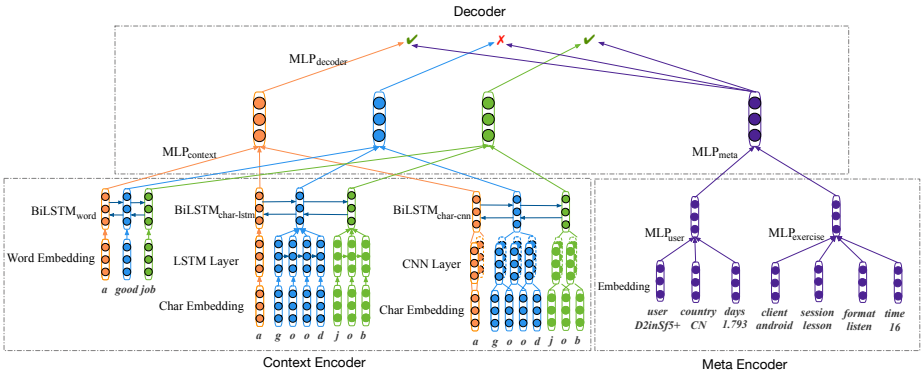

Figure 2: Illustration of our encoder-decoder structure

The meta information contains two user-related information: (1) user: the unique identifier for each student, e.g., *D2inf5*, (2) country: student's country, e.g., *CN*, and the following five exercise-related information: (1) days: the number of days since the student started learning this language, e.g., *1.793*, (2) client: the student's device platform, e.g., *android*, (3) session: the session type, e.g., *lesson*, (4) format (or type): exercise type, e.g., *Listen*, (5) time: the amount of time in seconds it took for the student to construct and submit the whole answer, e.g., *16s*. This is shared among all language datasets.

The information of the context in the exercise e_j^k includes the word sequence, that is $\{w^1_{e_j^k}, w^2_{e_j^k}, ..., w^l_{e_j^k}\}$, and word's linguistic sequences, such as $\{p^1_{e_j^k}, p^2_{e_j^k}, ..., p^l_{e_j^k}\}$, which is the POS-tagging of each word. This is unique to each language-learning dataset.

At last, e_j^k has a word level label sequence $\{y^1_{e_j^k}, y^2_{e_j^k}, ..., y^l_{e_j^k}\}$, where $y_{e_j^k} \in \{0, 1\}$. $y_{e_j^k} = 0$ means this word is answered correctly, and $y_{e_j^k} = 1$ means the opposite.

Our task is to build a model based on users' exercises, and further to predict word-level label sequence of future exercises.

3.2. Encoder and Decoder Structure

Our model is an encoder-decoder structure with two encoders, i.e., a meta encoder, a context encoder, and a decoder. We use the meta encoder to learn the non-linear relationship between meta information, use the context encoder to learn the representation of a sequence of words and use the decoder to generate the final prediction of each word. The overall structure of the proposed model is shown in Fig. 2.

Meta Encoder: The meta encoder is a multi-layer perceptron (MLP) based neural network. This encoder takes the metadata as inputs. First, these inputs are converted into high-dimensional representations by the embedding layers, which are randomly initialized and will map each input into a 150-dimensional

vector. After the embedding step, we separately concatenate the user-related embeddings and the exercise-related embeddings, and send them into MLP_{user} and $MLP_{exercise}$ to get the representation of user-related meta information r^{user} and the representation of exercise-related meta information $r^{exercise}$, respectively. Finally, we concatenate r^{user} and $r^{exercise}$, and send the concatenated result to MLP_{meta} to obtain the representation of whole meta information r^{meta}. The meta encoder can be formulated as

$$\begin{aligned} s &= [x^{user}, x^{countries}, x^{days}] \\ r^{user} &= MLP_{user}(s) \\ t &= [x^{format}, x^{session}, x^{client}, x^{time}] \\ r^{exercise} &= MLP_{exercise}(t) \\ r^{meta} &= MLP_{meta}([r^{user}, r^{exercise}]) \end{aligned} \tag{1}$$

where for the sake of simplicity, the variables are omitted from the subscript e_j^k, and $x^{(\cdot)}$ is the embedded representation of each meta information.

Context Encoder: The context encoder consists of three sub-encoders, i.e., a word level context encoder, a char level Long Short Term Memory (LSTM) context encoder, and a char level Convolutional Neural Network (CNN) context encoder. The word level encoder can capture better semantics and longer dependency than the character level encoders [10]. By modeling the character sequence, we can partially avoid the out-of-vocabulary (OOV) problem [26, 27]. Furthermore, we only use the word sequence in the datasets without using any of the provided linguistic information here. The previous work [8] has pointed out that the linguistic information given by the datasets has mistakes. So, through two character level encoders, we can learn certain word information and linguistic rules.

Given the word sequence $\{w^1_{e_j^k}, w^2_{e_j^k}, ..., w^l_{e_j^k}\}$, the word level context encoder is computed as

$$\begin{aligned} x_t &= Embedding^{word}(w_t) \\ (g_1, g_2.., g_l) &= BiLSTM_{word}(x_1, x_2, .., x_l) \end{aligned} \tag{2}$$

where w_t is the t^{th} word in the sequence, and $Embedding^{word}$ is the word embedding. Here, we use the pre-trained ELMo [28] as the look-up table. g_t is the concatenated result of the last layer's t^{th} hidden state of the forward and the backward cells of $BiLSTM_{word}$. It is also the output of the word level context encoder.

The char level LSTM context encoder is computed according to the sequence characters of word $w_t = \{c_1, c_2, ..., c_M\}$. This can be formulated as

$$\begin{aligned} m_i &= Embedding^{char}(c_i) \\ \hat{h}_{w_t} &= LSTM(m_1, m_2, .., m_l) \\ (\hat{g}_1, .., \hat{g}_l) &= BiLSTM_{char-lstm}(\hat{h}_{w_1}, .., \hat{h}_{w_l}) \end{aligned} \tag{3}$$

where $\hat{h}$ is the last hidden state of the last layer of $LSTM$. $\hat{g}_t$ is the concatenated result of the last layer's t^{th} hidden state of the forward and the backward cells of $BiLSTM_{char-lstm}$. It is also the output of the char level LSTM context encoder.

The char level CNN context encoder can be similarly formulated as

$$\begin{aligned} \tilde{h}_{w_t} &= CNN(m_1, m_2, .., m_l) \\ (\tilde{g}_1, ..., \tilde{g}_l) &= BiLSTM_{char-cnn}(\tilde{h}_{w_1}, ..., \tilde{h}_{w_l}) \end{aligned} \tag{4}$$

where $\tilde{h}$ is the result of CNN encoder. $\tilde{g}_t$ is the concatenated result of the last layer's t^{th} hidden state of the forward and the backward cells of $BiLSTM_{char-cnn}$. It is also the output of the char level CNN context encoder.

The final output of the context encoder is generated by a single-layer MLP, and the concatenation of g_t, $\hat{g}_t$ and $\tilde{g}_t$ is fed as the input. The process is formulated as

$$r_t^{context} = MLP_{context}([g_t, \hat{g}_t, \tilde{g}_t]) \tag{5}$$

where $r_t^{context}$ is the final context representation of the word w_t.

Decoder: The decoder takes the output of meta encoder r^{meta} and the output of context encoder $r_t^{context}$ as inputs, the prediction of word w_t is computed with a MLP. It is formulated as

$$p_t = MLP_{decoder}([r_t^{context}, r^{meta}]) \tag{6}$$

where the activation function of $MLP_{decoder}$ is sigmoid function.

3.3. Multi-Task Learning

As is shown in Fig. 3, suppose there are N languages, and each has a corresponding dataset, i.e., $\{D_1, D_2, ..., D_N\}$. Since our task is to predict the exercise accuracy of language learners on each language, we can regard these predictions as different tasks. Therefore, there are N tasks.

We defined the cross-entropy loss for each task, which encourages the correct predictions and punishes the incorrect ones. Specifically, for the k^{th} task, we have

$$\begin{aligned} Loss_{D_k} = -\frac{1}{N}\sum_{t=1}^{N}(&\alpha y_t \cdot log(p_t) \\ &+ (1-\alpha)(1-y_t) \cdot log(1-p_t)) \end{aligned} \tag{7}$$

where α is the hyper parameter to balance the negative and positive samples.

In multi-task learning, the parameters in meta encoder and decoder are shared, and each task only has its own parameters of the context encoder part, so the whole model has only one meta encoder, one decoder and N context encoders. In this way, the common patterns extracted from all language datasets can be utilized simultaneously by the shared meta encoder and decoder.

In the training process, one mini batch contains data of N datasets and they will all be sent to the same meta encoder and decoder, but will be sent to their

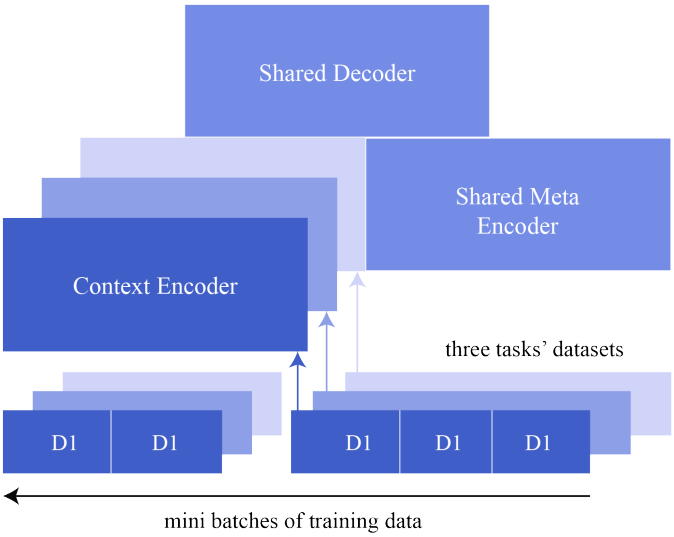

Figure 3: Illustration of multi-task learning

Table 1: The statistics of Duolingo SLA modeling dataset

	en_es	*es_en*	*fr_en*
#Exercises (Train)	824,012	731,896	326,792
#Exercises (Dev)	115,770	96,003	43,610
#Exercises (Test)	114,586	93,145	41,753
#Unique words	2,226	2,915	2,178
#Unique users	2,593	2,643	1,213
#words / exercise	3.18	2.7	2.84
%OOV radio (Test)	4.5%	10.0%	5.9%
%Correct radio	87%	86%	84%
%Incorrect radio	13%	14%	16%

corresponding context encoder according to their language type. Thus, the final loss with N tasks is calculated as

$$Loss_{final} = \sum_{k=1}^{N} Loss_{D_k} \tag{8}$$

Finally, we use Adam algorithm [29] to train the model.

4. Experiments

4.1. Datasets and Settings

We conduct experiments on Duolingo SLA modeling shared datasets, which have three datasets and are collected from English students who can speak

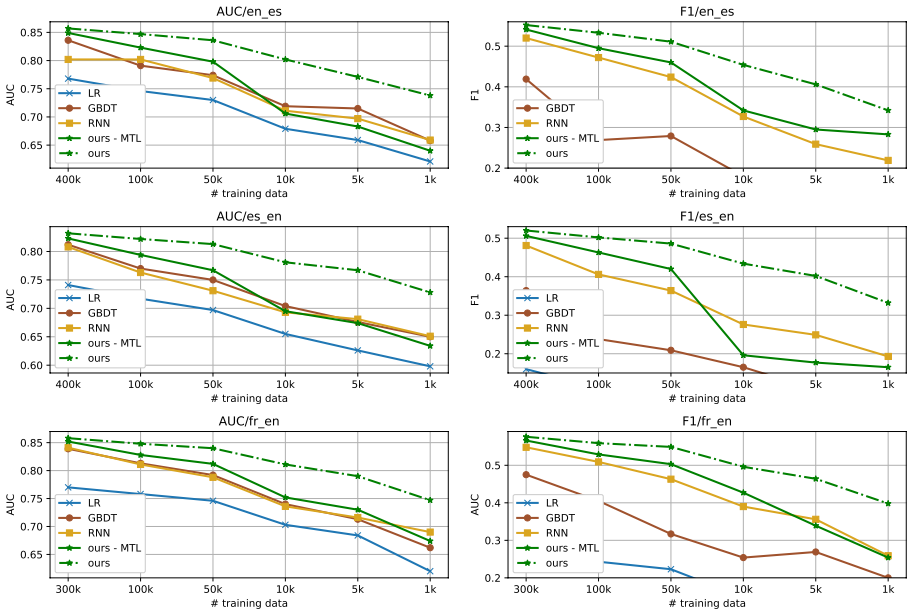

Figure 4: Comparison of our method and baselines on training data of different sizes

Spanish (*en_es*), Spanish students who can speak English (*es_en*), and French students who can speak English (*fr_en*) [30]. Table 1 shows basic statistics of each dataset.

We compare our method with the following state-of-the-art baselines:

- LR Here, we use the official baseline provided by Duolingo [30]. It is a simple logistic regression using all the meta information and context information provided by datasets.
- GBDT Here, we use NYU's method [8], which is the best method among all tree ensemble methods. It uses an ensemble of GBDTs with existing features of dataset and manually constructed features based on psychological theories.
- RNN Here, we use singsound's method [31], which is the best method among all sequence modeling methods. It uses an RNN architecture which has four types of encoders, representing different types of features: token context, linguistic information, user data, and exercise format.
- ours-MTL It is our encoder-decoder model **without** multi-task learning. Thus, we will separately train a model for each language-learning dataset.

In the experiments, the embedding size is set to 150 and the hidden size is also set to 150. Dropout [32] regularization is applied, where the dropout rate is set to 0.5. We use the Adam optimization algorithm with a learning rate of 0.001.

Table 2: Comparison of our method with existing methods on different language datasets

Methods	*en_es*		*es_en*		*fr_en*	
	AUC	F_1	AUC	F_1	AUC	F_1
LR [30]	0.774	0.190	0.746	0.175	0.771	0.281
GBDT[8]	0.859	0.468	0.835	0.420	0.854	0.493
RNN [10]	0.861	0.559	0.835	0.524	0.854	0.569
GBDT+RNN [31]	0.861	0.561	0.838	**0.530**	0.857	0.573
ours-MTL	0.863	**0.564**	0.837	0.527	0.857	0.575
ours	**0.864**	**0.564**	**0.839**	**0.530**	**0.860**	**0.579**

Table 3: Comparison of encoder removal

Methods	*en_es*		*es_en*		*fr_en*	
	AUC	F_1	AUC	F_1	AUC	F_1
ours - meta encoder	0.743	0.353	0.716	0.320	0.750	0.478
ours - word level context encoder	0.862	0.559	0.838	0.526	0.858	0.575
ours - char level LSTM context encoder	0.863	0.563	0.838	0.526	0.860	0.579
ours - char level CNN context encoder	0.863	0.564	0.838	0.528	0.860	0.559
ours - char level context encoder all	0.863	0.562	0.838	0.526	0.859	0.579
ours	**0.864**	**0.564**	**0.839**	**0.530**	**0.860**	**0.579**

4.2. Metric

SLA modeling is actually the word level classification task, so we use area under the ROC curve (AUC) [33] and F_1 score [34] as evaluation metric.

- AUC is calculated as:

$$AUC = P(s(x_1) > s(x_2)) \tag{9}$$

  where $P(\cdot)$ is the probability, $s(\cdot)$ is the trained classifier, x_1 is the instance randomly extracted from positive samples, and x_2 is the instance randomly extracted from negative samples.

- F_1 is calculated as

$$F_1 = 2 \times \frac{precision * recall}{precision + recall} \tag{10}$$

  where *precision* and *recall* are the precision rate and recall rate of the trained model.

4.3. Experiment on Small-scale Datasets

We first verify the advantages of our method in cases where the training data of the whole language-learning dataset is insufficient.

Specifically, we gradually decrease the size of training data from 400K (300K for *fr_en*) to 1K and keep the development set and test set. For all

Table 4: The statistics of two users (the following number is the number of words in exercises)

User	Dataset	Train	Dev	Test
RWDt7srk	*es_en*	361	68	19
	fr_en	519	80	51
t6nj6nr/	*es_en*	562	245	274
	fr_en	998	0	0

Table 5: Comparison of our method and baselines in the cold start scenario

Methods	AUC	F_1
LR [30]	0.765	0.083
GBDT [8]	0.751	0.187
RNN [31]	0.771	0.276
ours-MTL	0.770	0.210
ours	**0.881**	**0.411**

baseline methods, since they only use the single language dataset for training, we hence only reduce the data of corresponding language data. For our multi-task learning method, we reduce the training data of one language dataset and keep the remaining other two datasets unchanged.

The experimental results are shown in Fig. 4. It can be found that our method outperforms all the state-of-the-art baselines when the training data of a language dataset is insufficient, which is a huge improvement compared with the existing methods. For example, as shown in *AUC/en_es* in Fig. 4, using only 1K training data, our multi-task learning method still could get the AUC score of 0.738, while the AUC score of ours-MTL is only 0.640, and existing RNN, GBDT and LR methods are 0.659, 0.658 and 0.650 respectively. Therefore, the performance of introducing the multi-task learning **increases by nearly ten percent**. Moreover, to achieve the same performance as our multi-task learning on 1K training data, the methods without multi-task learning require more than 10K training data, which is **ten times more than ours**. Thus, multi-task learning utilizes data from all language-learning datasets simultaneously and effectively alleviate the problem of lacking data in a single language-learning dataset.

At the same time, we notice that ours-MTL is slightly worse than the RNN and GBDT when the amount of training data is very small (1K, 5K, 10K). This is because our model does not utilize the linguistic related features of the dataset, and the deep model will be over-fitting when the amount of training data is insufficient. However, as the training data improves (>10K), ours-MTL becomes better than the existing RNN and GBDT. Thus, our encoder-decoder structure is very competitive with existing methods even without multi-task learning.

4.4. Experiment in the Cold Start Scenario

Further, we can consider directly predicting a user's answer on a language without any training exercises of this user on this language at all. This is cold start scenario and also the situation that the language-learning platforms must consider.

Specifically, it can be found that user *RWDt7srk* and *t6nj6nr/* are all English speakers and learn both Spanish and French, so they have data both in the dataset *es_en* and *fr_en*. The statistics are shown in Table 4. For baseline methods, we remove the data of these two users on the training set as well as development set of *es_en*, and then train a model. At last, we use the trained model to directly predict the data of this two users on the *es_en* test set. Similarly, we use our multi-task method to do the same experiment, and the training data of these two users is also removed from the *es_en* data set, but *fr_en* and *en_es* are unchanged.

The experimental results are shown in Table 5. If we do not use multi-task learning to predict the new users directly, the performance will be very poor. Compared with the method without multi-task learning, such as ours-MTL, our multi-task learning method increases by **11%** on ACU and **20%** on F_1. Because of the multi-task learning, the user information of these two users has been learned through the *fr_en* dataset. Therefore, although there is no training data of these two users on *es_en*, we can still obtain good performance with mult-task learning.

4.5. Experiment in the Non-low-resource Scenario

The experiments above show that our method has a huge advantage over the existing methods in low-resource scenarios. In this section, we will observe the performance of our method in the non-low-resource scenario.

Specifically, we use all the data on the three language datasets to compare our methods with existing methods. This experiment is exactly 2018 public SLA modeling challenge held by Duolingo.[3] Here, we add a new baseline GBDT+RNN. This is SanaLabs's method [31] which combines the prediction of a GBDT and an RNN, and it is also the current best method on the 2018 public SLA modeling challenge.

As shown in Table 2, it can be found that although the improvement is not very big, our method surpasses all existing methods on all three datasets and refreshes the best scores on all three datasets. Especially for the smallest dataset *fr_en*, our method obtains the most improvement than ours-MTL. As for the largest dataset *en_es*, our method also improves the AUC score by 0.003 over the best existing method GBDT+RNN. Therefore, our method also gains improvement slightly in the non-low-resource scenario.

[3] http://sharedtask.duolingo.com/

5. Model Analysis

5.1. Component Analysis

Our encoder-decoder structure contains two encoders, i.e., meta encoder and context encoder, where the context encoder includes three encoders, i.e., word level context encoder, char level LSTM context encoder and char level CNN context encoder. In order to explore the importance of each encoder, we do a component removal analysis experiment.

Specifically, we remove each encoder component, train a model, and record the performance on test set. We also remove both two char level context encoders and do the same experiment.

The experimental results are shown in Table 3. It can be found that the meta information is critical to the final result, much more important than the context encoder. If the meta encoder is removed, the result will be sharply reduced. The reason is that: if there is only a context encoder, it is equal to modeling the global word error distribution, completely ignoring the individual's situation, which violates adaptive learning.

For context encoder, word level encoder has a greater impact than char level encoder on the performance of our model.

5.2. Metadata Analysis

The analysis above has proven that meta information is important for predicting results. Obviously, different features of meta information have different influence. Therefore, feature removal analysis is made to find important features. Specifically, we remove each meta feature and get the performance of the model without this feature.

As shown in Fig. 5, the most important feature is the user (id). Without user (id), the model performance declines rapidly, because user information is the key to building user-adaptive learning. This also shows that the most common pattern between learning different languages is the students themselves. Besides, it can be found that learning format and spent time also make significant influences on the model.

5.3. Visualization

In this part, we will show what meta encoder has learned from three datasets by multi-task learning.

We cluster the user embedding with k-means algorithm ($k = 4$), and calculate the average accuracy of each user and the overall average accuracy of each cluster. Embeddings are processed by t-SNE [35] for visualization, as shown in Fig. 6, every point represents a user and its color represents the average accuracy of this user. Red means low accuracy and blue means high. The four large points indicate the center of clustering, and the value pointing to the point is the overall average accuracy of the corresponding cluster. It can be found that students with good grades and students with poor grades can be distinguished very well according to their user embeddings, so the user embedding trained by our model contains rich information for the final prediction.

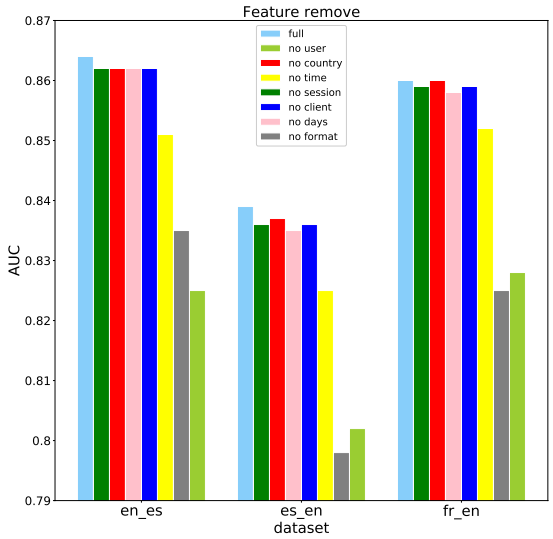

Figure 5: Analysis of meta features removal

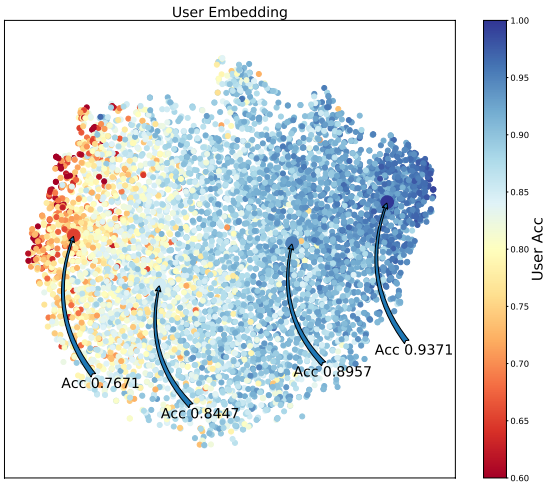

Figure 6: User embedding cluster

6. Conclusion

In this paper, we have proposed a novel multi-task learning method for SLA modeling. As far as we know, this is the first work applying multi-task neural network to SLA modeling and study the common patterns among different languages. Extensive experiments show that our method performs much better than the state-of-the-art baselines in low-resource scenarios, and it also obtains improvement slightly in the non-low-resource scenario.

7. Acknowledgments

The work is supported by NKRD(No. 2018YFB1005100), NSFC (No. 61772076 and 61751201), NSFB (No. Z181100008918002), Major Project of Zhijiang Lab (No. 2019DH0ZX01), and Open fund of BDAlGGCNEL and CETC Big Data Research Institute Co., Ltd (No. w-2018018).

References

[1] K. Bauman, A. Tuzhilin, Recommending learning materials to students by identifying their knowledge gaps., in: RecSys Posters, 2014.

[2] R. Pelánek, Bayesian knowledge tracing, logistic models, and beyond: an overview of learner modeling techniques, User Modeling and User-Adapted Interaction 27 (3-5) (2017) 313–350.

[3] D. Larsen-Freeman, M. H. Long, An introduction to second language acquisition research, Routledge, 2014.

[4] S. Klerke, H. M. Alonso, B. Plank, Grotoco@ slam: Second language acquisition modeling with simple features, learners and task-wise models, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 206–211.

[5] N. V. Nayak, A. R. Rao, Context based approach for second language acquisition, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 212–216.

[6] Y. Bestgen, Predicting second language learner successes and mistakes by means of conjunctive features, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 349–355.

[7] B. Tomoschuk, J. Lovelett, A memory-sensitive classification model of errors in early second language learning, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 231–239.

[8] A. Rich, P. O. Popp, D. Halpern, A. Rothe, T. Gureckis, Modeling second-language learning from a psychological perspective, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 223–230.

[9] G. Chen, C. Hauff, G.-J. Houben, Feature engineering for second language acquisition modeling, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 356–364.

[10] S. Xu, J. Chen, L. Qin, Cluf: a neural model for second language acquisition modeling, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 374–380.

[11] Z. Yuan, Neural sequence modelling for learner error prediction, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 381–388.

[12] M. Kaneko, T. Kajiwara, M. Komachi, Tmu system for slam-2018, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 365–369.

[13] Y. Liu, R. Song, R. Bucknall, X. Zhang, Intelligent multi-task allocation and planning for multiple unmanned surface vehicles (usvs) using self-organising maps and fast marching method, Information Sciences 496 (2019) 180–197.

[14] H. He, L. Du, Y. Liu, J. Ding, Similarity preserving multi-task learning for radar target recognition, Information Sciences 436 (2018) 388–402.

[15] Y. Jiang, Z. Deng, K.-S. Choi, F.-L. Chung, S. Wang, A novel multi-task tsk fuzzy classifier and its enhanced version for labeling-risk-aware multi-task classification, Information Sciences 357 (2016) 39–60.

[16] R. Collobert, J. Weston, A unified architecture for natural language processing: Deep neural networks with multitask learning, in: Proceedings of the 25th international conference on Machine learning, ACM, 2008, pp. 160–167.

[17] P. Liu, X. Qiu, X. Huang, Recurrent neural network for text classification with multi-task learning, arXiv preprint arXiv:1605.05101.

[18] D. Dong, H. Wu, W. He, D. Yu, H. Wang, Multi-task learning for multiple language translation, in: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), Vol. 1, 2015, pp. 1723–1732.

[19] L. Deng, G. Hinton, B. Kingsbury, New types of deep neural network learning for speech recognition and related applications: An overview, in: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, IEEE, 2013, pp. 8599–8603.

[20] S. Kim, T. Hori, S. Watanabe, Joint ctc-attention based end-to-end speech recognition using multi-task learning, in: 2017 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, 2017, pp. 4835–4839.

[21] Z. Wu, C. Valentini-Botinhao, O. Watts, S. King, Deep neural networks employing multi-task learning and stacked bottleneck features for speech synthesis, in: 2015 IEEE international conference on acoustics, speech and signal processing (ICASSP), IEEE, 2015, pp. 4460–4464.

[22] Y. Chen, D. Zhao, L. Lv, Q. Zhang, Multi-task learning for dangerous object detection in autonomous driving, Information Sciences 432 (2018) 559–571.

[23] W. Guo, G. Chen, Human action recognition via multi-task learning base on spatial–temporal feature, Information Sciences 320 (2015) 418–428.

[24] Z. Zhang, P. Luo, C. C. Loy, X. Tang, Facial landmark detection by deep multi-task learning, in: European conference on computer vision, Springer, 2014, pp. 94–108.

[25] S. Ruder, An overview of multi-task learning in deep neural networks, CoRR abs/1706.05098. arXiv:1706.05098.
URL http://arxiv.org/abs/1706.05098

[26] M.-T. Luong, I. Sutskever, Q. V. Le, O. Vinyals, W. Zaremba, Addressing the rare word problem in neural machine translation, arXiv preprint arXiv:1410.8206.

[27] M. Ballesteros, C. Dyer, N. A. Smith, Improved transition-based parsing by modeling characters instead of words with lstms, arXiv preprint arXiv:1508.00657.

[28] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in: Proc. of NAACL, 2018.

[29] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980.

[30] B. Settles, C. Brust, E. Gustafson, M. Hagiwara, N. Madnani, Second language acquisition modeling, in: Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications, 2018, pp. 56–65.

[31] A. Osika, S. Nilsson, A. Sydorchuk, F. Sahin, A. Huss, Second language acquisition modeling: An ensemble approach, arXiv preprint arXiv:1806.04525.

[32] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, R. Salakhutdinov, Dropout: a simple way to prevent neural networks from overfitting, The Journal of Machine Learning Research 15 (1) (2014) 1929–1958.

[33] J. A. Hanley, B. J. McNeil, The meaning and use of the area under a receiver operating characteristic (roc) curve., Radiology 143 (1) (1982) 29–36.

[34] C. Goutte, E. Gaussier, A probabilistic interpretation of precision, recall and f-score, with implication for evaluation, in: European Conference on Information Retrieval, Springer, 2005, pp. 345–359.

[35] L. v. d. Maaten, G. Hinton, Visualizing data using t-sne, Journal of machine learning research 9 (Nov) (2008) 2579–2605.