Strategize Before Teaching: A Conversational Tutoring System with Pedagogy Self-Distillation

Lingzhi Wang[1,2], **Mrinmaya Sachan**[3], **Xingshan Zeng**[4], **Kam-Fai Wong**[1,2]
[1]The Chinese University of Hong Kong, Hong Kong, China
[2]MoE Key Laboratory of High Confidence Software Technologies, China
[3]Department of Computer Science, ETH Zurich
[1,2]{lzwang,kfwong}@se.cuhk.edu.hk
[3]msachan@ethz.ch, [4]zxshamson@gmail.com

Abstract

Conversational tutoring systems (CTSs) aim to help students master educational material with natural language interaction in the form of a dialog. CTSs have become a key pillar in educational data mining research. A key challenge in CTSs is to engage the student in the conversation while exposing them to a diverse set of teaching strategies, akin to a human teacher, thereby, helping them learn in the process. Different from previous work that generates responses given the strategies as input, we propose to jointly predict teaching strategies and generate tutor responses accordingly, which fits a more realistic application scenario. We benchmark several competitive models on three dialog tutoring datasets and propose a unified framework that combines teaching response generation and pedagogical strategy prediction, where a self-distillation mechanism is adopted to guide the teaching strategy learning and facilitate tutor response generation. Our experiments and analyses shed light on how teaching strategies affect dialog tutoring.

1 Introduction

Decades of research effort (Carbonell, 1970; Richardson, 1988; Brown, 2009) has been put into building intelligent tutoring systems (ITSs). An important feature of these systems is the ability to customize the instructional activities and strategies based on the learner's characteristics and needs (Keleş et al., 2009). Conversational tutoring systems (CTSs) that aim to offer automated tutoring through natural language dialog is a key pillar of ITS research. Earlier work in CTSs was based on conventional techniques such as Bayesian techniques with rule engines (Jeon and Su, 2010; Weragama and Reye, 2014) and hybrid neural networks (Kose and Arslan, 2017; Stasaski et al., 2020). While various advanced neural approaches have been applied to open-domain (Sordoni et al., 2015; Serban et al., 2016; Xing et al., 2017) and task-oriented dialogue systems (Zhao et al., 2017; Lei et al., 2018; Peng et al., 2020), conversational tutoring systems have not benefited from the development of these technologies (Macina et al., 2023).

Teaching Strategy	**Tutor Response**
Restating	Let me say back what I heard.
Pressing for accuracy	Can you tell us the steps you used to find the answer?

(a) Two examples of teaching strategy and tutor response

Tutor:	Ok, now we have 'get cut off', 'put someone through' and 'get through'	scaffolding
Tutor:	I was talking to her, but suddenly we ___ ___ ___ (I couldn‘t hear her anymore) Can you choose one?	eliciting
Student:	got cut off	
Tutor:	yes, good! Well done today! Have a lovely day!	closing

(b) An example of interactions between tutor and student

Figure 1: Examples of teaching strategy and interactions between tutor and student. Teaching strategies in Figure 1(b) are in red.

Human teachers use a number of nuanced teaching strategies in the classroom during interactions with students; these strategies are tailored to keep the students engaged in the conversation and learn knowledge efficiently. We show some examples of teaching strategies and interactions between the tutor and the student in Fig. 1. Previous work has attempted to model these teaching strategies in different ways – e.g., Suresh et al. (2019) contributed a teaching strategy classification model and Stasaski et al. (2020) proposed a response generation model based on given teaching strategies of next response.

In this work, we benchmark several neural dialog models on three conversational tutoring datasets, CIMA (Stasaski et al., 2020), TSCC (Caines et al., 2020) and TalkMoves (Suresh et al., 2019, 2022), and contribute a unified framework based on pretrained language models, where teaching strat-

Findings of the Association for Computational Linguistics: EACL 2023, pages 2268–2274
May 2-6, 2023 ©2023 Association for Computational Linguistics

egy prediction and response generation are jointly trained. As predicting a teaching strategy merely by the historical context is more difficult than when we are also given the target tutor response, we also propose a pedagogy distillation mechanism that allows teaching strategy prediction to learn from the soft labels which are produced by the prediction with target response. The soft labels learned from the target response provides the model knowledge about various interrelationships between teaching strategies that hard labels lack. This approach is believed to be able to alleviate the learning difficulty (Hinton et al., 2015), which is particularly important, especially when the data imbalance and scarcity issues are severe – often the case in conversational tutoring data.

In summary, we are the first to benchmark[1] several competitive models for conversation tutoring system on all three datasets that are currently available. Besides, we propose a unified framework that can predict teaching strategy and generate tutoring responses accordingly, which is enhanced by a self-distillation mechanism. Our experiments validate the positive effects of teaching strategy to guide generation and the importance of predicting strategy first and then generate response accordingly.

2 Related Work

A classical Intelligent Tutoring System generally has three modules (Brown, 2009; Polson and Richardson, 2013): (i) expert module that includes the knowledge that the student wants to learn (Carter, 2014)). (ii) student module that can adjust the level of student (e.g., primary/middle school, non-native/native speaker), student's knowledge deficiency, etc. (iii) pedagogical module that focuses on the strategies of teaching. In expert module, the knowledge is usually domain specific, such as computer programming (Costello, 2012), mathematics (Grawemeyer et al., 2016; Suresh et al., 2022), Italian (Stasaski et al., 2020), English (Caines et al., 2020). Many technologies have been used in the expert module, such as Bayesian techniques with rule engines (Jeon and Su, 2010; Weragama and Reye, 2014) and hybrid neural networks (Kose and Arslan, 2017; Stasaski et al., 2020). For pedagogical module, to our best knowledge, there are only three publicly available datasets that provide the pedagogy information. They are CIMA (Stasaski et al., 2020), TSCC (Caines et al., 2020) and TalkMoves (Suresh et al., 2022) datasets and all of them are based on single pedagogy. There has been very little work on neural dialog tutoring. Two exceptions to this are Suresh et al. (2022), who propose a simple BiLSTM-based module to predict the pedagogy of the next sentence that teachers are meant to say, and Stasaski et al. (2020) who use various generative models to generate responses given the pedagogical strategies. In contrast, in this work, we propose a joint approach for modelling the pedagogy and response generation that outperforms the previous approaches using a novel pedagogy distillation mechanism.

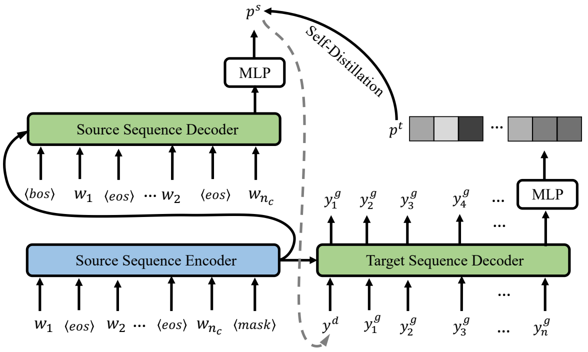
p^s
Self-Distillation
MLP
Source Sequence Decoder
p^t
⟨bos⟩ w_1 ⟨eos⟩ ··· w_2 ⟨eos⟩ w_{n_c}
y_1^g y_2^g y_3^g y_4^g ···
MLP
Source Sequence Encoder
Target Sequence Decoder
w_1 ⟨eos⟩ w_2 ··· ⟨eos⟩ w_{n_c} ⟨mask⟩
y^d y_1^g y_2^g y_3^g ··· y_n^g

Figure 2: Our overall framework. The self-distillation leverages predictions based on target to improve predictions based on source. The enhanced strategy prediction is further utilized to facilitate the generation.

3 Our Model

3.1 Problem Formulation

Our conversational tutoring system takes conversation context C and teaching strategy list D as input. C is formalized as a sequence of turns $\{t_1, t_2, ..., t_{n_c}\}$ where n_c represents the number of turns. t_i $(1 \leq i \leq n_c)$ denotes the i-th turn of the conversation, and we use $\boldsymbol{w}_i$ to indicate the word tokens contained in it. The teaching strategy list D covers all the possible strategies and contain n_d teaching strategies. Our model will first output one or several strategy labels, each $y^d \in \{1, 2, ..., n_d\}$, to indicate what teaching strategy to use. Then the generation module generates a target response $y^t = (y^t_1, \ldots, y^t_{n_t})$ based on the predicted strategy.

3.2 Conversational Tutoring System (CTS)

PLM-based Generation Module. The generation module follows a Transformer (Vaswani et al., 2017) sequence-to-sequence framework. As the currently available tutoring datasets are quite small

[1] The code can be found in https://github.com/Lingzhi-WANG/TutorSystem

(containing about 3k conversations), we choose to finetune pretrained language models (PLM) to alleviate data scarcity and enhance context modeling. We finetune BART (Lewis et al., 2020) and multilingual BART(mBART) (Liu et al., 2020) models for our generation module. During finetuning, we concatenate the utterances t_i $(1 \leq i \leq n_c)$ in context C with appended $\langle\text{eos}\rangle$ tokens in their chronological order as input, and maximize the probability of the ground-truth target sequence. The whole process is summarized as follows:

$$\boldsymbol{H}^c = \text{Transformer\_Encoder}(\boldsymbol{w}^c) \tag{1}$$

$$y_k^t = \text{Transformer\_Decoder}(y_{<k}^t, \boldsymbol{H}^c) \tag{2}$$

$$\mathcal{L}_{target}^{gen} = \sum_{k=1}^{n_t} -\log(p(y_k^t | y_{<k}^t, \boldsymbol{H}^c)) \tag{3}$$

where $\boldsymbol{w}^c = [\boldsymbol{w}_1; \langle\text{eos}\rangle; \boldsymbol{w}_2; ..; \boldsymbol{w}_{n_c}; \langle\text{mask}\rangle]$, and $y_{<k}^t$ represents the target tokens before y_k^t. We add $\langle\text{mask}\rangle$ at the end of context, to simulate the operation in pre-training (Schick and Schütze, 2021).

Besides, to summarize the representation of the conversation context, we employ an additional source sequence decoder as follows:

$$y_k^s = \text{Transformer\_Decoder}(y_{<k}^s, \boldsymbol{H}^c) \tag{4}$$

$$\mathcal{L}_{source}^{gen} = \sum_{k=1}^{n_s} -\log(p(y_k^s | y_{<k}^s, \boldsymbol{H}^c)) \tag{5}$$

where $y_{<k}^s$ represents the source tokens before y_k^s.

Teaching Strategy Prediction Module. We use the representation of the $\langle\text{eos}\rangle$ token (i.e. the final token) produced by the decoder as the representation for teaching strategy prediction, denoted as $\boldsymbol{h}^{\langle\text{eos}\rangle}$. This is fed into a two-layer MLP for prediction:

$$\boldsymbol{r}^d = \boldsymbol{W}_2 \times \alpha(\boldsymbol{W}_1 \boldsymbol{h}^{\langle\text{eos}\rangle} + \boldsymbol{b}_1) + \boldsymbol{b}_2 \tag{6}$$

where $\boldsymbol{W}_1$, $\boldsymbol{W}_2$, $\boldsymbol{b}_1$ and $\boldsymbol{b}_2$ are learnable parameters, and α is a non-linear activation function. The output representation $\boldsymbol{r}^d$ will be an n_d-dimension vector and the probability for each teaching strategy in list D is computed based on $\boldsymbol{r}^d$:

$$p(y^d = j) = \text{softmax}(\boldsymbol{r}^d)_j \tag{7}$$

where y^d denotes the predicted strategy and $j \in \{1, 2, ..., n_d\}$.

We denote $\boldsymbol{h}^{\langle\text{eos}\rangle}$ produced by source and target generation as $\boldsymbol{h}_s^{\langle\text{eos}\rangle}$ and $\boldsymbol{h}_t^{\langle\text{eos}\rangle}$, respectively. With $\boldsymbol{h}_s^{\langle\text{eos}\rangle}$, it means that we predict the teaching strategy without knowing the corresponding content; while with $\boldsymbol{h}_t^{\langle\text{eos}\rangle}$, we summarize the teaching strategy based on the target content. Obviously, predicting with $\boldsymbol{h}_s^{\langle\text{eos}\rangle}$ is what we need, but this is quite challenging. Thus we design a self-distillation mechanism which uses prediction based on $\boldsymbol{h}_t^{\langle\text{eos}\rangle}$ for enhancing the generation model.

Teaching Strategy Enhancement with Distillation. We denote the predicted probability for each strategy (derived with Eq. 7) using $\boldsymbol{h}_s^{\langle\text{eos}\rangle}$ and $\boldsymbol{h}_t^{\langle\text{eos}\rangle}$ as $p_s(\cdot)$ and $p_t(\cdot)$, respectively. Our self-distillation is defined as guidance from $p_t(\cdot)$ to $p_s(\cdot)$:

$$\mathcal{L}^{sd} = -\sum_{j=1}^{n_d} p_s(y^d = j) \log p_t(y^d = j) \tag{8}$$

where we define $p_t(\cdot)$ as teacher distribution and $p_s(\cdot)$ as student distribution, and Eq. 8 makes the student distribution similar to the teacher distribution. In this way, our teaching strategy prediction model can also learn from the soft labels produced by the target sequence.

Multiple Teaching Strategies Guided Generation. To guide the response generation with teaching strategy, we regard the teaching strategies as prompt tokens and display them at the beginning of generation. In this way, the target tokens will be generated autoregressively according to the giving teaching strategy. Specifically, during training, we use the ground-truth strategy (denoted as d^c, and it will be masked in distillation to avoid information leakage) for teacher forcing (i.e. $y_0^t = d^c$ in Eq. 3); during inference, we use the predicted strategies produced by the prediction module as prompt tokens.

To enable multiple teaching strategies guidance, we define a threshold θ, where all the strategies satisfying $p_s(y^d = j) \geq \theta$ $(1 \leq j \leq n_d)$ will be used to guide the response generation. To that end, we weightedly sum over the embeddings of those strategies as prompt based on their predicted probabilities produced by Eq. 7 and then use it to guide the generation.

3.3 Learning Objectives

The learning objective for teaching strategy prediction is defined as follows:

$$\mathcal{L}^{pred} = -(\log p_s(y^d = d^c) + \log p_t(y^d = d^c)) + \lambda \cdot \mathcal{L}^{sd} \tag{9}$$

where d^c is the ground-truth strategy for context C and λ is a hyper-parameter to control the weights of self-distillation loss. Our model is jointly trained on both generation and prediction, with the overall objective summarized as:

$$\begin{aligned}\mathcal{L} &= \mathcal{L}^{gen} + \gamma \cdot \mathcal{L}^{pred} \\ &= \mathcal{L}^{gen}_{target} + \delta \cdot \mathcal{L}^{gen}_{source} + \gamma \cdot \mathcal{L}^{pred}\end{aligned} \tag{10}$$

where δ and γ are tradeoff hyper-parameters.

4 Experimental Setup

Datasets. We use three datasets to do the experiments. They are CIMA (Stasaski et al., 2020), TSCC (Caines et al., 2020) and TalkMoves (Suresh et al., 2019, 2022). CIMA contains one-to-one conversations that focus on teaching students to translate a phrase from English to Italian. TSCC focuses on teaching English for eight non-native English-speaking students. TalkMoves is constructed by transcripts of math classrooms.

Parameter Setting. Our implementation is based on Fairseq (Ott et al., 2019). We split the data into 8:1:1 for training, validation and test. All the hyper-parameters are chosen by grid-search based on the validation performance.

We use BART-Base[2] and mBART-Large[3] models to initialize our model, respectively. BART-Base model has 6 layers of encoder and decoder with 768 hidden dimension, while mBART-Large has 12 layers of encoder and decoder with 1024 hidden dimension. The parameter sizes for the two models initialized with BART and mBART are 199M and 816M, respectively.

We use one NVIDIA RTX 3090 GPU to train our model. During training, we set the max tokens of each batch to 1024 (for BART, or 512 for mBART) with an update frequency of 4. We adopt Adam optimizer (Kingma and Ba, 2015) with learning rate selected in {1e-4, 5e-5, 2e-5, 1e-5} and warm-up updates selected in {200, 500, 1000} followed by a polynomial decay scheduler. Dropout strategy (Srivastava et al., 2014) with dropout rate selected in {0.2, 0.4} and L_2 regularization with 0.01 effect value, as well as early stoping based on validation performance, are used to alleviate overfitting. We set the tradeoff values among the losses as $\lambda = 1.0$, $\gamma = 1.0$ and $\delta = 0.2$. During inference, predicting threshold $\theta = 0.3$ and beam size is set to 5.

5 Experimental Results

5.1 Teaching Strategy Prediction Results

We report the accuracy and Macro F1 scores for teaching strategy prediction task in Table 1. We can find that prediction based on the target tutor response performs much better than merely on source context (comparing BART† and BART), which indicates that prediction with target content is much easier and also validates our motivation of the self-distillation mechanism. With the help of our proposed distillation mechanism, our models with pre-trained BART or mBART achieve the best performance in the prediction based on source context.

5.2 Tutor Response Generation Results

We then report case-sensitive detokenized sacreBLEU (Post, 2018) and BERTScore (Zhang et al., 2019) for tutor response generation in Table 2.

Three Evaluation Settings. We show results in three settings in Table 2. “W/O TS” means we don’t include teaching strategy (TS) labels in training and testing. “With Golden TS” means providing ground truth TS labels for training and testing. “Need TS Prediction” means models have to predict TS labels in testing and generate the follow-up tutor responses based on the predicted TS labels.

Analysis on Generation Results. From Table 2, we can draw the following main observations.

• *Teaching strategy shows positive effects in generation.* By comparing the results in “W/O TS” and “With Golden TS” settings, we observe that guidance from golden teaching strategies improves the generation performance in general, which validates the effects of teaching strategy in guiding generation. Besides, our models further improve their corresponding baselines (e.g. Our Model(BART) v.s. BART), which should result from the joint learning of generation and strategy prediction.

• *Successful guidance requires accurate teaching strategies.* By comparing results in “With Golden TS” and “Need TS Predict”, we can find that most of the models perform worse when they need to predict strategies first, especially for the baselines with poor strategy prediction performance (refer to results of BiLSTM and Transformer in Table 1). This shows that guidance from inappropriate strategies might even hurt performance, which raises the need for accurate prediction in real-world applications and our proposed method can alleviate the gap significantly.

[2] https://github.com/facebookresearch/fairseq/tree/main/examples/bart

[3] https://github.com/facebookresearch/fairseq/tree/main/examples/mbart

Models	**CIMA**		**TSCC**		**TalkMoves**	
	Acc	F1	Acc	F1	Acc	F1
BART	64.3	31.5	59.1	11.6	55.2	31.1
BART†	**82.3**	**57.1**	**64.4**	**18.9**	**75.9**	**50.5**
Frequency	62.7	15.4	58.4	4.1	52.5	11.5
BiLSTM	57.3	30.1	56.5	11.2	50.1	25.6
Transformer	63.3	33.9	57.2	16.2	53.6	30.7
Our Model(BART)	69.7	39.2	60.6	17.4	57.8	35.5
Our Model(mBART)	70.4	39.8	60.4	17.0	59.6	37.6

Table 1: Teaching strategy prediction results (in %). † indicates the prediction is based on the target tutor response. The best and second-best results in each column are in **bold** and underlined respectively.

	Models	**CIMA**		**TSCC**		**TalkMoves**	
		BLEU	BERT	BLEU	BERT	BLEU	BERT
W/O TS	BiLSTM	9.08	**72.6**	1.04	69.0	0.43	73.2
	Transformer	10.1	72.2	1.53	70.4	0.74	74.9
	BART	6.77	71.9	1.27	**71.2**	0.85	78.0
	mBART	**10.6**	70.9	**1.96**	68.6	**2.95**	**78.1**
With Golden TS	BiLSTM	8.61	71.8	1.32	69.1	1.42	75.8
	Transformer	11.2	72.8	1.99	69.9	2.35	77.4
	BART	9.17	70.8	1.47	68.6	2.93	78.0
	mBART	11.1	72.3	1.57	69.5	3.38	75.7
	Our Model(BART)	10.8	71.4	2.02	70.6	3.18	78.0
	Our Model(mBART)	**12.1**	**73.8**	**2.93**	**72.6**	**5.47**	**79.7**
Need TS Predict	BiLSTM	7.65	69.8	0.68	68.2	0.48	74.7
	Transformer	8.04	68.6	0.79	69.3	2.05	76.8
	BART	7.64	69.5	1.13	69.4	1.49	73.8
	mBART	7.77	70.2	1.57	69.7	2.44	77.1
	Our Model(BART)	8.67	70.8	2.83	70.0	2.22	77.5
	Our Model(mBART)	**11.9**	**73.0**	**2.98**	**71.9**	**4.51**	**78.6**

Table 2: Generation results (in %). The best results in each setting are in **bold**. Our full model achieves significantly better performance than the baselines with the same architecture in the same settings (paired t-test $p < 0.05$).

Student: how to say under in Italian?		
CTS:	[Hint] [Question]	"Is under the" is "e sotto il". Do you know how to say box?
CTS:	[Hint]	"Is under the" is "e sotto il".
Student: La pianta e accanto al congilio giallo.		
CTS:	[Correction]	You're very close. but remember that adjective follows the noun.
CTS:	[Confirmation]	Yes, that's right!

Figure 3: Our CTS generates different responses when giving different teaching strategies (in red).

5.3 Effects of Teaching Strategy

We explore how teaching strategy affects the generation in Fig. 3. We feed our conversational tutoring system (CTS) with different teaching strategies and find that CTS generates totally different responses regarding the same context input. This also validates that teaching strategy is important for a CTS and strategizing before teaching is also essential.

6 Conclusion

In this work, we benchmarked neural models on various conversational tutoring datasets and proposed a self-distillation based model that jointly trains a teaching strategy prediction model and a response generation model. Experiments on three conversational tutoring datasets show that our model outperforms various standard baselines by a significant margin. Finally, we ended with an interesting case study to demonstrate the importance of strategizing before teaching.

Limitations

There are only three publicly available datasets (CIMA, TSCC and TalkMoves) for conversational tutoring task and they are quite small (less than 10K instances). There are significant data imbalance problems in these datasets – some teaching strategies occur much more frequently than others. These small and imbalanced datasets bring a lot of challenges to this task, but we did not discuss these issues in our paper due to the space limit. Besides, there are no standard teaching strategy annotation schemes, which prevents us from combining these three datasets together for more interesting experimental analyses. Another limitation of our work is that we only evaluate our approaches on automatic generation metrics. In the future, it would be interesting to also evaluate the model on learning related evaluations.

References

Quincy Brown. 2009. Mobile intelligent tutoring system: moving intelligent tutoring systems off the desktop.

Andrew Caines, Helen Yannakoudakis, Helena Edmondson, Helen Allen, Pascual Pérez-Paredes, Bill Byrne, and Paula Buttery. 2020. The teacher-student chatroom corpus. In *Proceedings of the 9th Workshop on NLP for Computer Assisted Language Learning*, pages 10–20.

Jaime R Carbonell. 1970. Ai in cai: An artificial-intelligence approach to computer-assisted instruction. *IEEE transactions on man-machine systems*, 11(4):190–202.

Elizabeth Emily Carter. 2014. An intelligent debugging tutor for novice computer science students.

Robert Costello. 2012. *Adaptive intelligent personalised learning (AIPL) environment*. Ph.D. thesis, University of Hull.

Beate Grawemeyer, Manolis Mavrikis, Wayne Holmes, Sergio Gutierrez-Santos, Michael Wiedmann, and Nikol Rummel. 2016. Affecting off-task behaviour: how affect-aware feedback can improve student learning. In *Proceedings of the sixth international conference on learning analytics & knowledge*, pages 104–113.

Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7).

Sanghyun S Jeon and Stanley YW Su. 2010. Adaptive e-learning using ecpaa rules, bayesian models, and group profile and performance data. *International Journal of Learning Technology*, 5(4):415–434.

Aytürk Keleş, Rahim Ocak, Ali Keleş, and Aslan Gülcü. 2009. Zosmat: Web-based intelligent tutoring system for teaching–learning process. *Expert Systems with Applications*, 36(2):1229–1239.

Diederick P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*.

Utku Kose and Ahmet Arslan. 2017. Optimization of self-learning in computer engineering courses: An intelligent software system supported by artificial neural network and vortex optimization algorithm. *Computer Applications in Engineering Education*, 25(1):142–156.

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 2018. Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447, Melbourne, Australia. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742.

Jakub Macina, Nico Daheim, Lingzhi Wang, Tanmay Sinha, Manu Kapur, Iryna Gurevych, and Mrinmaya Sachan. 2023. Opportunities and challenges in neural dialog tutoring. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*. Association for Computational Linguistics.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model. *arXiv preprint arXiv:2005.05298*.

Martha C Polson and J Jeffrey Richardson. 2013. *Foundations of intelligent tutoring systems*. Psychology Press.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels. Association for Computational Linguistics.

Jeffrey Ralph James Richardson. 1988. *Foundations of intelligent tutoring systems*. Psychology Press.

Timo Schick and Hinrich Schütze. 2021. Few-shot text generation with natural language instructions. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 390–402.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C. Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 3776–3784. AAAI Press.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205, Denver, Colorado. Association for Computational Linguistics.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958.

Katherine Stasaski, Kimberly Kao, and Marti A. Hearst. 2020. CIMA: A large open access dialogue dataset for tutoring. In *Proceedings of the Fifteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 52–64, Seattle, WA, USA â†' Online. Association for Computational Linguistics.

Abhijit Suresh, Jennifer Jacobs, Charis Harty, Margaret Perkoff, James H. Martin, and Tamara Sumner. 2022. The TalkMoves dataset: K-12 mathematics lesson transcripts annotated for teacher and student discursive moves. pages 4654–4662.

Abhijit Suresh, Tamara Sumner, Jennifer Jacobs, Bill Foland, and Wayne Ward. 2019. Automating analysis and feedback to improve mathematics teachers' classroom discourse. In *Proceedings of the the Ninth AAAI Symposium on Educational Advances in Artificial Intelligence*, volume 33, pages 9721–9728.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Dinesha Weragama and Jim Reye. 2014. Analysing student programs in the php intelligent tutoring system. *International Journal of Artificial Intelligence in Education*, 24(2):162–188.

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017. Topic aware neural response generation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 3351–3357. AAAI Press.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2019. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*.

Tiancheng Zhao, Allen Lu, Kyusong Lee, and Maxine Eskenazi. 2017. Generative encoder-decoder models for task-oriented spoken dialog systems with chatting capability. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 27–36, Saarbrücken, Germany. Association for Computational Linguistics.