Lexical Complexity Controlled Sentence Generation for Language Learning

Jinran Nie[1], Liner Yang[1]*, Yun Chen[2], Cunliang Kong[1], Junhui Zhu[1], Erhong Yang[1]
[1]Beijing Language and Culture University
[2]Shanghai University of Finance and Economics
njrbarry@gmail.com

Abstract

Language teachers spend a lot of time developing good examples for language learners. For this reason, we define a new task for language learning, lexical complexity controlled sentence generation, which requires precise control over the lexical complexity in the keywords to examples generation and better fluency and semantic consistency. The challenge of this task is to generate fluent sentences only using words of given complexity levels. We propose a simple but effective approach for this task based on complexity embedding while controlling sentence length and syntactic complexity at the decoding stage. Compared with potential solutions, our approach fuses the representations of the word complexity levels into the model to get better control of lexical complexity. And we demonstrate the feasibility of the approach for both training models from scratch and fine-tuning the pre-trained models. To facilitate the research, we develop two datasets in English and Chinese respectively, on which extensive experiments are conducted. Experimental results show that our approach provides more precise control over lexical complexity, as well as better fluency and diversity.

1 Introduction

In the fields of language teaching and acquisition, language instructors and textbook compilers need to make teaching materials with example sentences, either synthetically designed or from authentic resources (Caro and Mendinueta, 2017; Lu et al., 2019). In most cases, they are required to create appropriate example sentences that only use the words at particular complexity for language learners passing through different learning levels (Nordlund and Norberg, 2020; Laufer, 2021), which is very time-consuming and exhausting. Automatically generating good examples can support educators and language learners in obtaining, analyzing, and selecting proper example sentences. Besides, it can also assist in the development of graded reading materials (Ryu and Jeon, 2020; Al-Jarf, 2021; Amer, 2021).

For language learners, good examples are not only required to be fluent and diverse but also match the level of the learners, especially the level of vocabulary. Therefore, it is necessary to effectively control the lexical complexity in good examples generation, which is a task of controllable text generation.

Controllable text generation (CTG), a significant area of natural language generation, contains a series of tasks that aim to generate text according to the given controlled requirements (Prabhumoye et al., 2020; Zhang et al., 2022). CTG systems usually focus on controlling text attributions such as sentiment (Hu et al., 2017; Zhang et al., 2019; Samanta et al., 2020), topic (Dathathri et al., 2019; Tang et al., 2019; Khalifa et al., 2020) or keywords (He, 2021; Zhang et al., 2020; He and Li, 2021), generating poems or couplets with specific formats (Chen et al.,

* Corresponding author:Liner Yang

This work was supported by the funds of Research Project of the National Language Commission No. ZDI145-24.

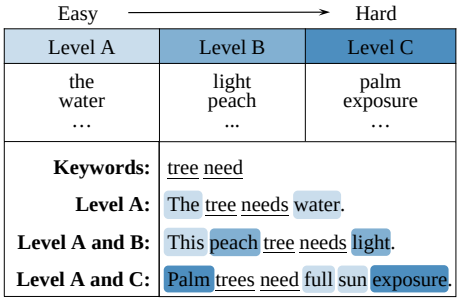

Figure 1: An example for lexical complexity controlled sentence generation. There are three complexity levels (A, B, and C) from easy to hard. Given the keywords "tree" and "need", we will generate "The tree needs water." if required to use all words from level A and generate "This peach tree needs light." if required to use words from both level A and B as both "peach" and "light" are in level B.

2019; Shao et al., 2021; Sheng et al., 2021), and even predicting descriptions from structured data (Zhao et al., 2020; Su et al., 2021; Ribeiro et al., 2021). However, few works have been devoted to strict control over the lexical complexity for text generation. Although lexical simplification has been paid attention to the text simplification task through substitution (Kriz et al., 2018), it cannot strictly control the lexical complexity levels of the generated sentence.

To this end, we propose a new task of lexical complexity controlled sentence generation, which requires that keywords and complexity levels be given to generate a sentence including the keywords and consisting of the words in the given complexity levels. For example, as shown in Figure 1, we assume that there are three complexity levels (A, B, and C) from easy to hard. Given the keywords, we can generate sentences consisted with words of different complexity according to the given levels.

It is challenging to generate fluent sentences for given keywords while using the words only at specific complexity levels. This can be regarded as an extension and a particular case of lexical CTG task (He and Li, 2021; Miao et al., 2019; Zhang et al., 2020). Differently, it combines two aspects of constraints during generation: keywords constraint the semantics, and lexical complexity levels constraint the surface form. It is difficult for the model to select suitable words from a specific subspace satisfying the above two constraints in each generation process. We formulate this problem in Section 2.1.

Some previous works can be customized as solutions to this problem, which are divided into three branches: controlled decoding, prompting, and reranking. The first method forces to change the probability distribution during the decoding phase to ensure that only words of the specified levels are used in the generation (Dathathri et al., 2019; Post and Vilar, 2018). But the hard constraint may lead to poor quality generation quality. The second one considers lexical complexity through prompting (Brown et al., 2020; Raffel et al., 2020; Li and Liang, 2021) in the input of the model, which introduce coarse grained information of training and inference. The method of reranking is to select the sentence that best meets the lexical complexity requirements from the candidates (Ravaut et al., 2022; Pandramish and Sharma, 2020), which executes after decoding and does not consider lexical complexity in the training time.

The complexity constraint requires models to aware of lexical complexity and respond to complexity control signals. Therefore, we use two mechanisms as enhancements to the transformer-based models. *For the complexity awareness*, we propose the Complexity Embedding (**CE**) method, which represents the complexity levels with trainable embeddings. We incorporate the CEs into both training and prediction processes by fusing the CEs and word embeddings as token representations, which is simple but effective. *For responding to complexity control signals*, we concatenate special tokens corresponding to specific complexity levels with the keywords as the input sequence. To combine the awareness and response, we use CEs to represent these

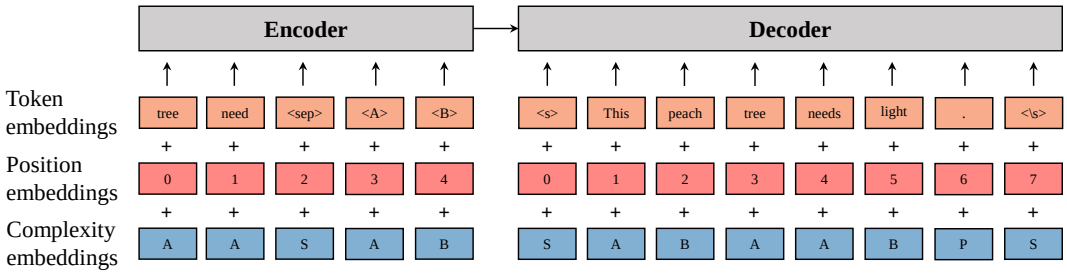

Figure 2: Encoder-Decoder model with our proposed CE method. The representation of each input token is a summary of three embeddings, which are token embedding, position embedding, and complexity embedding. And we concatenate the keywords and complexity level tokens as the input sequence of the encoder. Note that the special tokens correspond to the complexity level of "S", and the punctuation correspond to "P".

special tokens. The experiments show that our proposed method is effective for both training from scratch and fine-tuning the pre-trained language models. And compared to the baseline methods, our method achieves significant improvement in the restriction of lexical complexity levels and generation quality. Our main contributions include:

- We propose a new task of lexical complexity controlled sentence generation and two datasets in English and Chinese for this task. To evaluate the satisfaction of the lexical complexity constraint, we develop four metrics.
- We propose a new method for this task based on complexity embedding.
- The experimental results show that the complexity embedding method we proposed significantly outperforms the baseline methods which are implemented for this task.

2 Method

2.1 Problem Definition

Lexical Complexity Controlled Sentence Generation aims at keywords to sentence generation with desired complexity levels. First, we give the keywords set $K = \{k_1, k_2, ..., k_m\}$ and the complexity levels $L = \{l_1, l_2, ..., l_n\}$ which correspond to a subset $D = \{W_1 \cup W_2 \cup ... \cup W_n\}$ of the whole vocabulary V and W_i is the word set of complexity level l_i. The control elements in this task include three parts:

First, we define a predicate $F(K, Y)$ to be a boolean function indicating the occurrence of keyword k_i in a generated sequence $Y = y_1, y_2, ..., y_t$, and t is the sequence length.

$$C_1 = F(K, Y) \tag{1}$$

$$F(K, Y) \equiv \forall\ i, k_i \in Y \tag{2}$$

where C_1 is the keywords constraint which means the keywords are required to be included in the generated sentence.

Second, we define a predicate $G(Y, D)$ to be a boolean function indicating the occurrence of a word y_i which is a word of the sentence Y in a word set D.

$$C_2 = G(Y, D) \tag{3}$$

$$G(Y, D) \equiv \forall\ i, y_i \in D \tag{4}$$

where C_2 is the complexity constraint on word which means the words in the generated sentence are required to be the words of the given complexity levels.

Then, we define a predicate $H(Y, W_i)$ to be a boolean function indicating that there exist at least one word in the generated sentence in the W_i.

$$C_3 = H(Y, W_1) \wedge H(Y, W_2) ... \wedge H(Y, W_n) \tag{5}$$

$$H(Y, W_i) \equiv \exists\ j, y_j \in W_i \tag{6}$$

where C_3 is the constraint on the species of complexity level which means the lexical levels of the generated sentence need cover all the given levels.

The task requires to seek optimal sequences in which all constraints are satisfied as much as possible. The formula is as follows:

$$\hat{Y} = \arg\max_{Y \in \mathcal{Y}} \log P_\theta\big(Y|K, L\big) \quad \text{where} \quad \sum_{i=1}^{N} C_i = N \tag{7}$$

where N is the number of constraints and $N = 3$.

2.2 Complexity Embedding

As illustrated in Figure 2, our model is based on the encoder-decoder architecture. To make the model aware of the complexity levels, we fuse the complexity into the task by designing a lexical complexity embedding for each token. To make the model respond to specific complexity levels, we insert special tokens corresponding to complexity levels into the input sequence as controllable elements. This section introduces these two key components as well as the training and inference strategy.

We initialize a learnable matrix $\mathbf{M} \in \mathbb{R}^{U \times dim}$ as representations of complexity levels, where U is the total number of complexity levels, and dim is the dimensions of each embedding. For each token input to the encoder and decoder, we retrieve a predefined hash-table to obtain its complexity level l_i. Then we get the corresponding complexity embedding by $com_i = \mathbf{M}_i$. The final embedding of this token emb_i is as following:

$$emb_i = tok_i + pos_i + com_i \tag{8}$$

where tok_i and pos_i are token and positional embeddings, which are obtained according to Transformer model(Vaswani et al., 2017).

For example, as shown in Figure 2, when two keywords "tree" and "need" along with two complexity levels A and B are required, the sentence "This peach tree needs light." is generated which satisfies both constraints. We use different complexity representations (mapping into a complexity embedding) for words of different complexity levels. And the complexity representations of special tokens and punctuation are also different.

In practice, we apply the BPE (byte pair encoding) (Sennrich et al., 2015) algorithm to split words into sub-word tokens to mitigate the OOV (out-of-vocabulary) problem. We mark each sub-word with the same complexity level as the original word. More details about the complexity levels can be found in the Appendix A.

2.3 Controllable Elements

As illustrated in Equation 4, each word in the sentence Y is constrained to the word set D. To achieve this, we design a set of special tokens $Z = \{z_1, z_2, \ldots, z_n\}$, where each token corresponds to a complexity level in L.

We concatenate the keywords and the special tokens as the input sequence $X = [K; \langle sep \rangle; Z]$. And we refer the special tokens Z as controllable elements, as they control the complexity of the generated sentence. Note that the complexity embedding of z_i is that of the level l_i.

2.4 Training Complexity Embedding

We train the complexity embedding in the Transformer model from scratch or fine-tune the pre-trained model discriminatively as there is no complexity embedding layer in the pre-trained process. If a model is trained from scratch, the parameters of complexity embedding will be trained the same as other parameters in the model. If the complexity embedding is added to a pre-trained model for fine-tuning, we first train the complexity embedding layer by fixing the original parameters of the pre-trained model and then fine-tune the whole model.

During the training process, in fact, both the word embedding and the complexity embedding are in a teach-forcing pattern through the ground truth. At the time of inference, the next word embedding at each step will be predicted by the probability distribution of the vocabulary of the model. Since the complexity level of the next word is unknown at each step of the inference stage, we utilize a look-up table method to map the predicted token id to complexity id. The table is a mapping relation between the token id and its complexity id on the whole vocabulary. At each step, the token id will be predicted by the model. We get its complexity id through its token id and the table. The complexity id and token id will then be given as the input for the next step of inference.

2.5 Length and Syntactic Complexity Control

The length of the generated text is also a factor that language learners may consider, and there is a correlation between text length and syntactic complexity. From a statistical view, text length and syntactic complexity are generally positively correlated. Thus, we design a method to dynamically control text length and syntactic complexity, which is used in the decoding stage. We set three sentence length modes: short, normal, and long, and the sentence length mode also corresponds to the syntactic complexity. We introduce length penalties to beam search in the decoding time in different modes. The formula for calculating the penalty coefficient is as follows:

$$Penalty = N^{pen} \tag{9}$$

where N is the counts of keywords, $pen = -1, 0, 1$ if the mode is short, normal or long respectively. We have observed from statistics that the larger the number of given keywords leads the longer the generated sentences. Therefore, we set the relationship between the length penalty and the number of keywords. In the mode of short or long, if the number of keywords is larger, the greater the penalty required.

3 Datasets and Evaluation Metrics

3.1 Dataset Construction

We present two datasets for lexical complexity controlled sentence generation in English and Chinese. The English raw corpus is collected from the monolingual English News dataset in ACL2019 WMT. The Chinese raw corpus is collected from 500 textbooks for Chinese L2 learners. We adopt the English word complexity levels in the Common European Framework of Reference for Languages (CEFR) [0] which is divided into six complexity levels (A1, A2, B1, B2, C1, and C2). The word complexity levels in Chinese Proficiency Grading Standards for International Chinese Language Education (CPGS) [1] is divided into seven complexity levels (1 to 7). The process for cleaning data is divided into three steps: split the raw data into sentences and choose the proper sentences; obtain the keywords from the sentences; get the lexical complexity levels from the sentences. More details of the two datasets are in the Appendix B.

3.2 Evaluation Metrics

Generated Quality To evaluate the quality of generated text, we employ some automatic evaluate metrics in three aspects. 1) N-gram Similarity with References: we use **BLEU** (Papineni et

[0] https://www.englishprofile.org/wordlists/evp
[1] http://www.chinesetest.cn

al., 2002), **METEOR** (Lavie and Agarwal, 2007), and **NIST** (Doddington, 2002) evaluate the difference between generated texts and reference texts, which are commonly utilized in machine translation and text generation. 2) Diversity: We use 2-gram and 4-gram of **Entropy** (Zhang et al., 2018) and 1-gram and 2-gram of **Distinct** (Li et al., 2015) to evaluate lexical diversity. 3) Fluency: Following previous works (Zhang et al., 2020; He and Li, 2021), to assess the fluency of generated sentences, we report the perplexity (**PPL**) over the test set using the pre-trained GPT-2 (Radford et al., 2019) large model.

Satisfaction of Lexically Controlling The control elements of lexical complexity controlled sentence generation have introduced in the Section 2.1. Our metrics are corresponding to the three constraints.

- **Keywords Constraint**. For this aspect, we introduce Keywords Constraint (**K-C**) satisfaction metric on word-level, which is computed using the percentage of the keywords contained in the generated sentences. The formular describe is as below:

$$K - C = \frac{1}{N} \sum_{i=1}^{N} \text{count}_i^{C_1} / m_i \tag{10}$$

  where N is the total number of samples in the test dataset, $\text{count}_i^{C_1}$ is the number of keywords included in the generated sentence of the i-th sample, which satisfy the constraint of C_1, and m_i is the number of the keywords of the input on the i-th sample.

- **Word Complexity Constraint**. The purpose of this metric is to calculate the Accuracy (**ACC**) of the words that meet the lexical complexity levels requirement in the generated sentence. As shown in the following formula:

$$ACC = \frac{1}{N} \sum_{i=1}^{N} \text{count}_i^{C_2} / t_i \tag{11}$$

  where $\text{count}_i^{C_2}$ is the number of the words that satisfy the constraint C_2 of the i-th sample, and t_i is the length of the generated sentence of the i-th sample.

- **Complexity Levels Constraint**. We propose three metrics to evaluate the satisfaction of the species of the required complexity levels. It is unreasonable that the ACC is still 100% if given two complexity levels but the words of generated sentence only covers one of the levels. Thus we design the metrics of Precision (**P**), Recall (**R**), and **F1** to calcuate the saticfaction of complexity level constraint. The formular discribes are as follows:

$$P = \frac{1}{N} \sum_{i=1}^{N} \text{count}_i^{C_3} / g_i \tag{12}$$

$$R = \frac{1}{N} \sum_{i=1}^{N} \text{count}_i^{C_3} / n_i \tag{13}$$

$$F1 = \frac{2}{N} \sum_{i=1}^{N} \text{count}_i^{C_3} / (n_i + g_i) \tag{14}$$

  where $\text{count}_i^{C_3}$ is the number of the complexity levels satisfy the constraint C_3 of the i-th sample, n_i is the number of the complexity levels given in the source of the i-th sample, and g_i is the number of the complexity levels of the generated sentence of the i-th sample.

4 Experiments

Our experiments are based on the two datasets introduced in Section 3. Besides the strong baselines of controlled decoding, prompting and reranking mentioned in Section 4.2, we generate the sentence by setting the keys as the input directly as the basic baseline (K2S). This baseline does not require complexity levels, which are just learnt from the data. Our evaluations include automatic evaluation and human evaluation. The automatic metrics have been introduced in the Section 3.

Metrics	BLEU(%)		NIST(%)		METEOR(%)	Entropy(%)		Distinct(%)		PPL
	B-2	B-4	N-2	N-4		E-2	E-4	D-1	D-2	
Training Transformer from scratch										
K2S	16.58	4.57	3.14	3.27	15.23	8.20	10.23	**5.93**	24.76	74.91
Ctrl-decoding	12.12	3.16	2.45	2.61	11.72	7.28	9.22	5.27	20.14	286.50
Prompting	18.19	5.73	3.57	3.64	15.93	8.30	10.36	6.10	25.55	52.10
Reranking	**18.47**	6.27	3.52	3.60	15.99	7.87	9.79	5.93	22.70	47.81
CE (ours)	18.37	**6.66**	**3.64**	**3.69**	**16.06**	**8.43**	**10.47**	5.80	**25.75**	**42.06**
Fine-tuning BART										
K2S	17.40	5.96	3.20	3.26	15.60	8.60	10.52	6.36	28.53	33.11
Ctrl-decoding	14.17	3.55	2.73	2.48	13.15	8.03	9.87	5.96	21.96	223.43
Prompting	19.36	6.88	3.59	3.67	16.09	**8.93**	**10.81**	**7.22**	**33.84**	39.65
Reranking	18.95	6.54	3.54	3.58	16.03	8.72	10.67	6.60	30.09	34.24
CE (ours)	**19.80**	**7.22**	**3.61**	**3.69**	**16.34**	8.50	10.48	6.41	27.56	**28.48**

Table 1: Generation quality evaluation results on English dataset.

Metrics (%)	K-C	ACC	P	R	F1
Training Transformer from scratch					
K2S	96.93	95.68	89.03	83.27	84.93
Ctrl-decoding	85.56	99.02	97.84	83.51	89.19
Prompting	96.85	98.91	97.35	90.86	93.46
Reranking	97.33	96.80	91.81	87.97	88.98
CE (ours)	**98.00**	**99.10**	**98.09**	**92.84**	**94.96**
Fine-tuning BART					
K2S	97.51	95.26	88.79	84.63	85.58
Ctrl-decoding	89.73	**99.34**	**98.57**	84.19	90.33
Prompting	96.57	97.79	95.77	90.17	92.25
Reranking	98.52	96.10	92.36	88.96	91.87
CE (ours)	**98.68**	99.13	98.54	**93.72**	**95.77**

Table 2: Satisfaction of controlling evaluation results on English dataset.

4.1 Experimental Setup

Our experimental setup contains two aspects:training from scratch and fine-tuning. From scratch training experiments are on the Transformer model (Vaswani et al., 2017), which is the most widely used model in text generation. The fine-tuning experiments are on the pre-trained model of BART (Lewis et al., 2019), which has superior generation ability. During inference, we run greedy decoding on all models for a fair comparison. We implement all models with the Fairseq library [2] and the BART pre-trained model is from HuggingFace Transformers library [3] (Wolf et al., 2019). All models are trained and tested on NVIDIA TITAN Xp GPU.

From Scratch Training Setup We adopt the typical Transformer (Vaswani et al., 2017) as the model trained from scratch. We utilize a learning rate of 3e-4 and set the warming-up schedule with 4000 steps for training. We train our model for around 100 epochs. The optimization algorithm is Adam (Kingma and Ba, 2014). We set the maximum number of input tokens as 8192, which is the same as transformer-based baselines. **Fine-tuning Setup** We initialize our model with BART-base (Lewis et al., 2019), which has comparable parameters to generation baselines. For generation baselines and our models, we use Adam (Kingma and Ba, 2014) with an initial learning rate of 1e-5 to update parameters for four epochs and choose the checkpoints with the lowest validation loss. We train our model for around 30 epochs. We set the maximum number of input tokens as 2048.

[2]https://github.com/pytorch/fairseq
[3]https://github.com/huggingface/transformers

Metrics	BLEU(%)		NIST(%)		METEOR(%)	Entropy(%)		Distinct(%)		PPL
	B-2	B-4	N-2	N-4		E-2	E-4	D-1	D-2	
Training Transformer from scratch										
K2S	13.92	4.17	2.73	2.76	15.00	8.83	10.20	8.60	37.70	48.32
Ctrl-decoding	12.84	3.57	2.48	2.50	13.70	8.70	10.30	6.08	34.90	224.59
Prompting	13.90	3.81	2.70	2.73	14.35	8.53	10.05	7.47	33.35	45.61
Reranking	15.46	5.37	**2.98**	**3.02**	15.34	8.84	10.15	9.13	37.88	38.56
CE (ours)	**15.69**	**6.27**	2.91	2.94	**16.04**	**9.28**	**10.58**	**10.68**	**47.71**	**34.53**
Fine-tuning BART										
K2S	14.97	4.39	3.08	3.10	16.56	8.60	10.06	9.91	37.13	**21.76**
Ctrl-decoding	12.54	3.71	2.38	2.55	14.04	8.73	10.25	9.96	37.85	129.86
Prompting	16.81	5.47	3.15	3.17	16.24	8.69	10.13	10.04	38.33	31.75
Reranking	16.53	6.42	**3.29**	**3.36**	16.61	8.81	10.08	10.15	38.96	53.47
CE (ours)	**17.07**	**6.46**	3.18	3.26	**16.73**	**9.34**	**10.27**	**10.55**	**48.76**	26.52

Table 3: Generation quality evaluation results on Chinese dataset.

Metrics (%)	K-C	ACC	P	R	F1
Training Transformer from scratch					
K2S	87.36	92.74	85.40	68.40	73.75
Ctrl-decoding	71.83	**99.96**	**99.96**	61.79	74.73
Prompting	85.54	98.88	97.79	80.23	86.88
Reranking	88.22	96.70	93.05	75.74	81.59
CE (ours)	**89.61**	98.87	97.49	**88.80**	**92.17**
Fine-tuning BART					
K2S	92.12	93.73	86.88	68.87	74.37
Ctrl-decoding	82.52	**99.18**	**98.65**	65.26	76.41
Prompting	86.94	98.73	97.98	81.78	88.02
Reranking	90.14	97.21	95.44	76.78	83.95
CE (ours)	**92.58**	99.07	97.91	**89.34**	**92.85**

Table 4: Satisfaction of controlling evaluation results on Chinese dataset.

4.2 Baseline

Controlled decoding We consider a strategy of controlled decoding (Dathathri et al., 2019) to realize the generated sentence consists of the words belonging to the given complexity levels. Since we know the words of the complexity level to be used in the sentence, we can restrict the words of the subset of the vocabulary to only be used in the decoding stage. The specific method is to set the probability of words outside the subset to zero so that they can meet the requirements of the word complexity level.

Prompting Prompting is another feasible method for controlled text generation (Zou et al., 2021). Inspired by the prefix-tuning (Li and Liang, 2021), which uses continuous vectors as prompts, we add the required complexity levels as the prefix for controlling in the input of the generation model.

Reranking Inspired by previous works (Ravaut et al., 2022; Pandramish and Sharma, 2020), we select the sentence that best meets the lexical complexity requirements from the N-best candidates. We take the score that is the sum of ACC score and $F1$ score on the test reference hypothesis from this N-best list and choose the candidate that has the largest score. The detail of the re-ranking method is shown as the Algorithm 1 in Appendix C.

4.3 Experimental Results

The experimental results on English dataset are shown in Table 1 and Table 2. From the evaluation of generation quality in Table 1, it can be seen that the method of complexity embedding has competitive results in different aspects, especially on fluency. In general, the CE method

Metrics (%)	Semantics	Fluency	Diversity
English dataset			
Ctrl-decoding	2.68	2.40	2.92
Prompting	**4.63**	3.25	3.45
Reranking	4.60	3.39	3.40
CE (ours)	4.62	**3.82**	**3.54**
Chinese dataset			
Ctrl-decoding	3.89	2.82	3.27
Prompting	4.23	3.08	3.02
Reranking	4.37	3.29	3.16
CE (ours)	**4.57**	**3.80**	**3.71**

Table 5: Human evaluations for fine-tuning BART model on two datasets.

has better performance in the control of lexical complexity, especially on the metrics of R and F1. The method of controlled decoding has poor performance on PPL because it forces the distribution of the logits to concentrate on the words of given complexity levels in the decoding stage. This hard constraint pattern will impact the fluency of the generated sentences. But its performances on the metrics of ACC and P are better than other methods from Table 2. The methods of prompting and reranking are two competitive baselines. The prompting method has better performance in the control of the word complexity because it has considered the word complexity levels in training. But the reranking method has better generation quality on the whole metrics of Table 1.

The experimental results on Chinese dataset are shown in Table 3 and Table 4. We can draw similar conclusions from these two tables. Our approach performs well in terms of both text generation quality and lexical complexity control. The rerank approach outperforms prompt in all aspects of generation quality, both in terms of similarity to ground truth and in diversity and fluency, and even achieves the best NIST metrics for the Chinese dataset.

4.4 More Analyses and Discussion

The CE method we proposed has an excellent performance in controlling lexical complexity. The reason is that the CE method not only keeps the consistency of training and prediction but also considers the information of the complexity at the token level. Thus, it has more precise control of lexical complexity. And it also has competitive generation quality in the aspect of fluency and similarity with the reference. From the metrics of Entropy and Distinct, its diversity has a little poor performance in terms of the fine-tuning pattern on the English dataset. We think the main reason is that the vocabulary of the English word complexity levels is less than which of the Chinese, so the token level restrictions of complexity embedding will impact the diversity of the sentences. The Chinese dataset, on the other hand, has a much larger coverage of voabulary with complexity and the dataset comes from the field of second language teaching, so the diversity of our model is better. It is worth noting that our CE method performs best in terms of lexical complexity control, especially the metrics of K-C, R, and F1, compared to the baseline model. This indicates that the CE method has higher coverage on complexity levels due to it takes into account the complexity of each word.

4.5 Length and Syntactic Complexity Control

We evaluate the length and the depth of the syntactic tree of generated text in the modes of short, normal and long, which can reflect the complexity of the generated text. As shown in the table 6, the experiment of controlling sentence length and syntactic complexity is on the English dataset. In the long mode, the generated sentences are longer, and the syntactic tree is deeper. In the short mode, the generated sentences are shorter, and the syntactic tree depth is smaller. The length penalty in the decoding stage can effectively control the sentence length while affecting the complexity of the syntax.

Metric/Mode	Short	Normal	Long
Length	15.3	24.6	36.8
Syn-Depth	9.3	11.1	13.5

Table 6: The length and depth of syntactic tree of generated sentences in different modes.

4.6 Human Evaluation

We conduct a human evaluation to further compare our model with the three baselines with fine-tuning the BART model on two datasets. For each model, we randomly select 200 generated sentences from the test set for each dataset and invite three annotators to label the sentences, who are postgraduates of the major in linguistics. To evaluate the quality of the sentences, annotators rate the sentences on three dimensions: semantic consistency between the keywords and sentence; the fluency of the sentence; the diversity of the sentence (Zhang et al., 2020). The score is range from 0 to 5. As shown in Table 5, our method has better performance at the three aspects of human evaluation, especially the fluency and diversity. We give some real cases of two datasets in the Appendix D. From the cases study we can find that the CE method can cover more lexical complexity levels than the baseline methods. This also confirms the reason why the CE method that we proposed has a better performance on R and F1 metrics of the automatic evaluation.

5 Related Work

Lexical constraint text generation is to generate a complete text sequence, given a set of keywords as constraints (Zhang et al., 2020). Previous works involve enhanced beam search (Post and Vilar, 2018; Hu et al., 2019) and the stochastic search methods (Zhang et al., 2020; Sha, 2020). Currently, Seq2Seq-based models such as Transformer and pre-trained models have been increased in generation with lexical constraint (Wang et al., 2021b; Liu et al., 2020; Wang et al., 2021a; Fan et al., 2020; Liu et al., 2021). But lexically constrained text generation is not able to control the complexity of words used in the generation, which is different from our work.

Text readability assess research has shown that lexical complexity is also a crucial aspect of evaluating the complexity of a text for text readability assess task (Chakraborty et al., 2021). In the relevant study of sentence-level readability, it is generally accepted that apart from sentence length, the most predictive indicator is the number of difficult words in the sentence (Weiss and Meurers, 2022). In our work, we follow the definition and vocabulary of lexical complexity of text readability assess.

Text simplification In text simplification field, lexical substitution, the replacement of complex words with simpler alternatives, is an integral part of sentence simplification and has been the subject of previous work (Alonzo et al., 2020; Nishihara et al., 2019). Differently, our work can strictly control the lexical complexity levels of the generated sentence, not only simplify the lexical complexity.

6 Conclusions

To summarize, we introduce a new task of lexical complexity controlled sentence generation, where word complexity must be strictly controlled in generating. To promote the development of this task, we develop two datasets and four metrics for the controlled element. In this paper, we also develop a series of alternate solutions for this task and propose a novel method based on complexity embedding to obtain better control of lexical complexity in a generation. Our results indicate that the complexity embedding method has better performance in controlling the lexical complexity and competitive generation quality.

References

Reima Al-Jarf. 2021. Efl students' difficulties with lexical and syntactic features of news headlines and news stories. *Technium Soc. Sci. J.*, 17:524.

Oliver Alonzo, Matthew Seita, Abraham Glasser, and Matt Huenerfauth. 2020. Automatic text simplification tools for deaf and hard of hearing adults: Benefits of lexical simplification and providing users with autonomy. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13.

Mohammad Ahmad Bani Amer. 2021. Lexical density and readability of secondary stage english textbooks in jordan. *International Journal for Management and Modern Education*, 2(2):11–20.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Keiby Caro and Nayibe Rosado Mendinueta. 2017. Lexis, lexical competence and lexical knowledge: a review. *Journal of Language Teaching & Research*, 8(2).

Susmoy Chakraborty, Mir Tafseer Nayeem, and Wasi Uddin Ahmad. 2021. Simple or complex? learning to predict readability of bengali texts. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12621–12629.

Huimin Chen, Xiaoyuan Yi, Maosong Sun, Wenhao Li, Cheng Yang, and Zhipeng Guo. 2019. Sentiment-controllable chinese poetry generation. In *IJCAI*, pages 4925–4931.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145.

Zhihao Fan, Yeyun Gong, Zhongyu Wei, Siyuan Wang, Yameng Huang, Jian Jiao, Xuanjing Huang, Nan Duan, and Ruofei Zhang. 2020. An enhanced knowledge injection model for commonsense generation. *arXiv preprint arXiv:2012.00366*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2020. Making pre-trained language models better few-shot learners. *arXiv preprint arXiv:2012.15723*.

Xingwei He and Victor OK Li. 2021. Show me how to revise: Improving lexically constrained sentence generation with xlnet. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 12989–12997.

Xingwei He. 2021. Parallel refinements for lexically constrained text generation with bart. *arXiv preprint arXiv:2109.12487*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *International conference on machine learning*, pages 1587–1596. PMLR.

J Edward Hu, Huda Khayrallah, Ryan Culkin, Patrick Xia, Tongfei Chen, Matt Post, and Benjamin Van Durme. 2019. Improved lexically constrained decoding for translation and monolingual rewriting. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 839–850.

Kenji Imamura and Eiichiro Sumita. 2017. Ensemble and reranking: Using multiple models in the nict-2 neural machine translation system at wat2017. In *Proceedings of the 4th Workshop on Asian Translation (WAT2017)*, pages 127–134.

Muhammad Khalifa, Hady Elsahar, and Marc Dymetman. 2020. A distributional approach to controlled text generation. *arXiv preprint arXiv:2012.11635*.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Reno Kriz, Eleni Miltsakaki, Marianna Apidianaki, and Chris Callison-Burch. 2018. Simplification using paraphrases and context-based lexical substitution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 207–217.

Batia Laufer. 2021. Lexical thresholds and alleged threats to validity: A storm in a teacup? *Reading in a Foreign Language*, 33(2):238–246.

Alon Lavie and Abhaya Agarwal. 2007. Meteor: An automatic metric for mt evaluation with high levels of correlation with human judgments. In *Proceedings of the second workshop on statistical machine translation*, pages 228–231.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055*.

Ye Liu, Yao Wan, Lifang He, Hao Peng, and Philip S Yu. 2020. Kg-bart: Knowledge graph-augmented bart for generative commonsense reasoning. *arXiv preprint arXiv:2009.12677*.

Yixian Liu, Liwen Zhang, Wenjuan Han, Yue Zhang, and Kewei Tu. 2021. Constrained text generation with global guidance–case study on commongen. *arXiv preprint arXiv:2103.07170*.

Dawei Lu, Xinying Qiu, and Yi Cai. 2019. Sentence-level readability assessment for l2 chinese learning. In *Workshop on Chinese Lexical Semantics*, pages 381–392. Springer.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842.

Daiki Nishihara, Tomoyuki Kajiwara, and Yuki Arase. 2019. Controllable text simplification with lexical constraint loss. In *Proceedings of the 57th annual meeting of the association for computational linguistics: Student research workshop*, pages 260–266.

Marie Nordlund and Cathrine Norberg. 2020. Vocabulary in efl teaching materials for young learners. *International Journal of Language Studies*, 14(1):89–116.

Vinay Pandramish and Dipti Misra Sharma. 2020. Checkpoint reranking: An approach to select better hypothesis for neural machine translation systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 286–291.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. *arXiv preprint arXiv:1804.06609*.

Shrimai Prabhumoye, Alan W Black, and Ruslan Salakhutdinov. 2020. Exploring controllable text generation techniques. *arXiv preprint arXiv:2005.01822*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Mathieu Ravaut, Shafiq Joty, and Nancy F Chen. 2022. Summareranker: A multi-task mixture-of-experts re-ranking framework for abstractive summarization. *arXiv preprint arXiv:2203.06569*.

Leonardo FR Ribeiro, Yue Zhang, and Iryna Gurevych. 2021. Structural adapters in pretrained language models for amr-to-text generation. *arXiv preprint arXiv:2103.09120*.

Jisu Ryu and Moongee Jeon. 2020. An analysis of text difficulty across grades in korean middle school english textbooks using coh-metrix. *Journal of Asia TEFL*, 17(3):921.

Bidisha Samanta, Mohit Agarwal, and Niloy Ganguly. 2020. Fine-grained sentiment controlled text generation. *arXiv preprint arXiv:2006.09891*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703.

Yizhan Shao, Tong Shao, Minghao Wang, Peng Wang, and Jie Gao. 2021. A sentiment and style controllable approach for chinese poetry generation. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, pages 4784–4788.

Zhonghao Sheng, Kaitao Song, Xu Tan, Yi Ren, Wei Ye, Shikun Zhang, and Tao Qin. 2021. Songmass: Automatic song writing with pre-training and alignment constraint. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 13798–13805.

Yixuan Su, David Vandyke, Sihui Wang, Yimai Fang, and Nigel Collier. 2021. Plan-then-generate: Controlled data-to-text generation via planning. *arXiv preprint arXiv:2108.13740*.

Hongyin Tang, Miao Li, and Beihong Jin. 2019. A topic augmented text generation model: Joint learning of semantics and structural features. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5090–5099.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Han Wang, Yang Liu, Chenguang Zhu, Linjun Shou, Ming Gong, Yichong Xu, and Michael Zeng. 2021a. Retrieval enhanced model for commonsense generation. *arXiv preprint arXiv:2105.11174*.

Yufei Wang, Ian Wood, Stephen Wan, Mark Dras, and Mark Johnson. 2021b. Mention flags (mf): Constraining transformer-based text generators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 103–113.

Zarah Weiss and Detmar Meurers. 2022. Assessing sentence readability for german language learners with broad linguistic modeling or readability formulas: When do linguistic insights make a difference? In *Proceedings of the 17th Workshop on Innovative Use of NLP for Building Educational Applications (BEA 2022)*, pages 141–153.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Yizhe Zhang, Michel Galley, Jianfeng Gao, Zhe Gan, Xiujun Li, Chris Brockett, and Bill Dolan. 2018. Generating informative and diverse conversational responses via adversarial information maximization. *arXiv preprint arXiv:1809.05972*.

Rui Zhang, Zhenyu Wang, Kai Yin, and Zhenhua Huang. 2019. Emotional text generation based on cross-domain sentiment transfer. *IEEE Access*, 7:100081–100089.

Yizhe Zhang, Guoyin Wang, Chunyuan Li, Zhe Gan, Chris Brockett, and Bill Dolan. 2020. Pointer: Constrained progressive text generation via insertion-based generative pre-training. *arXiv preprint arXiv:2005.00558*.

Hanqing Zhang, Haolin Song, Shaoyu Li, Ming Zhou, and Dawei Song. 2022. A survey of controllable text generation using transformer-based pre-trained language models. *arXiv preprint arXiv:2201.05337*.

Chao Zhao, Marilyn Walker, and Snigdha Chaturvedi. 2020. Bridging the structural gap between encoding and decoding for data-to-text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2481–2491.

Xu Zou, Da Yin, Qingyang Zhong, Hongxia Yang, Zhilin Yang, and Jie Tang. 2021. Controllable generation from pre-trained language models via inverse prompting. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 2450–2460.

A Complexity Embedding Id

The English words have six levels. And the Chinese words have seven levels (Diff 1-7). We give the design of the complexity embedding id for this two language in the table 7. Note that, if a word is out of the complexity level vocabulary, its complexity is "$\langle out \rangle$" which is mapping into id 7 in English corpus and 8 in Chinese corpus. In addition, the special tokens such as "$\langle s \rangle$" "$\langle pad \rangle$" "$\langle \backslash s \rangle$" "$\langle unk \rangle$" are the common meaning in data preprocessing for model training.

English		Chinese	
Token	Id	Token	Id
Punctuation	0	Punctuation	0
A1-C2	1-6	Diff 1-7	1-7
$\langle out \rangle$	7	$\langle out \rangle$	8
$\langle sep \rangle$	8	$\langle sep \rangle$	9
$\langle s \rangle$	8	$\langle s \rangle$	9
$\langle pad \rangle$	8	$\langle pad \rangle$	9
$\langle \backslash s \rangle$	8	$\langle \backslash s \rangle$	9
$\langle unk \rangle$	8	$\langle unk \rangle$	9

Table 7: Complexity Embedding Id.

B Details of Datasets Construction

B.1 English Dataset

We adopt the English word complexity levels in the Common European Framework of Reference for Languages (CEFR) [4] which is divided into six complexity levels (A1, A2, B1, B2, C1, and C2). First, we need to restrict the words in the corpus to ensure most of the words are in the complexity level vocabulary. Then, we need to extract keywords from the sentences. In this process, we command the number of keywords is related to the length of the sentence, and the number of keywords is between 1 to 5. Finally, we obtain the complexity information of each sentence through the complexity level vocabulary. The English raw corpus is collected from the monolingual English News dataset in ACL2019 WMT. We select those sentences which have 90% words in the complexity level vocabulary of CEFR. After the processes mentioned above, we get 199k samples in the English corpus, and we split the train, validation and test dataset as shown in the Table 8.

B.2 Chinese Dataset

The word complexity levels in Chinese Proficiency Grading Standards for International Chinese Language Education (CPGS) [5] is divided into six complexity levels (1 to 7). The Chinese raw corpus is collected from 500 textbooks for Chinese learners. These textbooks contain two types of text: essay and dialogue. We split these texts into sentences and throw away those short sentences. If the raw text is a dialogue, after splitting, we need to remove the speaker's name to guarantee it is a proper sentence. Then, we command the number of keywords is related to the length of the sentence, and the number of keywords is between 1 to 5. After the processes mentioned above, we get 156k samples in the Chinese corpus, as shown in the Table 8.

[4] https://www.englishprofile.org/wordlists/evp
[5] http://www.chinesetest.cn

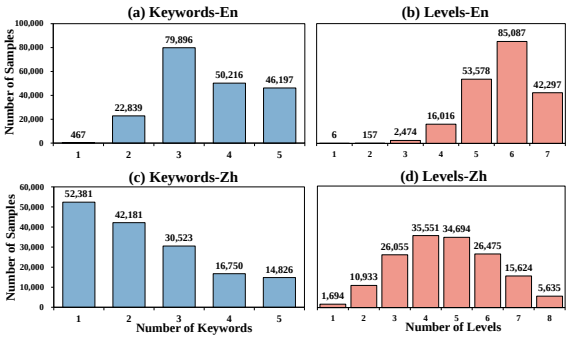

Figure 3: Distributions of the number of keywords and complexity levels.

Dataset	Train	Valid	Test	Total
English	180,000	16,000	3,615	199,615
Chinese	140,000	14,000	2,661	156,661

Table 8: Statistics of the two datasets.

B.3 Analysis of the Datasets

B.3.1 Coverage of Words with Levels

We first analyze the two datasets from the coverage rate of complexity level vocabulary. Due to the requirement of complexity level, the target text is proper to cover most of the vocabulary of complexity level. Both of the two datasets have covered over 93% of the vocabulary of complexity levels.

B.3.2 Distributions of the Number of Keywords and Complexity Levels

One or multiple complexity levels and keywords are given as the input to generate sentences. We give the distribution of the number of keywords and the complexity levels in Figure 3. From the statistics of (a) and (c) in Figure 3, the number of keywords in all samples has covered the range of 1 to 5 both in the English and Chinese datasets, but the distributions are quite different. On account of the average sentence length of English news data is longer than the Chinese corpus, the number of keywords in English is larger. From the statistics in (b) and (d) of Figure 3, the number of complexity levels distribution of the Chinese dataset is close to a standard normal distribution, and the English dataset concentrates on a wider range of complexity levels. This indicates that in the English dataset it tends to use more words of different complexity levels in the same sentence.

C Algorithm of Reranking

The algorithm is the detail of reranking method. We select the sentence that best meets the lexical complexity requirements from the N-best candidates, and $N = 10$. On the test set, We take the sum of ACC score and $F1$ score. The, we choose the candidate that has the largest score.

D Case Study

We choose some cases of the fine-tuning pattern from two datasets. The English cases are in the Table 9, and the Chinese cases are in the Table 10. In both tables, the required keywords as well as appearing in the sentences are shown in blue font, and certain given grades as well as words actually appearing in the sentences for the corresponding grade are shown in red font.

Algorithm 1 Reranking Method

Input: Generated n best candidate sentences $H = (h_0, h_1, h_2, ..., h_{n-1})$ for given keywords and $n = 10$

Output: Sentence having highest score

Let $score = 0$
for each sentence h_j in H **do**
    $ACC = F_{acc}(h_j)$
    $F1 = F_{f1}(h_j)$
    $score_j = ACC + F1$
    if $score_j > score$ **then**
        $score = score_j$
        $ret = h_j$
    end if
end for
return ret

E Related Methods

E.1 Controlled Decoding

The gradients of an external discriminator is directly used to the generation of a pre-trained language model toward the target topic(Dathathri et al., 2019). The output probabilities of a language model is modified by using the output of a discriminator that determines whether the future text will contain the desired attribute. Different from the controlled decoding methods, our method considers the constraint of lexical complexity during both training and prediction.

E.2 Prompting

The prompting method has emerged as a new way to perform natural language processing by conditioning on extra information. Brown et al. propose to use a task description and a few examples to adapt the GPT-3 model to downstream tasks, which is referred to as in-context learning (Brown et al., 2020). Their prompts are manually designed. Gao et al. present LM-BFF for automatic prompts generation(Gao et al., 2020). Liang et al. propose prefix-tuning, which uses continuous vectors as prompts(Li and Liang, 2021). Compared to the prompting method, our method fuses more fine-grained information on lexical complexity in model training.

E.3 Reranking

The reranking approach has been proved to have excellent performance in machine translation (Pandramish and Sharma, 2020) and text generation (Ravaut et al., 2022). The reranking method rescores the n-best candidates through a model or a function and selects the highest scoring candidate as the final prediction (Imamura and Sumita, 2017). Unlike the reranking method, our method do not need to process the outputs after decoding.

F Limitation

Our proposed task has wide applications in the field of language teaching, and the proposed method has precise control over lexical difficulty. However, the task requires that the lexical complexity is known first. The vocabulary difficulty table is the experience summed up by the predecessors, and it is difficult to apply to all vocabulary. Therefore, we are actively exploring how to make the model automatically understand all vocabulary difficulties so that it can cover a wider vocabulary at generation.

Keywords: refuse, accept, country ; **Complexity Levels:** A1, A2, B1, C1	
Ctrl-decoding	I refuse to accept that this is a matter of time for the country, but I do not accept it.
Prompting	I refuse to accept that I am the only one who has been elected to the country, but I am not.
Reranking	I refuse to accept that this is the best thing I can do for the country, ” he said.
CE (ours)	I refuse to accept that this is a country that is going to be a place where people are going to be unsafe.
Keywords: public, maintain, lose ; **Complexity Levels:** A1, A2, B1, B2	
Ctrl-decoding	I do not think the public will maintain the power to lose, but I do not think it will.
Prompting	The public will maintain the public's ability to lose, and the public will not lose, and they will not lose.
Reranking	I don't want to be in public, but I don't want to maintain my weight and lose.
CE (ours)	The public must maintain their faith and not lose , and we will continue to do everything we can to protect them.
Keywords: football, Leicester, City, magical ; **Complexity Levels:** A1, A2, B1, B2, C2	
Ctrl-decoding	I think football is a great way to play for the game and to be able to play for the best of the game against the game against the game against the game and the way we play against the game against the game against the game against the game is not the same, but the way we are magical.
Prompting	I think football is going to be the best of Leicester City, but I think it's a magical moment for us to have to play with.
Reranking	This is a football game for Leicester City, and it's a magical moment for us to be part of the game and be a part of it.
CE (ours)	It's a football club that has been around for a long time and to see Leicester City win the title is magical.

Table 9: Generated examples from the English dataset.

Keywords: 生活 ; **Complexity Levels:** 1, 2, 3, 7	
Ctrl-decoding	生活中，我最喜欢的就是吃面。
Prompting	生活中的一切都是为了生活。
Reranking	我们的生活很美好，也很快乐。
CE (ours)	没有工作压力的生活多让人羡慕啊！
Keywords: 春天, 姑娘 ; **Complexity Levels:** 1, 2, 3	
Ctrl-decoding	春天来了，姑娘又来了，他还在那儿等着我呢。
Prompting	春天来了，姑娘又来了，他的眼睛也变得很美丽。
Reranking	春天到了，一个年轻的姑娘爱上了他。
CE (ours)	春天来了，我认识了一位美丽的姑娘，他非常喜欢这个姑娘。
Keywords: 喜欢 ; **Complexity Levels:** 1, 7	
Ctrl-decoding	我喜欢吃中国菜，我喜欢吃中国菜。
Prompting	我喜欢看中国电影。
Reranking	你喜欢什么样的女孩子呢?
CE (ours)	我最喜欢逗它们玩儿。
Keywords: 别人, 可能, 意思; **Complexity Levels:** 1, 2	
Ctrl-decoding	别人看我懂了，我觉得我有可能去做一个人，意思是要看他。
Prompting	别人可能不会说，如果你觉得自己可能有可能，你可能会觉得自己是个很难的意思。
Reranking	如果别人问你一个问题，你的意思是什么?
CE (ours)	别人可能不知道你的意思，你要做我喜欢的，要我愿意跟别人说。

Table 10: Generated examples from the Chinese dataset.