Controllable Dictionary Example Generation: Generating Example Sentences for Specific Targeted Audiences

Xingwei He, Siu Ming Yiu
Department of Computer Science,
The University of Hong Kong,
Hong Kong, China
hexingwei15@gmail.com, smyiu@cs.hku.hk

Abstract

Example sentences for targeted words in a dictionary play an important role to help readers understand the usage of words. Traditionally, example sentences in a dictionary are usually created by linguistics experts, which are labor-intensive and knowledge-intensive. In this paper, we introduce the problem of *dictionary example sentence generation*, aiming to automatically generate dictionary example sentences for targeted words according to the corresponding definitions. This task is challenging especially for polysemous words, because the generated sentences need to reflect different usages and meanings of these targeted words. Targeted readers may also have different backgrounds and educational levels. It is essential to generate example sentences that can be understandable for different backgrounds and levels of audiences. To solve these problems, we propose a controllable target-word-aware model for this task. Our proposed model can generate reasonable examples for targeted words, even for polysemous words. In addition, our model allows users to provide explicit control over attributes related to readability, such as length and lexical complexity, thus generating suitable examples for targeted audiences. Automatic and human evaluations on the Oxford dictionary dataset show that our model can generate suitable examples for targeted words with specific definitions while meeting the desired readability.

1 Introduction

A dictionary usually consists of targeted words, part-of-speech (POS) tags, definitions and corresponding example sentences. Definitions and their corresponding examples enable audiences to better master new words, understand unfamiliar texts and the usage of the words in typical sentences, where a definition is a simple description for the meaning of the targeted word, and an example shows audiences how to use the word under this definition. Both definitions and examples are critical, playing an important role in language acquisition and natural language understanding. However, it is often the case that audiences cannot find satisfactory example sentences for rarely used or newly coined words. On the other hand, it is time-consuming for experts to create dictionary examples for these words. With the advancement of AI technologies, it is a natural direction to study how to generate dictionary examples automatically, to assist dictionary compilation and help humans understand the corresponding targeted words.

Dictionary example sentence generation aims to generate example sentences for targeted words to reflect their definitions and usages automatically. Recently, definition generation (Noraset et al., 2017; Gadetsky et al., 2018; Ishiwatari et al., 2019) has been extensively studied, yet generating example sentences is not well-studied. To the best of our knowledge, we are the first group to introduce this challenging problem. One main challenge for this task is that targeted words must appear in outputs. Another challenge is that polysemous words (e.g. *'bank'*), which have multiple senses, even multiple POS tags, are ubiquitous. Thus, a polysemous word in generated examples should convey the given sense and POS tag.

Lexically constrained text generation is meant to incorporate some specific keywords into outputs, which has been widely studied. Previous lexically constrained models inject the given keywords into outputs either by manipulating the decoding process (Mou et al., 2015; Hokamp and Liu, 2017), or using the keywords as the initial state and refining it with a series of actions, such as insertion and replacement until it is completed (He and Li, 2021). It is natural to use these lexically constrained models as baselines, since they have solved the first challenge. In response to the second one, we further extend these lexically constrained models by feeding the definition into the encoder and then

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 610 - 627

injecting the targeted word during decoding. However, these models have two inherent drawbacks for this task: (1) During inference, these models generate a sentence based on the definition and force the targeted word to appear in outputs. However, they fail to explore the correlation between the targeted word and input, thus sacrificing the generation quality to ensure the targeted word appears in outputs. (2) These models are computation-intensive, as they need to manipulate the decoding process.

To circumvent these problems, the proposed model is expected to understand this task so that there is no need to interfere with the decoding process. To achieve this goal, we directly feed the targeted word and definition into the model. This simple change brings two advantages over lexically constrained generation models: (1) During training, our model fully explores the correlation between targeted words and definitions, and gradually acquires this task. As a result, even the proposed model does not control decoding, outputs contain targeted words in 99.6% of cases. (2) With the release of control over decoding, our model significantly improves the generation quality and dramatically reduces the inference latency.

Apart from the above two challenges, the proposed model should generate suitable examples to match the readability levels of different audiences, such as children and college students. To address the third challenge, the proposed model is expected to control the readability-related attributes of outputs, namely length and lexical complexity. Inspired by Keskar et al. (2019), the proposed model is trained on discrete control tokens, which are related to the length and lexical complexity of gold example sentences. By doing so, the proposed model will learn to associate the control tokens with the length and lexical complexity of outputs. As a result, we can control the readability of outputs by varying the length and lexical control tokens.

Our contributions are summarized as follows: (1) We introduce the *dictionary example sentence generation* task. (2) We propose a large dataset for dictionary example generation. (3) We propose a controllable target-word-aware model and several baselines for this task[1]. (4) We propose two BERT-based classifiers to automatically evaluate whether the target word in the generated example conveys the given sense and POS tag, respectively. (5) Our experiment results on the Oxford dictionary dataset show that our model outperforms baselines in terms of generation quality, diversity, POS and definition accuracy. More importantly, our model can tailor examples to fit the needs of targeted audiences by controlling the length and lexical complexity.

[1]Our dataset and code are available at https://github.com/NLPCode/CDEG.

2 Problem Statement

Dictionary Example Sentence Generation aims to generate a fluent example $E = \{e_1, \ldots, e_T\}$ for the targeted word w^* under a specific definition $D = \{d_1, \ldots, d_S\}$, where w^* should appear in E and convey D. During training, this task aims to maximize the conditional probability of E:

$$p(E|w^*, D; \theta) = \prod_{t=1}^{T} p(e_t|e_{i<t}, w^*, D; \theta). \quad (1)$$

3 Methodology

3.1 Motivation

Our motivation is to make the model understand *dictionary example sentence generation* so that we do not need to interfere with the decoding process. Intuitively, if the model has mastered the requirements of this task, the model will know reasonable outputs should contain the targeted word under the specific sense when seeing the target word and definition. Driven by this motivation, we use an encoder-decoder architecture, initialized with BART (Lewis et al., 2020), where the encoder directly takes the targeted word and definition as inputs. During training, the model gradually learns to incorporate the targeted word under the specific meaning into output, otherwise, it will suffer a large cross-entropy loss between the predicted distributions of the decoder and golden examples.

To gain control over the readability of outputs, the model is also trained on the readability-related control tokens of gold examples. In this way, the model will gradually learn to correlate the special token with a readability attribute of outputs, otherwise, it will also suffer a large cross-entropy loss. See Section 3.2 for readability-related control tokens. The overview of the proposed model is shown in Figure 1. The encoder input consists of five parts: the targeted word, POS tag, length, lexical complexity, and definition. Each part begins with a special token, indicating the start of this part. For example, <Word> means the following content is the targeted word. The decoder aims to generate examples based on the encoder inputs.

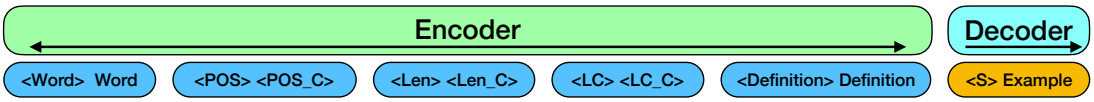


Figure 1: The overview of our proposed model. <Word>, <POS>, <Len>, <LC> and <Definition> are special tokens, which are used to separate the different parts of an encoder input. <POS_C> is the POS tag of a targeted word. <Len_C> and <LC_C> refer to the length and lexical complexity of an example. <S> denotes the start of a sentence.

Words (tokens)	Examples (tokens)
'banked' (**b, anked**)	'a banked racetrack' (a, **Ġbank**, **ed**, Ġrac, etr, ack)
Add an initial space to words/examples	
' banked' (**Ġbank, ed**)	' a banked racetrack' (Ġa, **Ġbank**, **ed**, Ġrac, etr, ack)

Table 1: Tokens are achieved by using the BART tokenizer to tokenize the inputs w/o (top) or w/ (bottom) a leading space. BART uses 'Ġ' to denote a space.

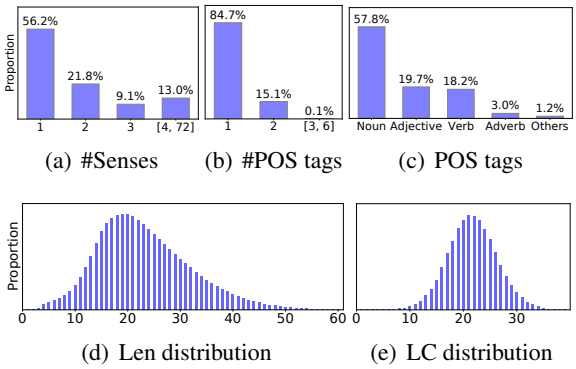
3.2 Readability-Related Control Tokens

To control the readability of outputs, we need to find out which attributes of outputs are related to readability. Flesch-Kincaid Grade Level (FKGL) and Flesch Reading-Ease Score (FRES) (Kincaid et al., 1975) are widely used to assess the difficulty of English text. Both metrics are related to the average sentence length and assume that the longer the sentence, the more difficult the text is to understand. On the other hand, lexical complexity also affects readability (Shardlow, 2014). For example, too many complicated words appearing in a text may hinder audiences' understanding of the text.

Length (Len). Len denotes the number of tokens in a tokenized[2] example. Figure 2 (d) shows that example lengths range from 3 to 60. Hence, we add 58 learnable Len control tokens to the vocabulary.

Lexical Complexity (LC). Word frequencies are the most reliable predictor of word complexity (Paetzold and Specia, 2016). Given this, we use word frequencies as a proxy of LC. In the following, we will show how to compute the LC of an example. First, we tokenize all examples in the training set with NLTK word tokenizer[3]. Next, we rank unique words by word frequencies in descending order. Then, we compute the word ranks for all words in one example. After that, we calculate the third-quartile of log-ranks and use it as the LC for the example. Finally, we discretize all LC values into 40 discrete LC labels. LC label distribution is shown in Figure 2 (e). Therefore, we add 40 trainable LC control tokens (0-39) to the vocabulary.

[2]Sentences are tokenized by BART Tokenizer.
[3]https://www.nltk.org/api/nltk.tokenize.html



(a) #Senses (b) #POS tags (c) POS tags

(d) Len distribution (e) LC distribution

Figure 2: Subfigures (a) and (b) show the unique lemma distribution over #Senses, and #POS tags in the training set, respectively. #Senses denotes the number of unique definition triplets (lemma, POS, definition). In this paper, we use a definition triplet to denote a sense for a lemma. Subfigure (c) shows the distinct definition distribution over POS tags in the training set. Subfigures (d) and (e) show the Len and LC label distributions of examples in the training set.

3.3 Improve the Word Coverage

Since we utilize the BART tokenizer, feeding the original form of a targeted word into the encoder may hinder the decoder from injecting it into the output. As shown in Table 1, the token sequence for the targeted word 'banked' does not appear in the tokenized example (row 1). Therefore, the model must learn to map {**b**, **anked**} to {**Ġbank**, **ed**} to include it in outputs, which undoubtedly increases the difficulty of incorporating the word into outputs. This problem is caused by the discrepancy between the token sequences of the targeted word and example. To solve this, we add an initial space to the targeted word and example so that the tokenized word appears in the tokenized example (row 2). By doing so, the decoder can copy the targeted word from the encoder to outputs instead of mapping, thus improving the word coverage by 8.6%.

3.4 Training and Inference

During training, we feed the targeted word, golden POS, Len and LC labels of examples into the en-

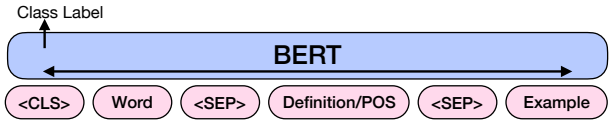


Figure 3: The overview of the BERT-based definition/POS evaluation model. <CLS> is the special symbol for classification output, and <SEP> is the special symbol to separate different parts.

Partition	Training	Validation	Test	Total
#Lemmas	47,889	6,311	6,256	48,352
#Senses	90,044	7,931	7,843	105,818
#Examples	1,138,316	87,373	87,050	1312739
Avg. Sense Len	11.92	11.29	11.31	11.83
Avg. Example Len	20.72	20.56	20.57	20.70
Avg. #Senses p. Lemma	2.19	5.29	5.32	2.84

Table 2: Statistics of the Oxford dictionary dataset. #Lemmas and #Senses denote the number of unique lemma and definition triplets (lemma, POS, definition), respectively. #Examples is the number of examples.

coder, and then fine-tune the model by minimizing the cross-entropy loss. During inference, we set Len and LC to fixed values to generate examples with expected Len and LC.

3.5 Assess the Definition and POS Accuracy

In text style transfer, Shen et al. (2017), Hu et al. (2017) and Li et al. (2018) used a pre-trained classifier to assess whether outputs have the desired attribute. Inspired by this, we propose a definition classifier to evaluate whether the targeted word w^* in the example E conveys the given meaning D. The definition model takes a triple of word, definition and example (w^*, D, E) as input. To train the definition model, we first create the synthetic data $\{(w^*, D, E, L)\}$. If w^* in E conveys D, the label L is 1, denoting the data instance is positive. Otherwise, L is 0, denoting the data instance is negative. We directly select the positive data instance (w^*, D, E) from the Oxford training or validation set. Then, we create three kinds of negative data instances based on a positive data instance by (1) replacing w^* with another word in E or vocabulary; (2) replacing D with another definition of w^* or other words; (3) replacing E with another sentence, which does not contain w^*. For ease of understanding, we show several synthetic data instances in Table 12 in the Appendix. We fine-tune BERT-base (Devlin et al., 2019) on the synthetic training set (see Figure 3 for the model input), achieving 89.9% F1 on the validation set. Similarly, we train a BERT-based POS classifier to assess whether w^* in E reflects the given POS tag, which achieves 98.5% F1 on the synthetic validation set. We show the statistics of synthetic data for the definition and POS models, and their performance on the validation set in Appendix A and B.

4 Experiments

4.1 Experiment Setups

Dataset and Pre-processing. We evaluate our proposed model on Oxford Dictionary[4]. Gadetsky et al. (2018) released a dataset based on this resource for definition generation. However, this dataset is unsuitable for dictionary example generation due to the following limitations (Chang et al., 2018): (1) each definition has only one example sentence; (2) some examples in their dataset do not contain targeted words. To solve these problems, we collect a new Oxford dataset by filtering out definitions with the number of examples less than two, and examples not containing the targeted word. In addition, we remove targeted words containing letters less than two or greater than 20. Each data instance is a quadruplet, containing a targeted word, POS tag, definition and examples of the word usage. We split the dataset into training, validation and test sets based on the triplets (lemma, POS, definition), which are mutually exclusive across three sets (see Table 2 for statistics of this dataset).

Different from the training set, the validation/test set only contains polysemous words with at least two definitions, since it is more challenging to generate examples for polysemy. During training, each sense along with all corresponding example sentences will be used to update models. During inference, we will generate only one example sentence for each sense, but each lemma in English may have multiple inflections (For example, inflected forms of the verb 'bank' include 'banked', 'banking', etc). Given that we use BLEU to evaluate the generation quality, we only keep the word form with most example sentences for each definition tuple (lemma, POS, definition) in the validation/test set. For the sense ('bank', Verb, 'heap (a substance) into a mass or mound') in the test set, two examples contain 'banked' and only one example contains 'banking', so we keep examples containing 'banked'.

The data distributions of the training set are shown in Figure 2. See Appendix C for the data distributions of validation and test sets.

Baselines. We first implement two retrieval baselines by randomly selecting examples containing

[4]https://en.oxforddictionaries.com/

# Models/Metrics	Coverage ↑	POSA ↑	DefA ↑	B-2 ↑	B-4 ↑	SB-4 ↓	D-2 ↑	D-4 ↑	AveLen	Latency ↓
Retrieval Models										
1 One-Billion-Word	96.4%	82.9%	35.6%	12.5%	1.6%	18.7%	53.4%	76.6%	28.7	9.033
2 Training set	97.3%	84.0%	35.6%	17.3%	6.8%	18.2%	**54.3%**	77.1%	27.3	0.371
Lexically Constrained Models without Definitions										
3 sep-B/F	**100.0%**	86.1%	32.6%	25.1%	4.7%	44.8%	29.3%	61.0%	18.2	0.964
4 asyn-B/F	**100.0%**	86.1%	32.3%	24.5%	4.5%	43.0%	30.1%	63.1%	19.6	0.931
5 GBS	**100.0%**	83.8%	33.4%	17.0%	2.5%	61.6%	23.7%	44.4%	19.9	7.854
6 X-MCMC-C	**100.0%**	0.1%	7.5%	15.6%	2.3%	**15.1%**	53.2%	**95.0%**	12.1	24.23
Lexically Constrained Models with Definitions										
7 sep-B/F	**100.0%**	87.7%	77.1%	27.9%	6.4%	30.0%	43.5%	83.2%	15.5	1.002
8 asyn-B/F	**100.0%**	89.6%	77.5%	27.8%	6.2%	30.0%	42.9%	83.9%	16.9	0.991
9 GBS	**100.0%**	91.3%	77.5%	26.3%	6.1%	28.1%	44.6%	84.2%	15.9	8.025
Our Models + Word + POS + Len$_{14}$ + LC$_{25}$										
10 Random (greedy)	99.8%	96.9%	81.8%	28.0%	5.4%	40.5%	34.3%	69.9%	14.0	0.164
11 BART-base (greedy)	99.6%	97.2%	87.7%	28.8%	7.6%	23.9%	49.5%	86.4%	14.0	**0.161**
12 BART-base (beam 5)	99.5%	**97.4%**	**87.8%**	**31.4%**	**9.6%**	26.2%	46.8%	84.2%	14.1	0.195

Table 3: Results on the Oxford test set. For our model, the subscript integers denote the selected control labels for Len and LC, with which the model performs best on the validation set. 'greedy' and 'beam 5' denote generating sentences using greedy or beam search with a beam size of 5. 'AveLen' means the average length of examples. 'Latency' is the average decoding time (second) per sentence computed on the test set without mini-batching.

the targeted words from the One-Billion-Word[5] corpus or the training set, respectively. We adopt four lexically constrained generation models: two variants of the backward forward model (sep-B/F and asyn-B/F) (Mou et al., 2015), grid beam search (GBS) (Hokamp and Liu, 2017) and X-MCMC-C (He and Li, 2021). We implement the former three baselines based on GPT-2 small (117M). We train X-MCMC-C with the code provided by He and Li (2021), which is based on XLNet-base (110M). These methods generate sentences containing targeted words without considering definitions. To remedy this, we re-implement the former three models based on BART-base (139M), where the encoder takes the definition as input and the decoder incorporates the word during inference.

Implementation Details. We initialize our model with BART-base, which has comparable parameters to generation baselines. For generation baselines and our models, we use AdamW (Loshchilov and Hutter, 2019) with an initial learning rate of $1e-5$ to update parameters for four epochs and choose the checkpoints with the lowest validation loss.

During inference, we run beam search decoding with beam width = 5 on generation baselines and our model. We also run greedy decoding on our model. Following He and Li (2021), we run X-MCMC-C for 200 steps and select the example with the lowest negative log-likelihood (NLL) as output. To discourage the generation of repetitive tokens, we apply the repetition penalty strategy Keskar et al. (2019) with the penalized parameter = 1.3 to all models. We implement all models with the HuggingFace Transformers library (Wolf et al., 2019). All models are trained and tested on a single GeForce RTX 2080 Ti GPU.

Evaluation Metrics. We evaluate the generated examples from four aspects: **Q1:** Whether the generated example contains the targeted word? **Q2:** Whether the targeted word in the generated example conveys the given sense? **Q3 & Q4:** Whether the outputs are fluent and diverse? First, we check whether the targeted word appears in the example, indicated as word **Coverage**. If so, we will further assess whether the targeted word conveys the given POS tag and sense with the BERT-based POS and definition models, called POS Accuracy (**POSA**) and Definition Accuracy (**DefA**).

As for Q3, it is non-trivial to evaluate the generation quality. In this paper, we do not use NLL as a metric for sentence fluency, since lower NLL does not always denote better sentence quality (Holtzman et al., 2020). We use **BLEU** (Papineni et al., 2002) to measure the n-gram similarity between the generated examples and human references, which is a widely-used automatic metric for generation quality. One concern is that BLEU may be not ideal for dictionary example generation, since there may exist many sentences that could be appropriate for a given word and definition. To remedy this, each sense (i.e., definition triplet) in the validation and test sets contains an average of 11 examples, which provide a richer and more diverse test-bed for further automatic evaluation. To answer Q4, we use Self-BLEU (Zhu et al., 2018) and Distinct n-gram (Li et al., 2016) to measure the generation diversity. Self-BLEU-4 (**SB-4**) is computed by treating one sentence as the hypothesis, and the first 1K generated sentences excluding the hypothesis as

[5]http://www.statmt.org/lm-benchmark/

references. Distinct bigram (**D-2**) and 4-gram (**D-4**) indicate the proportions of unique bigrams and 4-grams, respectively.

4.2 Experimental Results

Table 3 reports the main experiment results on the test set, from which we can draw four conclusions:
(1) **Generation models are critical.** We cannot retrieve examples for all words. For example, only 97.3% of the words in the test set appear in the training set. We do not see any improvements with a larger dataset (rows 2), yet brings a much higher retrieval latency. By comparison, the generation models have the potential to generate examples for unseen words, thus greatly improving coverages.
(2) **The definition is helpful.** Compared with generation baselines w/o definitions (rows 3-6), their counterparts w/ definitions (rows 7-9) significantly improve DefA to around 77%. As we have mentioned before, all words in the test set are polysemous (see Figure 6 (a)). That is why the definition is useful and indispensable for this task.
(3) **The pre-trained model does matter.** Compared with the random counterpart (row 10), our model initialized with the BART-base model (row 11) can generate more fluent (B-4) and diverse (SB-4, D2, D4) sentences while improving DefA by around 6%. That is possibly because BART acquires some syntactic and semantic knowledge during pre-training, which is useful for this task.
(4) **The proposed models outperform other generation baselines in most metrics.** One problem with lexically constrained generation models (rows 7-9) is that they do not explicitly explore the correlation between the targeted word and input. When feeding a definition into these models, they just generate a sentence based on the encoder input and force the targeted word to appear in outputs. By interfering with decoding, they can achieve 100% word coverage, yet this is achieved at the cost of generation quality, POSA and DefA. Another problem is that their manipulations of decoding cause higher inference latency.

By comparison, our proposed model directly takes the targeted word as input instead of compulsorily injecting it into outputs during inference. This simple change brings two advantages over the lexically constrained generation methods: (1) Our model can fully explore the correlation between the targeted word and the definition, and gradually acquires this task during training. As a result, when feeding a targeted word and a definition into the proposed model, it will understand that the reasonable outputs should contain the targeted word under the specific sense. That is why even the proposed model does not control the decoding process, it does not sacrifice the word coverage (e.g. 99.6% word coverage in row 11). (2) Eliminating interference to decoding brings substantial improvements in generation quality (B-4), POSA, and DefA, and dramatically reduces inference latency.

#	Variants	Coverage↑	POSA↑	DefA↑	B-4↑	D-4↑
1	Full model	99.6%	97.2%	87.7%	7.6%	86.4%
2	– Word	14.5%	17.3%	16.1%	3.6%	85.9%
3	– POS	99.6%	96.6%	87.6%	7.5%	86.6%
4	– Definition	99.4%	97.4%	35.8%	4.3%	73.2%

Table 4: Results of ablation study on the test set. Compared with the full model (row 11 of Table 3), the metric with the largest change in each row is underlined.

Space	Pointer	Coverage↑	POSA↑	DefA↑	B-4↑	D-4↑
		91.0%	89.1%	81.2%	7.1%	87.1%
√		99.6%	97.2%	87.7%	7.6%	86.4%
	√	89.7%	88.0%	79.4%	7.5%	86.5%
√	√	99.5%	97.2%	86.9%	8.2%	85.1%

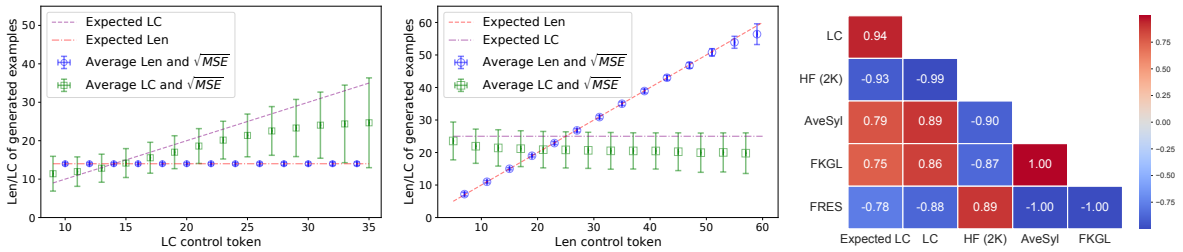
Table 5: Results of ablation study on the test set.

4.3 Ablation Study

We perform an ablation study to demonstrate the importance of each design. We first train variants of the full model by removing the word, POS, and definition, and then run greedy decoding on the well-trained models to generate examples. We show the results on the test set[6] in Table 4. Compared with the full model (row 1), we note that: (1) removing the targeted word significantly decreases the word coverage (row 2). (2) POS helps to improve the POSA (row 3). (3) the definition improves the DefA (row 4). These observations verify the effectiveness of these components. Len and LC control tokens are mainly used to control the readability of outputs, which do not degrade the generated examples (see Table 13 in the Appendix).

We also test the effect of leading space. As shown in Table 5, adding the space increases the word coverage by 8.6%, establishing the importance of this design. The pointer network (Gulcehre et al., 2016; See et al., 2017) is used to copy content from the source into outputs. However, only using the pointer network cannot improve the word coverage, as it does not solve the mapping issue mentioned in Section 3.3. Therefore, we do not use the pointer network.

[6]We observe similar results on the validation set.



(a) Effect of the LC control token (b) Effect of the Len control token (c) Pearson correlation coefficient

Figure 4: The impact of control tokens on generated examples. Subfigures (a) and (b) demonstrate the statistics of the corresponding attributes of the generated examples on the test set, including average, standard deviation (std), and mean squared error (MSE) values. Subfigure (c) shows the Pearson correlation coefficients for LC and a set of metrics. All examples are generated by running greedy decoding on the model (row 11 of Table 3) with Len_{14}+LC (Subfigures (a) and (c)), and Len+LC_{25} (Subfigure (b)) control tokens. Expected LC/Len is the gold LC/Len label used to generate examples; LC/Len means the average lexical complexity/length of outputs.

Cases	Coverage↑	POSA↑	DefA↑	B-4↑	#POS	#Senses
2	99.6%	**97.7%**	**92.2%**	6.1%	1.2	2.0
3	**99.7%**	96.6%	88.8%	6.6%	1.4	3.0
≥4	99.5%	97.4%	85.4%	**8.6%**	1.9	10.6
Noun	99.7%	**98.7%**	88.5%	7.1%	1.5	6.3
Adjective	99.2%	95.1%	87.5%	6.9%	1.6	5.1
Verb	**99.8%**	98.6%	**89.1%**	**9.6%**	1.8	10.2
Adverb	99.6%	84.9%	75.7%	4.5%	1.6	6.1

Table 6: Results on different cases, where the test set is separated according to the #Senses (part one), and POS tags (part two), respectively. All examples are generated by the model (row 11 of Table 3) with Len_{14}+LC_{25}. #Sense and #POS denote the average number of senses and POS tags owned by each word.

4.4 More Analysis and Discussion

Effect of Control Tokens. In Table 13, we have shown that Len and LC control tokens affect readability via HF, but two questions are still unclear: **Q1:** Whether these control tokens have the desired effects on their associated attributes, length and lexical complexity? **Q2:** What is the correlation between LC and readability? To answer Q1, we generate examples by running greedy decoding on our model (row 11 of Table 3) with different control tokens. From Figure 4 (a) and (b), we see that: (1) the average LC and Len of outputs increase linearly with the gold LC and Len labels; (2) the MSE values between Len and gold Len labels are negligible, while the MSE values between LC and gold LC labels are relatively large, especially when LC $>$ 30 indicating that the control ability of the model on LC decreases, possibly due to the limited training data (see Figure 2 (e)). Therefore, we can conclude that Len and LC control tokens do affect their associated attributes.

To answer Q2, we compute the Pearson correlation coefficients (PCC) between LC and two widely used readability metrics, FKGL and FRES. Since LC is based on the word frequency, we compute the PCC between LC and the proportion of high-frequency words with a word rank lower than 2K (HF (2K)). FKGL and FRES are related to the average number of syllables of outputs (AveSyl), so we also compute PCC between LC and AveSyl. As shown in Figure 4 (c), LC values of outputs are strongly positively correlated with the gold/expected LC labels, which again verifies our model's control ability over LC. We also notice that PCC between LC and HF is -0.99, proving that LC can control the other readability-related metrics of outputs by controlling HF. We show more results of control tokens in Appendix H.

Effect of the Number of Senses. As shown in the first part of Table 6, with the increase of the number of definitions, it becomes more and more challenging to generate examples satisfying the definition(s), thus causing a decrease in DefA.

Effect of POS Tags. As shown in the second part of Table 6, our model performs worst on the adverb case, especially in POSA and DefA. We found that our model may ignore the adverb POS tag and use the adjective POS tag (see the targeted word '*worse*' in Table 15). We presume that there are two possible reasons: (1) in the training set, the adverb training data is far less than the adjective data (see Figure 2 (c)), so the adverb embedding may not be well learned and updated; (2) for some adverb, such as *'worse'*, the adjective definition is much more common, so the pre-trained model, BART, may bias towards the adjective meaning.

Overall speaking, as shown in Table 7, our model can generate high-quality examples for different

Words	POS	Definitions	Generated examples
banks	Noun	the land alongside or sloping down to a river or lake	The river **banks** are a natural habitat for wildlife.
banked	Verb	heap (a substance) into a mass or mound	The ground was **banked** with mud and the water had been pumped into a bucket.
star	Noun	an outstandingly successful person or thing in a group	The **star** of the show is a young man who has been in the spotlight for years.
star	Noun	a fixed luminous point in the night sky . . .	The **star** is a bright, luminous object that shines brightly in the night sky.
satisfy	Verb	meet the expectations, needs, or desires of (someone)	The company has been trying to **satisfy** customers with its new products.
satisfy	Verb	fulfil (a desire or need)	The only way to **satisfy** this desire is by making a new one.
sentences	Noun	a set of words that is complete in itself . . .	The first two **sentences** are the same, but they have different meanings.
sentence	Verb	declare the punishment decided for (an offender)	The court will **sentence** him to life in prison.
sentenced			The offender was **sentenced** to a total of six months in prison.
sentencing			The judge was **sentencing** the offender to a maximum of five years in prison.
sentences			The court **sentences** him to life in prison without parole.

Table 7: Examples generated with different words, POS tags and definitions from the test set. All examples are generated by running greedy decoding on the model trained with the targeted word, POS tag, and definition.

Word: banks **POS:** Noun	**Definition:** the land alongside or sloping down to a river or lake
Human	Massive housing projects are springing up on the banks of lakes.
Len$_5$	The banks of the river.
Len$_{10}$	The river banks are a ***haven*** for the fish.
Len$_{20}$	The river banks are a natural habitat for wildlife, and the water is not too ***polluted*** or ***salty***.
Len$_{14}$**+LC**$_{10}$	The river banks are the only way to get water from this area.
Len$_{14}$**+LC**$_{20}$	The river banks are a natural habitat for the birds and their ***larvae***.
Len$_{14}$**+LC**$_{30}$	The river banks are the most ***productive*** of all the ***estuaries***.

Table 8: The impact of control tokens on generated examples. All examples are generated by the model (row 11 of Table 3) with different control tokens. Text in bold and italics denotes low-frequency ***words*** with the word rank higher than 5K.

Models	Fluency	Definition	POS
asyn-B/F	4.08	2.19	77.3%
GBS	4.43	2.27	84.0%
Our model	**4.83**	**2.59**	**90.7%**
Our model	LC_{10}	LC_{20}	LC_{30}
Readability	1.08	1.44	1.68

Table 9: Human evaluation results on the test set for fluency, definition and POS scores, and readability of our model with different LC control tokens are shown at the top and bottom. The differences between models' scores and baselines' are statistically significant due to the paired t-test comparisons (p-value$<$0.05).

words and the same word with different definitions, such as *'star'* and *'satisfy'*. Moreover, our model can generate plausible examples for different inflected forms of words, such as *'sentence'*. Table 8 shows that we can control the length and lexical complexity of examples generated with our model by varying the control labels.

To summarize, our model not only can generate meaningful examples for existing words, but also has a strong control ability over the length and lexical complexity of outputs.

Please refer to Appendix E, F and G for the effect of word frequencies, unseen words and the size of training data. Please refer to Appendix I for more detailed sample analysis.

4.5 Human Evaluation

We conduct a human evaluation to further compare our model with asyn-B/F and GBS (rows 8, 9 and 11 of Table 3). For each model, we randomly select 50 generated examples and invite three annotators [7] to label the sentences. Annotators first rate the sentence fluency on a 5-point Likert scale from 1 (not fluent) to 5 (extremely fluent). Then, annotators assess whether the meaning of the targeted word in the output is the same as the given definition on a 3-point Likert scale, from 1 (totally different) to 3 (exactly the same). Finally, annotators judge whether the POS of the targeted word in the output is consistent with the given POS. We show the detailed annotation method in Appendix D. As shown in Table 9, our proposed model outperforms baselines in human evaluation on all metrics.

PCCs between two automatic evaluation metrics (DefA, POSA) and related human evaluation scores are 73.5% and 90.3% (p-value$<$0.05), indicating positive and strong correlations.

We also conduct a human evaluation to assess our model's control ability over readability. We first randomly select 50 groups of examples generated by our model with different LC (10, 20,

[7] All annotators are Ph.D. students and are independent of our research group.

30) + LC_{14}. Then, we ask annotators to rank the sentences on readability in each group. The most difficult sentence receives a score of 3, the others receive scores of 2, 1. Annotators can give the same rank to different examples if they have no preference. As shown at the bottom of Table 9, the difficulty of the generated sentences increases with LC, verifying that our model can control the readability of outputs via LC control tokens. Inter-rater agreement measured by Fleiss' *kappa* (Fleiss, 1971) is 0.51, 0.73, 0.90 and 0.60 for fluency, definition, POS and readability, indicating moderate, substantial, almost perfect and moderate inter-rater agreement, according to Landis and Koch (1977).

5 Related Work

Word Sense Disambiguation. Word Sense Disambiguation (WSD) (Navigli, 2009) is a fundamental task and long-standing challenge in NLP, which aims to associate an ambiguous word in context with the exact sense from a finite set of possible choices. Previous work formulates the task as a token classification problem (Raganato et al., 2017) or sentence-pair (context and gloss pair) classification problem (Huang et al., 2019). WiC (Pilehvar and Camacho-Collados, 2019) is framed as a binary classification problem, which aims to identify if the occurrences of the targeted word in the first context and second context correspond to the same meaning or not. Our proposed work is related to word disambiguation, yet it is a generation task, which is more challenging.

Controllable Text Generation. Controllable text generation aims to generate text in a controlled way, which has attracted wide attention. One line of research injects pre-specified keywords into outputs by controlling the decoding process (Mou et al., 2015; Hokamp and Liu, 2017; Post and Vilar, 2018) or refining candidate outputs iteratively (Miao et al., 2019; Sha, 2020; He and Li, 2021; He, 2021). Another kind of work uses control tokens to manipulate text attributes, such as the length (Kikuchi et al., 2016; Fan et al., 2018), topic (Ficler and Goldberg, 2017; Keskar et al., 2019), and grade level for text simplification (Scarton and Specia, 2018; Nishihara et al., 2019).

In this paper, we first introduce the dictionary example generation task, which also requires the targeted word to appear in outputs. To this end, we use a target-word-aware model to generate examples for given words. Different from the former line of work, our proposed model does not interfere with the decoding process, thus reducing the inference time and improving the generation quality. Moreover, we expect to tailor-made outputs for different audiences. Inspired by the latter kind of work, our model takes readability-related control tokens to generate suitable example sentences with the desired readability.

Dictionary Example Generation. Two recent works are related to dictionary example generation. One work is GPT-3 (Brown et al., 2020), a large-scale autoregressive language model. To qualitatively test GPT-3's ability for the few-shot task of using a new/nonexistent word in a sentence, Brown et al. (2020) gave GPT-3 the definition of a nonexistent word, such as "screeg", and then asked GPT-3 to use it in a sentence. However, they did not formally define this task.

Similar to our work, another concurrent work (Barba et al., 2021) also gives a formal statement of the dictionary example generation task. However, they did not evaluate the quality of generated examples directly. In their work, they aimed to improve WSD models by augmenting WDS datasets with the generated examples. Compared with their work, we directly evaluate whether the targeted work in the generated example reflects the given sense and POS tag with the proposed BERT-based classifiers. Our work also explores how to generate suitable examples for different targeted audiences.

6 Conclusions

In this paper, we first introduce the dictionary example sentence generation problem, and propose a controllable target-word-aware model and several strong baselines for it. We propose two BERT-based classifiers to evaluate the definition and POS accuracy of generated examples. Our experiment results on the Oxford dictionary dataset show that our model outperforms baselines in most metrics and can generate appropriate examples meeting different audiences' understanding levels.

Acknowledgments

We would like to thank the anonymous reviewers for their constructive and informative feedback. This project is supported by the funding from HKU-SCF FinTech Academy.

References

Edoardo Barba, Luigi Procopio, Caterina Lacerra, Tommaso Pasini, and Roberto Navigli. 2021. Exemplification modeling: Can you give me an example, please. In *Proceedings of IJCAI*, pages 3779–3785.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Ting-Yun Chang, Ta-Chung Chi, Shang-Chi Tsai, and Yun-Nung Chen. 2018. xsense: Learning sense-separated sparse representations and textual definitions for explainable word sense networks. *arXiv preprint arXiv:1809.03348*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, pages 45–54.

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. In *Proceedings of the Workshop on Stylistic Variation*, pages 94–104.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5):378–382.

Artyom Gadetsky, Ilya Yakubovskiy, and Dmitry Vetrov. 2018. Conditional generators of words definitions. In *Proceedings of ACL*, pages 266–271.

Caglar Gulcehre, Sungjin Ahn, Ramesh Nallapati, Bowen Zhou, and Yoshua Bengio. 2016. Pointing the unknown words. In *Proceedings of ACL*, pages 140–149.

Xingwei He. 2021. Parallel refinements for lexically constrained text generation with bart. In *Proceedings of EMNLP*, pages 8653–8666.

Xingwei He and Victor O.K. Li. 2021. Show me how to revise: Improving lexically constrained sentence generation with XLNet. In *Proceedings of AAAI*, pages 12989–12997.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of ACL*, pages 1535–1546.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *ICLR*.

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of ICML*, pages 1587–1596.

Luyao Huang, Chi Sun, Xipeng Qiu, and Xuanjing Huang. 2019. GlossBERT: BERT for word sense disambiguation with gloss knowledge. In *Proceedings of EMNLP*, pages 3509–3514.

Shonosuke Ishiwatari, Hiroaki Hayashi, Naoki Yoshinaga, Graham Neubig, Shoetsu Sato, Masashi Toyoda, and Masaru Kitsuregawa. 2019. Learning to describe unknown phrases with local and global contexts. In *Proceedings of NAACL*, pages 3467–3476.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. CTRL: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling output length in neural encoder-decoders. In *Proceedings of EMNLP*, pages 1328–1338.

J. Kincaid, R. P. Fishburne, R. L. Rogers, and B. Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel. *Technical Report*.

J Richard Landis and Gary G Koch. 1977. The measurement of observer agreement for categorical data. *Biometrics*, 33(1):159–174.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of ACL*, pages 7871–7880.

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A diversity-promoting objective function for neural conversation models. In *Proceedings of NAACL-HLT*, pages 110–119.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: A simple approach to sentiment and style transfer. In *Proceedings of NAACL-HLT*, pages 1865–1874.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. In *ICLR*.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. CGMH: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of AAAI*, pages 6834–6842.

Lili Mou, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. 2015. Backward and forward language modeling for constrained sentence generation. *arXiv preprint arXiv:1512.06612*.

Roberto Navigli. 2009. Word sense disambiguation: A survey. *ACM computing surveys (CSUR)*, 41(2):1–69.

Daiki Nishihara, Tomoyuki Kajiwara, and Yuki Arase. 2019. Controllable text simplification with lexical constraint loss. In *Proceedings of ACL: Student research workshop*, pages 260–266.

Thanapon Noraset, Chen Liang, Larry Birnbaum, and Doug Downey. 2017. Definition modeling: Learning to define word embeddings in natural language. In *Proceedings of AAAI*.

Gustavo Paetzold and Lucia Specia. 2016. Semeval 2016 task 11: Complex word identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 560–569.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL*, pages 311–318.

Mohammad Taher Pilehvar and Jose Camacho-Collados. 2019. WiC: the word-in-context dataset for evaluating context-sensitive meaning representations. In *Proceedings of NAACL*, pages 1267–1273, Minneapolis, Minnesota.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. In *Proceedings of NAACL-HLT*, pages 1314–1324.

Alessandro Raganato, Claudio Delli Bovi, and Roberto Navigli. 2017. Neural sequence learning models for word sense disambiguation. In *Proceedings of EMNLP*, pages 1156–1167.

Carolina Scarton and Lucia Specia. 2018. Learning simplifications for specific target audiences. In *Proceedings of ACL*, pages 712–718.

Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get to the point: Summarization with pointer-generator networks. In *Proceedings of ACL*, pages 1073–1083.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of EMNLP*, pages 8692–8703.

Matthew Shardlow. 2014. A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications*, 4(1):58–70.

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *NIPS*, pages 6830–6841.

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

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. In *SIGIR*, pages 1097–1100.

Models	Labels	Training	Validation
POS	Negative	3,337,293	417,733
	Positive	1,138,316	141,626
Definition	Negative	4,219,163	532,272
	Positive	1,138,316	141,626

Table 10: Statistics of the synthetic training and validation sets for BERT-based POS and definition models.

Models	Labels	P	R	F1
POS	Negative	0.994	0.991	0.993
	Positive	0.974	0.983	0.978
	Macro-average	0.984	0.987	0.985
Definition	Negative	0.968	0.944	0.956
	Positive	0.806	0.882	0.843
	Macro-average	0.887	0.913	0.899

Table 11: Results of the BERT-based POS and definition models on the synthetic validation set. "P" and "R" denote precision and recall.

A BERT-based Definition Evaluation Classifier

The definition classifier takes a triple of word, definition and example (w^*, D, E) as input, which aims to assess whether the targeted word w^* in the example E conveys the given meaning D. To train the definition model, we should create the synthetic data $\{(w^*, D, E, L)\}$. If w^* in E conveys D, the label L is 1, denoting the data instance is positive. Otherwise, the label L is 0, denoting the data instance is negative.

We can directly select the positive data instance (w^*, D, E) from the Oxford training or validation set. For each positive data instance, we first create one negative data instance by replacing the targeted word w^* with another word in the example E with a 50% probability or another word in the vocabulary with a 50% probability. Next, we create at most two negative instances by replacing the definition D with any two definitions of the word. Then, we construct a negative instance by replacing the definition D with any definition of other words. Finally, we create a negative instance by replacing the example E with another sentence with a 50% probability, which does not contain the targeted word w^*. For ease of understanding, we show several synthetic data instances for the BERT-based definition model in Table 12. We show the statistics of the synthetic data in Table 10.

We fine-tune the BERT-base-cased model on the synthetic training set for two epochs with the initial learning rate of $1e{-}5$ and select the best checkpoint on the validation set. We show the performance of the definition model on the synthetic validation set in Table 11.

B BERT-based POS Evaluation Classifier

The POS model takes a triple of word, POS tag, and example (w^*, P, E) as input, which aims to assess whether the targeted word w^* in the example E conveys the given POS tag P. To train the POS model, we should create the synthetic data $\{(w^*, P, E, L)\}$. If w^* in E conveys P, the label L is 1, denoting the data instance is positive. Otherwise, the label L is 0, denoting the data instance is negative.

We can directly select the positive data instance (w^*, P, E) from the Oxford training or validation set. For each positive data instance, we first create several negative data instances by replacing the POS tag P with all other POS tags of the targeted word. Then, we construct at most two negative instances by replacing the POS tag P with any two POS tags not belonging to the targeted word. Finally, we create a negative instance by replacing the example E with another sentence with a 50% probability, which does not contain the targeted word w^*. We show the statistics of the synthetic data in Table 10.

We resort to the same training strategy with the definition model and show the performance of the POS model on the synthetic validation set in Table 11.

C Data Distributions of Validation and Test Sets

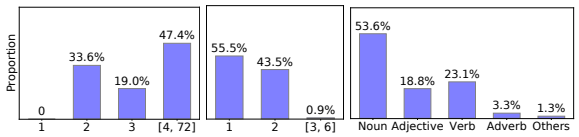
Figure 5/6 (a) and (b) show the unique lemma distribution over the number of senses, and the number of POS tags in the validation/test set, respectively. Similar to the training set, the validation and test sets have ten POS tags (noun, adjective, verb, adverb, preposition, interjection, numeral, pronoun, determiner, conjunction). Figure 5/6 (c) shows the distinct definition distribution over POS tags in the validation/test set.

D Details on Human Evaluation

For human evaluation, we first show graders the inputs used to generate example sentences, consisting of the targeted word, POS tag, and specific definition. Next, we show them a group of sentences generated by asyn-B/F, GBS and our proposed model. To avoid bias, sentences in each

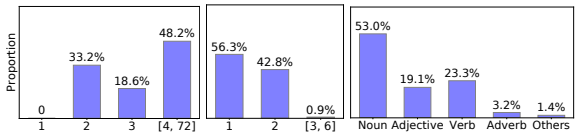
Word	Definition	Example	Label
bank	The land alongside or sloping down to a river or lake.	Willows lined the bank of the stream.	1
stream	The land alongside or sloping down to a river or lake.	Willows lined the bank of the stream.	0
bank	Heap (a substance) into a mass or mound.	Willows lined the bank of the stream.	0
bank	The land alongside or sloping down to a river or lake.	I'm happy with his performance.	0

Table 12: Synthetic data instances for the BERT-based definition model.



(a) #Sense (b) #POS tags (c) POS tags

Figure 5: Subfigures (a) and (b) show the unique lemma distribution over #Senses, and #POS tags in the validation set, respectively. Subfigure (c) shows the distinct definition distribution over POS tags in the validation set.



(a) #Sense (b) #POS tags (c) POS tags

Figure 6: Subfigures (a) and (b) show the unique lemma distribution over #Senses, and #POS tags in the test set, respectively. Subfigure (c) shows the distinct definition distribution over POS tags in the test set.

group are shuffled before annotation. Then, annotators should compare these sentences and score them on three criteria: sentence fluency, POS and definition accuracy.

D.1 Fluency

To evaluate the sentence fluency, graders should answer the first question:
Q1: How fluent do you think the sentence is?

Specifically, graders are asked to score the sentence fluency on a 5-point Likert scale from 1 to 5, based on the following rules:
1: the sentence cannot be understood and all segments are not fluent;
2: the sentence cannot be understood, but some segments are fluent;
3: the sentence can be understood to some extent, but with many grammatical errors;
4: the sentence can be understood with several grammatical errors;
5: the sentence is extremely fluent without any grammatical errors.

D.2 Definition Accuracy

To assess the definition accuracy, graders need to answer the second question:
Q2: How consistent do you think the meaning of the targeted word in the generated sentence is with respect to the given definition?

To finish this task, we ask graders to refer to all definitions and examples of the targeted word on Oxford Dictionary. Concretely, graders are asked to score the definition accuracy on a 3-point Likert scale from 1 to 3, based on the following rules:
1: the meaning reflected by the targeted word in the generated sentence is totally different from the given definition;
2: the meaning reflected by the targeted word in the generated sentence is similar or relevant to the given definition;
3: the meaning reflected by the targeted word in the generated sentence is exactly the same as the given definition.

Suppose that the targeted word, POS tag, given definition are *'bank'*, verb, *'Heap up (a fire) with tightly packed fuel so that it burns slowly.'* We will ask graders to label the following four examples as 1, 1, 2 and 3.
(1) *'a grassy bank'*
In this example, the POS tag of *'bank'* is a noun different from the given POS tag, so graders need to label this example as 1.
(2) *'I banked the aircraft steeply and turned.'*
In this example, the POS tag of *'bank'* is a verb, yet it conveys an entirely different meaning *'(with reference to an aircraft or vehicle) tilt or cause to tilt sideways in making a turn.'* Therefore, we ask graders to label this sentence as 1.
(3) *'Purple clouds banked up over the hills.'*
Graders need to label this as 2, since *'bank'* conveys a relevant meaning (*'Form into a mass or mound.'*) to the given definition.
(4) *'She banked up the fire.'*
Graders are asked to label this as 3, since *'bank'* exactly reflects the given definition.

# Variants	Coverage↑	POSA↑	DefA↑	B-4↑	D-4↑
1 BART-base	99.6%	97.2%	87.7%	7.6%	86.4%
2 – Word	14.5%	17.3%	16.1%	3.6%	85.9%
3 – POS	99.6%	96.6%	87.6%	7.5%	86.6%
4 – Definition	99.4%	97.4%	35.8%	4.3%	73.2%
5 – Len	99.6%	97.4%	87.4%	7.2%	83.5%
6 – LC	99.7%	97.2%	88.1%	7.7%	84.2%

Table 13: Results of ablation study on the test set.

Cases	Coverage↑	POSA↑	DefA↑	B-4↑	#POS	#Senses
[1, 1K]	99.5%	94.3%	74.3%	7.8%	2.2	19.8
(1K, 2K]	99.3%	97.1%	83.5%	8.9%	1.9	13.9
(2K, 5K]	99.1%	97.6%	83.5%	8.8%	1.7	10.7
(5K, 10K]	99.6%	97.5%	85.7%	**9.1%**	1.6	8.0
>10K	**99.7%**	**97.7%**	**91.0%**	7.0%	1.5	4.1
Seen	99.6%	97.1%	87.1%	**7.9%**	1.6	7.7
Unseen	**99.7%**	**98.8%**	**92.3%**	6.0%	1.5	2.9

Table 14: Results on different cases, where the test set is separated according to word frequencies (part one) and unseen/seen words (part two), respectively. All examples are generated by running greedy decoding on the model (row 11 of Table 3) with Len_{14}+LC_{25}. #Sense and #POS denote the average number of definitions and POS tags owned by each word.

These examples and definitions are extracted from Oxford Dictionary.

D.3 POS Accuracy

As for POS, annotators should judge whether the POS tag of the targeted word in the generated example is consistent with the given POS tag.

E Effect of Word Frequencies

As shown in the first part of Table 14, high-frequency words have more definitions and POS tags, and in turn have lower POSA and DefA. As for the sentence quality, high-frequency words have more definitions, while low-frequency (rare) words may not appear in the training set. Both factors may hinder the model from generating satisfactory sentences. That is why words in range $(5K, 10K]$ have the best generation quality (B-4).

F Effect of Unseen and Seen Words

We split the words in the test set into seen and unseen. If a word in the test set has at least one definition in the training set, it will be regarded as a seen word. Otherwise, it will be treated as unseen. The bottom of Table 14 shows that seen words have more #Def than unseen words, since most seen words are high-frequency words, resulting in a lower DefA.

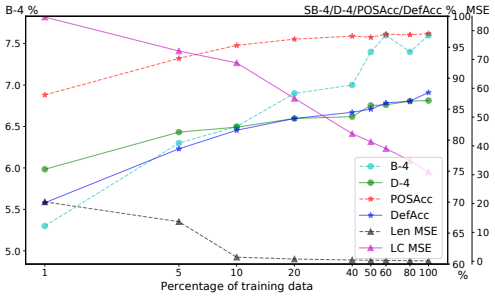


Figure 7: Results of our proposed model variants (row 11 of Table 3) trained with different size of training data.

G Effect of the Size of Training Data

From Figure 7, we see that the size of training data matters, as there is a clear performance gain in sentence quality (B-4), diversity (D-4), definition accuracy and POS accuracy when the data size increases. In addition, the control ability of our model over Len and LC also improves with the increase of the data size.

H Effect of Control Tokens

To further evaluate the effect of control tokens, we generate examples by running greedy decoding on the model (row 11 of Table 3) with different control tokens and show the results on the test set in Figure 8. We witness that the proportion of high-frequency (HF 2K) words significantly reduces from around 90% to 30% with the increase of LC, which again verifies that LC affects the readability of generated examples (see row 4, col 1-3 of Figure 8). However, there is an opposite trend for NLL, in line with the conclusion given by Holtzman et al. (2020) that generic (high-frequency) text tends to have low NLL. As for the generation diversity, the model achieves the highest D-4 when LC is around 25, where the model can balance the high-frequency and low-frequency words.

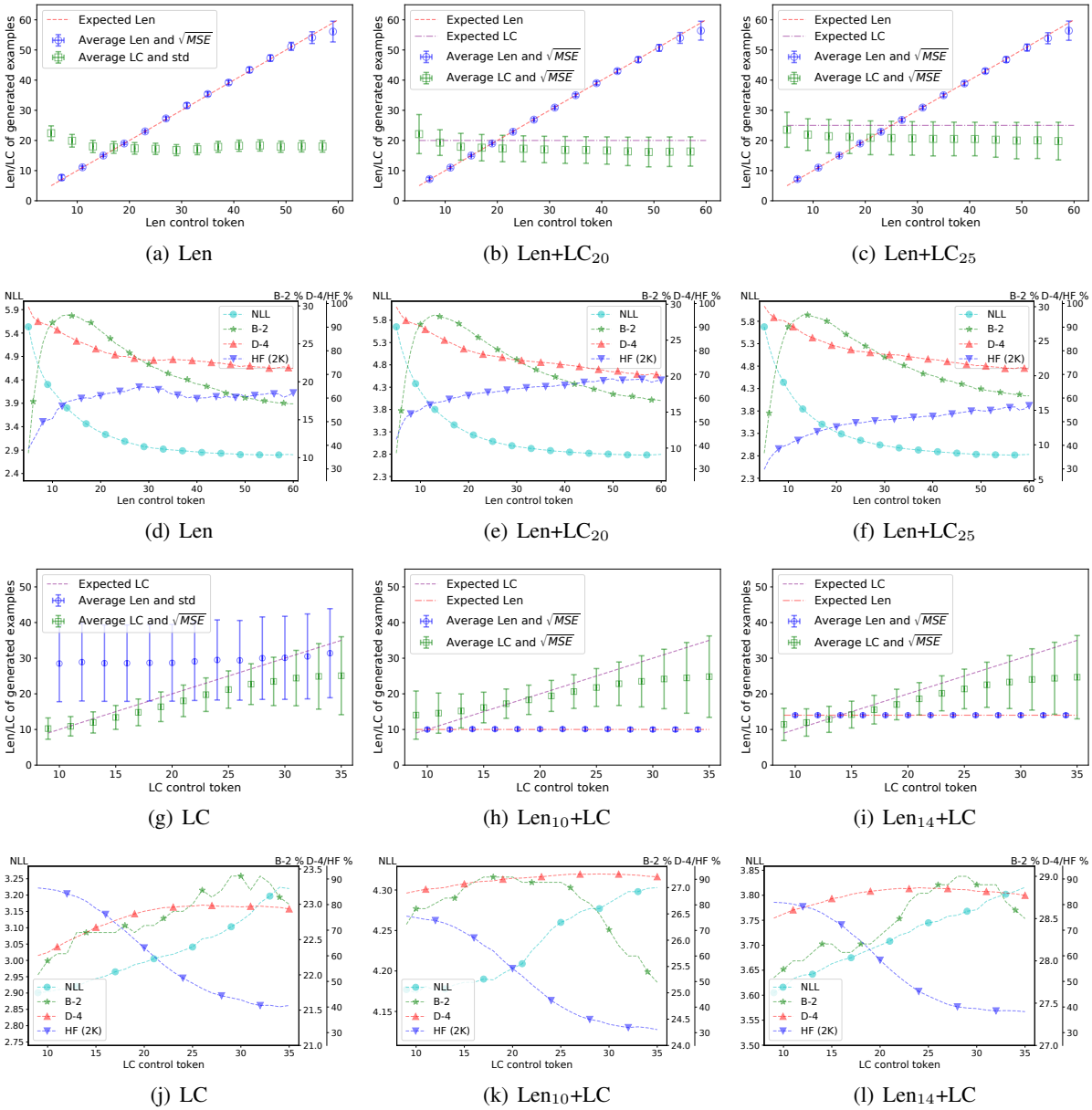
I Further Sample Analysis

We show some example sentences generated by different models in Table 15. Compared with baselines, our models can generate fluent example sentences, and the targeted words in the generated sentences can reflect the given meanings in most cases. For example, for the first case (*'happy'*), both sep-B/F and asyn-B/F fail to generate meaningful sentences. Although GBS can generate a

fluent sentence, the meaning reflected by the targeted word is not the same as the given definition. As for the fourth case (*'plain'*), all baselines ignore the given definition and use the more general definition (*'Not decorated or elaborate; simple or basic in character.'*) and adjective POS tag to generate examples. By comparison, all our models can generate satisfying example sentences.

However, for the last case (*'worse'*), our models seem to ignore the adverb POS tag and use the adjective POS tag. We presume that there are two possible reasons: (1) in the training set, the adverb training data is far less than the adjective data, so the adverb embedding may not be well learned and updated; (2) for the given word, *'worse'*, the adjective definition is much more common, so the pretrained model, BART, may have a bias towards the adjective meaning. Therefore, it is still challenging to generate example sentences for polysemous words with uncommon definitions and POS tags.

To demonstrate the impact of Len and LC control tokens on generated examples, we show some examples generated by running greedy decoding on our proposed model (row 11 of Table 3) with different control labels in Table 16.



(a) Len (b) Len+LC$_{20}$ (c) Len+LC$_{25}$

(d) Len (e) Len+LC$_{20}$ (f) Len+LC$_{25}$

(g) LC (h) Len$_{10}$+LC (i) Len$_{14}$+LC

(j) LC (k) Len$_{10}$+LC (l) Len$_{14}$+LC

Figure 8: The impact of control tokens on generated examples. The first and third rows demonstrate the statistics of the corresponding attributes of the generated examples on the test set, including average, standard deviation (std), and mean squared error (MSE) values. The second and fourth rows illustrate the results on NLL, B-2, D-4 and HF (2K) of the test set. All examples are generated by running greedy decoding on the model (row 11 of Table 3) with **Len** (rows 1-2, col 1), **Len+LC$_{20}$**, (rows 1-2, col 2), **Len+LC$_{25}$**, (rows 1-2, col 3), **LC** (rows 3-4, col 1), **Len$_{10}$+LC**, (rows 3-4, col 2), and **Len$_{14}$+LC**, (rows 3-4, col 3) control tokens. **Len** means sentences are generated using different Len without using LC. **Len+LC$_{20}$** denotes sentences are generated using different Len and a fixed LC of 20. The meaning of other abbreviations can be inferred from these two abbreviations.

Word	POS	Definition
happy	Adjective	satisfied with the quality or standard of
Human reference		Are you **happy** with the role of these representative groups in the diversity programme?
Lexically Constrained Models w/ Definitions		
sep-B/F		A quality **happy** hour. It was well worth the wait, and we were very pleased with ourselves for our efforts.
asyn-B/F		A quality **happy** hour. The food was good and the service excellent, but it wasn't cheap either.
GBS		The quality of the food is not very good, but it's **happy** to eat what you like.
Our Models+Word+POS+Len$_{14}$+LC$_{25}$		
BART-base (greedy)		I'm not **happy** with the way they've been treated by me.
BART-large (greedy)		I'm not **happy** with the quality of my own e-mail.
BART-large (beam 5)		I wasn't **happy** with the quality of the food and the service.
BART-large (top-k 50)		I am **happy** with the quality of the product and its ergonomics.
happy	Adjective	willing to do something
Human reference		United Future is always pleased and **happy** to support sensible legislation.
Lexically Constrained Models w/ Definitions		
sep-B/F		That's not to say that he is one of the **happy** ones, but I think it would be better for him.
asyn-B/F		That's not to say that he is one of the **happy** people I've ever met.
GBS		I am **happy** to say that the Government has decided not to do anything about it.
Our Models+Word+POS+Len$_{14}$+LC$_{25}$		
BART-base (greedy)		I'm **happy** to be a part of the new millennium's celebrations.
BART-large (greedy)		I'm **happy** to do that, but don't be so harsh.
BART-large (beam 5)		I'm **happy** to do that, but I don't like it.
BART-large (top-k 50)		I'm **happy** to do that as long as it doesn't hurt.
sample	Verb	try the qualities of (food or drink) by tasting it
Human reference		There will be cookery demonstrations, videos, information and, of course, a chance to **sample** some of the delicious food that will be on display.
Lexically Constrained Models w/ Definitions		
sep-B/F		You will be able to taste and **sample** a variety of wines from the region.
asyn-B/F		You will be able to taste and **sample** a range of wines, beers and spirits from around the world.
GBS		The **sample** was tastefully tanned and tasted very well.
Our Models+Word+POS+Len$_{14}$+LC$_{25}$		
BART-base (greedy)		We **sample** the wines and savoury snacks of our local restaurants.
BART-large (greedy)		The judges **sample** the wines and then invited their guests to sample them.
BART-large (beam 5)		Guests will be able to **sample** a variety of cheeses and wines.
BART-large (top-k 50)		We sampled the wines and appetizers and had a chance to **sample**.
plain	Adverb	used for emphasis
Human reference		Your statement on Nicaragua shows how utterly naive and just **plain** stupid you are.
Lexically Constrained Models w/ Definitions		
sep-B/F		This is in stark contrast to the **plain** fact that it's not just a matter of whether or not you want them.
asyn-B/F		This is in stark contrast to the **plain** fact that most of us have no idea what we are talking about.
GBS		I'm not sure if it's a good thing or bad, but there is something **plain** and simple about this.
Our Models+Word+POS+Len$_{14}$+LC$_{25}$		
BART-base (greedy)		The whole thing is just **plain** uninteresting, and it's frustrating.
BART-large (greedy)		I'm just **plain** tired of the constant barrage of e-mails.
BART-large (beam 5)		It's just **plain** tacky, and I don't like it.
BART-large (top-k 50)		It was just **plain** rude, and I didn't mean to offend.
worse	Adverb	more seriously or severely
Human reference		At this point, Kohaku's stomach had already began hurting far **worse** than Muteki's.
Lexically Constrained Models w/ Definitions		
sep-B/F		He said he was seriously **worse** off than before and had to be taken out of hospital for treatment.
asyn-B/F		He said he was seriously **worse** affected by the accident and had been taken to hospital for treatment.
GBS		He was seriously injured in a car accident on the way home from work and **worse** still had to be taken by ambulance.
Our Models+Word+POS+Len$_{14}$+LC$_{25}$		
BART-base (greedy)		The situation has gotten **worse** since the end of the Cold War.
BART-large (greedy)		The situation is getting **worse**, and the people are being scapegoated.
BART-large (beam 5)		He was in a wheelchair and his condition was getting **worse** and worse.
BART-large (top-k 50)		He was seriously ill, but his condition didn't get **worse** overnight.

Table 15: Example sentences generated by different models with definition triplets (word, POS, definition) extracted from the test set. 'top-k 50' refers to running top-k decoding with k=50 on our model.

Word: banks **POS:** Noun	**Definition:** the land alongside or sloping down to a river or lake
Human reference	Massive housing projects are springing up on the banks of lakes.
Len$_5$	The banks of the river.
Len$_{10}$	The river banks are a ***haven*** for the fish.
Len$_{15}$	The river banks are a natural habitat for the birds and their ***migratory***.
Len$_{20}$	The river banks are a natural habitat for wildlife, and the water is not too ***polluted*** or ***salty***.
Len$_{25}$	The river banks are a natural habitat for wildlife, and the water is not too ***salty*** to be used as an ***aquatic*** environment.
Len$_{30}$	The river banks are a natural habitat for wildlife, and the water is not too ***salty*** to be used as an ***aquatic feeder*** or even a ***fertilizer***.
Len$_5$**+LC**$_{25}$	The banks of the ***Thames***.
Len$_{10}$**+LC**$_{25}$	The banks of the River ***Thames*** are also ***flooded***.
Len$_{15}$**+LC**$_{25}$	The river banks are a ***haven*** for the wildlife and its ***inhabitants***, too.
Len$_{20}$**+LC**$_{25}$	The river banks are a ***haven*** for the wildlife, and they're also an important habitat for birds.
Len$_{25}$**+LC**$_{25}$	The river banks are a ***haven*** for the wild, and there is no need to worry about the ***dangers*** of ***overfishing***.
Len$_{30}$**+LC**$_{25}$	The river banks are a ***haven*** for the wild, and there is no need to worry about the ***waterlogged*** trees that litter the ***shoreline***.
Len$_{14}$**+LC**$_{10}$	The river banks are the only way to get water from this area.
Len$_{14}$**+LC**$_{20}$	The river banks are a natural habitat for the birds and their ***larvae***.
Len$_{14}$**+LC**$_{30}$	The river banks are the most ***productive*** of all the ***estuaries***.
Word: banked **POS:** Verb	**Definition:** deposit (money or valuables) in a bank
Human reference	Obviously, we banked the money because we didn't know whether or not it was a donation.
Len$_5$	A ***banked*** account.
Len$_{10}$	The money was ***banked*** in a ***Swiss franc***.
Len$_{15}$	The money was ***banked*** in a trust account and the company's shares.
Len$_{20}$	The money was ***banked*** in a trust account and the company's accounts were transferred to its ***subsidiary***.
Len$_{25}$	The money was ***banked*** in a trust account, which is now being used to pay for the purchase of new homes and ***apartments***.
Len$_{30}$	The ***banked*** money was ***deposited*** in a safe ***deposit*** box, which is where the cash will be stored for up to three months before it goes into ***circulation***.
Len$_5$**+LC**$_{25}$	A ***banked*** loan.
Len$_{10}$**+LC**$_{25}$	The money was ***banked*** in a ***Swiss franc***.
Len$_{15}$**+LC**$_{25}$	The money was ***banked*** in a trust account at the Bank of England.
Len$_{20}$**+LC**$_{25}$	The money was ***banked*** in a trust account and the company's assets were transferred to its ***subsidiaries***.
Len$_{25}$**+LC**$_{25}$	The money was ***banked*** in a trust account, which is now owned by the Bank of England and ***administered*** through its ***subsidiaries***.
Len$_{30}$**+LC**$_{25}$	The money was ***banked*** in a trust account, which is now owned by the Bank of England and has been ***deposited*** into an ***escrow*** fund.
Len$_{14}$**+LC**$_{10}$	The money was ***banked*** in the first place and sent to us.
Len$_{14}$**+LC**$_{20}$	The money was ***banked*** in a trust account and sent to China.
Len$_{14}$**+LC**$_{30}$	The money was ***banked*** in the Bank of England's ***Money Reserve***.

Table 16: The impact of control tokens on generated examples. All examples are generated by running greedy decoding on our proposed model (row 11 of Table 3) with different control tokens. Text in bold and italics denotes low-frequency ***words*** with the word rank higher than 5,000.