In-sample Curriculum Learning by Sequence Completion for Natural Language Generation

Qi Jia[1], **Yizhu Liu**[2], **Haifeng Tang**[3], **Kenny Q. Zhu**[4*]
[1,4]Shanghai Jiao Tong University, Shanghai, China
[2]Meituan, Shanghai, China
[3]China Merchants Bank Credit Card Center, Shanghai, China
[1]Jia_qi@sjtu.edu.cn, [2]liuyizhu@meituan.com
[3]thfeng@cmbchina.com, [4]kzhu@cs.sjtu.edu.cn

Abstract

Curriculum learning has shown promising improvements in multiple domains by training machine learning models from easy samples to hard ones. Previous works which either design rules or train models for scoring the difficulty highly rely on task-specific expertise, and cannot generalize. Inspired by the "easy-to-hard" intuition, we propose to do in-sample curriculum learning for natural language generation tasks. Our learning strategy starts training the model to generate the last few words, i.e., do sequence completion, and gradually extends to generate the whole output sequence. Comprehensive experiments show that it generalizes well to different tasks and achieves significant improvements over strong baselines.

1 Introduction

Curriculum learning (CL) proposed by Bengio et al. (2009) provides performance improvements on a number of machine learning tasks. It mimics the learning process of humans by training models with samples in a more meaningful order, i.e., from the easy ones to the hard ones. Therefore, ranking training samples by difficulty lies in the core of CL, which is also the key challenge when it's applied to natural language generation (NLG) tasks.

Previous work on CL for NLG focuses on measuring the difficulty of training samples in two ways. One is to resort to human-crafted rules based on various linguistic features and human observations (Liu et al., 2018; Kocmi and Bojar, 2017). The other uses models either trained from outside data or the same data but in previous epochs/steps (Zhou et al., 2020; Kumar et al., 2019; Shen and Feng, 2020). Either way seeks to produce a numeric score for each training sample relying on domain expertise so that it can be ranked, making it difficult to generalize to different tasks. For example, summarization focuses more on generating concise outputs while style transfer emphasizes style changes. So the former should pay attention to the ratio between the lengths of the output and the input (the more compressed the more difficult), while the latter should focus on differences in style between the input and output (the more different the more difficult). Designing a comprehensive or universal scoring function is difficult or even impossible under this definition of CL.

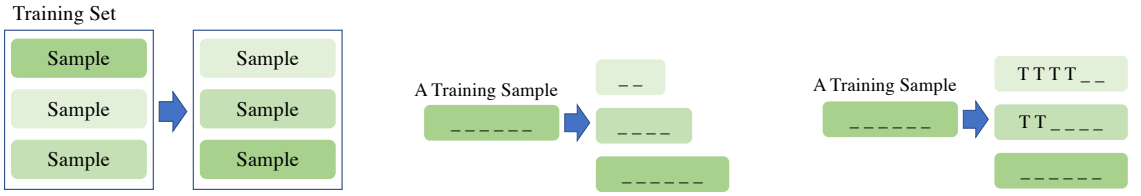
In this paper, we propose an alternative to sample-wise CL, which we call in-sample CL (ICL). ICL re-orders the learning sequence within the sample. One particular ICL re-ordering strategy which we find effective is to predict the last few tokens given a long prefix first from the original output, and then gradually increase the number of tokens at the end while shortening the prefix, to create an easy-to-hard training order. Such a curriculum learning strategy focuses more on the difficulty of language generation itself, leading to a better generalization ability among tasks.

Actually, we are not the first to propose the idea of ICL. Liang et al. (2021) introduced the notion of "token-wise curriculum learning(TCL)". Illustrations of TCL, ICL and the traditional CL are shown in Figure 1. Their work considers generating the first few tokens in the output sequence to be easier than generating a longer sequence in the output. Based on this idea, they proposed a "hard" version of TCL that creates training samples of increasing output length by cutting the sentences short. In this way, TCL is similar to data augmentation with incomplete and even "incorrect" samples, while our ICL considers each training sample in full length. A "soft" version of TCL that places decaying weights on the end tokens instead of cutting short is introduced as a mitigation to avoid incomplete samples, which was proved to uniformly outperform the "hard" version.

To validate the advantage of ICL, we conduct ex-

* The corresponding author.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 11937–11950



(a) Sample-wise Curriculum Learning (b) Token-wise Curriculum Learning (c) In-sample Curriculum Learning

Figure 1: Illustrations of the main ideas of traditional sample-wise CL, Liang et al. (2021)'s TCL and our ICL. Green box refers to samples in different difficulty levels. The darker, the harder. "T" refers to a known token while "_" refer to a token to be generated in the output sequence of a sample during training.

tensive experiments on a range of natural language generation tasks, including reading comprehension, dialogue summarization, style transfer, question generation and news summarization, with different backbone models, such as BART, UniLM and GPT-2. The results show the favorable performance of ICL over the strong baselines.

In a word, our contributions are:

- We propose an improved in-sample curriculum learning strategy for text generation by doing sequence completion (Section 2.1).
- We propose a novel ICL learning algorithm (Section 2.2). Together with our sequence completion ICL curriculum, it achieves significant improvements over the strong baselines on different NLG tasks, demonstrating strong generalization ability (Section 3).
- Our approach can be combined with traditional CL for further performance gains (Section 4.3).

2 Approach

We present an ICL strategy in the context of the vanilla sequence-to-sequence (Seq2Seq) training objective with a detailed learning algorithm.

2.1 ICL by Sequence Completion

Today, NLG tasks are generally solved by Seq2Seq models, especially the pre-trained language models. Vanilla Seq2Seq models are trained to predict the output $Y = \{y_1, ..., y_n\}$ given the input X by minimizing the negative log-likelihood:

$$L_{orig} = -\frac{1}{n}\sum_{t=1}^{n} \log P(y_t|y_{<t}, X) \quad (1)$$

Traditional CL manipulates the selection of training pair (X, Y) from easier pairs to harder ones for different tasks with this vanilla loss function.

In contrast, ICL digs into the output sequence itself and exploits the difficulty of language generation within each training sample. We segment Y into two sub-sequences by a cutting point c, where $1 \leq c \leq n$. The sub-sequence before c is called the *prefix*, and the one after (and including) c is the *target*. According to the Shannon Information Theory, the entropy goes down when more related information is given. Thus, the difficulty of the sequence completion task that generates the target will decrease when a longer prefix is given. In other words, we can manipulate c to vary the difficulty of samples during training.

Based on this intuition, we modify the vanilla loss as:

$$L_{icl} = -\frac{1}{n-c+1}\sum_{t=c}^{n} \log P(y_t|y_{<t}, X) \quad (2)$$

i.e., given X and the prefix as inputs to the encoder and decoder respectively, we only calculate the loss for predicting the target. At the beginning of the training process, we use a larger c to train the model to predict the target with only the last few words. Then, we gradually decrease c, until the prefix reduces to an empty sequence. In this way, the model grows stronger with more difficult generation objectives and learns to generate the whole output in the end. An illustration is in Figure 2.

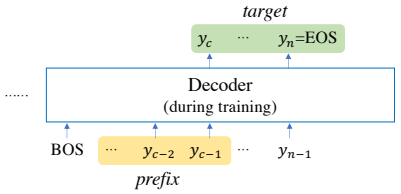


Figure 2: The decoder of a NLG model. BOS and EOS are special tokens representing the beginning and the end of the output. It's suitable for different model architectures, including encoder-decoder, decoder-only, etc.

2.2 ICL Algorithm

Since the output length varies from sample to sample, it's hard to set c as a constant for all samples. If so, samples with short outputs will be neglected when c is large at the beginning, and the model will eventually bias to training samples with long outputs as they are shown more times. In light of this, we proposed to determine c sample by sample relative to their output lengths.

We define a start point p_{start} and a stride s for controlling c, where $0 \leq p_{start}, s \leq 1$. The training process starts with:

$$c = n \times p_{start} \tag{3}$$

After each epoch or a number of updating steps, we validate the model on the validation set. If the performance on the validation set no longer increases, we introduce a more difficult generation task by removing s from p_{prev}:

$$p_{new} = \begin{cases} p_{prev} - s, & \text{if } p_{prev} > s \\ 0, & \text{else} \end{cases}$$

and update c by Equation 3. The training process terminates until there are no improvements on the validation set with c equaling 0. More details are included in Algorithm 1.

Algorithm 1 The ICL training algorithm.

Input: the model to be fine-tuned M_{in}, the training set D_t, the validation set D_v
Parameter: a start point p_{start}, a stride s
Output: the final model M_{out}

procedure ICL($M_{in}, D_t, D_v, p_{start}, s$)
  $p = p_{start}$
  $M_{out} = M_{in}$
  for training epoch $e = 1, ...$ **do**
    ▷ Training process
    for training steps in an epoch **do**
      Randomly sample a batch B from D_t
      for Each sample (X, Y) in B **do**
        $c = n \times p$
        Calculate L_{icl} by Eq. 2
      end for
      Update M_{in} based on $\frac{1}{|B|}\sum_{|B|} L_{icl}$
    end for
    ▷ Validation process
    Calculate M_{in}'s performance on D_v.
    if M_{in} gets improvements on D_v **then**
      $M_{out} = M_{in}$
    else
      Update p according to Eq. 3
    end if
  end for
  return M_{out}
end procedure

3 Experiment

In this section, we first present the experimental setups for different tasks. Then, we show the quantitative and qualitative results together with comprehensive analysis and case studies.

3.1 Experimental Setups

We did experiments on five commonly-researched natural language generation tasks as follows:

Reading comprehension is the task that answering questions about a piece of text. We use the DREAM dataset (Sun et al., 2019) where questions are about corresponding dialogues and the answer is a complete sentence in natural language. We neglect the negative choices in the original dataset and formulate it as a NLG task. We adopt the pre-trained language model BART [1] (Lewis et al., 2020) as the baseline. The generated answers are evaluated by BLEU scores (Papineni et al., 2002) widely used for QA systems, together with Meteor and Rouge-L F1 (Fabbri et al., 2021). We evaluate the model after each training epoch and the early-stop patience will be added 1 if there is no improvement in the perplexity on the validation set. The training process terminates when the early-stop patience equals or is larger than 3. During the inference, the minimum and maximum output length are set to 5 and 100, with no_repeat_ngram_size=3, length_penalty=1.0 and num_beams=4.

Dialogue summarization is to generate a concise summary covering the salient information in the input dialogue. The preceding model BART has shown to be a strong baseline for this task.We experiment with SAMSum dataset (Gliwa et al., 2019) for daily-chat dialogues. The generated summaries are evaluated by comparing with the reference through evaluation metrics, including Rouge-1/2/L F1 scores (Lin, 2004), Meteor (Banerjee and Lavie, 2005) and BertScore F1. The parameters are the same as reading comprehension, except that the early-stop is activated if there is no improvement according to the Rouge-2 F1 score.

Style transfer preserves the semantic meaning of a given sentence while modifying its style, such as positive to negative, formal to informal, etc. We adopt the Shakespeare author imitation dataset (Xu et al., 2012), containing William Shakespeare's original plays and corresponding modernized versions. Krishna et al. (2020) proposed to do unsupervised style transfer by training paraphrase models

[1] https://huggingface.co/facebook/bart-large

Task	Dataset	Model	#Train	#Val	#Test	Input	Output	Avg	Std
Reading Comprehension	DREAM	BART	6,116	2,040	2,041	"Q:"+ question + dialogue	answer	5.59	2.61
Dialogue Summarization	SAMSum	BART	14,732	818	819	dialogue	summary	24.99	13.06
Style Transfer	Shakespeare	STRAP (GPT-2)	36,790	2,436	2,924	original /modern	modern /original	11.63	8.19
Question Generation	SQuAD1.1	UniLM	75,722	10,570	11,877	passage + [SEP] + answer	question	13.09	4.27
News Summarization	CNNDM	BART	287,227	13,368	11,490	document	summary	70.97	29.59

Table 1: A summary of tasks and datasets. #Train, #Val and #Test refers to the number of samples in the corresponding dataset. Avg and Std are the statistics for the number of output tokens. "+" is the concatenation operation.

based on the GPT-2 language model (Radford et al., 2019). We re-implemented their approach STRAP. Evaluation metrics include transfer accuracy(ACC), semantic similarity(SIM), Fluency(FL) and two aggregation metrics, i.e., geometric averaging(GM) and their proposed $J(\cdot)$ metric [2]. In the training stage, we evaluate the model after updating every 500 steps. The perplexity on the validation set is used to activate the early-stop which equals 3. The inference is done as default.

Question generation (Zhou et al., 2017) aims at generating a question given an input document and its corresponding answer span. SQuAD 1.1 (Rajpurkar et al., 2016) is generally used for evaluation. We adopt the data split as in (Du et al., 2017) and fine-tune the pre-trained UniLM (Dong et al., 2019) as the strong baseline. Generated questions are evaluated by metrics including BLEU-1/2/3/4, Meteor and Rouge-L with the provided scripts. The model is evaluated every 1000 steps and the early-stop equaling 5 is associated with the perplexity on the validation set. Other parameters are unchanged following the official guideline.

News summarization differs from dialogue summarization where the input is a document instead of a dialogue. We adopt the same strong baseline BART and evaluation metrics as dialogue summarization. Experiments are done with CNNDM dataset (Hermann et al., 2015) consisting of news articles and multi-sentence summaries. The model is evaluated every 3000 steps and the early-stop equaling 3 is associated with the Rouge-2 on the validation set. During the inference, the minimum and maximum output length is set to 45 and 140 respectively, with no_repeat_ngram_size=3, length_penalty=2.0 and num_beams=4 [3].

A summary of these tasks is in Table 1 and the specific packages we adopted are in the Appendix. For fair comparisons, we re-implement these baselines on our machine. Then, we further arm them with different in-sample curriculum settings without changing corresponding hyperparameters. Specifically, we distinguish Liang et al. (2021)'s work and our method in detail from two aspects, including the curriculum criterion denoted by SG or SC and the training algorithm denoted by TCL or ICL [4], which results in the following 4 combinations:

- **TCL-SG**: the token-wise curriculum learning algorithm(TCL) with sub-sequence generation(SG) criterion proposed by Liang et al. (2021) with their best soft setting. The hyperparameters are set as $\gamma_0 = 0.7$ and $\alpha_0 = 25$ following the original paper.

- **TCL-SC**: we modified the TCL-SG by incorporating our sequence completion(SC) criterion in Section 2 with the hard setting [5] where $\lambda_0 = 0.1$ following the original paper.

- **ICL-SG**: we implemented the SG criterion by using our ICL algorithm in Section 2 which calculating the loss with $1 \leq t \leq c$ in (2).

- **ICL-SC**: our final approach. Both TCL-SC and ICL-SG are ablations for it. The settings of newly introduced p_{start} and s are specified and discussed in Section 4.2.

All of the approaches are trained with the same max training epochs with the early-stop for preventing from over-fitting. The experiments are done on a single RTX 3090 with 24G GPU memory. The results are averaged over three runs. We open-source all of codes and results at `https://github.com/JiaQiSJTU/InsampleCurriculumLearning`.

[2] GM calculate the geometric mean of the corpus-level ACC, SIM and FL. $J(\cdot)$ first calculates the multiplication of sample-level ACC, SIM and FL, then get the average score across the test corpus.

[3] Inference parameters are borrowed from `https://github.com/pytorch/fairseq/blob/main/examples/bart/summarize.py`.

[4] In the rest sections, we use TCL and ICL to refer to the corresponding training algorithms specifically.

[5] The soft setting will hurt our ordering criterion according to preliminary studies in Appendix.

Method	B1	B2	B3	B4	Met	RL
w/o CL	32.03	16.01	8.77	**4.80**	19.84	38.89
TCL-SG	32.35	16.38	8.86	4.69	19.95	39.27
TCL-SC	33.44	16.90	8.93	4.66	20.45	40.55
ICL-SG	32.80	16.32	8.88	4.75	19.96	39.72
ICL-SC	**33.99**	**17.43**	**9.18**	4.64	**20.60**	**40.78**

(a) Reading Comprehension

Method	R1	R2	RL	Met	BertS
w/o CL	51.88	27.30	42.77	24.75	71.38
TCL-SG	52.43	27.65	43.56	25.17	71.86
TCL-SC	52.69	**28.28**	43.89	25.08	71.95
ICL-SG	52.95	28.07	**43.91**	25.67	72.01
ICL-SC	**53.07**	28.23	43.83	**26.12**	**72.17**

(b) Dialogue Summarization

Method	ACC	SIM	FL	GM	J
w/o CL	70.49	55.70	85.98	69.63	33.72
TCL-SG	**76.09**	53.79	82.97	69.76	34.02
TCL-SC	73.27	54.84	85.49	70.03	34.56
ICL-SG	74.60	55.75	84.89	**70.68**	35.64
ICL-SC	73.72	**55.91**	**86.30**	70.60	**35.81**

(c) Style Transfer.

Method	B1	B2	B3	B4	Met	RL
w/o CL	50.36	35.81	27.46	21.62	24.56	50.88
TCL-SG	50.47	35.96	27.57	21.69	24.66	50.76
TCL-SC	50.48	36.04	27.67	21.79	24.70	51.17
ICL-SG	50.89	36.28	27.83	21.92	24.82	51.16
ICL-SC	**51.02**	**36.39**	**27.90**	**21.96**	**24.90**	**51.29**

(d) Question Generation

Method	R1	R2	RL	Met	BertS
w/o CL	43.07	20.01	35.94	**21.44**	63.72
TCL-SG	43.03	20.19	36.22	19.58	63.84
TCL-SC	43.63	20.69	36.70	19.84	64.18
ICL-SG	**43.76**	**20.81**	**36.88**	19.69	**64.31**
ICL-SC	43.60	20.66	36.73	19.64	64.20

(e) News Summarization

Table 2: Results on different NLG tasks. w/o CL and TCL-SG are two previous strong baselines. Both TCL-SC and ICL-SG are variations of our final approach ICL-SC. Scores underlined of ICL-SC are statistically significantly better than both baselines in the first two lines with $p < 0.05$ according to the t-test.

3.2 Automatic Evaluations on Different Tasks

The performances on different NLG tasks are shown in Table 2. These tasks not only focus on solving different problems, but also has a various amount of training data as well as output lengths according to Table 1. Besides, the basic models are also different, including BART, GPT-2 and UniLM. Our approach **ICL-SC achieves significant improvements over the strong baselines** among different tasks on most evaluation metrics, which shows that our method not only works well, but also has strong generalization abilities. It should be noted that GM and J are two comprehensive evaluation metrics for style transfer, with our approach topping the ranks with significant improvements.

To disentangle factors of learning curriculum and training algorithms, we conduct variations of ICL-SC for detailed comparisons to TCL-SG. More observations are as follows.

$*$-SC outperforms $*$-SG for both training algorithms, showing that our proposed sequence completion curriculum is a more effective way of doing curriculum learning within a single sample. The only exception is that ICL-SG performs better than ICL-SC for news summarization in Table 2e. The reason is that multi-sentence summaries in CNNDM are more extractive and cover different salient information in each sentence. Human agreement on salient information is relatively low as shown in Table 3. Consequently, the prefix of a summary can also be a reasonable and more concise reference summary with one or more complete sentences. The nature of $*$-SG happens to take advantage of this property.

ICL-$*$ is better than TCL-$*$ with better performance and less computational costs. For TCL training algorithm adopted in Liang et al. (2021), it separates the whole training process into curriculum and ordinary training. The curriculum length is an important hyper-parameter that is required to be estimated by finishing the training of the baseline model and computing the number of steps it takes to reach approximately 70% of final scores. It intensively aggravates the computational costs. Besides, this estimation rule can not generalize well to different tasks (More in Appendix). We choose to set curriculum steps to 2 or 3 epochs, approximately to the same amount of samples with different difficulty levels in ICL-SC. Taking dialogue summarization as an example, TCL-SG takes around 15.67 epochs (6 for the curriculum step estimation, 3 for curriculum and 6.67 for ordinary training) while our ICL-SC takes only 11.67 epochs to get the final results (More in Appendix). In a word, our ICL-$*$ do the curriculum and ordinary training in a unified manner, requiring less computational costs in total. Moreover, ICL-$*$ moves to the next difficulty level after the model has fully been trained on that judging by the performance on the validation set, which is more similar to the education process in real life and leads to better results.

3.3 Human Evaluations

To further prove the improvement of our approach, we asked three proficient English speakers from Asia for human evaluation. 100 samples from the test set of each task are randomly selected, ignoring the ones with totally same generations among three models, including the vanilla model, TCL-SG and

ICL-SC. The original input, reference output and three generations are shown to annotators together, while the order of the three generations is unknown and different among samples. 3-point Likert Scale is adopted for scoring each generation (Gliwa et al., 2019), where [5, 3, 1] represent excellent, moderate and disappointing results respectively. The average scores and annotator agreements are in Table 3.

Tasks	w/o CL	TCL-SG	ICL-SC	Agree
Reading Comprehension	3.42	3.39	3.94	0.64
Dialog Summarization	3.01	3.51	3.6	0.41
Style Transfer	2.85	2.67	3.02	0.43
Question Generation	3.77	3.81	3.93	0.40
News Summarization	3.13	3.04	3.43	0.23

Table 3: Human evaluations. The agreement (Agree) is calculated by Fleiss Kappa.

The Fleiss Kappa on the first four tasks indicates moderate agreements. It shows the promising improvement of ICL-SC over the vanilla model and TCL-SG, which is consistent with the conclusion based on automatic metrics. The poor agreement on news summarization reflects the diverse concerns of summarization from different annotators.

The drop of TCL-SG over the baseline on style transfer is apparent. Although TCL-SG achieves significant improvements in accuracy, the generated contents with less semantic similarities and poor fluency are not preferred by annotators. Examples will be discussed in Section 3.4.

3.4 Case Studies

We show some cases in Table 4.

In the first case from reading comprehension, our ICL-SC reasoned correctly while the baseline model raised a wrong answer. TCL-SG also answered incorrectly by merging both keywords. Such ability is not suitable for generating a precise answer. In contrast, ICL-SC successfully incorporated more salient information in a single sentence for dialogue summarization, which performs better than both baselines. The vanilla model did poorly on coreference resolution among dialogue utterances and generated "this" without a clear referent. ICL-SC also generated a more accurate question in Table 4d compared with strong baselines, although it's not the same as the reference.

For transferring style from modern to Shakespeare's style, the model generated results are all acceptable while ICL-SC performs slightly better for being more polite. Both TCL-SG and ICL-SC even generated the more professional word "prithee" which is widely used in Shakespeare's

Dialogue	M: ... So **health is more valuable than anything else**. No matter how much money we have, ... W: ... **honors can never equal good health either.** M: ... we should try our best to keep us as healthy as possible.
Question	Which of the following did the man think the most important?
Reference	Good health.
w/o CL	***Honors***.
TCL-SG	***Honors*** and health.
ICL-SC	Health.

(a) Reading comprehension

Dialogue	Mike: <file_photo> **woke up** like this :/ Emma: omg what is this??? **allergy**? Mike: no idea... probably... but **no idea** to what :/
Reference	Mike suspects he might have had an allergic reaction to something.
w/o CL	Mike **woke up** like ***this***.
TCL-SG	Mike has an **allergic reaction**.
ICL-SC	Mike **woke up** with an **allergic reaction**.

(b) Dialogue summarization

Modern	Excuse me , sir , do you know how to read ?
Original	I pray , sir , can you read ?
w/o CL	I **pray you** , can you read ?
TCL-SG	I **prithee** , ***read ?***
ICL-SC	I **prithee** , **sir** , can you read ?
Original	My **dismal** scene I needs must act alone .
Modern	In my **desperate** situation , I have to act alone .
w/o CL	I have to act alone in my **gloomy** scene .
TCL-SG	***It's my own fault , my own fault , that I'm the one who's in a d***
ICL-SC	I have to act alone in my **gloomy** scene .

(c) Style Transfer

Document	Plymouth is home to Plymouth Argyle F.C., who play in **the fourth tier of English football league known as Football League Two.** ...
Answer	Football League Two
Reference	What level of the football league does Plymouth Argyle F.C. operate in?
w/o CL	What is the fourth tier of English football league ?
TCL-SG	What is ***the fourth tier of English football*** ?
ICL-SC	What is the fourth tier of English football league called ?

(d) Question Generation

Document	a man has been arrested ... an imam found dead in his car . ‖ abdul hadi arwani was found slumped in the back seat of his black volkswagen passat on tuesday morning in wembley , north west london . ‖ the 48-year-old syrian national was an outspoken critic of the assad regime and ' actively ' campaigned against extremist , his family have since revealed . ‖ on monday morning scotland yard confirmed that a 46-year-old had been arrested in brent , north west london , on suspicion of conspiracy to murder ‖ ... ‖ he is being questioned ... while officers ... for witnesses ‖ ... ‖ counter-terrorism investigators were drafted ...
Reference	abdul hadi arwani was found dead in his car on tuesday in wembley . counter terrorism police were drafted in to lead investigation into death . a 46-year-old man has been arrested on suspicion of conspiracy to murder .
w/o CL	abdul ... london. The 48 ... revealed. A man ... car.
TCL-SG	abdul ... london. the 48... revealed. on monday ... arrested on suspicion of conspiracy to murder. he ... witnesses.
ICL-SC	abdul ... back seat of his car in wembley. the syrian national was ... assad regime. a 46-year-old man has been arrested on suspicion of conspiracy to murder. he ... witnesses.

(e) News Summarization

Table 4: Case studies. **Keywords** are in bold. ***Doubtful generations*** are italic. '‖" marks sentence boundaries. Unnecessary words in the document and identical words among generations are folded with "...".

time. A bad case is the second case of Table 4c. ICL-SC didn't make any improvements over the baseline. TCL-SG even got out of control.

Generated summaries in Table 4e cover different parts of information in the original document. The vanilla output is just a reordering of the first three sentences. ICL-SC did better by omitting too detailed content compared to the two baselines.

In a word, the results show that **ICL-SC can capture the characteristics of different tasks and do better language modeling**. Besides, by comparing the improvements among these five tasks with different output length, we conclude that our **ICL-SC is more competitive with tasks having shorter outputs.** Long outputs, such as summaries in news summarization, bring additional difficulties on the arrangement of multiple salient contents and cross-sentence relations, which can't be well solved with such a simple in-sample curriculum and will be considered in the future.

4 Analysis

For a better understanding of ICL-SC, we did comprehensive ablation studies and combined it with the traditional CL. The experiments in this section are done on dialogue summarization, which is representative due to the medium output length.

4.1 Ablations on the Training Strategy

To examine the design of decreasing the prefix for ICL-SC, we introduce the alternatives as follows:

- **Decrease** refers to the Algorithm 1. Taking $p_{start} = 0.6$ and $s = 0.3$ as an example, the prefix percentage p varies as $0.6 \rightarrow 0.3 \rightarrow 0.0$ during training.
- **Increase** means that we gradually increase the length of prefix by increase p following $0.0 \rightarrow 0.3 \rightarrow 0.6$.
- **Random** is that we randomly pick p from the set $\{0.0, 0.3, 0.6\}$ in this example.

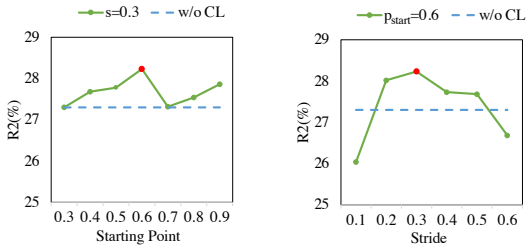
Strategy	R1	R2	RL	Met	BertS
Decrease	**53.07**	**28.23**	**43.83**	**26.12**	**72.17**
Increase	51.43	27.35	42.97	24.32	71.25
Random	51.80	27.69	43.27	24.59	71.51

Table 5: Ablations on ICL strategies. The starting point and the stride are 0.6 and 0.3 respectively.

The results are shown in Table 5, with Decrease ranking first and Increase ranking the worst. Decrease significantly outperforms other ablations, showing that our sequence completion criterion of shrinking the prefix does work by means of learning from easy to hard.

4.2 Parameter Search of the Starting Point and the Stride

To better understand how the ICL-SC manipulates the difficulty of samples during the training process, we further did experiments on different settings of two newly-introduced hyper-parameters p_{start} and s. The results are in Figure 3.



(a) Starting Point (b) Stride

Figure 3: Parameter search of the starting point p_{start} and the stride s. The "w/o" CL representing the BART baseline is drawn for comparison.

We can see that the performance drops with either a too large or too small p_{start}. The former one starts training with only predicting the last 1 or 2 tokens according to the average length of reference output shown in Table 1. Most of the time, they are punctuation marks that do not carry any important semantic information, leading to a bad warm-up. The latter one requires the model to predict more than half of the output, which are too difficult as a beginning learning target. Besides, a larger p_{start} which is divisible by s is more competitive.

The trend is the same for using different stride values. The performance drops with s equaling 0.1 or 0.6. The smaller ones lead to too tiny changes, which not only excessively prolongs the required training time but also leads to server outfitting on the training set. The larger ones greatly enlarge the gap between training targets which degrades to 0.0 directly. It also harms the performances.

In a word, the training should start with a medium difficulty training objective and the gap between training objectives shouldn't be too large. Both parameters are closely related to the output length of different tasks. We suggest using ($p_{start} = 0.6$, $s = 0.3$) for NLG tasks with multi-sentence outputs, and ($p_{start} = 0.5$, $s = 0.5$) for NLG tasks with single-sentence outputs. All of our experiments are done based on this guideline.

4.3 Combinations with the Traditional CL

Since our ICL-SC is orthogonal to sample-wise CL and designing an appropriate sample-wise curriculum is not easy, we choose dialogue summarization as a representative task, design several traditional CL strategies empirically, and further apply our ICL-SC on top of them for comparisons. 4 different traditional CL strategies are as follows:

- **Input length (InLen)** refers to the number of tokens in the input dialogue. The longer a dialogue is, the more complex a sample is.
- **Output length (OutLen)** is the number of tokens in a reference summary, which is also proportional to the difficulty of a sample.
- **Compression ratio (CompR)** equals the output length divided by the input length. More compressed training pairs are harder.
- **Abstractiveness (Abstr)** represents the percentage of novel words in the reference summary which are not in the dialogue. We measure it by Rouge-2 recall, which is inversely proportional to the difficulty level.

Method	R1	R2	RL	Met	BertS
w/o CL	51.88	27.30	42.77	24.75	71.38
ICL-SC	**53.07**	**28.23**	**43.83**	**26.12**	**72.17**
InLen	52.19	**27.73**	**43.50**	25.57	71.73
InLen+	**52.56**	27.60	43.43	**25.77**	**71.92**
OutLen	41.38	20.88	31.77	**27.95**	67.21
OutLen+	**43.96**	**22.14**	**33.05**	26.39	**67.64**
CompR	39.68	19.28	34.73	14.41	65.96
CompR+	**41.59**	**20.78**	**36.62**	**15.22**	**67.19**
Abstr	**44.61**	20.10	36.93	**17.34**	68.29
Abstr+	44.41	**20.64**	**37.29**	17.25	**68.33**

Table 6: Performances with traditional CL strategies. "+" represents experiments further armed with ICL-SC.

The results based on the ordered training samples according to these intuitive CL strategies are shown in Table 6. It shows that only InLen improves the vanilla model, but it still lags behind the pure ICL-SC. Other strategies failed mainly due to the low data quality at the beginning or the end of training. Taking Abstr as an example, samples with the highest Rouge-2 recall are gathered at the beginning where their inputs and outputs are almost the same. This leads to a bad initialization for models learning the summarization ability.

Besides, some strategies are incompatible, such as OutLen and CompR. Samples with the shortest output length are always too compressed. Therefore, developing a comprehensive score for a better ranking is difficult. It should be also noticed that most of these strategies are designed for summarization, which are not suitable for generalization.

In a word, it's hard to develop a comprehensive strategy for one task or a unified strategy for different NLG tasks with traditional CL. ICL-SC not only outperforms these CL strategies, but also improves them when easily combined.

5 Related Work

Natural language generation has received great attention with deep neural networks, especially pre-trained language models. It refers to the task where expected outputs for different purposes are in natural language (Dong et al., 2022). The inherent characteristic of having more than one correct output given the same input is the core challenge of solving this kind of task, especially for evaluation (Singh et al., 2018).

Curriculum learning (Bengio et al., 2009) boost models' performances in a range of machine learning areas (Liu et al., 2021; Varshney et al., 2022) by reordering the training samples. It meets great obstacles when applying to NLG tasks as it's hard to evaluate the difficulties of training samples. Different rules are developed for different tasks (Platanios et al., 2019; Chang et al., 2021). For example, (Liu et al., 2018) measures the complexity of question-answering pairs from the view of frequency and grammar simply for answers. (Kocmi and Bojar, 2017) focuses more on POS features and the length of translation pairs. Other works utilize additional models or targeting models in the previous training step (Zhang et al., 2018). Shen and Feng (2020) reorder samples by the accuracy from an independent emotion classifier for response generation. However, such salient features do not always exist or can be well classified. There is also work (Zhou et al., 2020) using either the reference perplexity or generations evaluated by corresponding metrics for ranking during training, while these scores are not ideal due to the one-to-many characteristic of NLG. Thus, designing a CL strategy generalizing well for NLG is difficult.

Instead of figuring out the oracle scoring function for training samples, we propose to measure the language generation difficulty within a sample. Liang et al. (2021) did something similar though their approach amounts to data augmentation by doing sub-sequence generation, which is not exactly curriculum learning. We, on the other hand,

train on the original sample with a decreasing prefix length and thus learn from easy to hard.

6 Conclusion

This paper defines a kind of curriculum learning strategy for NLG tasks called in-sample curriculum learning (ICL) by manipulating the difficulty of training within a training sample instead of ranking among samples. We propose the ICL algorithm with the sequence completion curriculum which boosts the performance of strong baselines on a wide range of tasks, showing the effectiveness and strong generalization ability of our approach. More training strategies under ICL digging the inherent difficulties of generating a language sequence are expected in the future.

Limitations

Tasks	w/o CL	TCL-SG	ICL-SC
Reading Comprehension	6.67 ep	13.00 ep	7.67 ep
Dialog Summarization	6.00 ep	15.67 ep	11.67 ep
Style Transfer	6.50k st	14.78k st	9.67k st
Question Generation	17.67k st	37.73k st	21.00k st
News Summarization	21.00k st	47.20k st	36.00k st

Table 7: Average number of training steps for different approaches. "ep" and "st" are short for "epochs" and "steps" respectively.

One limitation of our approach is that in-sample curriculum learning methods (both TCL-SG and ICL-SC) always incur extra overhead during training compared with the vanilla model shown in Table 7. Nevertheless, the inference time of different approaches is the same as the vanilla model. In a word, it's worthwhile because (1) ICL-SC can perform significantly better than both baselines without additional computational requirements during inference in real applications; (2) ICL-SC doesn't rely on task-specific expertise and has strong generalization ability.

Due to the limited computational resources, we were unable to do experiments on machine translation. According to the implementation details in Liang et al. (2021), all of their machine translation experiments were done on 32G NVIDIA V100 GPUs which are much more powerful than a single RTX 3090. Even for the low resource setting with around 133K to 612K training samples, they used dynamic batching with 4096 maximum tokens and trained for 60 epochs. This will either lead to an out-of-memory error or take us several weeks or even months to get the results of a single run on our machine. Instead, we tried our best to cover a range of representative natural language generation tasks and corresponding datasets with different characteristics, such as sizes and output lengths (Table 1).

Acknowledgments

This work was generously supported by the CMB Credit Card Center & SJTU joint research grant, and Meituan-SJTU joint research grant.

References

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48.

Ernie Chang, Hui-Syuan Yeh, and Vera Demberg. 2021. Does the order of training samples matter? improving neural data-to-text generation with curriculum learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 727–733.

Chenhe Dong, Yinghui Li, Haifan Gong, Miaoxin Chen, Junxin Li, Ying Shen, and Min Yang. 2022. A survey of natural language generation. *ACM Computing Surveys*, 55(8):1–38.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 13063–13075.

Xinya Du, Junru Shao, and Claire Cardie. 2017. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352.

Alexander R Fabbri, Wojciech Kryściński, Bryan McCann, Caiming Xiong, Richard Socher, and Dragomir Radev. 2021. Summeval: Re-evaluating summarization evaluation. *Transactions of the Association for Computational Linguistics*, 9:391–409.

Bogdan Gliwa, Iwona Mochol, Maciej Biesek, and Aleksander Wawer. 2019. Samsum corpus: A human-annotated dialogue dataset for abstractive summarization. In *Proceedings of the 2nd Workshop on New Frontiers in Summarization*, pages 70–79.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015*.

Tom Kocmi and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 379–386.

Kalpesh Krishna, John Wieting, and Mohit Iyyer. 2020. Reformulating unsupervised style transfer as paraphrase generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*, pages 737–762.

Gaurav Kumar, George Foster, Colin Cherry, and Maxim Krikun. 2019. Reinforcement learning based curriculum optimization for neural machine translation. 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 2054–2061.

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

Chen Liang, Haoming Jiang, Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao, and Tuo Zhao. 2021. Token-wise curriculum learning for neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3658–3670. Association for Computational Linguistics.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Cao Liu, Shizhu He, Kang Liu, Jun Zhao, et al. 2018. Curriculum learning for natural answer generation. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pages 4223–4229.

Fenglin Liu, Shen Ge, and Xian Wu. 2021. Competence-based multimodal curriculum learning for medical report generation. In *ACL/IJCNLP (1)*, pages 3001–3012.

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.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabás Poczós, and Tom Mitchell. 2019. Competence-based curriculum learning for neural machine translation. 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 1162–1172.

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.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Lei Shen and Yang Feng. 2020. Cdl: Curriculum dual learning for emotion-controllable response generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 556–566.

Sandhya Singh, Kevin Patel, Pushpak Bhattacharyya, Krishnanjan Bhattacharjee, Hemant Darbari, and Seema Verma. 2018. Does curriculum learning help deep learning for natural language generation? In *15th International Conference on Natural Language Processing*, page 97.

Kai Sun, Dian Yu, Jianshu Chen, Dong Yu, Yejin Choi, and Claire Cardie. 2019. Dream: A challenge dataset and models for dialogue-based reading comprehension. *Transactions of the Association for Computational Linguistics*, 7:217–231.

Neeraj Varshney, Swaroop Mishra, and Chitta Baral. 2022. Let the model decide its curriculum for multitask learning. In *Proceedings of the Third Workshop on Deep Learning for Low-Resource Natural Language Processing*, pages 117–125. Association for Computational Linguistics.

Wei Xu, Alan Ritter, William B Dolan, Ralph Grishman, and Colin Cherry. 2012. Paraphrasing for style. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 2899–2914.

Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J Martindale, Paul McNamee, Kevin Duh, and Marine Carpuat. 2018. An empirical exploration of curriculum learning for neural machine translation. *arXiv preprint arXiv:1811.00739*.

Xuan Zhang, Pamela Shapiro, Gaurav Kumar, Paul McNamee, Marine Carpuat, and Kevin Duh. Curriculum learning for domain adaptation in neural machine translation. 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 1903–1915. Association for Computational Linguistics.

Qingyu Zhou, Nan Yang, Furu Wei, Chuanqi Tan, Hangbo Bao, and Ming Zhou. 2017. Neural question generation from text: A preliminary study. In *National CCF Conference on Natural Language Processing and Chinese Computing*, pages 662–671. Springer.

Yikai Zhou, Baosong Yang, Derek F Wong, Yu Wan, and Lidia S Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6934–6944.

A Packages used for Baselines

The packages we adopted to re-implement the baseline are listed as follows:

Reading Comprehension

- Dataset: https://github.com/nlpdata/dream/tree/master/data
- Baseline Code: https://github.com/huggingface/transformers
- Evaluation Metric: https://github.com/tensorflow/nmt/blob/master/nmt/scripts/bleu.py

Dialogue Summarization

- Dataset: https://arxiv.org/src/1911.12237v2/anc/corpus.7z
- Baseline Code: https://github.com/huggingface/transformers
- Evaluation Metric: https://github.com/pltrdy/files2rouge; https://github.com/Yale-LILY/SummEval

Style Transfer

- Dataset: https://github.com/martiansideofthemoon/style-transfer-paraphrase
- Baseline Code: https://github.com/martiansideofthemoon/style-transfer-paraphrase
- Evaluation Metric: https://github.com/martiansideofthemoon/style-transfer-paraphrase

Question Generation

- Dataset: https://github.com/microsoft/unilm/tree/master/unilm-v1
- Baseline Code: https://github.com/microsoft/unilm/tree/master/unilm-v1
- Evaluation Metric: https://github.com/microsoft/unilm/tree/master/unilm-v1

News Summarization

- Dataset: https://drive.google.com/file/d/0BzQ6rtO2VN95a0c3TlZCWkl3aU0/view?resourcekey=0-toctC3TNM1vffPCZ7XT0JA
- Baseline Code: https://github.com/huggingface/transformers
- Evaluation Metric: https://github.com/pltrdy/files2rouge; https://github.com/Yale-LILY/SummEval

B Preliminary Studies on TCL

Preliminary studies on dialogue summarization for TCL under different settings are shown in Table 8. We can see that the "soft" setting does help the TCL with sub-sequence generation curricula, which is consistent with the results in Liang et al. (2021). Results are opposite for TCL with our proposed sequence completion curricula. The "soft" setting considering the loss from prefix tokens actually hurts the intuition that "the shorter the target is, the easier the tasks is". As a result, SC-hard performs better than SC-soft.

	R1	R2	RL	Met	BertS
w/o CL	51.88	27.30	42.77	24.75	71.38
SG-hard	50.70	27.31	43.00	23.47	70.85
SG-soft	52.43	27.65	43.56	25.17	71.86
SC-hard	52.69	28.28	43.89	25.08	71.95
SC-soft	51.39	27.53	43.06	23.84	71.35

Table 8: Ablations on TCL learning algorithm with different settings.

Experiments on the sensitivity of curriculum step in TCL-SG (Liang et al., 2021) are in Table 9. It consistently has improvements on dialogue summarization compared with the baseline. However, the performances also vary a lot with different curriculum steps, especially on R1, Meteor and BertScore. The estimation rule proposed in Liang et al. (2021) of computing the number of steps it takes to reach approximately 70% of final scores doesn't perform well for dialogue summarization. So, we choose to set curriculum steps to 3 epochs

for dialogue summarization and news summarization, and 2 epochs for reading comprehension and style transfer, which not only achieve better results, but also are fairer for comparisons. For news summarization, we still adopted their estimation rule and trained with 5200 curriculum steps.

Curriculum Step	R1	R2	RL	Met	BertS
w/o CL	51.88	27.30	42.77	24.75	71.38
1 epoch	52.48	27.86	43.47	25.50	71.83
1.58 epoch(70%)	51.89	27.64	43.51	24.37	71.55
2 epoch	51.93	27.75	43.37	24.73	71.57
3 epoch	52.43	27.65	43.56	25.17	71.85

Table 9: Performances on TCL-SG with different curriculum steps.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
It is the section after the conclusion and before the references.

☐ A2. Did you discuss any potential risks of your work?
Not applicable. We propose a method for better natural language generation.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
Section 1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☑ Did you use or create scientific artifacts?

Section 3.1

☑ B1. Did you cite the creators of artifacts you used?
Section 3.1

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
All of the datasets are publicly available and the source link for downloading them are in the Appendix A. We will only release the codes and results for our work (Section 3.1).

☑ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Section 3.1

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. We adopted the widely-used publicly available datasets.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. We are not a dataset paper. We provided necessary information about the datasets in Section 3.1. More information please refer to their original dataset paper.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Section 3.1, Table 1

C ☑ Did you run computational experiments?

Section 3.2

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 3.1 and Limitations

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 3.1 and Section 4.2

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Section 3.1

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Section 3.1 and Appendix A

D ☑ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Section 3.3

☑ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Section 3.3

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. We had student volunteers to do the human evaluation.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. The volunteers knew how the data would be used before doing the human evaluation.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. We did not collect new datasets, only a simple human evaluation.

☑ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Section 3.3 for human evaluation.