CoNT: Contrastive Neural Text Generation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recently, contrastive learning attracts increasing interests in neural text generation as a new solution to alleviate the *exposure bias* problem. It introduces a sequence-level training signal which is crucial to generation tasks that always rely on auto-regressive decoding. However, previous methods using contrastive learning in neural text generation usually lead to inferior performance. In this paper, we analyse the underlying reasons and propose a new **Co**ntrastive **N**eural **T**ext generation framework, CoNT. CoNT addresses bottlenecks that prevent contrastive learning from being widely adopted in generation tasks from three aspects – the construction of contrastive examples, the choice of the contrastive loss, and the strategy in decoding. We validate CoNT on five generation tasks with ten benchmarks, including machine translation, summarization, code comment generation, data-to-text generation and commonsense generation. Experimental results show that CoNT clearly outperforms the conventional training framework on all the ten benchmarks with a convincing margin. Especially, CoNT surpasses previous the most competitive contrastive learning method for text generation, by 1.50 BLEU on machine translation and 1.77 ROUGE-1 on summarization, respectively. It achieves new state-of-the-art on summarization, code comment generation (without external data) and data-to-text generation.

1 Introduction

Contrastive learning has achieved great success in representation learning [6, 41, 42]. It also attracts enormous interests in neural text generation recently. By creating positive and negative examples in response to unseen (or erroneous) inputs [21], contrastive learning offers a new solution to alleviate the *exposure bias* problem [3, 33] – an autoregressive model trained only using the ground truths exhibits inferior generalization performance. Apart from that, contrastive learning also introduces a sequence-level loss in addition to the conventional token-level language model loss with maximum likelihood estimation (MLE). This is crucial to most conditional text generation tasks (e.g., machine translation and summarization) which are evaluated on sequence-level metrics (e.g., BLEU [30]).

However, it is non-trivial to get contrastive learning working on neural text generation. If we simply use from-batch positive-negative samples following simCLR [6], and adopt the InfoNCE loss [12, 42] which ignores the difference between negative samples (§2.2: Naive CL), the improvement over non-contrastive baselines on generation tasks is rather marginal. Previous work attempts to build better contrastive samples by disturbing the ground truth [9, 21, 28] in the discrete space or the continuous embedding space, but when it comes to text generation tasks, their performance gains are still far from satisfactory.

In this work, we propose a new contrastive neural text generation framework, CoNT. CoNT does three different things from previous frameworks that make suboptimal use of contrastive learning. First, CoNT samples contrastive examples from its own predictions (e.g., through beam search algorithm). This training procedure exposes the model to its mistakes in the inference stage and

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

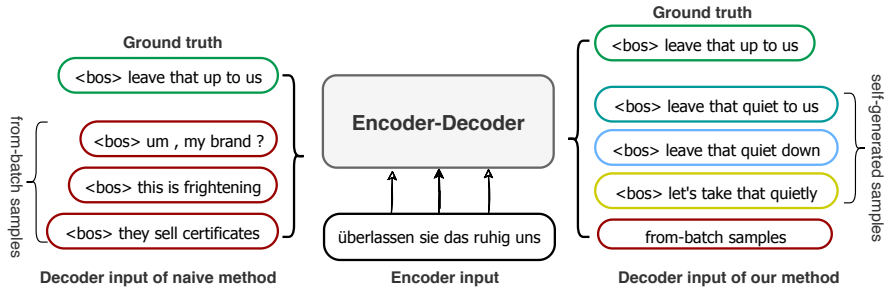


Figure 1: A case study from IWSLT'14 De-En translation task. The naive setting uses from-batch samples following SimCLR [6]. Compared with the naive method, CoNT both incorporates self-generated samples and from-batch samples. The border color indicates the acutual distance between the ground truth and the contrastive example.

effectively alleviate the exposure bias problem. We show a comparison between negative samples in CoNT and in Naive CL in Figure 1. Second, we use a N-pairs contrastive loss which gives a fine-grained treatment to the contrastive examples based on their sequence-level scores (e.g., BLEU). It allows the model to fully leverage the supervision from the ground truth example (and its own generated examples) to learn a better sequence-level distance function between the source and the target representation. Third, we directly incorporate the learned sequence similarity score from the distance function into the inference stage. This helps the model to find a better global configuration, than merely follows the language model likelihood objective in decoding.

We validate CoNT on various important conditional language generation tasks (§4.2), including machine translation, summarization, code comment generation, data-to-text generation, and commonsense generation. Extensive experiments demonstrate that CoNT greatly improve the conventional MLE baselines and significantly outperforms all previous contrastive generation models. CoNT establishes new state-of-the-art results on summarization, code comment generation (without external data), and data-to-text generation. Particularly, on data-to-text generation and commonsense generation, CoNT achieves on-par performance with the powerful large pre-trained models: T5-large, T5-3B [34] with only the base version of T5 while maintaining the efficiency of lightweight models.

2 Background

2.1 Neural Conditional Text Generation

A neural sequence-to-sequence model [40] $\mathcal{M} = (f, g)$ generates the target sequence conditioning on a source sequence, where f and g denote the encoder and encoder, respectively. It is typically trained using the language model objective with the maximum likelihood estimation (MLE). Given a source sequence $\boldsymbol{x} = \{x_i\}_{i=0}^{M}$ and its target sequence $\boldsymbol{y} = \{y_i\}_{i=0}^{N}$, we minimize the following negative log likelihood (NLL) loss:

$$\mathcal{L}_{\text{NLL}} = -\sum_{t=1}^{N} \log p_\theta(y_t|\boldsymbol{x}, \boldsymbol{y}_{<t}). \tag{1}$$

At training stage, it predict the next word based on previous ground truth input $\boldsymbol{y}_{<t} \in \boldsymbol{y}$, but at inference stage, tokens of $\boldsymbol{y}_{<t}$ are predicted by itself, this introduces the *exposure bias*.

2.2 Naive Contrastive Learning for Text Generation

Contrastive text generation introduces a constrastive term in addtion to the original NLL loss. In Naive CL, we simply follows SimCLR [6] and use from-batch negative samples $\mathcal{B}$ in the InfoNCE loss [12, 42]:

$$\mathcal{L}_{\text{NCE}} = -\log \frac{\exp(\cos(\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\boldsymbol{y}})/\tau)}{\sum_{\boldsymbol{y}' \in \mathcal{B}} \exp(\cos(\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\boldsymbol{y}'})/\tau)}, \tag{2}$$

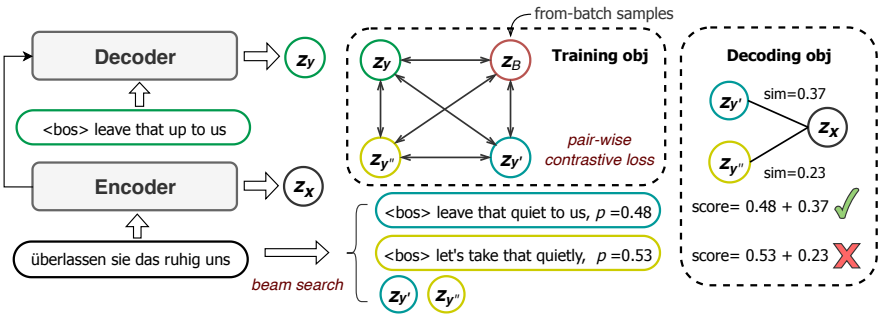


Figure 2: An overview of CoNT. $\mathbf{z}_{\boldsymbol{x}}$, $\mathbf{z}_{\boldsymbol{y}}$ is the representation of source sequence $\boldsymbol{x}$ and its target sequence $\boldsymbol{y}$. $\boldsymbol{y}'$ and $\boldsymbol{y}''$ with their representations $\mathbf{z}_{\boldsymbol{y}'}$, $\mathbf{z}_{\boldsymbol{y}''}$ are returned by beam search algorithm. The feature representations come from pooling the output of the encoder (source sequence) or decoder (target sequence). Our training objective is obtained by comparing by all contrastive samples in pair. The decoding objective not only considers the likelihood of each sequence, but also the sequence similarity score modeled in training.

where $\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\boldsymbol{y}}, \mathbf{z}_{\boldsymbol{y}'} \in \mathbb{R}^d$ denote the vector representation of input $\boldsymbol{x}$, ground truth $\boldsymbol{y}$ and negative sample $\boldsymbol{y}' \in \mathcal{B}$, respectively. τ is the temperature and $\cos(\cdot,\cdot)$ defines the cosine similarity. Intuitively, the contrastive loss $\mathcal{L}_{\text{NCE}}$ seeks to learn a similarity function that drives the distance between the source sequence representation $\mathbf{z}_{\boldsymbol{x}}$ and its ground-truth target sequence representation $\mathbf{z}_{\boldsymbol{y}}$ closer.

3 Method

In this section, we present our new contrastive neural text generation framework, CoNT . CoNT advances the Naive CL (§2.2) in three aspects. First, CoNT uses negative examples from its own predictions (§3.1) to construct the set $\mathcal{B}$. Second, CoNT replaces the InfoNCE loss (Eq.2) with a N-pairs constrastive loss (Eq.3) which leverages a finer-grained supervision given by the the sequence-level scores of all pairs (§3.2). Third, CoNT incorporates the learned similarity function into its inference score directly (§3.3). An overview of our approach can be found in Figure 2.

3.1 Contrastive Examples from Predictions

Instead of only using contrastive examples from the same batch [6], we propose to add new contrastive examples from the model's own predictions. Kalkstein et al. [17] point that using diverse contrastive samples helps the generalization ability of the model. Therefore, we use the diverse beam search algorithm [46] to create contrastive examples from the top-K list of the model's lastest predictions and then append them to the from-batch samples to form the constrastive examples. A warm-up stage where the model is only supervised by $\mathcal{L}_{\text{NLL}}$ is recommended as it guarantees the quality of the examples from the model's prediction. These self-generated contrastive examples alleviate the model's exposure bias. Besides, with the model's performance improving gradually, this approach creates high-quality hard negative examples that is known to be important in contrastive learning [15, 35].

3.2 N-Pairs Contrastive Loss

One major drawback of the InfoNCE loss is that it has the same treatment for all negative samples. In text generation, this means that the relative difference between the ground truth and the contrastive examples is ignored, while this can be easily quantified using a sequence level score (e.g. BLEU) and the quality of these contrastive examples varies. To mitigate this problem, we propose to employ a pair-wise margin loss. We first rank all the contrastive examples based on an oracle function $o(\cdot, \boldsymbol{y})$, which computes a sequence-level score with the ground truth $\boldsymbol{y}$. Given a input sequence $\boldsymbol{x}$, the ground truth $\boldsymbol{y}$, and a set of K contrastive samples $\mathcal{B} = \{\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_K\}$, we can create a series of example pairs $(\boldsymbol{y}^+, \boldsymbol{y}^-) \in \mathcal{P}$, where $+$ and $-$ are determined by their ranks.[1] The contrastive learning objective is formulated as a margin loss according to their cosine similarity to the source

[1] $\mathcal{P}$ contains C_K^2 pairs constructed from $\mathcal{B}$, ground truth $\boldsymbol{y}$, and from-batch examples.

Algorithm 1 Inference algorithm: Given an input sequence $\boldsymbol{x}$, a contrastive generation model $\hat{\mathcal{M}} = (\hat{f}, \hat{g})$; return the output sequence.

procedure BEAMSEARCH(g, $H_{\boldsymbol{x}}$, b) ▷ beam search algorithm
  return Text, likelihood, logits of the b hypotheses

procedure INFERENCE($\hat{G}$, $\boldsymbol{x}$)
  $\mathbf{H}_{\boldsymbol{x}} \leftarrow \hat{f}(\boldsymbol{x})$, $b \leftarrow$ beam size, $\alpha \leftarrow$ balance factor $\in (0, 1)$
  $\boldsymbol{y}^{1:b}, \mathbf{P}_{\boldsymbol{y}}^{1:b}, \mathbf{H}_{\boldsymbol{y}}^{1:b}$ = BEAMSEARCH($\hat{g}, \mathbf{H}_{\boldsymbol{x}}, b$) ▷ Get b candidates with beam search
  $\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\boldsymbol{y}}^{1:b} \leftarrow \text{Avg}(\mathbf{H}_{\boldsymbol{x}}), \text{Avg}(\mathbf{H}_{\boldsymbol{y}}^{1:b})$ ▷ Avg($\cdot$) is an average pooling function
  $\mathbf{D}_{\boldsymbol{y}}^{1:b} \leftarrow$ Cosine distance between $\mathbf{z}_{\boldsymbol{x}}$ and representation of hypotheses $\mathbf{z}_{\boldsymbol{y}}^{1:b}$
  $\mathbf{P}_{\boldsymbol{y}}^{1:b} \leftarrow$ Likelihood of hypotheses returned by beam search
  $k = \arg\max_{i=1..b}\{\alpha * \mathbf{D}_{\boldsymbol{y}}^{i} + (1-\alpha) * \mathbf{P}_{\boldsymbol{y}}^{i}\}$
  return $\boldsymbol{y}^{k}$

representation $\mathbf{z}_{\boldsymbol{x}}$:

$$\mathcal{L}_{\text{N-Pairs}} = \sum_{(\boldsymbol{y}^+, \boldsymbol{y}^-) \in \mathcal{P}} \mathcal{L}(\boldsymbol{y}^+, \boldsymbol{y}^-) = \sum_{(\boldsymbol{y}^+, \boldsymbol{y}^-) \in \mathcal{P}} \max\{0, \cos(\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\boldsymbol{y}^-}) - \cos(\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\boldsymbol{y}^+}) + \xi\}. \quad (3)$$

We further set $\xi = \gamma * (\text{rank}(\boldsymbol{y}^-) - \text{rank}(\boldsymbol{y}^+))$ following Zhong et al. [53] to reflect the quality difference in these pairs, where γ is a hyperparameter controlling the the strength. Full details of the training algorithm can be found in Algorithm 2, Appendix B.

3.3 Inference with Learned Similarity Function

Previous inference algorithm for contrastive text generation method [21] usually remains the same with non-contrastive approaches. In CoNT, we incorporate the similarity function learned in the N-pairs contrastive loss into the decoding stage. Despite such a inference objective can be generalized to other contrastive learning methods as long as the vector representations for source and target sequence pair exist, the design of CoNT can better make use of the learned similarity function (§4.3). The decoding objective in CoNT is to find the sequence $\boldsymbol{y}^*$ that maximizes both the learned similarity score and the conventional language model likelihood:

$$\boldsymbol{y}^* = \arg\max_{\hat{\boldsymbol{y}}}\{\alpha \cdot \cos(\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\hat{\boldsymbol{y}}}) + (1-\alpha)\prod_{t=0}^{n} p(\hat{y}_t|\boldsymbol{x}, \hat{\boldsymbol{y}}_{<t})\}, \quad (4)$$

where $\mathbf{z}_{\boldsymbol{x}}, \mathbf{z}_{\hat{\boldsymbol{y}}} \in \mathbb{R}^d$ is the vector representation of $\boldsymbol{x}$, $\hat{y}$, and α is the hyperparameter that balances the contribution of each term. In most cases, α can be directly set to 0.5, tuning α on the validation set will usually get better results. Algorithm 1 illustrates the inference stage in CoNT in details.

The relationship between different modules of CoNT is summarized in Figure 3.

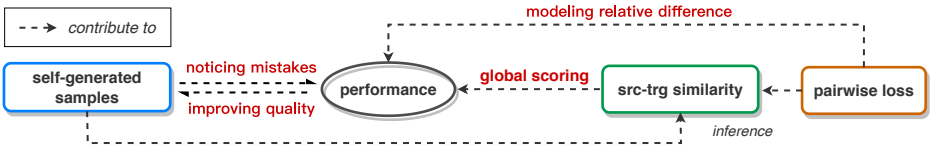


Figure 3: Relationship between different modules in CoNT. Both the design of the pairwise loss function and self-generated samples could contribute the source-target similarity function that computes the sequence-level score at inference stage. With the performance improved, self-generated contrastive samples tend to be more indistinguishable.

4 Experiments

We experiment CoNT on 5 downstream tasks with 10 different benchmarks. Our contrastive learning framework supports most sequence-to-sequence models at multiple scales. Concretely, we experiment

CoNT on 4 kinds of base models: (a) Transformer-samll (60M) [44] and transformer-base (220M), (b) T5-small (60M) and T5-base (220M) [34], (c) CodeT5-base (220M) [48] , (d) PEGASUS-large (560M) [52]. Details of our experimental setup for each benchmarks can be found in Appendix C.

On WMT'16 Ro-En (machine translation) and XSum (summarization) which are also used in previous contrastive text generation frameworks [21], results show that CoNT is able to substantially improve the MLE baseline and clearly outperform all previous contrastive baselines by a large margin. We also build CoNT on state-of-the-art (SOTA) baselines: PEGASUS-large (summarization), CodeT5-base (code comment generation) and achieve new SOTA. Moreover, on data-to-text generation and commonsense generation, CoNT also shows its superior performance over strong MLE baselines.

4.1 Baselines

1. **Naïve CL** [6]: Naive CL denotes the naive contrastive learning approach that treats the ground truth as the positive sample and the target sequences from the same mini-batch as the negative examples. The training object of naive CL takes the form of Eq. 2. We also implement Naive CL with N-Pairs contrastive loss, it can be viewed as an ablation study when setting beam size of CoNT to 0 during training.
2. **SSMBA CL** [28]: Compared with naive CL, SSMBA builds more positive samples via disturbing the ground truth in the discrete space. Concretely, SSMBA first randomly masks 25% tokens in the target sequence and then reconstructs the ground truth with a masked language model BERT.
3. **Dropout CL** [9]: Dropout CL enhances the positive samples by using dropout mechanism on the target sequence. We use the default dropout rate of standard transformer decoder [44] and input the ground truth to the decoder twice.
4. **CLAPS** [21]: CLAPS is previous the best contrastive learning framework for conditional text generation task. In order to provide more challenging contrastive examples, CLAPS propose to simultaneously create extra positive and negative pairs by adding perturbations to the ground truth sequence in the continuous embedding space.
5. **CoNT** (this work): CoNT is the contrastive neural text generation framework proposed in this work. We implement its InfoNCE version by treating ground truth as positive sample and self-generated samples are also treated as negative samples.

4.2 Quantitative Results

Machine Translation For machine translation, we evaluate CoNT on WMT 2016 Romanian-to-English translation task (WMT'16 Ro-En), WMT 2014 English-to-German translation task (WMT'14 En-De) and IWSLT 2014 German-to-English translation task (IWSLT'14 De-En). We use BLEU as the evaluation metric. Results in Table 1 (rows with gray background) indicates our model CoNT significantly improves the traditional maximum likelihood estimation training and inference framework. On WMT'16 Ro-En, CoNT outperforms previous the best contrastive learning approach CLAPS by 1.50 BLEU and exceeds the MLE baseline by **2.70** BLEU with the same base model T5-small. We also compare the infoNCE loss used in previous methods with the N-Pairs margin loss described in Eq. 3. Results show that the N-pairs contrasting samples generally works better than dividing all samples into predefined positive-negative categories. Similar to CLAPS and Naive CL, only incorporating contrastive learning into training improves the performance of T5-small baseline on WMT'16 Ro-En to 30.55 (+**2.34**) BLEU. If we further add learned target-source similarity as decoding target as Eq. 4, the result is further boosted to 30.91 BLEU. We observe that the benefits of introducing sequence similarity into inference is more obvious on IWSLT'14 De-En – the additional decoding target improves the vanilla beam search algorithm up to 0.86 BLEU.

Summarization For summarization, we use the XSum [26] dataset collected from BBC News whose reference summaries are provided by human writers. We also evaluate CoNT on a multi-document summarzation dataset multi-news [8] consisting of news articles from the site newser.com. Compared with the common summarization task, multi-document is more challenging where the model need to automatically summarize several articles and usually has to handle long input sequence and target sequence. Experimental results are in Table 2. The first block includes the performance of different contrastive frameworks with T5-small. On XSum, it shows that our proposed model strongly outperform previous contrastive frameworks by about 2.0 ROUGE-1 score. We also illustrate our

Table 1: BLEU on WMT'16 Ro-En, IWSLT'14 De-En and WMT'14 En-De translation tasks. For IWSLT'14 De-En and WMT'14 En-De, we use Transformer-small (**Tr-small**) and Transformer-base (**Tr-base**) as baselines. For WMT'16 Ro-En, we add a pre-trained baseline **T5-small**. *w/o seq sim* means we use the origin beam search without target-source representation similarity. The best results in each block are underlined and the best results are in bold. Rows in gray denotes the contrastive learning based model strongly outperforms its MLE version. Results with † are token from [21].

Model	**WMT'16 Ro-En**		**IWSLT'14 De-En**	**WMT'14 En-De**
	Tr-small	**T5-small**	**Tr-small**	**Tr-base**
MLE	25.78	28.21	34.18	27.30
Contrastive loss: InfoNCE loss				
Naive CL	25.49	27.79	34.45	27.28
SSMBA CL	25.98	28.48	34.32	27.16
Dropout CL	<u>26.01</u>	29.10	34.41	27.34
CLAPS†	23.59	29.41	–	–
CoNT	25.74	<u>29.64</u>	<u>34.46</u>	<u>27.35</u>
Contrastive loss: N-Pairs loss				
Naive CL	26.15	29.86	34.47	27.41
w/o seq sim	26.27	29.74	34.26	27.45
CoNT	**27.70**	**30.91**	**35.55**	**28.04**
w/o seq sim	27.42	30.54	34.69	27.77

Table 2: ROUGE score on Summarization datasets. Results with † are token from [21] and results with * are from [52]. Current state-of-the-art models and the best results are in bold. Earlier SOTA means the best results before PEGASUS.

Model	**XSum**			**Multi-News**		
	R-1	**R-2**	**R-L**	**R-1**	**R-2**	**R-L**
T5-small	36.10	14.72	29.16	42.36	15.34	21.91
T5-SSMBA CL	36.58	14.81	29.68	42.06	14.98	21.73
T5-Dropout CL	36.82	14.93	29.26	42.43	15.32	21.95
T5-CLAPS†	37.89	15.78	30.59	–	–	–
T5-Naive CL	36.34	14.81	29.41	42.20	15.18	21.78
T5-Naive CL (N-Pairs)	37.76	15.48	30.15	43.04	15.83	22.03
T5-CoNT	<u>39.66</u>	<u>16.96</u>	<u>31.86</u>	<u>44.08</u>	<u>16.39</u>	<u>22.58</u>
Earlier SOTA*	45.14	22.27	37.25	43.47	14.89	17.41
PEGASUS (base)*	39.79	16.58	31.70	42.24	13.27	21.44
PEGASUS (large)*	47.21	24.56	39.25	47.52	18.72	**24.91**
PEGA-CoNT	**47.76**	**24.69**	**39.46**	**48.68**	**19.29**	24.58

method is not restricted to the small model. By employing CoNT on state-of-the-art base model PEGASUS, it is able to establish new state-of-the-art on the two summarization benchmarks.

Code Comment Generation Code comment generation aims to generate an English description for a function-level code snippet. We test our method on two widely used datasets Java and Python from the CodeXGLUE benchmark [25]. Results are shown in Table 3. Our model is built upon state-of-the-art pre-trained model on program language model CodeT5-base. CodeT5-Dual-Gen means they further involve a comment-to-code task which is the best model on Python and Java without using external data. We also include the results of earlier strong pre-trained baselines: PLBART and CodeBERT. We also report some data augmentation methods in the third block of Table 3. CodeT5-Multi-Task makes use of training datasets of other program languages. REDCODER [32]

Table 3: BLEU on two code comment generation datasets Java and Python. Results with † and * are from [48]

Model	Python	Java
CodeBERT †	19.06	17.65
PLBART †	19.30	18.45
CodeT5 †	20.01	20.31
CodeT5-Dual-Gen†	20.11	20.41
With N-Pairs CL		
CodeT5-Naive CL	20.26	20.31
CodeT5-CoNT	20.43	20.56
With External Training Data		
CodeT5-Multi-Task†	20.36	20.46
REDCODER*	**21.01**	**22.94**

Table 4: BLEU on data-to-text generation dataset WikiBio. We run our model three times and report the mean and variance of the BLEU metric. Results with † are token from [23] and results with * are from [2].

Model	BLEU
Table NLM †	34.70 ±0.36
vanilla Seq2Seq †	42.06 ±0.32
StructureAware †	44.89±0.33
R2D2 *	46.23±0.15
T5-small †	46.02±0.36
With N-Pairs CL	
T5-small-Naive CL	46.50±0.24
T5-small-CoNT	**47.17**±0.19

Table 5: Results on the dev set and test set Totto. PAR is short for PARENT score. Dev Set (Non) means the non-overlap subset of the dev set. results with † are reported in [16].

Model	Dev Set (All)		Dev Set (Non)		Test Set (All)		
	BLEU	PAR	BLEU	PAR	BLEU	PAR	BLEURT
BERT-to-BERT†	44.0	52.6	34.8	46.7	44.0	52.6	0.121
T5-large†	48.1	57.3	39.8	52.8	–	–	–
T5-3B†	48.4	57.8	40.4	53.3	**49.5**	58.4	0.230
T5-base†	47.7	57.1	39.6	52.6	–	–	–
T5-base-CoNT	**49.2**	**59.38**	**41.5**	**55.0**	49.1	**58.9**	**0.238**

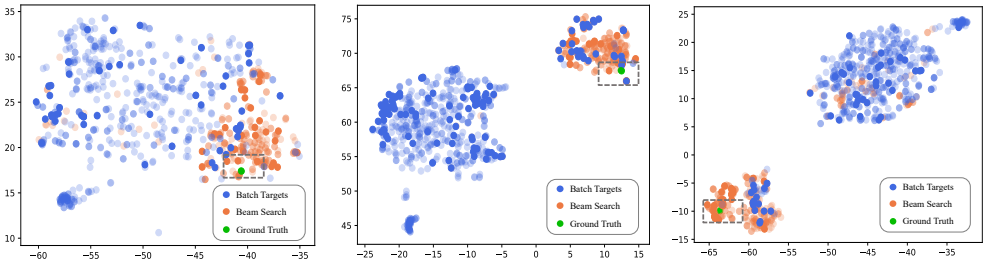
uses retrieval to enhance the task with open-source code base and achieve the bset results on this task. Our model is orthogonal to these methods and clearly outperforms all baselines without external data.

Data-to-text Generation Data-to-text generation aims to produce text from non-linguistic input. The first benchmark we use is WikiBio [20] consisting of biography pairs from English Wikipedia where the infobox is treated as input sequence, and the target sequence is the first sentence of the biography. Totto [31] is also collected from Wikipedia whose input is a table with its highlighted cells and target sequences are professionally annotated by human. Results on WikiBio is shown in Table 4, the performance of some popular baselines (first block) are token from [23]. R2D2 [2], using XLNet [50]-large as base model, is previous state-of-the-art model on WikiBio. We experiment CoNT on T5-small, and results show that we exceed R2D2 by about 0.94 BLEU. The test set and dev set of Totto are both split into two parts - overlap and non-overlap. The non-overlap part contains out-of-domain samples from the training set. The test set of Totto is invisible and we report the results on dev set and test set by the feedback of the Totto authors. We also add PARENT [7] and BLEURT [38] as evaluation metrics. PARENT is a word-overlap based metric that designed to evaluate the factual accuracy of generation results. BLEURT is trained under human supervision and correlates well with human judgement. As we can see from Table 5, by comparing our model with different T5 variants, we show that CoNT is able to greatly outperform the large version of T5 even built on a T5-base model. Compared with previous state-of-the-art model T5-3B, our model still show advantage in PARENT and BLEURT.

Commonsense Generation The task of commonsense generation aims to explicitly test the ability of machines on commonsense reasoning. The source sequence consists of a set of concepts and the target sequence is a fluent sentence mentioning all the input concepts. We evaluate CoNT on the CommonGen [22] benchmark with a hidden test set and the final results is obtained with the help of the authors of CommonGen. In addition to the mostly used metrics, CIDEr [45] and

Table 6: Results on CommonGen. Results with † are reported in [22]. The metrics used in the official leaderboard are in bold. Human performance is also reported as an upper bound.

Model	ROUGE-2/L		BLEU-3/**4**		METEOR	**CIDEr**	**SPICE**	Coverage
GPT-2†	16.85	39.01	33.92	23.73	26.83	12.19	23.57	79.09
BART†	**22.02**	41.78	39.52	29.01	31.83	13.98	28.00	**97.35**
T5-large†	21.74	42.75	**43.01**	**31.96**	31.12	15.13	28.86	92.29
T5-base†	14.63	34.56	28.76	18.54	23.94	9.40	19.87	76.67
T5-base-CoNT	20.96	**43.15**	42.60	31.42	**32.05**	**15.96**	**28.95**	96.55
Human	36.72	54.45	52.55	46.49	38.79	37.64	52.43	99.33



(a) MLE (b) Naive CL (c) CoNT

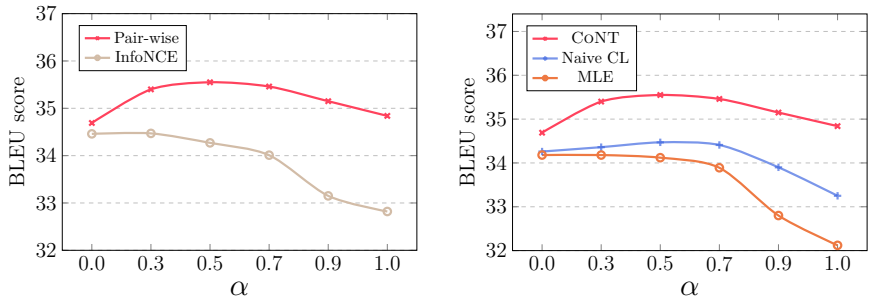
Figure 4: T-SNE experiments on IWSLT'14 De-En. Each point represents a target sequence. Batch target is blue; beam search hypothesis is orange; ground-truth sequence is green. Darker points indicate sequences with higher BLEU.

SPICE [1], concerning to evaluating semantic faithfulness, are highlighted by the leaderboard of commonGen. In Table 6, we demonstrate that the lightweight T5-base model is able to greatly benefit from our contrastive learning framework. Moreover, CoNT not only surprisingly outperforms its MLE baseline but also surpass the large version of T5 in terms of CIDEr and SPICE metrics.

4.3 Discussion

Discrimination of Hard Negative Samples To deeply look into the learnt representations, we visualize target sequences, that is trained by MLE, naive CL and CoNT on IWSLT'14 De-En, with the t-SNE algorithm [43]. The visualized sequences consist of three groups of target sequence: a) batch targets that is mostly unrelated to ground -truth target; b) beam search hypothesis that could be of high/low quality; c) ground truth target. As can be seen in Figure 4a, the representations trained by MLE are distributed almost uniformly in the vector space, and there are no clear boundary between one group to another. With naive CL, we find there is clear boundary between batch tokens and others. Naive CL does contribute to discriminating related sequences with unrelated ones, but it still cannot distinguish hypothesis of high quality from the others. Even without contrastive learning, the generation model trained has already pulled from-batch samples away from the ground truth and the naive contrastive learning procedure is only to make the margin more obvious. As for CoNT , a set of hypotheses of low quality are excluded from the neighborhoods of the ground-truth targets. The experimental results verifies that CoNT enables better representations in sequence generation.

Sequence Likelihood and Sequence Similarity We also perform ablation study on our ranking objective and self-generated negative samples on IWSLT'14 De-En with Transformer-small as base model. For the ranking objective, we compare our N-pairs contrastive objective with InfoNCE with increasing α. Figure 5a that N-pairs contrastive loss consistently outperform InfoNCE. With $\alpha \in [0.3, 0.7]$, N-pairs contrastive loss can benefit from target-source representation similarity is inference process, while InfoNCE cannot. For the sampling strategy, we compare CoNT with naive CL and vanilla MLE. CoNT use self-generated hypotheses as negatives while naive CL only use samples within the same mini-batch. Figure 5b shows that contrastive learning with self-generated hypotheses is more effective than using batch samples. With $\alpha \in [0.3, 0.7]$, CoNT gains about 1.0 BLEU improvements, while naive CL is only improved with less than 0.5 BLEU.



(a) Relationship between α and BLEU with different contrastive loss.

(b) Relationship between α and BLEU with different training methods.

Figure 5: Ablation study on the balance factor α on the test set of IWSLT'14 De-En where $\alpha = 0.0$ means selecting output only relying on likelihood and $\alpha = 1.0$ means choosing output with only sequence similarity.

5 Related Work

Contrastive Learning Contrastive learning [6, 12, 37] aims to learn a better representation via contrasting positive and negative samples. It also has been widely used in the field of natural language processing [9, 13, 19]. Jiang et al. [14] show that contrastive learning helps learn a robust pre-trained model and Lee et al. [21] first introduce contrastive learning into text generation to mitigate the exposure bias problem. They propose an adversarial method to build more challenging positive-negative samples in additional to the from-batch samples.

Adopting binary supervision in contrastive loss is originally proposed in FaceNet [37] which helps learn the face recognition of the same person in various positions and angles. Given an anchor face image $\mathbf{x}$, a positive sample $\mathbf{x}^+$ (usually the same person) and a negative sample $\mathbf{x}^-$ from other people, the triplet loss makes $\mathbf{x}^+$ become close to $\mathbf{x}$ and maximize the distance between $\mathbf{x}$ and $\mathbf{x}^-$. The pair-wise contrastive loss has also been widely used in metric learning Chen and Deng [5], Kim et al. [18], Wang et al. [47]. Sohn [39] extend the triplet loss to multi-class and multi-pair. Recent work thinks the margin value between samples should not be fixed. Zhou et al. [54] divide the sample set into multiple subsets and assign different margin value to different subsets. Ha and Blanz [11], Zhong et al. [53] suggests dynamically adjust the margin value via a determination metric.

Post-generation Re-ranking Models Post-generation re-ranking re-score the multiple output sequences via training another re-ranking module. Noisy Channel Modeling (NCM) [27, 51] is a widely-used re-ranking scheme for neural machine translation. NCM parameterizes the noisy channel probability with a sequence-to-sequence model. There also various structures to instantiate the re-ranking module: Gulcehre et al. [10] select the candidate with a language model, Bhattacharyya et al. [4] leverage an energy-based model in NMT and Liu and Liu [24], Salazar et al. [36] re-score candidates with masked language models such as BERT. Despite this paradigm achieves impressive results while having a large size of candidate sequences, most of post-generation re-ranking systems trade efficiency and simplicity for accuracy.

6 Conclusion

We introduce a new contrastive neural text generation framework called CoNT. It models an additional contrastive learning objective to provide a sequence-level supervision for auto-regressive neural text generation models. We explore three shortcomings that limit the development of contrastive learning on text generation tasks. Results on five generation tasks with ten different benchmarks show that CoNT not only clearly beats all previous contrastive generation models, but also boosts the performance of state-of-the-art large models to a new level. CoNT practically does not have a negative impact on decoding speed. Nevertheless, CoNT suffers from the training inefficiency problem. In general, the total training time of CoNT is about 2∼4 times more than that of a MLE based model. A detailed discussion and some speed-accuracy trade-off tricks can be found in Appendix B. Speeding up the training stage without losing accuracy is the next important step to improve CoNT.

References

[1] Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. Spice: Semantic propositional image caption evaluation. In *European conference on computer vision*, pages 382–398. Springer, 2016.

[2] Aryan Arbabi, Mingqiu Wang, Laurent El Shafey, Nan Du, and Izhak Shafran. R2d2: Relational text decoding with transformers. *arXiv preprint arXiv:2105.04645*, 2021.

[3] Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

[4] Sumanta Bhattacharyya, Amirmohammad Rooshenas, Subhajit Naskar, Simeng Sun, Mohit Iyyer, and Andrew McCallum. Energy-based reranking: Improving neural machine translation using energy-based models. *arXiv preprint arXiv:2009.13267*, 2020.

[5] Binghui Chen and Weihong Deng. Energy confused adversarial metric learning for zero-shot image retrieval and clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8134–8141, 2019.

[6] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.

[7] Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William W Cohen. Handling divergent reference texts when evaluating table-to-text generation. *arXiv preprint arXiv:1906.01081*, 2019.

[8] Alexander Richard Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir Radev. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1074–1084, 2019.

[9] Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

[10] Caglar Gulcehre, Orhan Firat, Kelvin Xu, Kyunghyun Cho, and Yoshua Bengio. On integrating a language model into neural machine translation. *Computer Speech & Language*, 45:137–148, 2017.

[11] Mai Lan Ha and Volker Blanz. Deep ranking with adaptive margin triplet loss. *arXiv preprint arXiv:2107.06187*, 2021.

[12] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[13] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Philip Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *International Conference on Learning Representations*, 2018.

[14] Ziyu Jiang, Tianlong Chen, Ting Chen, and Zhangyang Wang. Robust pre-training by adversarial contrastive learning. *Advances in Neural Information Processing Systems*, 33:16199–16210, 2020.

[15] Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. *Advances in Neural Information Processing Systems*, 33:21798–21809, 2020.

[16] Mihir Kale and Abhinav Rastogi. Text-to-text pre-training for data-to-text tasks. *arXiv preprint arXiv:2005.10433*, 2020.

[17] David A Kalkstein, David A Bosch, and Tali Kleiman. The contrast diversity effect: Increasing the diversity of contrast examples increases generalization from a single item. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, 46(2):296, 2020.

[18] Sungyeon Kim, Minkyo Seo, Ivan Laptev, Minsu Cho, and Suha Kwak. Deep metric learning beyond binary supervision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2288–2297, 2019.

[19] Lingpeng Kong, Cyprien de Masson d'Autume, Lei Yu, Wang Ling, Zihang Dai, and Dani Yogatama. A mutual information maximization perspective of language representation learning. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. URL `https://openreview.net/forum?id=Syx79eBKwr`

[20] Rémi Lebret, David Grangier, and Michael Auli. Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213, 2016.

[21] Seanie Lee, Dong Bok Lee, and Sung Ju Hwang. Contrastive learning with adversarial perturbations for conditional text generation. In *International Conference on Learning Representations*, 2020.

[22] Bill Yuchen Lin, Wangchunshu Zhou, Ming Shen, Pei Zhou, Chandra Bhagavatula, Yejin Choi, and Xiang Ren. Commongen: A constrained text generation challenge for generative commonsense reasoning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1823–1840, 2020.

[23] Tianyu Liu, Kexiang Wang, Lei Sha, Baobao Chang, and Zhifang Sui. Table-to-text generation by structure-aware seq2seq learning. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[24] Yixin Liu and Pengfei Liu. Simcls: A simple framework for contrastive learning of abstractive summarization. *arXiv preprint arXiv:2106.01890*, 2021.

[25] Shuai Lu, Daya Guo, Shuo Ren, Junjie Huang, Alexey Svyatkovskiy, Ambrosio Blanco, Colin Clement, Dawn Drain, Daxin Jiang, Duyu Tang, et al. Codexglue: A machine learning benchmark dataset for code understanding and generation. *arXiv preprint arXiv:2102.04664*, 2021.

[26] Shashi Narayan, Shay B Cohen, and Mirella Lapata. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, 2018.

[27] Nathan Ng, Kyra Yee, Alexei Baevski, Myle Ott, Michael Auli, and Sergey Edunov. Facebook fair's wmt19 news translation task submission. *arXiv preprint arXiv:1907.06616*, 2019.

[28] Nathan Ng, Kyunghyun Cho, and Marzyeh Ghassemi. Ssmba: Self-supervised manifold based data augmentation for improving out-of-domain robustness. *Empirical Methods in Natural Language Processing, EMNLP*, 2020.

[29] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 48–53, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-4009. URL https://aclanthology.org/N19-4009.

[30] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 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, 2002.

[31] Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. Totto: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1173–1186, 2020.

[32] Md Rizwan Parvez, Wasi Uddin Ahmad, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. Retrieval augmented code generation and summarization. *arXiv preprint arXiv:2108.11601*, 2021.

[33] Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*, 2017.

[34] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*, 2019.

[35] Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. *arXiv preprint arXiv:2010.04592*, 2020.

[36] Julian Salazar, Davis Liang, Toan Q Nguyen, and Katrin Kirchhoff. Masked language model scoring. *arXiv preprint arXiv:1910.14659*, 2019.

[37] Florian Schroff, Dmitry Kalenichenko, and James Philbin. Facenet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823, 2015.

[38] Thibault Sellam, Dipanjan Das, and Ankur P Parikh. Bleurt: Learning robust metrics for text generation. *arXiv preprint arXiv:2004.04696*, 2020.

[39] Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objective. *Advances in neural information processing systems*, 29, 2016.

[40] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks, 2014. URL https://arxiv.org/abs/1409.3215.

[41] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[42] Aaron Van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv e-prints*, pages arXiv–1807, 2018.

[43] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.

[44] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[45] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015.

[46] Ashwin K Vijayakumar, Michael Cogswell, Ramprasath R Selvaraju, Qing Sun, Stefan Lee, David Crandall, and Dhruv Batra. Diverse beam search: Decoding diverse solutions from neural sequence models. *arXiv preprint arXiv:1610.02424*, 2016.

[47] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1386–1393, 2014.

[48] Yue Wang, Weishi Wang, Shafiq Joty, and Steven CH Hoi. Codet5: Identifier-aware unified pre-trained encoder-decoder models for code understanding and generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8696–8708, 2021.

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

[50] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32, 2019.

[51] Kyra Yee, Yann Dauphin, and Michael Auli. Simple and effective noisy channel modeling for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5696–5701, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1571. URL https://aclanthology.org/D19-1571.

[52] Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, pages 11328–11339. PMLR, 2020.

[53] Ming Zhong, Pengfei Liu, Yiran Chen, Danqing Wang, Xipeng Qiu, and Xuanjing Huang. Extractive summarization as text matching. *arXiv preprint arXiv:2004.08795*, 2020.

[54] Mo Zhou, Zhenxing Niu, Le Wang, Zhanning Gao, Qilin Zhang, and Gang Hua. Ladder loss for coherent visual-semantic embedding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 13050–13057, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes] See Appendix B.

(c) Did you discuss any potential negative societal impacts of your work? [Yes] See Appendix E

(d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]

2. If you are including theoretical results...

(a) Did you state the full set of assumptions of all theoretical results? [N/A]

(b) Did you include complete proofs of all theoretical results? [N/A]

3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] We include them in our supplemental materials.

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Appendx C

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No] We only report the error bars for WikiBio following previous work. Since we have 10 benchmarks and 4 base models and some experiments requires several hundred GPU hours to run, so it is quite costly to compute error bars

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Appendx C

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [Yes] Yes See Appendix C

(b) Did you mention the license of the assets? [Yes] We forked transformers [49] and fairseq [29] and the license for them can be found in their offical site.

(c) Did you include any new assets either in the supplemental material or as a URL? [Yes] We submit our code in the supplemental material.

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]