CoP: Factual Inconsistency Detection by Controlling the Preference

Shuaijie She, Xiang Geng, Shujian Huang*, Jiajun Chen

National Key Laboratory for Novel Software Technology, Nanjing University
{shesj, gx}@smail.nju.edu.cn, {huangsj,chenjj}@nju.edu.cn

Abstract

Abstractive summarization is the process of generating a summary given a document as input. Although significant progress has been made, the factual inconsistency between the document and the generated summary still limits its practical applications. Previous work found that the probabilities assigned by the generation model reflect its preferences for the generated summary, including the preference for factual consistency, and the preference for the language or knowledge prior as well. To separate the preference for factual consistency, we propose an unsupervised framework named CoP by controlling the preference of the generation model with the help of prompt. More specifically, the framework performs an extra inference step in which a text prompt is introduced as an additional input. In this way, another preference is described by the generation probability of this extra inference process. The difference between the above two preferences, i.e. the difference between the probabilities, could be used as measurements for detecting factual inconsistencies. Interestingly, we found that with the properly designed prompt, our framework could evaluate specific preferences and serve as measurements for fine-grained categories of inconsistency, such as entity-related inconsistency, coreference-related inconsistency, etc. Moreover, our framework could also be extended to the supervised setting to learn better prompt from the labeled data as well. Experiments show that our framework achieves new SOTA results on three factual inconsistency detection tasks.

Introduction

Current abstractive summarization models (Lewis et al. 2020) can generate fluent summaries which are rich in information. However, over 70% of the summaries generated by existing models contain factual inconsistency, which greatly damages the quality of the summary and limits the practical applications (Maynez et al. 2020; Pagnoni, Balachandran, and Tsvetkov 2021). The first step to solve this problem is to evaluate the factual consistency of the summary, in other words, to detect inconsistency (Goodrich et al. 2019).

Some research directly uses the preference of the probabilistic generation model, i.e. the generation probability, as an indicator for factor inconsistency (Yuan, Neubig, and Liu 2021). But, since the generation model (Lewis et al. 2020) is usually pre-trained on large-scale corpora, the model would also have a preference for language or knowledge prior. To solve the problem, Xie et al. (2021) introduces an extra inference with a partially masked document as input and uses the probability difference as the indicator for inconsistency. However, since the masked document is inherently against the language prior, due to its nonfluency, the evaluation is still entangled by the preference of the model. Besides, it is also difficult to decide which word to mask in the document.

In this paper, we propose an unsupervised framework to elaborately control the preferences with the help of prompt, namely CoP. CoP employs a fluent text prompt, i.e. the summary itself, as the additional input for the extra inference step, which is easily to obtain and less relevant to the preference of the model. The resulting generation probability is conditioned on both the document and the prompt. So that we can use the differential probability between the two inference processes to detect factual inconsistency.

One advantage of the proposed framework is that the preference could be easily controlled by varying the prompt. With proper designed prompts, CoP could evaluate the specific preference and serve as measurements for fine-grained categories of inconsistencies, such as the entity-related inconsistency, the coreference-related inconsistency (Pagnoni, Balachandran, and Tsvetkov 2021), etc.

Our framework not only works well in the unsupervised manner; its performance could be further improved by supervised training with prompt learning techniques (Li and Liang 2021; Liu et al. 2021b). Compared with previous work (Zhou et al. 2020; Goyal and Durrett 2021) which fine-tune a large model, we could fine-tune a prompt vector with limited labeled data, which enjoys advantages in both effectiveness and efficiency.

Experiments are conducted on token-level and summary-level inconsistency detection and inconsistency category detection, where CoP achieves remarkable improvements over several strong baselines in the unsupervised settings. For detecting inconsistency categories, using specifically designed prompts brings further improvements without any additional training, revealing the flexibility of the prompt design and the great potential of our approach. With supervised training of the prompt, large improvements are achieved with only 0.02% of the total parameters updated, showing an efficient

*Corresponding author

strategy for the low-resource problem. In all the above experiments, our framework achieves better performance than the current state-of-the-art, to our best knowledge.[1]

Preliminaries

Abstractive Summarization

The essence of abstractive summarization is a conditional generation process: generating the target summary $Y = \{y_1, y_2, \ldots, y_n\}$ with a given document $X = \{x_1, x_2, \ldots, x_m\}$ as input. In the summarization process, there is a abstractive summarization model M with parameter θ. In each inference step i, the model will give the probability distribution of the current token y_i based on the following probability: $P(y_i|X, y_{<i}; \theta)$, where $y_{<i} = \{y_1, y_2, \ldots, y_{i-1}\}$ denotes the prefix tokens (Cao, Dong, and Cheung 2021; Xie et al. 2021).

Factual Inconsistency Detection

Given a document-summary pair (X, Y), the factual inconsistency detection task is to decide whether C is supported by the document (Maynez et al. 2020; Zhou et al. 2020), where C can be defined in multiple granularities. In summary-level, C is the whole summary, meaning that the task is to decide whether the whole summary is consistent with the given document. In this setting, a factual score F is usually assigned to each summary. In a finer granularity, C could be each token in the summary, meaning that the task is to check token-level factual consistency, which assigns a factuality label for each token.

Document:
Mr Charney, who also founded the company, was ousted last year because of the employee complaints and amid accusations of misuse of company funds. In the San Francisco court filing, the board said it did not expect ...
Summary (with Token-Level Labels):
The former chief executive of the San Francisco court , Charney , has been cleared by a court in California.
Summary-Level Labels : 0 (inconsistent)
Category Labels : EntE, OutE

Table 1: An example of factual inconsistency detection in multiple granularities (underlined words are inconsistent words in the token level; EntE and OutE are inconsistency categories, denoting Entity Error and Out of Article Error).

Factual inconsistencies could also be related to different categories. Besides sentence-level and token-level labels, another task is to specify the detailed type T of inconsistency. In this paper, we follow the typology definition of factual inconsistency from Pagnoni, Balachandran, and Tsvetkov (2021). Table 1 shows an example: the four underlined tokens, "San Francisco court" and "cleared" are token-level inconsistent contents C. The summary-level score F indicates that the summary is inconsistent. Finally, the category label EntE represents the summary containing entity-related inconsistency.

[1]Code will be released at https://github.com/NJUNLP/CoP

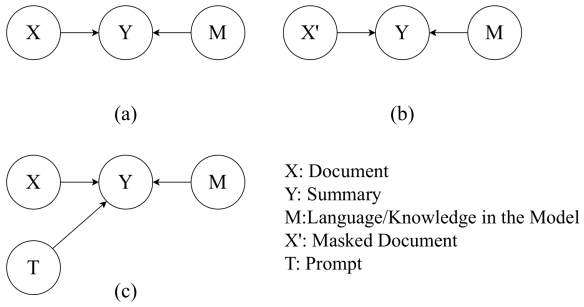

Figure 1: Illustration of different inference processes: (a) the regular inference, where the generation of Y is determined by the document and the pre-trained model; (b) the inference in Xie et al. (2021) with a partially masked document; (c) our extra inference step with an additional prompt.

Our Method

Motivation

The generation of a summarization involves at least the following two factors. The document X provides the important source of factual information to support the consistent summary, leading to a preference for factually consistent summaries. On the other hand, the model M, usually pre-trained on larger texts, provides the essential language prior knowledge to make the summary fluent in the generation process, i.e. preferring fluent summarization result. Therefore, the generation probability p of each summary token is jointly decided by the model M and source document X. The above causal relations are presented in Figure. 1(a).

The essence of the factual inconsistency detection is to evaluate how much the summary Y is supported by the document X, that is to estimate the causal effect of $X \rightarrow Y$. Obviously, simply using the probability from the regular inference process, such as BARTScore (Yuan, Neubig, and Liu 2021), cannot discriminate the preferences from X and M and may lead to errors in detection. For example, some factually consistent but not fluent summaries will be misjudged as inconsistency.

The probability differential approach employs probability from an extra inference process to disentangle the preference from X and M. Xie et al. (2021) propose to introduce a partially masked document as the input for the extra inference (Figure 1(b)). However, because the masked document is usually less fluent which also contradicts the language prior knowledge, the generation probability will still be entangled with M. Moreover, it is difficult to decide which document token to mask precisely.

We propose a different extra inference process to obtain the differential probabilities. Instead of disturbing the original input X, we keep X as it is, but employ an extra prompt as the additional input (Figure 1(c)). The prompt could be

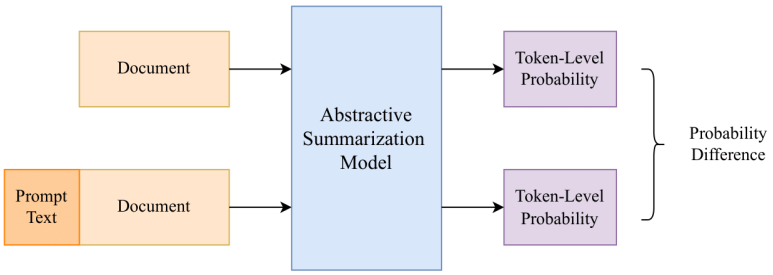

Figure 2: Our framework of controlling the preference: CoP

fluent text, e.g. the summary itself, which does not affect the preference from M. So the differential probability would be mainly affect by the preference of X. We present our framework CoP, and its advantages in the rest of this section.

Controlling the Preference by an Extra Inference with Prompt

Our framework consists of two inference processes (Figure 2). The first process is the same with regular inference, where the document is used as input and the decoder is forced to generate the given candidate summary using forced-decoding (Yuan, Neubig, and Liu 2021). The log probability of generating each token in the candidate summary is denoted as P_1. The second inference process takes an text prompt T together with the document X as the input. With the same forced-decoding process, a second log probability is obtained, denoted as P_2.

The choice of T is flexible and may adapt to different scenarios. In the simplest case, we take the candidate summary Y as the prompt (also called prompt text specifically). The intuition lies in how the prompt affects the generation probability. Intuitively, the consistent part in the summary is redundant to the generation model and thus will not bring large changes in probability. On the other hand, the inconsistent part in the summary will bring a more significant variation in its generation probability. In other words, the probability difference is caused by the model's preference for factual consistency, which filters out the irrelevant preferences such as fluency. In practice, we can calculate P_{diff} of each token in the summary by following equation:

$$P_{\text{diff}}(y_i) = P_2(y_i) - P_1(y_i) \quad (1)$$

Tokens with higher P_{diff} are more likely to be factual inconsistent with the document. According to the certain applications, we can set different thresholds to control the proportion of predictions, and the tokens above the threshold are identified as inconsistent. For example, we can choose a relatively low threshold for inconsistent error correction task which higher recall is preferred.

Please note that in the area of prompt learning, prompt is the extra input added to the original input, which motivates the model to use its pre-trained abilities to accomplish specific tasks (Liu et al. 2021a). Here we use prompt as an variant of input, which motivates the model to generate different probabilities. The method is simple and requires no additional training.

Design Prompt for Fine-Grained Category

Previous work (Pagnoni, Balachandran, and Tsvetkov 2021) defined a typology of the factual inconsistency, annotated datasets and analyzed the distribution of inconsistency category. However, existing factuality assessment methods usually overlook these category annotations and only use the overall score for evaluation. We argue that the detailed category information of inconsistency will be helpful to analyze inconsistency tendency of existing models and provide guidance for future improvement. According to Pagnoni, Balachandran, and Tsvetkov 2021, EntE (Entity-related Error), CorefE (Coreference-related Error) and OutE (Not in Article Error) appear with a high frequency of 36%, 10% and 27%, respectively. We try to address this issue and first consider the three most frequent inconsistency categories to illustrate how our framework works. Since OutE corresponds to errors that are not in the article, we can solve this inconsistent category with the candidate summary as prompt. So we also included it in the experiment as a comparison.

In addition to using our framework directly, we further design various prompts to control the preference of generation model. The base version of prompt text is the whole candidate summary that cover all inconsistent tokens in the summary. For detecting inconsistencies in each category, we could add fact information related to the specific category to improve the detection process. The intuition is similar: the consistent category-related information is redundant to the generation model, while the inconsistent category-related information will bring a more significant category-targeted probability variation.

In this case, we can extract entities from the summary and append the entity token list after the original prompt text. If the probability of the summary is greatly affected by additional entity information, we can consider that this summary contains inconsistency of entity category. For CorefE, we can start with reference resolution of the summary and add these fact information: insert resolved reference token after the pronoun tokens. We believe that the addition of entities adds more entity information to the prompt text, bringing more entity-targeted probability changes. And the coreference information can help model analyze and compare the co-reference facts in the abstract.

At the moment we are still getting token-level scores, and category annotations are at the summary-level. The most straightforward way is to simply average all the token-level

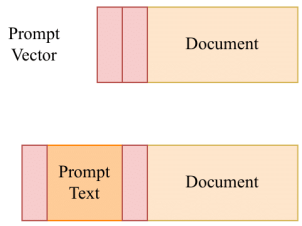

Figure 3: Illustration of prompt vector (marked in red).

scores (all tokens are equally weighted). It's worth noting that our framework can finely check the factuality of each token in the summary, including the entity tokens and pronoun tokens. We can make better use of this information. We simply double the weights of entity tokens and pronoun tokens to help the model focus on the specific category.

Learn From Labeled Data with Prompt Tuning

Although our framework already works well in training-free style by exploiting the preference in summarization model, it is still possible to take advantage of the labeled data, which maybe extremely scarce. Thanks to the flexible structure of our framework, we can integrate the prompt tuning technique, which achieves great success in few-shot tasks.

To learn a prompt text in discrete tokens is difficult, instead, we propose to a small-sized continuous task-specific vector (we call it **prompt vector** as opposed to **prompt text**). We expect the prompt vector to help the model distinguish the prompt text and document, and guide the model to make a fine inferential comparison on factuality between the candidate summary and the document. More specifically, we add the same prompt vector before and after the prompt text in the extra inference step as shown in Figure 3. To keep the inference similar in the two processes, the same prompt vector are added to the regular inference process with no prompt text between them. To train the prompt vectors, we froze the whole summarization model and only update the small-scale prompt vector with the following loss function:

$$\mathcal{L} = \sum_i P_{\text{diff}}(y_i) * label \tag{2}$$

$label$ is the token-level label of 1 and -1, indicating whether y_i is consistent or inconsistent, respectively. Our loss function is designed succinctly for task purposes: further maximizing the P_{diff} of inconsistent tokens and minimizing the P_{diff} of consistent tokens. Minimizing the loss with labeled data strengthens the preference for factual consistent and provides a clearer discriminating boundary.

Experiment Setup

Detect Inconsistency in Unsupervised Manner

Dataset: XSum Hallucination Annotations (Maynez et al. 2020) sampled 500 document-summary pairs from the summarization dataset XSUM (Narayan, Cohen, and Lapata 2018) and explored the fact consistency of summary generated by four popular models (PtGen, TConvS2S, TranS2S, BERTS2S). The annotator marks each token with 0 or 1 to indicate its factuality. In addition to the above token-level dataset, we also take two popular summary-level datasets into consideration: QAGS (Wang, Cho, and Lewis 2020) and FRANK (Pagnoni, Balachandran, and Tsvetkov 2021) datasets. Both two datasets contain summary-level factuality annotations on CNNDM and XSUM datasets respectively.
Implementation Details: We take released BARTCNN[2] (BART (Lewis et al. 2020) fine tuned on CNNDM dataset) as the backbone of our framework. Our framework ranks tokens with the P_{diff} as score and consider tokens with scores above the threshold as inconsistency. For a fair comparison, we follow the setting in DHC (Zhou et al. 2020) by setting the threshold to predict a close proportion of inconsistency.

Improve Performance with Efficient Prompt Tuning

Dataset: Following previous work, we use the token-level dataset as in Section . Intuitively, token-level inconsistency detection is more difficult and requires the fine-grained analysis capabilities of the model.
Implementation Details: In full-shot setting, the dataset is split into three subsets: training (1200), validation (400), and test (400) sets which is similar to DAE (Goyal and Durrett 2021). We cut the training set to 300 for the few-shot settings. The length of prompt vector is set to 40 and 5 for full-shot and few-shot respectively. The prompt tuning process uses AdamW optimizer with 1e-3 learning rate and the experiment are conducted on single TITAN-RTX GPU.

Detailed Inconsistency Category Evaluation

Dataset: In addition to the factual consistency scores at the summary level, FRANK (Pagnoni, Balachandran, and Tsvetkov 2021) dataset provides more detailed assessment information by showing the types of inconsistency in each summary. On the FRANKCNN set, the candidate summary is longer and there are richer and more comprehensive inconsistency types, so we use this dataset to verify the effectiveness of our method.
Implementation Details: For CorefE, we use SpanBERT (Joshi et al. 2019) to generate coreference targeted prompt text. According to the definition of CorefE, We filtered the summaries without pronouns. For EntE, we use Spacy[3] to extract entity prompt text. The method of using the above strategy is called Ours Variant.

Experiment Results and Comparison

Detect Inconsistency in Unsupervised Manner

Following DAE and DHC, we report F1 on each data split in Table 2 and corpus-level F1 in Table 3. From a train-free perspective, Ours Zero-Shot achieves impressive performance, increasing the F1 score by **4.64** compared to the best method BARTSc-Ours. It is a very intuitive demonstration of the effectiveness of the extra inference and prompt.

[2]https://huggingface.co/facebook/bart-large-cnn
[3]https://spacy.io/

Model	PtGen	TConvS2S	TranS2S	BERTS2S	Average F1
Alignment-based *	38.92	37.94	34.47	35.81	36.78
Overlap-based *	57.22	54.25	53.79	55.13	55.09
Synonym-based *	59.54	63.73	58.66	53.07	58.75
DHC Model	64.72	69.37	63.88	56.49	63.61
BARTSc-Ours *	60.59	63.41	59.76	51.82	58.90
Ours Zero-Shot *	63.08	68.61	63.90	58.58	63.54
Ours Few-Shot	68.27	70.31	66.03	60.19	66.20
Ours Full-Shot	**71.06**	**72.91**	**67.64**	**65.67**	**69.32**

Table 2: F1 $(\times 100)$ on each benchmark split, * denotes that this method is training-free

Model	Corpus F1
DAE-Weak	59.10
BARTSc-Ours *	59.25
EntFA	60.23
Ours Zero-Shot *	**63.72**
DAE	65.00
Ours Few-Shot	**66.56**
Ours Full-Shot	**69.61**

Table 3: Corpus-Level F1 $(\times 100)$ Performance, * denotes that this method is training-free

Compared with some methods training with large-scale pseudo-data, Our Zero-Shot still shows very competitive results, even outperforming DAE-Weak by **4.62** in corpus-level F1. Moreover, our model has a steady improvement on each data subset, further proving the generalization of our model to handle different styles of candidate summaries generated by different summarization models well.

Table 4 shows the pearson correlation with human score in summary-level inconsistency detection task. Our model achieve SOTA results in all four datasets. It's worth noting that our model leads SOTA by a larger margin in two XSUM datasets: our method exceeded BARTScore by **3.98** and **5.34** on QAGSXSUM and FRANKXSUM respectively. This phenomenon may be related to the fact that XSUM is more abstractive and contains more noise which proves that the CoP can better separate the unrelated preference for fluency and focus on the verification of fact consistency.

Improve Performance with Efficient Prompt Tuning

We further validate the effectiveness of our designed prompt tuning in the token-level inconsistency detection task as shown in Table 2 and Table 3. Using only 300 annotated data significantly improved the performance of our model to a new SOTA level, exceeding DAE, which uses 2K annotated data and DHC, which requires 960K pseudo-data. This suggests that CoP can learn more efficiently from small amounts of data.

As the scale of annotated data increases, the performance of our model is further improved. When we train with the entire training set (1200), the corpus level F1 came to **69.61**, which is an improvement of 5.89 (9.24%) over the impressive Our Zero-shot. Thanks to the effectiveness of the prompt tuning we designed, we were able to fully tap the potential of CoP with the scarce data. Another thing worth noting is that compared with DAE, which used more annotated data, we improved 4.61, showing a strong effectiveness.

Detailed Inconsistency Category Evaluation

The experimental results of Table 6 and Table 5 show that our baseline method is not only good at detecting fine-grained fact inconsistency but also good at detecting certain types of factual inconsistency errors. When Ours Base has exceeded the previous work, Ours Variants can even further improve the pearson correlation with human annotation in different categories of inconsistency. It is surprising enough to note that we did not have additional training to achieve this, but with the flexible design of the prompt.

What is more surprising is that when the designed prompt improves the evaluation of specific categories, it also affects the pearson correlation of Overall and OutE. We think this is due to: (1) EntE is a fairly frequent inconsistency. When we improve the detection of this category, the model's ability to identify overall inconsistencies will also be improved. (2) There is some relation between inconsistency categories, such as between EntE and OutE. If the model fails to understand entities in document, it is also more likely to generate OutE. That's why a prompt for one category can help detect other inconsistency categories.

Analysis

Case Study

We sampled two examples in Table 7 and compared the assessment results of CoCo and our proposed CoP. For the Summary1 , it is factually consistent with redundant generation which is a common problem in abstractive summary. If the evaluation model has difficulty separating fluency preferences, redundant spans that reduce the fluency will mislead the model into thinking that the summary is inconsistent. Obviously, our model is better to separate preferences and accurately judge that the summary is more consistent.

Another example, the sentence is quite fluent, but it has a common factual inconsistency: Entity Error. Such errors are frequent and subtle, making discrimination difficult. CoCo

Model	QAGSCNN	QAGSXSUM	FRANKCNN	FRANKXSUM
BERTScore	27.63	2.51	32.85	18.63
QAGSScore	54.53	17.49	30.45	-5.847
BARTScore	71.43	22.65	55.84	17.43
CoCoScore	58.84	24.08	-	-
Ours Zero-shot	73.02	26.63	56.09	22.77

Table 4: Summary-level pearson correlation($\times 100$) between metrics and human judgments of factuality

Model	Overall	EntE	OutE
BERTScore	23.62	14.57	0.092
QAGSScore	30.15	21.40	12.19
BARTScore	55.98	32.58	18.56
Ours Base	56.07	33.32	23.47
Ours Variant	57.66	34.66	26.09

Table 5: Pearson correlation($\times 100$) between metrics and human judgments of EntE

Model	Overall	CorefE	OutE
BERTScore	31.33	24.65	15.86
QAGSScore	29.67	10.09	12.28
BARTScore	56.38	29.29	34.76
Ours Base	59.76	34.77	37.89
Ours Variant	60.57	35.47	36.55

Table 6: Pearson correlation($\times 100$) between metrics and human judgments of CorefE

gives Summary2 a even higher score than Summary1 which is more consistent. On the contrary, CoP shows better ability in evaluation factual consistency.

Robustness on Different Backbones

We tested our method based on various backbones on QAGSCNN dataset. We choose BART (Lewis et al. 2020), PEGASUS (Zhang et al. 2020), BRIO (Liu et al. 2022), T5 (Raffel et al. 2020) and the experimental results are shown in Table 8. CoP achieves stable and impressive improvements on different backbones against strong baselines.

Flexible Length of Prompt Vector

Since we are the first to propose this novel training schema in factual consistency task, we analyzed the important properties of prompt vector: prompt length. As the length of the prompt vector increases, it means that there are more trainable parameters and the model will have more expressive ability. The Figure 4 shows that performance increases as the length of prompt vector increases up to a threshold. We also observed the slight performance degradation above the threshold. We think this is mainly because in this low-resource and noisy dataset, the model suffers from a risk of overfitting to the noise in the data. Fortunately, our adjustable training parameters help us deal with scenarios with different size of data, while the previous work can only fine tune from a fixed-size PLM. Moreover, even with an unreasonable prompt length (such as 100), our framework still outperforms the previous SOTA model which further demonstrates the robustness of our framework.

Document1: Trevor Deely, 22, was last seen walking home from a Christmas party in December 2000. A search of a site in Chapelizod in Dublin started ...
Summary1: A search is under way for a man missing in the republic of Ireland in the republic of Ireland.
CoCo: -4.45
Ours: -2.28
Human Annotation: consistent

Document2: The win means Egypt play Ghana on Wednesday with top spot of Group D still at stake, while Uganda are out of their first tournament since 1978.
Summary2: Uganda was knocked out of the Africa Cup of Nations as they were held to a goalless draw by Uganda at the Africa Cup of Nations.
CoCo: -2.87
Ours: -4.12
Human Annotation: inconsistent

Table 7: Comparison of CoCo and Ours (CoP). Higher score indicates that the model considers the summary to be more consistent. (underlined words are inconsistent)

Model	BART	PEGA	BRIO	T5
BARTSc	71.42	72.10	56.03	43.82
CoCo	58.84	58.96	-	56.13
Ours	73.01	74.01	65.97	69.29

Table 8: Summary-level pearson correlation($\times 100$) on QAGSCNN of CoP and baselines on different backbones.

Significantly Fewer Parameter for Efficiency

The number of trainable parameters greatly affects the training speed and the size of needed GPU memory. Moreover, for the low resource task as factual consistency evaluation, previous methods have to construct large-scale pseudo-data for training. A large amount of pseudo-data further increases the cost of training and brings an irreparable performance gap compared to real data (Goyal and Durrett 2021). We

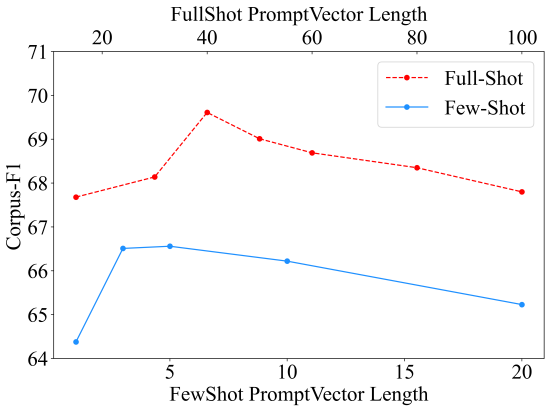

Figure 4: Prompt vector length and corpus-level F1. Two X-axis corresponding to two training settings.

Model	Param Size
DAE	109M
DHC	330M
EntFA	406M
Ours Zero-Shot	0M
Ours Few-Shot	0.01M
Ours Full-Shot	0.08M

Table 9: Trainable parameter size of different methods.

compare trainable parameter size between our framework and other methods in Table 9. The results of the comparison show that we only use **0.02%** of the parameters to exceed the results of previous work which demonstrate that our framework is quite parameter-efficient.

Prompt Tuning Contributes to Clearer Decision Boundaries

The distribution of token-level score is visualized in Figure 5. For comparison, we normalized the scores (P_{diff}). The certain difference in distribution between factual and unfactual tokens explains why CoP is valid under zero-shot. After enhancement by prompt tuning, the distribution presents a better decision boundary, thus distinguishing factual and unfactual tokens better. The distribution difference is a clear demonstration of the feasibility of our framework.

Related Work

Previous work (Pagnoni, Balachandran, and Tsvetkov 2021; Maynez et al. 2020) show that the common n-gram based evaluation toolkits ROUGE (Lin 2004), BLEU (Papineni et al. 2002) can not perform the fact consistency task well. BERTScore (Zhang et al. 2019), though, takes advantage of the contextual information and pre-trained model to improve the performance. However, its correlation with human judgments is still unsatisfactory.

Therefore, a series of work on the assessment of fact consistency has been proposed. Some work extracts facts for

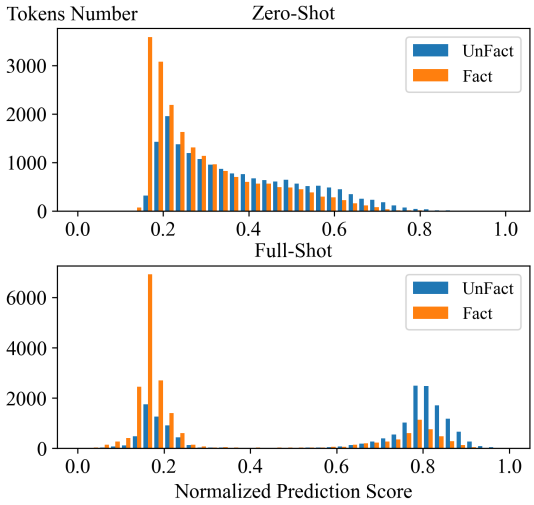

Figure 5: Normalized score distribution. Higher score denotes CoP thinks the token is more inconsistent.

matching, such as QAGS (Wang, Cho, and Lewis 2020) using QA and QG to extract facts from summaries and documents. Some work like FactCC (Kryscinski et al. 2020) directly translates factual consistency judgment into a sentence classification. Recently more and more work pay attention to the generation process. BARTScore (Yuan, Neubig, and Liu 2021) evaluates consistency with generation probability. And CoCo (Xie et al. 2021) models consistency problem by counterfactual and uses mask operations to estimate causal effects. These works mainly focus on coarse-grained at the summary level to measure factual consistency.

In terms of fine-grained factual consistency assessment, DHC (Zhou et al. 2020) constructs large-scale pseudo-data and trains a token-level classifier with Roberta (Liu et al. 2019). Goyal and Durrett (2021) full tuned Electra (Clark et al. 2020) on pseudo-data and small batches of real data. Experimental results show that the huge difference in distribution between pseudo-data and real data greatly limits the effectiveness of current models which encourages us to explore a novel training scheme.

Conclusion

In this paper, we propose CoP that detects factual inconsistency by controlling the preference with the help of prompt. By separating irrelevant preference, CoP can accurately measure factual inconsistency without training. Moreover, CoP could evaluate the specific preference and detect detailed inconsistency categories without training. We also explore to improve the performance by prompt tuning with labeled data which achieved both efficiency and effectiveness. Experimental results on token-level and summary-level factual inconsistency detection and detailed inconsistency category detection demonstrates the effectiveness of our work.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments. Shujian Huang is the corresponding author. This work is supported by National Science Foundation of China (No. 62176120), the Liaoning Provincial Research Foundation for Basic Research (No. 2022-KF-26-02).

References

Cao, M.; Dong, Y.; and Cheung, J. C. K. 2021. Inspecting the Factuality of Hallucinated Entities in Abstractive Summarization. *CoRR*, abs/2109.09784.

Clark, K.; Luong, M.; Le, Q. V.; and Manning, C. D. 2020. ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. *CoRR*, abs/2003.10555.

Goodrich, B.; Rao, V.; Liu, P. J.; and Saleh, M. 2019. Assessing the factual accuracy of generated text. In *proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, 166–175.

Goyal, T.; and Durrett, G. 2021. Annotating and Modeling Fine-grained Factuality in Summarization. *CoRR*, abs/2104.04302.

Joshi, M.; Chen, D.; Liu, Y.; Weld, D. S.; Zettlemoyer, L.; and Levy, O. 2019. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *CoRR*, abs/1907.10529.

Kryscinski, W.; McCann, B.; Xiong, C.; and Socher, R. 2020. Evaluating the Factual Consistency of Abstractive Text Summarization. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 9332–9346. Online: Association for Computational Linguistics.

Lewis, M.; Liu, Y.; Goyal, N.; Ghazvininejad, M.; Mohamed, A.; Levy, O.; Stoyanov, V.; and Zettlemoyer, L. 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*, 7871–7880. Online: Association for Computational Linguistics.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. *CoRR*, abs/2101.00190.

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

Liu, P.; Yuan, W.; Fu, J.; Jiang, Z.; Hayashi, H.; and Neubig, G. 2021a. Pre-train, Prompt, and Predict: A Systematic Survey of Prompting Methods in Natural Language Processing. *CoRR*, abs/2107.13586.

Liu, X.; Zheng, Y.; Du, Z.; Ding, M.; Qian, Y.; Yang, Z.; and Tang, J. 2021b. GPT Understands, Too. *CoRR*, abs/2103.10385.

Liu, Y.; Liu, P.; Radev, D.; and Neubig, G. 2022. BRIO: Bringing Order to Abstractive Summarization. *arXiv preprint arXiv:2203.16804*.

Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; and Stoyanov, V. 2019. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *CoRR*, abs/1907.11692.

Maynez, J.; Narayan, S.; Bohnet, B.; and McDonald, R. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 1906–1919. Online: Association for Computational Linguistics.

Narayan, S.; Cohen, S. B.; and Lapata, M. 2018. 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*, 1797–1807. Brussels, Belgium: Association for Computational Linguistics.

Pagnoni, A.; Balachandran, V.; and Tsvetkov, Y. 2021. Understanding factuality in abstractive summarization with FRANK: A benchmark for factuality metrics. *arXiv preprint arXiv:2104.13346*.

Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P. J.; et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140): 1–67.

Wang, A.; Cho, K.; and Lewis, M. 2020. Asking and Answering Questions to Evaluate the Factual Consistency of Summaries. *CoRR*, abs/2004.04228.

Xie, Y.; Sun, F.; Deng, Y.; Li, Y.; and Ding, B. 2021. Factual Consistency Evaluation for Text Summarization via Counterfactual Estimation. *CoRR*, abs/2108.13134.

Yuan, W.; Neubig, G.; and Liu, P. 2021. Bartscore: Evaluating generated text as text generation. *Advances in Neural Information Processing Systems*, 34.

Zhang, J.; Zhao, Y.; Saleh, M.; and Liu, P. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning*, 11328–11339. PMLR.

Zhang, T.; Kishore, V.; Wu, F.; Weinberger, K. Q.; and Artzi, Y. 2019. BERTScore: Evaluating Text Generation with BERT. *CoRR*, abs/1904.09675.

Zhou, C.; Gu, J.; Diab, M. T.; Guzman, P.; Zettlemoyer, L.; and Ghazvininejad, M. 2020. Detecting Hallucinated Content in Conditional Neural Sequence Generation. *CoRR*, abs/2011.02593.