Enhance Incomplete Utterance Restoration by Joint Learning Token Extraction and Text Generation

Shumpei Inoue[1], **Tsungwei Liu**[1], **Nguyen Hong Son**[1], **Minh-Tien Nguyen**[1,2,*]
[1]Cinnamon AI, 10th floor, Geleximco building, 36 Hoang Cau, Dong Da, Hanoi, Vietnam.
{sinoue, tsungwei.liu, levi, ryan.nguyen}@cinnamon.is
[2]Hung Yen University of Technology and Education, Hung Yen, Vietnam.
tiennm@utehy.edu.vn

Abstract

This paper introduces a model for incomplete utterance restoration (IUR). Different from prior studies that only work on extraction or abstraction datasets, we design a simple but effective model, working for both scenarios of IUR. Our design simulates the nature of IUR, where omitted tokens from the context contribute to restoration. From this, we construct a Picker that identifies the omitted tokens. To support the picker, we design two label creation methods (soft and hard labels), which can work in cases of no annotation of the omitted tokens. The restoration is done by using a Generator with the help of the Picker on joint learning. Promising results on four benchmark datasets in extraction and abstraction scenarios show that our model is better than the pretrained T5 and non-generative language model methods in both rich and limited training data settings.[1]

1 Introduction

Understanding conversational interactions through NLP has become important with increasing connectivity and range of capabilities. The applications using natural conversations cover a wide range of solutions including dialogue systems, information extraction, and summarization. For example, Adiwardana et al. (2020); Su et al. (2020) aimed to build the dialogue system where an intelligent virtual agent answers human conversations and makes suggestions in an open/closed domain. Bak and Oh (2018); Karan et al. (2021) attempted to detect decision-related utterances from multi-party meeting recordings, while Tarnpradab et al. (2018) applied extractive summarization for online forum discussions. These features allow users to to quickly catch up with the current situation, decisions and next-action without having to follow a lengthy or comprehensive dialogue. However, utterances, the components of a conversation, are generally not self-contained and are difficult to understand by their own. This comes from the nature of multi-turn dialogue where each utterance contains co-references, rephrases, and ellipses (Figure 1). Su et al. 2019 also showed that co-references and ellipses occur in over 70% of utterances in conversations. This is a ubiquitous problem in conversational AI, making the challenge for building practical systems with conversations.

Incomplete Utterance Restoration (IUR) (Pan et al., 2019) is one solution to restore semantically underspecified utterances (i.e., incomplete utterances) in conversations. Figure 1 shows an example of IUR, in which the model rewrites the incomplete utterance to the reference. IUR is a challenging task due to two reasons. Firstly, the gold utterance (the reference) overlaps a lot of tokens with the pre-restored, incomplete utterance, while it overlaps only a few tokens with utterances in the context. We observed that for CANARD (Elgohary et al., 2019), 85% of tokens in incomplete utterances were directly cited for rewriting, while only 17% of tokens in context was cited for rewriting. Secondly, it is important to detect omitted tokens in incomplete utterances and to include them in the restoration process. In actual cases of IUR, no matter how fluent and grammatically correct the machine's generation is, it is useless as long as important tokens are left out.

Recent studies used several methods for IUR. It includes the extraction of omitted tokens for restoration (PAC) (Pan et al., 2019), two-stage learning (Song et al., 2020), seq2seq fine-tuning (Huang et al., 2021), semantic segmentation (RUNBERT) (Liu et al., 2020), or the tagger to detect which tokens in incomplete utterances should be kept, deleted or changed for restoration (SARG) (Huang et al., 2021). However, we argue that these methods can only work on neither extractive nor abstractive IUR datasets. For example, SARG and

*Corresponding Author.
[1]The code is available at https://github.com/shumpei19/jointiur

Context		*U1:* Orson Welles *U2:* Goodwill ambassador *U3:* What is a Goodwill ambassador? *U4:* The OCIAA sponsored cultural tours to Latin America and appointed goodwill ambassadors *U5:* What did he do? *U6:* In late November 1941, Welles was appointed as a goodwill ambassador to Latin America by Nelson Rockefeller, *U7:* Was he good at it? *U8:* Whitney wrote Welles, \"Personally believe you would make great contribution to hemisphere solidarity with this project.\" *U9:* What do you find interesting about the article? *U10:* In addition to working on his ill-fated film project, It's All True, Welles was responsible for radio programs, lectures, interviews and informal talks as part of his OCIAA-sponsored cultural mission
Incomplete Utterance		Did he have any other responsibilities?
Gold Utterance		In addition to radio programs, lectures, interviews and informal talks, did Orson Welles have any other responsibilities?
Tokens Defined by Hard Labeling		["addition", "radio", "programs", "lectures", "interviews", "informal", "talks", "Orson", "Welles", "informal", "responsible"]
Prediction	SARG	Did orson welles have any other responsibilities besides the all cultural?
	Seq2Seq-tuning	Did orson welles have any other responsibilities other than being a goodwill ambassador to latin america?
	T5	Did orson welles have any other responsibilities besides radio programs , lectures , interviews and informal talks as part of his ociaa-sponsored cultural mission?
	T5 + Picker	Did orson welles have any other responsibilities besides radio programs , lectures , interviews and informal talks?

Figure 1: The sample data from *CANARD*. IUR models rewrite the incomplete utterance to be as similar as possible to the reference. The blue tokens are omitted tokens (excluding stop words) in the incomplete utterance. The red tokens are defined by our hard labeling approach as an important token.

seq2seq achieve promising results on *Restoration 200k* (Pan et al., 2019) where omitted tokens can be directly extracted from the context (extraction). But they are not the best on CANARD (Elgohary et al., 2019), which requires more abstraction for restoration. In Figure 1,[2] we can observe that the output of SARG and seq2seq are worse than that of our T5+Picker model. SARG needs inference to deal with the change from *"responsible"* to *"responsibilities"* while seq2seq has the problem in picking omitted tokens. As the result, the generality of these methods is still an open question.

We introduce a simple but effective model to deal with the generality of IUR methods. The model is designed to work in various scenarios, from extractive to abstractive styles. To do that, we first address the problem of identifying omitted tokens from the dialogue context by introducing a picker. The picker uses a new matching method for dealing with various forms of tokens (Figure 1) in the extraction style. We next consider the abstraction aspect of restoration by offering a generator. The generator utilizes the power of the pre-trained T5 model to rewrite incomplete utterances. The picker and generator share the T5's encoder and are jointly trained in a unified model for IUR. This paper makes three main contributions:

- We design a simple but effective model based on T5 for utterance restoration in multi-turn conversations. The model jointly optimizes two tasks: picking important tokens (the picker) and generating re-written utterances (the generator). To our best knowledge, we are the first to utilize T5 for the IUR task.
- We design a method for identifying important tokens for training the picker. The method facilitates IUR models in actual cases, in which there are no (a few) existing gold labels.
- We demonstrate the validity of the model by comparing it to strong baselines from multiple perspectives such as limited data setting (Section 5.2), human evaluation (Section 5.4) and output observation (Section 5.5).

2 Related Work

Sentence rewriting IUR can be considered to be similar to the sentence rewriting task (Xu and Veeramachaneni, 2021; Lin et al., 2021; Chen and Bansal, 2018; Cao et al., 2018). Recent studies have been addressed the IUR task with various sophisticated methods. For example, Pan et al. 2019 introduced a pick-then-combine model for IUR. The model picks up omitted tokens which are combined with incomplete utterances for restoration. Liu et al. 2020 proposed a semantic segmentation method that segments tokens in an edit matrix then applied an edit operation to generate utterances. Huang et al. 2021 presented a complicated model which uses a tagger for detecting kept, deleted, or changed tokens for restoration. We share the idea

[2]The performance of RUN-BERT is limited on CANARD.

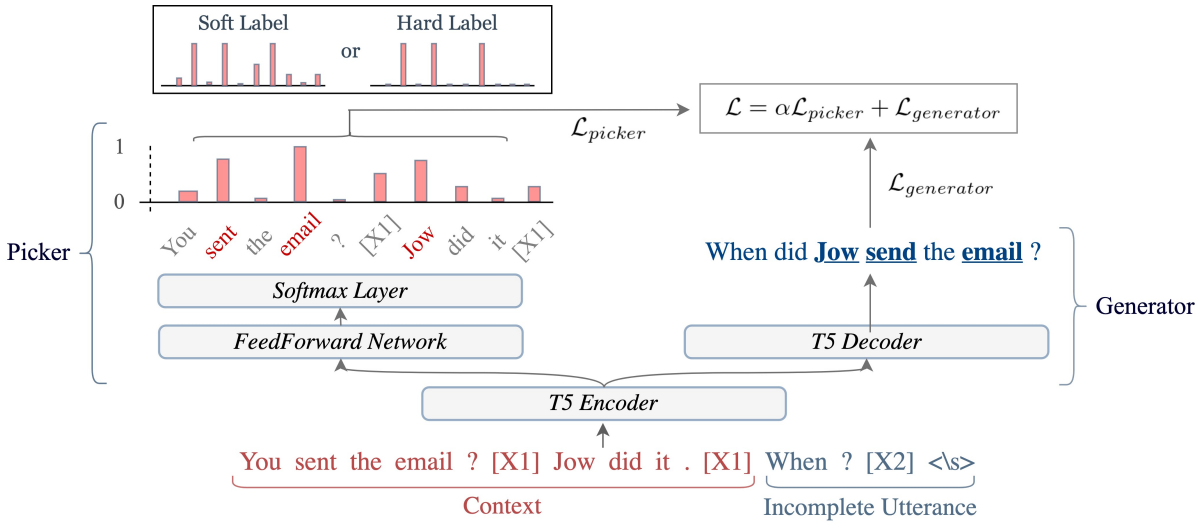

Figure 2: The proposed model for utterance restoration. The left is the picker and the right is the generator. The model jointly optimizes two tasks for doing restoration. The input format for our model is described in 3.2.1

of using a tagger with Huang et al. 2021 for IUR. However, we design a more simple but effective model which includes a picker (picking omitted tokens) and a generator for the restoration of incomplete utterances.

Text generation IUR can be formulated as text generation by using the seq2seq model (Pan et al., 2019; Huang et al., 2021). For the generation, several well-known pre-trained models have been applied (Lewis et al., 2020; Brown et al., 2020; Raffel et al., 2020) with promising results. We employ the T5 model (Raffel et al., 2020) as the main component to rewrite utterances. To address the problem of missing important tokens in model's rewriting, we enhance T5 by introducing a Picker and two labeling methods (Section 3.2).

3 The Utterance Restoration Model

3.1 Problem Statement

This work focuses on the incomplete utterance restoration of conversations. Let $H = \{h_1, h_2, ..., h_m\}$ be the history of the dialogue (context), $U = \{u_1, u_2, ..., u_n\}$ is the incomplete utterance that needs to be re-written. The task is to learn a mapping function $f(H, U|\Theta) = R$, where $R = \{r_1, r_2, ..., r_k\}$ is the re-written version of U. The learning of Θ is composed by only using utterance generation (the generator) or the combination of two tasks: important token identification (the picker) and utterance generation (the generator).

3.2 The Proposed Model

Our model is shown in Figure 2. The Picker receives the context to identify omitted tokens. The Generator receives incomplete utterances for restoration. The model jointly learns to optimize the two tasks. Our model distinguishes in three significant differences compared to PAC (Pan et al., 2019) and SARG (Huang et al., 2021). First, our model bases on a sing pre-trained model for both picker and generator while other models (i.e. PAC and SARG) use different architectures for the two steps. This makes two advantages for our model. (i) Our design can be easily adapted to create a new unified model for different tasks by using a single generative LM (Paolini et al., 2021). (ii) Our model can work well in several scenarios: extraction vs. abstraction (data characteristics) and full vs. limited training data (Section 5). Second, we design a joint training process to implicitly take into account the suggestion from the picker to the generator instead of using a two-step model as PAC which explicitly copes extracted tokens from the Pick for generation. Our joint training model can reduce the error accumulation compared to the two-step framework. Finally, we design a heuristic approach to build important tokens, which enable the model to work on a wider range of datasets and scenarios.

3.2.1 Input representation

As shown in Figure 2, we introduced three kinds of special tokens into the input text; $[X1]$, $[X2]$ and $<\backslash s>$. $[X1]$ and $[X2]$ are our newly defined special tokens and $<\backslash s>$ is the EOS token in the T5's

vocabulary. We inserted $[X1]$ at the end of each utterance in the context, $[X2]$ after the incomplete utterance and $<\backslash s>$ at the end of whole input. $[X1]$ and $[X2]$ convey two pieces of useful information to the model; the signal indicating the switch of speakers and the cue to distinguish whether the utterance is from context or incomplete utterance.

The embedding of each token in the entire input sequence $S = \{w_1, w_2, ..., w_l\}$ was obtained as $x_i = WE(w_i) + PE(w_i)$. Here, *WE* is *word embedding* initialized from a pretrained model by using a wordpiece vocabulary. *PE* is *relative position embedding* representing the position of each token in the sequence. These embeddings were fed into the L stacked Encoder of T5; $E^l = EncoderBlock(E^{l-1})$ where $E^0 = \{x_1, \ldots, x_l\}$. E^L is the contextual representation of the whole input used by Picker and Generator in next sections.

3.2.2 The Picker

It is possible to directly use T5 (Raffel et al., 2019) for IUR. However, we empower T5 with a Picker to implicitly take into account information from important tokens. The idea of selecting important tokens was derived from Pan et al. (2019), in which the authors suggested the use of important tokens contributes the performance of utterance restoration. We extend this idea by designing an end-to-end model which includes important token identification and generation, instead of using the two-step framework as Pan et al. (2019).

Given the context and the incomplete utterance, the Picker identifies tokens that are included in context utterances but omitted in the incomplete utterance. We call these tokens as important tokens. However, no important tokens are originally provided except for Restoration 200k in four datasets (please refer to Table 1). Besides, the form of important tokens could change after restoration such as from plural to singular or nouns to verbs (Figure 1). To overcome this issue, we introduce a label creation method that automatically identifies important tokens from the context for restoration.

Important token identification Since building a set of important tokens is time-consuming and important tokens are usually not defined in practical cases, we introduce a heuristic strategy to automatically construct important tokens. In the following processing, stop words in the context, incomplete utterances, and gold references are removed in advance, assuming that stop words are the out of scope of important tokens. In addition, we applied lemmatization and stemming, the process of converting tokens to their base or root form, to alleviate the spelling variants.

First, we extracted tokens, called “clue tokens", that exist in gold but not in incomplete utterances. If some tokens in context are semantically similar to some of the clue tokens, we can naturally presume that these tokens in the context are cited as important tokens for the rewriting. Therefore, we performed scoring by the distance d_{ij} between the word representations of i-th token in context h_i and j-th clue tokens c_j; $d_{ij} = \mathrm{cosine_sim}(\mathrm{h_i}, \mathrm{c_j})$ where cosine_sim() is the score of Cosine similarity. We used word representations of h_i and c_j from fastText (Bojanowski et al., 2017) trained on Wikipedia as a simple setting of our model.

According to the distance d_{ij}, we introduce two types of labels for the Picker, $soft_i$ as soft labels and $hard_i$ as hard labels.

$$soft_i = \max_j d_{ij}$$

$$hard_i = \begin{cases} 1 & \max_j d_{ij} = 1 \\ 0 & otherwise \end{cases}$$

Here, the $\max$ operation was applied based on the assumption that at most one clue token corresponds to a token in the context.

Intuitively, the soft label method takes into consideration the cases that could not be handled by lemmatization and stemming, such as paraphrasing by synonyms, and reflects them as the importance score in the range of 0 to 1. On the other hand, the hard label is either 0 or 1 where an important token is defined only when there is an exact matching between the context tokens and the clue tokens in the form after lemmatization and stemming. We provide the two methods to facilitate important token identification.

Important token selection The Picker takes encoded embeddings $E^L = \{E_1^L, ..., E_l^L\}$ and predicts the scores of the soft label or hard label corresponding to each input token.

$$p(y_i|E_i^L) = softmax(\mathrm{FNN}(E_i^L))$$

where $\mathrm{FNN}()$ is the vanilla feedforward neural network, which stands for projecting encoded embedding to the soft label or hard label space. Then

cross-entropy was adopted as the loss function.

$$\mathcal{L}_{picker} = -\sum_{i=1}^{l} q_i \log p(y_i|E_i^L)$$

where q_i is the picker's label for the i-th input token. To optimize loss function $\mathcal{L}_{picker}$ is equal to minimize the KL Divergence if the label is a soft label. In the hard labeling case, we assign three types of tags for tokens by following the BIO tag format as a sequence tagging problem.

3.2.3 The Generator

We explore the restoration task by using Text-to-Text Transfer Transformer (T5) (Raffel et al., 2019). This is because T5 provides promising results for the text generation task. We initialized transformer modules from T5-base, which uses 12 layers, and fine-tuned it for our IUR task.

For restoration, encoder's representation E^L was fed into a L stacked decoder with cross attention. $D_i^l = DecoderBlock(D_i^{l-1}, E^L)$ where $D_i^0 = R_{<i}$, with $R_{<i} = \{< s >, r_1..., r_{i-1}\}$ and $< s >$ is the SOS token. The probability p of a token t at the time step i was obtained by feeding the decoder's output D^L into the softmax layer.

$$p(t \mid R_{<i}, H, U) = softmax(linear(D_i^L)) \cdot v(t)$$

Here, $v(t)$ is a one-hot vector of a token t with the dimension of vocabulary size. The objective is to minimize the negative likelihood of conditional probability between the predicted outputs from the model and the gold sequence $R = \{r_1, r_2, ..., r_k\}$.

$$\mathcal{L}_{generator} = -\sum_{i=1}^{k} \log p(r_i \mid R_{<i}, H, U)$$

3.2.4 Joint learning

The joint model aims to optimize the Picker and the Generator jointly as a setting of Multi-Task Learning (MTL). Different from PAC (Pan et al., 2019) that directly copies extracted tokens to generation, our joint model can implicitly utilize knowledge from the Picker, in which the learned patterns of the Picker to identify important tokens can be leveraged by the Generator. It can reduce error accumulation in the two-step framework as PAC. The final loss of the proposed model is defined as follows.

$$\mathcal{L} = \alpha \mathcal{L}_{picker} + \mathcal{L}_{generator}$$

where the hyperparameter α balances the influence of the task-specific weight. Our simple setting enables us to implement minimal experiments to evaluate how much important token extraction makes the contribution to generation.

4 Settings and Evaluation Metrics

Data We conducted all experiments on four well-known datasets of utterance rewriting in Table 1.

Table 1: Four conversational datasets. **ext** is extraction and **abs** is abstraction. CN: Chinese; EN: English.

Data	train	dev	test	type	lang
Restoration 200k	194k	5k	5k	ext	CN
REWRITE	18k	0	2k	ext	CN
TASK	2.2k	0	2k	ext	EN
CANARD	32k	4k	6k	abs	EN

Restoration 200k (Pan et al., 2019) and REWRITE (Su et al., 2019) include Chinese conversations. TASK (Quan et al., 2019) and CANARD (Elgohary et al., 2019) are in English, in which CANARD includes English questions from QuAC (Choi et al., 2018). The datasets range from extraction to abstraction challenging UIR models.

Settings We used AdamW with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and a weight decay of 0.01 with a batch size of 12 in all experiments. The learning rate of $5e^{-5}$ is set for the joint model and the Generator, and $2e^{-3}$ for the Picker. We used 3 FFN layers (dimension as 768, 256, 64) with ReLu as the activation function. The final dimension is 1 for soft labeling and 3 for hard labeling. We set $\alpha = 1$ for the loss function. We applied beam search with the beam size of 8. For picker's label creation, we used stop words from NLTK for English and from stopwordsiso[3] for Chinese. For lemmatization and stemming, NLTK's WordNetLemmatizer and PorterStemmer were adopted for English, while lemmatization and stemming were skipped for Chinese. The pre-trained model was T5-base (English[4] and Chinese[5]). The epoch size of 6 was used for Restoration200k and CANARD and 20 for REWRITE and TASK in the full training data setting (Section 5.1), and the epoch size of 20 for all four datasets (Table 1) in the limited training data setting (Section 5.2). All models were trained on a single Tesla P100 GPU.

[3]https://pypi.org/project/stopwordsiso/
[4]https://huggingface.co/t5-base
[5]https://huggingface.co/lemon234071/t5-base-Chinese

Table 2: The comparison of our model and T5. **Bold numbers** show statistically significant improvements with $p \leq 0.05$. Underline is comparable (applied to Tables 3 and 4). The results come from the hard label method.

Data	Method	ROUGE-1	ROUGE-2	BLEU-1	BLEU-2	f1	f2	f3
Restoration 200k	T5-base	92.7	86.1	91.4	88.9	61.3	51.2	44.8
	T5+Picker	**93.1**	**86.9**	**92.0**	**89.6**	**63.0**	**53.3**	**47.1**
REWRITE	T5-base	95.5	90.3	92.8	90.5	89.0	82.1	77.2
	T5+Picker	**95.8**	**90.6**	**93.5**	**91.2**	**89.8**	**82.7**	**77.5**
TASK	T5-base	95.8	91.7	93.9	92.6	76.2	71.2	68.1
	T5+Picker	**96.1**	91.8	**94.3**	**93.0**	76.3	**72.1**	**69.6**
CANARD	T5-base	83.9	70.2	77.8	70.8	56.2	44.6	39.3
	T5+Picker	**84.3**	**71.1**	**78.8**	**72.0**	**57.3**	**45.9**	**40.7**

Table 3: The comparison of our model and strong baselines; For Restoration 200k, results were derived from Liu et al. (2020) and Huang et al. (2021). For CANARD, we reproduced strong baselines which output promising results on Restoration 200k. The results of RUN-BERT on CANARD were derived from the code of Liu et al. (2020). The results of our model come from hard labels.

Data	Method	ROUGE-1	ROUGE-2	BLEU-1	BLEU-2	f1	f2	f3
Restoration 200k	Syntactic	89.3	80.6	84.1	81.2	47.9	38.8	32.5
	CopyNet	89.0	80.9	84.7	81.7	50.3	41.1	34.9
	T-Ptr	90.1	83.0	90.3	87.4	51.0	40.4	33.3
	PAC	91.6	82.8	89.9	86.3	63.7	49.7	40.4
	Seq2Seq-tuning	91.4	85.0	90.8	88.3	56.8	46.4	39.8
	RUN	91.0	82.8	91.1	88.0	60.3	47.7	39.3
RUN-BERT	RUN-BERT	92.4	85.1	92.3	**89.6**	**68.6**	**56.0**	**47.7**
	SARG	92.1	86.0	92.2	**89.6**	62.4	52.5	46.3
	Our model	**93.1**	**86.9**	92.0	**89.6**	63.0	53.3	47.1
CANARD	Seq2Seq-tuning	83.1	69.0	78.6	71.2	55.1	43.2	37.9
	RUN-BERT	80.6	62.7	70.2	61.2	44.2	30.5	24.9
	SARG	80.3	63.7	70.5	62.7	42.7	30.5	25.9
	Our model	**84.3**	**71.1**	**78.8**	**72.0**	**57.3**	**45.9**	**40.7**

Evaluation metrics We followed prior work (Pan et al., 2019; Elgohary et al., 2019; Liu et al., 2020; Huang et al., 2021) to use three different metrics for evaluation, including ROUGE-scores, BLUE-scores, and f-scores.

5 Results and Discussion

5.1 Full Training Data Setting

We provide two scenarios of comparison with full training data: comparison with T5 and comparison with non-generative LM models.

Comparison with T5 We first compare our model against a strong pre-trained T5 model used for the generator as the first scenario. This scenario ensures fair comparison among strong pre-trained models for text generation and also shows the contribution of the Picker. Results in Table 2 show that the joint model is consistently better than T5 across all metrics on all four datasets. This is because the picker can pick up important omitted tokens, which are beneficial for restoration. For example, in Figure 1, the picker can identify three omitted tokens " *radio*", *"programs"*, or "*responsibilities*" from *"responsible"* for rewriting the incomplete utterance. These results prove joint learning can implicitly supports to capturing the hidden relationship between the picker and generator. Also, the promising results show that our labeling method can work in both extraction and abstraction datasets. The results of T5 are also competitive. The reason is that T5 (Raffel et al., 2019) was trained with a huge amount of data by using the generative learning process, which mimics the text generation task. As the result, it is appropriate for the restoration.

For other strong pre-trained models for text generation, we also test our joint learning framework with ProphetNet (Qi et al., 2020) but the results

are not good to report. We leave the comparison with UniLM (Dong et al., 2019) and ERNIGEN (Xiao et al., 2020) as a minor future task due to no pre-trained models for Chinese.

Comparison with non-generative LM models We next challenge our model to strong baselines which do not directly use generative pre-trained LMs, e.g. T5 for restoration. This is the second scenario that ensures the diversity of our evaluation. We leave the comparison of our model with BERT-like methods (e.g. SARG and RUN-BERT by using the T5 encoder) as a minor future task. For Restoration 200k and CANARD, we use the following baselines. **Syntactic** is the seq2seq model with attention (Kumar and Joshi, 2016). **CopyNet** is a LSTM-based seq2seq model with attention and the copy mechanism (Huang et al., 2021). **T-Ptr** employs transformer layers for encoder-decoder for restoration (Su et al., 2019). **PAC** is the two-stage model for utterance restoration (Pan et al., 2019). **Seq2seq-tuning** uses transformer blocks for fine-tuning the restoration task (Huang et al., 2021). **RUN-BERT** is an IUR model by using semantic segmentation (Liu et al., 2020). **SARG** is a semi autoregressive model for multi-turn utterance restoration (Huang et al., 2021).

Table 3 shows that our model outputs promising results compared to strong baselines. For Restoration 200k, our model is competitive with RUN-BERT, the SOTA for this dataset. For CANARD, our model is consistently better than the baselines. The improvements come from the combination of the picker and generator. It is important to note that RUN-BERT and SARG are behind our model significantly on the abstractive scenario (CANARD). It supports our statement in Section 1, in which the current strong models for IUR is overspecific for extractive datasets and their generality is limited.

We next report the comparison on REWRITE and TASK in another table due to a small number of evaluation metrics. Following Liu et al. (2020), we compare our model with RUN and two new methods: GECOR1 and GECOR2.

Results from Table 4 are consistent with the results in Tables 2 and 3. It indicates that our model outperforms the baselines on both TASK and REWRITE. For REWRITE, the EM (exact match) score of our model is much better than the baselines. It shows that the model can correctly restore incomplete utterances. These results confirm that our model can work well in the two scenarios over all four datasets.

Table 4: The comparison of our model and strong baselines on REWRITE and TASK; EM is exact match, B is BLEU, and R stands for ROUGE.

Data	Method	EM	B4	R2	f1
REWRITE	RUN	53.8	79.4	85.1	NA
	T-ptr+BERT	57.5	79.9	86.9	NA
	RUN-BERT	66.4	86.2	90.4	NA
	Ours	**69.1**	**86.6**	90.6	**89.8**
TASK	GECOR1	68.5	83.9	NA	66.1
	GECOR2	66.2	83.0	NA	66.2
	RUN	69.2	85.6	NA	70.6
	Ours	**79.6**	**90.9**	**91.8**	**76.3**

Important token ratio We observed how many important tokens are included in prediction on Restoration 200k. To do that, we defined two metrics, *pickup ratio* and *difference*. *pickup ratio* indicates the ratio of predictions that contains important tokens on test datasets. *difference* indicates the difference the character length between the prediction and the gold. Ideally, larger *pickup ratio* with smaller *difference* is desirable.

Table 5: The pickup ratio and difference of T5 and T5+Picker on Restoration 200k.

	pickup ratio (%)	difference
T5	29.0	1.28
Defined	29.9	1.30
Soft	26.6	1.28
Hard	30.4	1.21

Table 5 shows T5+Picker with hard labeling achieves better results on both metrics compared to single T5. This supports our hypothesis that the Picker contributes the Generator for the IUR task.

5.2 Limited Training Data Setting

We challenge our model in the limited training data setting. This simulates actual cases in which only a small number of training samples is available. We trained three strong methods: **SARG** (Huang et al., 2021), T5, and T5+Picker (Ours) on 10% of training data by using sampling. We could not run RUN-BERT due to errors in the original code.

As shown in Table 6, our model is consistently better than SARG with large margins. This is because our model is empowered by T5 which helps our model to work with a small number of training samples. This point is essential in actual cases. Our

Table 6: The comparison with limited training data.

Data	Method	R2	B2	f1	f2
Restoration 200k	SARG	82.8	87.4	52.4	**40.1**
	T5	84.5	87.2	53.8	37.2
	Ours	**84.6**	**88.0**	**56.5**	39.2
REWRITE	SARG	57.9	50.0	0.00	0.00
	T5	77.3	73.7	71.0	61.3
	Ours	**77.7**	**74.9**	**72.1**	**62.4**
TASK	SARG	76.4	44.5	12.0	0.14
	T5	86.0	85.8	52.7	48.3
	Ours	**87.0**	**86.9**	**55.7**	**51.2**
CANARD	SARG	58.6	46.4	41.8	25.9
	T5	68.5	70.0	54.1	42.2
	Ours	**69.2**	**71.3**	**55.9**	**43.6**

model is also better than T5, showing the contribution of the Picker. SARG is good at ROUGE-scores and BLUE-scores but worse at f-scores, e.g. on REWRITE. The reason is that SARG uses the pointer generator network, that directly copies input sequences for generation, but it learns nothing.

5.3 Soft Labels vs. Hard Labels

We investigated the efficiency of our labeling method in Section 3.2.2. We run the joint model (T5+Picker) with soft and hard labeling methods. We also include the results of the joint model on defined labels of Restoration 200k because this dataset already provides labels of important tokens.

Table 7: Soft vs. hard labeling methods. *Defined* is the ground truths. T5 does not use the labeling methods.

Data	Method	R2	B2	f1	f2
Restoration 200k	T5	86.1	88.9	61.3	51.2
	Defined	85.7	89.3	61.5	51.9
	Soft	86.2	87.9	57.4	47.7
	Hard	**86.9**	**89.6**	**63.0**	**53.3**
CANARD	T5	70.4	71.1	56.7	44.8
	Soft	70.8	71.4	57.1	45.5
	Hard	**71.1**	**71.8**	**57.6**	**45.6**

From Table 7 we can see the hard labeling method performs well on both datasets. Interestingly, the hard labeling method is even better than the one with defined labels on Restoration 200k. Although defined labels were manually created and of high quality, Restoration 200k defines at most one important token (some samples contain two or more omitted tokens) in one sample. We found the hard label method detects 164k omitted tokens while the originally defined tokens are about 120k, and tokens detected by hard labeling cover 42% of defined tokens. This suggests the hard label method extensively picks up important tokens even some important tokens are missing, and it can contribute to the enhancement of the joint model.

For the soft labeling method, it contributes to the f-scores on CANARD (=abstractive) while it exacerbates accuracy on Restoration 200k (=extractive). This implies soft label does not function well in the distinction case between important and unimportant tokens is clear as in Restoration 200k. The soft labeling method would need more exploration on abtractive scenarios that require more synonymous paraphrasing or creative summarization.

5.4 Human Evaluation

We report human evaluation with strong methods on CANRD because it is much more challenging than others. We asked three annotators who are well skilled in English and data annotation from the annotation team in our company. For the evaluation, we randomly selected 300 outputs from four models. Each annotator read each output and gave a score (1: bad; 2: acceptable; 3: good). Following Huang et al. (2021) we adopted **Text flow** and **Understandability** as our criteria. **Text flow** shows how the restoration utterance is correct grammatically and easy to understand. **Understandability** shows how much the predictions are similar to reference semantically. The agreement by Cohen's Kappa calculated from the average scores of sentence flow and comprehensibility is 0.794, which indicates that the annotation process is good.

Table 8: Human evalution on CANARD.

	SARG	seq2seq	T5	T5+Picker
Text flow	2.583	2.887	2.925	**2.933**
Understand	2.168	2.451	2.458	**2.496**

As shown in Table 8, the joint model obtains the highest scores on two criteria over other methods. It is consistent with the results of automatic evaluation in Tables 2 and 3. This is because our proposed model utilizes strong pre-trained weights which provide the ability of text generation on unseen tokens, especially for abstractive data. The scores of T5+Picker also show the contribution of the Picker compared to the T5 for restoration.

5.5 Output Observation

We observed the restoration outputs of different models in Figure 1. There exist 10 omitted tokens between the incomplete utterance and the reference. The SARG and seq2seq-tuning can restore 3 to 4 important tokens. T5 can restore all the important tokens but generates unnecessary words. Our proposed model can restore all the important tokens and have the same semantic meaning as the gold utterance. This is because our model is designed to pick up important tokens for restoration.

Table 9: The average BLEU score on CANARD.

Length	< 100	$100 \leq 200$	$200 <$
SARG	55.53	45.46	38.96
seq2seq	63.89	54.94	48.39
T5	**65.03**	55.69	51.25
T5+Picker	64.94	**56.56**	**52.84**

We also examined the ability of strong methods with different input lengths on CANARD. Results in Table 9 show that our model can deal with longer input sequences. Compared to SARG and seq2seq, the performance of our model is much better. This is because the implicit suggestion from the Picker combined with the ability to deal with long sequences of T5 increase the score.

6 Conclusion

This paper introduces a simple but effective model for incomplete utterance restoration. The model is designed based on the nature of conversational utterances, where important omitted tokens should be included in restored utterances. To do that, we introduce a picker with two labeling methods for supporting a generator for restoration. We found that the picker contributes to improve the generality of the model on four benchmark datasets. The model works well in English and Chinese, from extraction to abstraction, and in different training data settings, showing that it is flexible to be applied to different languages and is applicable for actual cases with limited training samples.

The future work will investigate the behavior of the model in other domains. The model also can be an important part of conversational AI systems, e.g. restore utterances for fluent conversations.

Acknowledgement

We would like to thank Yun-Nung Chen and anonymous ACL ARR reviewers who gave constructive comments for our paper.

References

Daniel Adiwardana, Minh-Thang Luong, Jamie Hall David R. So, Romal Thoppilan Noah Fiedel, and Zi Yang et al. 2020. Towards a human-like open-domain chatbot. In *arXiv preprint arXiv:2001.09977*.

JinYeong Bak and Alice Oh. 2018. Conversational decision-making model for predicting the king's decision in the annals of the joseon dynasty. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 956-961*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. *ArXiv*, abs/2005.14165.

Ziqiang Cao, Wenjie Li, Sujian Li, and Furu Wei. 2018. Retrieve, rerank and rewrite: Soft template based neural summarization. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 152–161, Melbourne, Australia. Association for Computational Linguistics.

Yen-Chun Chen and Mohit Bansal. 2018. Fast abstractive summarization with reinforce-selected sentence rewriting. *ArXiv*, abs/1805.11080.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2174-2184*.

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, pp. 13063-13075*.

Amed Elgohary, Denis Peskov, and Jordan Boyd-Graber. 2019. Can you unpack that? learning to rewrite questions-in-context. 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), pp. 5918-5924.

Mengzuo Huang, Feng Li, Wuhe Zou, and Weidong Zhang. 2021. Sarg: A novel semi autoregressive generator for multi-turn incomplete utterance restoration. In *Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 14, pp. 13055-13063*.

Mladen Karan, Prashant Khare, Patrick Healey, and Matthew Purver. 2021. Mitigating topic bias when detecting decisions in dialogue. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, pp. 542-547*.

Vineet Kumar and Sachindra Joshi. 2016. Non-sentential question resolution using sequence to sequence learning. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, pp. 2022-2031*.

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. *ArXiv*, abs/1910.13461.

Zhe Lin, Yitao Cai, and Xiaojun Wan. 2021. Towards document-level paraphrase generation with sentence rewriting and reordering. *ArXiv*, abs/2109.07095.

Qian Liu, Bei Chen, Jian-Guang Lou, Bin Zhou, and Dongmei Zhang. 2020. Incomplete utterance rewriting as semantic segmentation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pp. 2846–2857*.

Zhufeng Pan, Kun Bai, Yan Wang, Lianqiang Zhou, and Xiaojiang Liu. 2019. Improving open-domain dialogue systems via multi-turn incomplete utterance restoration. 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), pp. 1824-1833*.

Giovanni Paolini, Ben Athiwaratkun, Jason Krone, Jie Ma, Alessandro Achille, Rishita Anubhai, Cicero Nogueira dos Santos, Bing Xiang, and Stefano Soatto. 2021. Structured prediction as translation between augmented natural languages. In *arXiv preprint arXiv:2101.05779*.

Weizhen Qi, Yu Yan, Yeyun Gong, Dayiheng Liu, Nan Duan, Jiusheng Chen, Ruofei Zhang, and Ming Zhou. 2020. Prophetnet: Predicting future n-gram for sequence-to-sequence pre-training. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, pp. 2401-2410*.

Jun Quan, Deyi Xiong, Bonnie Webber, and Changjian Hu. 2019. Gecor: An end-to-end generative ellipsis and co-reference resolution model for task- oriented dialogue. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Confer- ence on Natural Language Processing (EMNLP- IJCNLP), pp. 4547–4557. Association for Computational Linguistics*.

C Raffel, N Shazeer, A Roberts, K Lee, S Narang, and others. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv*.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Shuangyong Song, Chao Wang, Qianqian Xie, Xinxing Zu, Huan Chen, and Haiqing Chen. 2020. A two-stage conversational query rewriting model with multi-task learning. In *Companion Proceedings of the Web Conference 2020, pp. 6-7*.

Hui Su, Xiaoyu Shen, Zhou Xia, Zheng Zhang, Ernie Chang, Cheng Zhang, Cheng Niu, and Jie Zhou. 2020. Moviechats: Chat like humans in a closed domain. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 6605-6619*.

Hui Su, Xiaoyu Shen, Rongzhi Zhan, Fei Sun, Pengwei Hu, Cheng Niu, and Jie Zhou. 2019. Improving multi-turn dialogue modelling with utterance rewriter. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pp. 22-31*.

Sansiri Tarnpradab, Fei Liu, and Kien A Hua. 2018. Toward extractive summarization of online forum discussions via hierarchical attention networks.

Dongling Xiao, Han Zhang, Yukun Li, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2020. Ernie-gen: An enhanced multi-flow pre-training and fine-tuning framework for natural language generation. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, pp. 3997-4003*.

Lei Xu and K. Veeramachaneni. 2021. Attacking text classifiers via sentence rewriting sampler. *ArXiv*, abs/2104.08453.