Long-term Control for Dialogue Generation: Methods and Evaluation

Ramya Ramakrishnan
ASAPP
rramakrishnan@asapp.com

Hashan Buddhika Narangodage
ASAPP
hnarangodage@asapp.com

Mauro Schilman
ASAPP
mschilman@asapp.com

Kilian Q. Weinberger
ASAPP, Cornell
kweinberger@asapp.com

Ryan McDonald
ASAPP
rmcdonald@asapp.com

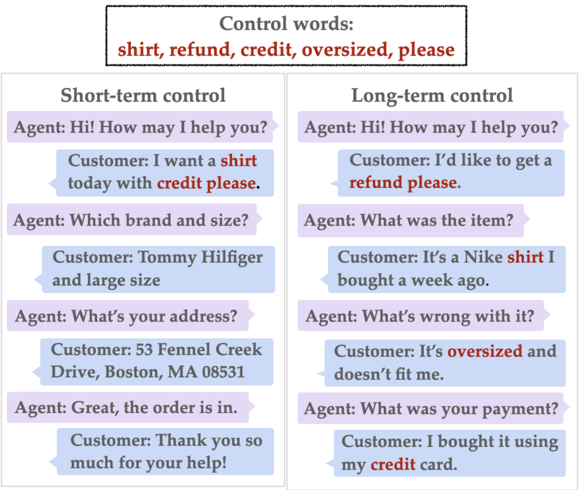
Abstract

Current approaches for controlling dialogue response generation are primarily focused on high-level attributes like style, sentiment, or topic. In this work, we focus on *constrained long-term* dialogue generation, which involves more fine-grained control and requires a given set of control words to appear in generated responses. This setting requires a model to not only consider the generation of these control words in the immediate context, but also produce utterances that will encourage the generation of the words at some time in the (possibly distant) future. We define the problem of constrained long-term control for dialogue generation, identify gaps in current methods for evaluation, and propose new metrics that better measure long-term control. We also propose a retrieval-augmented method that improves performance of long-term controlled generation via logit modification techniques. We show through experiments on three task-oriented dialogue datasets that our metrics better assess dialogue control relative to current alternatives and that our method outperforms state-of-the-art constrained generation baselines. [1]

1 Introduction

Despite recent advances in dialogue systems (Serban et al., 2016; Ham et al., 2020), *controlling* dialogue generation remains a significant challenge. Response generation in dialogue can be controlled towards different topics and styles (Madotto et al., 2020) or towards a set of hard constraints (i.e., lexical control words need to appear in the generated text) (Sha, 2020). We focus on the hard constraint setting, also known as *constrained* generation, as this provides a more fine-grained method of controlling dialogues.

For example, consider a customer service use case (Figure 1), in which an agent speaks to a customer about an issue. The goal is to generate a given set of control words in the responses of one of the speakers (agent or customer). Naive constrained generation approaches (Pascual et al., 2020; Miao et al., 2019) use methods like beam search and stochastic search to force the generation of these control words for short-term control, where control words need to appear in a single utterance or phrase. Because they do not consider the future, these approaches may generate the words all at once in a single response or not generate them at natural places in the conversation (Figure 1, left).



Agent: Hi! How may I help you?
Customer: I want a shirt today with credit please.
Agent: Which brand and size?
Customer: Tommy Hilfiger and large size
Agent: What's your address?
Customer: 53 Fennel Creek Drive, Boston, MA 08531
Agent: Great, the order is in.
Customer: Thank you so much for your help!

Agent: Hi! How may I help you?
Customer: I'd like to get a refund please.
Agent: What was the item?
Customer: It's a Nike shirt I bought a week ago.
Agent: What's wrong with it?
Customer: It's oversized and doesn't fit me.
Agent: What was your payment?
Customer: I bought it using my credit card.

Figure 1: Examples of short vs. long-term control for dialogue generation. **(Left)** In short-term control, many control words are generated initially, but the conversation is led away from the desired future. **(Right)** In long-term control, responses are generated with the future in mind with words generated at natural points in the conversation.

The above example highlights the challenges of applying existing constrained generation methods to long-term dialogue generation. First, since another speaker is involved in the dialogue, the model does not have full control of the generated text. Instead, the model can only control the dialogue indirectly. Second, dialogues can be long and

[1] Our code is available at https://github.com/asappresearch/constrained-dialogue-generation

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 738 - 753
July 10-15, 2022 ©2022 Association for Computational Linguistics

thus, controlling utterances several time steps into the future is non-trivial. In this work, we propose the problem of long-term dialogue control, where the goal is to generate a set of control words over many utterances in a dialogue, which requires appropriately timing the generation of control words (Figure 1, right). To the best of our knowledge, we are the first work to constrain *long-term dialogue generation* through lexical control words.

We begin by highlighting challenges with evaluation for this problem. Successful long-term control of dialogue can be difficult to measure. We describe current evaluation metrics for constrained text generation and show that these metrics can be gamed by generating all or many control words early in the conversation. To resolve this and measure how natural the control is, we propose a new set of metrics: long-term success rate, which measures the percentage of control words in simulated roll-outs of the conversation, and precision, recall, and F1-score, which compare control words in generated responses to those in reference responses from a historical dataset. The second set of metrics specifically help to capture whether the control words are generated at the right time.

Next, we propose a novel method to explicitly address long-term control. Prior methods are unable to handle this task as the number of possible future sequences is exponential. To alleviate this issue, we retrieve similar conversations from training and condition on them during generation. We first identify similar neighbors using a kNN-based approach and then guide the language model towards generating similar responses, inspired by plug-and-play methods (Madotto et al., 2021; Dathathri et al., 2019; Pascual et al., 2020). The motivation for this is that retrieved conversations guide the model to generate the control words at more natural points in the conversation.

We conduct experiments on multiple task-oriented dialogue datasets and show that our method outperforms several constrained text generation baselines on automated evaluation metrics as well as human evaluation. Specifically, we are able to generate 30-40% more control words on long-term success rate compared with baselines, while preserving fluency (scores of ≥ 4.3 out of 5), as measured by human evaluation.

2 Related work

Controllable text generation. Prior work has developed many methods for controllable text generation. These approaches can be categorized into three general areas. The first is altering decoding strategies (Grover et al., 2019; Deng et al., 2020), in which the sampling distribution can be modified (Ghazvininejad et al., 2017; Baheti et al., 2018) or hidden states in the models can be changed (Gu et al., 2017). The second area involves including prompts to guide text generation (Ribeiro et al., 2018; Jiang et al., 2020; Li and Liang, 2021), for example through universal trigger tokens (Wallace et al., 2019; Shin et al., 2020). Finally, fine-tuning can be used to guide language model outputs through the use of a latent variable (Fan et al., 2018; Peng et al., 2018) or through CTRL codes (Keskar et al., 2019). Our work differs from the broad area of controllable language generation in that 1) we require more fine-grained generation through lexical control words and 2) we focus on dialogue settings where another speaker can also change the course of the conversation.

Constrained text generation. The key difference between constrained text generation and controllable text generation is the focus on hard rather than soft constraints. Typically, there are two general methods for constrained generation: beam search (Hokamp and Liu, 2017; Post and Vilar, 2018; Pascual et al., 2020) and stochastic search (Miao et al., 2019; Sha, 2020). Directed Beam Search (DBS) (Pascual et al., 2020), modifies language model logits to encourage generation of a specified set of “guide words", or control words. A method based on stochastic search (Miao et al., 2019) uses Metropolis-Hastings with the constraint of keyword inclusion. These approaches do not apply to the dialogue setting where these constraints need to hold for many utterances into the future.

Dialogue response generation. While many works develop methods for unconstrained response generation (Budzianowski and Vulić, 2019; Peng et al., 2020; Cao et al., 2020; Hosseini-Asl et al., 2020; Yavuz et al., 2019), there is a subset of work more related to our problem focused on *controlling* response generation. In one work, transformer models are fine-tuned for dialogue through modifications of the inputs, for example by adding information about the user’s persona (Wolf et al., 2019). The work of Lippe et al. (2020) generates utterances by paraphrasing templated responses. Several works control generation through exemplar-guided methods (Cai et al., 2020; Gupta et al., 2020), which is a different setting from ours since we want to guide generation based on a set of control words rather than through a prototype. One work (Xu et al., 2019) controls response generation through meta-words that include desired attributes of the response (e.g., response length and specificity). Another work controls response generation through control words by adding inductive biases into training to guide generation (Wu et al., 2020). However, this work only controls generation for a single response, rather than controlling several utterances into the future. The closest work to ours is work by (Tang et al., 2019), which proposes a similar problem of long-term control towards a target subject. While the setup is similar, we learn to constrain dialogue responses given a set of control words rather than a target attribute, which also results in a different approach.

Retrieval-augmented generation. Another related area is retrieval-augmented language generation, which inspires our approach of using retrieval to control dialogue generation. REALM (Guu et al., 2020) uses a latent knowledge retriever to identify relevant documents and backpropagates through this retrieval step. In another work (Fan et al., 2020), relevant information is retrieved from an external knowledge base to guide dialogue generation. Several works by Khandelwal et al leverage nearest neighbor approaches to improve performance with no additional training (Khandelwal et al., 2019, 2020). While these works condition on retrieval for uncontrolled generation, we leverage ideas from this space specifically for control in dialogue.

3 Problem definition

We first define the problem of long-term constrained dialogue generation. A conversation $\mathcal{X} = \{s_1, u_1, s_2, u_2, ..., s_T, u_T\}$ is defined as a list of utterances generated by two speakers: the system s that we are trying to control and the user u, which we don't have explicit control over. T denotes the total number of turns in the conversation. Given the current dialogue context of a conversation $x = \{s_1, u_1, ..., s_t, u_t\}$ up until timestep t and a set of control words $\mathcal{W} = \{w_1, w_2, ..., w_M\}$, our goal is to generate the remaining responses of the conversation $\mathcal{R}_{t+1:T} = \{s_{t+1}, ..., s_T\}$ such that the control words $\mathcal{W}$ appear in the future generated responses. We consider a scenario in which someone provides a set of control words to be included in the conversation without assumptions on their order. This means methods need to handle control words given in any order.

We additionally assume access to a historical dataset of conversations $\mathcal{D} = \{x^{(i)}\}$, $i \in [1, ..., N]$ and a fine-tuned language model M on this dataset. We can leverage these inputs in order to control future responses $\mathcal{R}_{t+1:T}$. We focus on the plug-and-play setting (Pascual et al., 2020), in which approaches simply guide the given language model M towards generating the control words without any additional re-training.

4 Proposed metrics for evaluation

Directly evaluating the generated responses in terms of prior evaluation methods can lead to misleading results. Previous works on constrained text generation (Pascual et al., 2020) have used metrics like perplexity to measure fluency and success rate to measure the percentage of control words generated. However, these metrics are more relevant for short-term generation, as they can be gamed in settings where the control words would be naturally distributed across the full conversation. As shown in the left-hand side of Figure 1, when several words are forced into the first response, the conversation may move away from the desired future and control word generation could be inappropriately timed. To better evaluate how well the model generates the right words at the right time, we propose the following new metrics.

The first metric we propose is long-term success rate, which involves simulating conversations with a user language model and computing the percentage of generated control words in the system responses of these simulated roll-outs. Prior work (Ghandeharioun et al., 2019) has used self-play for evaluation, but they do not propose roll-outs as a way to measure dialogue control.

Long-term success rate: Our modified success rate metric is computed as the fraction of control words generated in a full simulated roll-out of the conversation. We compute this as: $s = \frac{n_w}{|\mathcal{W}|}$, where n_w is the number of control words that appear in all of the future system responses $\mathcal{R}_{t+1:T}$.

One limitation of long-term success rate is that it doesn't measure the timing of control words in the conversation. So next, we want to evaluate whether the methods generate control words at appropriate

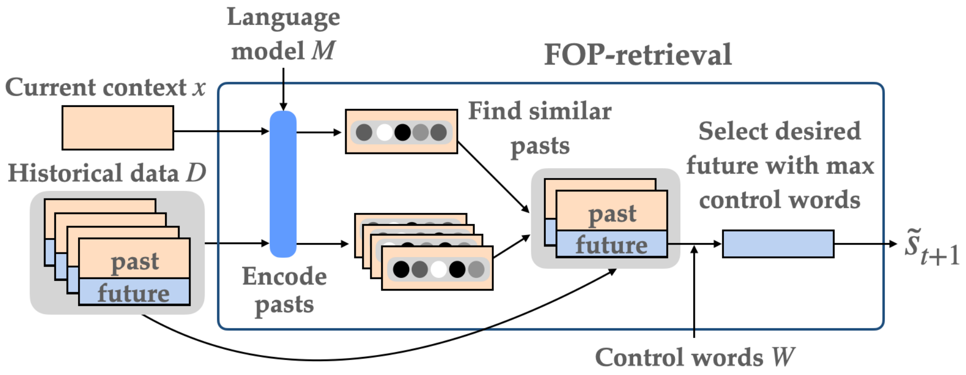


Figure 2: Visualization of FOP-retrieval. First, each conversation in the historical dataset $\mathcal{D}$ is split into many past-future conversation pairs. The current context x and the pasts are encoded using language model M. We use kNN search to identify pasts similar to context x and then select a desired future with the highest number of control words. The output is the first response in the selected future $\tilde{s}_{t+1}$.

points in the conversation. To measure this, we propose computing precision, recall, and F1-score for control words. This particular evaluation is not done in simulation. Instead, we consider each true system response in the evaluation dataset in isolation and generate a response for each, given the conversation history up until that point. We compute the number of generated control words that are correctly predicted, when compared with the control words in the ground truth response *in the same time step*.

For example, on the right side of Figure 1, when generating the second customer response (given the true conversation history up until then), we would count a “correct" prediction for P/R/F1 as a response that includes the word “shirt" (in any position in the response), as it is a control word that appears in the ground truth response in that time step. It is true that control words can also appear later in the conversation, but this setting is already evaluated by long-term success rate in simulated rollouts. After counting the number of correctly predicted control words for each response individually, we aggregate across all responses.

Precision: Precision is calculated at the corpus-level as the number of correctly predicted control words over the total number of predicted control words ($p = \frac{|\text{correct}|}{|\text{predicted}|}$).

Recall: Recall is similarly computed at the corpus-level as the number of correctly predicted control words over the total number of actual control words ($r = \frac{|\text{correct}|}{|\text{actual}|}$).

F1-score: Finally, F1-score combines precision and recall into one metric ($f1 = \frac{(2*p*r)}{(p+r)}$).

These metrics penalize models that condense all control words into one response. Instead, we want the models to naturally generate control words when they are relevant. These metrics evaluate whether control words are generated at the appropriate position in a conversation. To introduce some flexibility, an extension could be to compute a soft version of precision, recall, and F1-score that scores utterances based on whether control words appear within N utterances of the ground truth position.

Finally, we use human evaluation to evaluate how realistic and relevant the generated responses are. Specifically, we evaluate each conversation on fluency, consistency of control word generation, relevance, coherence, and diversity.

5 Retrieval-based Control

We now present our proposed approach for constrained dialogue generation. Inspired by work in retrieval-augmented generation (Guu et al., 2020; Fan et al., 2020), we retrieve similar pasts based on the current context x and use their futures to control dialogue response generation. The key insight here is that by looking at how people have used these control words in similar conversations in the past, we can bias the models towards more natural dialogues. In other words, we use futures of the past conversations to guide the current response generation. To better motivate the use of retrieval in our problem, consider the example conversation in Figure 1. The agent asks which item the customer wants to return, and there are many possible answers (e.g., “I want my pant refunded.", “I want to return gloves I bought yesterday."). Keyword-

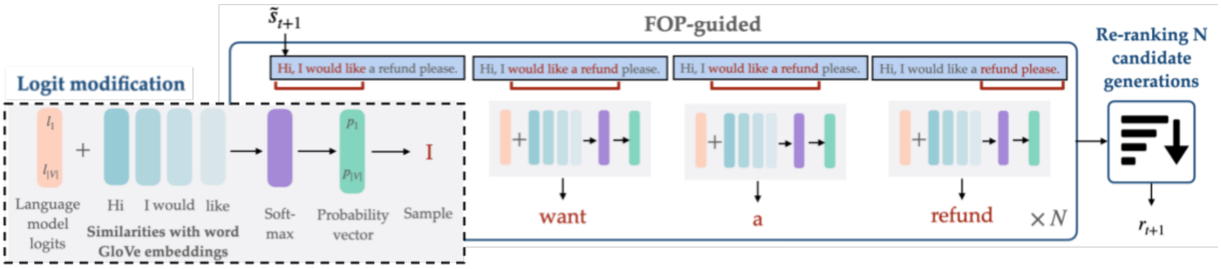

$\tilde{s}_{t+1}$
FOP-guided
Re-ranking N candidate generations
Logit modification
Hi, I would like a refund please.
Hi, I would like a refund please.
Hi, I would like a refund please.
Hi, I would like a refund please.
l_1
$l_{|V|}$
p_1
$p_{|V|}$
I
Language model logits
Hi
I would
like
Similarities with word GloVe embeddings
Soft-max
Probability vector
Sample
want
a
refund
× N
r_{t+1}

Figure 3: Visualization of FOP-guided. Language model logits are first modified using a window-based approach. All words (and similar words based on GloVe vector similarity) within the window are upweighted with a weight decay. Once any word in the window is generated, the window shifts until the full response is generated. After N generations, a re-ranking step selects the response with the highest number of control words and lowest loss.

based retrieval will surface a response about shirts, a control word, which encourages the model to generate a natural response with that word: “It’s a Nike shirt I bought a week ago."

We present two variants of our retrieval-inspired Futures of the Past (FOP) approach: 1) FOP-retrieval: we retrieve the desired future from historical data and simply use the retrieved utterance as the generated response and 2) FOP-guided: we use the utterance from FOP-retrieval as a reference sentence to guide the model towards similar responses.

The simple variant of our approach, FOP-retrieval, is shown in Figure 2. It focuses on identifying what the model should say now that will lead to the control words in the future. The reason we need to determine what to say now is that control words in our problem are distributed across a long dialogue conversation. One possible approach to generate the current response is to run many rollouts of the conversation and select the response that leads to the highest number of control words. However, this brute force approach is computationally expensive and will not be effective for rich, diverse conversations. Instead, we leverage historical conversation data to identify the most relevant futures given the current context and control words. The retrieved futures can guide the model towards what to say now that will lead to the desired future. The guided variant, shown in Figure 3, involves guiding the language model towards generating a response similar to the retrieved utterance.

Our proposed approaches address some of the challenges of long-term control for dialogue generation. First, another speaker can change the course of the conversation, which is why we retrieve a new set of similar past contexts at each time step to re-align with the current context. Second, to control responses many steps into the future, we retrieve historical conversations with the desired future (high percentage of control words) and gently nudge the conversation in that direction, thus controlling not only the current utterance but also the future of the conversation.

5.1 Retrieval Futures of the Past (FOP-retrieval)

For the retrieval component, the goal is to select futures that have relevant past contexts as well as desired futures based on the control words. To do this, we employ a multi-step approach. First, we split each conversation $x^{(i)}$ in the historical dataset $\mathcal{D}$ into a set of past-future conversation pairs $x^{(i)} = \{(p, f)^{(i,j)}\}$. We encode the current context $M(x)$ and each past conversation $M(p^{(i,j)})$ using the language model M. Then, we use kNN search based on FAISS, a library for fast nearest neighbor retrieval (Johnson et al., 2019), to identify k similar pasts from the historical data that closely match the current context x. We then filter the futures of these past conversations based on which have the highest percentage of control words.

$$\begin{aligned}\mathtt{KNN_x} &= \mathtt{faiss}(M(x), M(p^{(i,j)}), k)\\ f^* &= \mathtt{argmax}([\mathtt{count}(f^{(i,j)}, \mathcal{W})]_{f^{(i,j)} \in \mathtt{KNN_x}})\\ \tilde{s}_{t+1} &= f^*[0], f^* = \{s_1, u_1, ..., s_T, u_T\}\end{aligned}$$

In the above equations, the `count` function counts the number of control words $\mathcal{W}$ in the future $f^{(i,j)}$. The reference response $\tilde{s}_{t+1}$ is simply the first utterance of the retrieved future.

5.2 Guided Futures of the Past (FOP-guided)

Now that we have a candidate reference response $\tilde{s}_{t+1}$, we can guide the language model towards generating a similar response. To do this, we modify the logits from the language model to encourage generation of the control words or similar words. We start with the first word w_0 in $\tilde{s}_{t+1}$ and upweight logits in a way similar to DBS (Pascual et al., 2020) using similarity of GloVe vector embeddings:

$$l'_i = l_i + \lambda \cdot \texttt{min}(0, \texttt{cos}(\gamma(t_i), \gamma(w_j)))^2,$$

where γ represents GloVe embeddings, t_i is the ith token of the language model's vocabulary V, w_j is the current reference word, and λ is a hyperparameter specifying how much weight to put on generating words similar to w_j.

With this approach, we observed that sometimes the model got stuck on the first word and never moved on to later words. To enable more flexible control, instead of requiring every word to be generated before moving on to the next word, we include a window of size q and increase the logits of each word in the window, with a decay multiplier of $\frac{1}{2^i}$, $i \in q$. If any of the words in the window have been generated, the window is shifted beginning from the generated word with the same window size of q. The process repeats until the full response has been generated.

The decay multiplier is used to encourage the model to generate earlier words in the reference response and not skip words unless it's highly likely. We generate N such responses using this method and include an additional ranking step to select the best one. We first sort by the number of control words in the generated response. If multiple responses generate the highest number of control words, we sort by the loss from the model and select the response with the lowest loss l:

$$\begin{aligned}
\tilde{\mathcal{R}}_{t+1} &= \{M^j(l')|j \in [1, ..., N]\} \\
s^* &= \texttt{max}([\texttt{count}(r, \mathcal{W})]_{r \in \tilde{\mathcal{R}}_{t+1}}) \\
\hat{\mathcal{R}}_{t+1} &= \{r|\texttt{count}(r, \mathcal{W}) = s^*, r \in \tilde{\mathcal{R}}_{t+1}\} \\
r_{t+1} &= \texttt{argmin}([\texttt{loss}(r)]_{r \in \hat{\mathcal{R}}_{t+1}}),
\end{aligned}$$

where $\tilde{\mathcal{R}}_{t+1}$ is the set of N generated responses, using a model with logits l'. The final generated response r_{t+1} is selected based on the two-step ranking process. None of the other approaches include this ranking component.

6 Experimental setup

6.1 Task-Oriented Dialogue Datasets

Our problem and approach are applicable to any general dialogue control setting. In our experiments, we controlled the customer in task-oriented dialogue. This is useful for constructing a customer bot that imitates real-life customers. By controlling the customer simulator (for example through control words), we can develop a training environment for coaching customer service agents in a variety of diverse situations. For all datasets, we select control words from the utterances of the customer by selecting the top M ranked words based on tf-idf. For some real-world applications, control words can also be manually selected by a designer.

MultiWoz 2.3: The first dataset we evaluate on is MultiWoz 2.3 (Han et al., 2020), which is widely used in the dialogue community. The dataset has over 10K dialogues and 5 domains.

TaskMaster-3: The second is another commonly used task-oriented dialogue dataset TaskMaster-3 (Byrne et al., 2019). This dataset has 23,757 dialogues in the movie ticketing domain.

Action-Based Conversations Dataset (ABCD): The final dataset (Chen et al., 2021) includes a set of agent-customer conversations focused on solving customer problems. The dataset contains over 10k dialogues and is also focused on one domain.

6.2 Baselines

$\mathcal{W}_{\textbf{first}}$: The first baseline is a naive approach that outputs all control words in the first response of the conversation and nothing afterwards, which means words are not appropriately timed.

Fine-tuned: This approach simply generates responses using the fine-tuned language model M.

Prompt: This method is based on prompting approaches (Li and Liang, 2021; Ribeiro et al., 2018; Jiang et al., 2020; Madotto et al., 2021). Because we focus on the plug-and-play setting, we simply append control words to the beginning of the context and generate using this modified input.

Directed Beam Search (DBS): This is a constrained text generation approach (Pascual et al., 2020), in which keywords are generated using logit modification and beam search. It is not optimized for long-term control and is highly dependent on the ordering of control words.

Constrained Sentence Generation by Metropolis-Hastings Sampling (CGMH): This method (Miao et al., 2019) is based on stochastic search methods that insert, delete, and replace words in a sentence with the requirement

Methods	LT-SR	$f1$-score	Human eval	Overall average
Prompt	0.23	0.34	**0.87**	0.48
DBS	0.42	0.28	0.72	0.47
CGMH	**0.90**	0.17	0.3	0.46
FOP-retrieval	0.82	0.39	0.82	**0.68**
FOP-guided	0.74	**0.41**	0.81	0.67

Table 1: Summary table of results, including long-term success rate (LT-SR) from Figure 4 averaged over datasets for 9 control words, F1-score from the overall F1 column of Table 2 that averages F1 over datasets, and human eval from Table 3 averaged over all metrics and divided by 5 to get a number between 0 and 1.

that control words need to be present. It is neither optimized for long-term generation of control words nor forward generation and is particularly susceptible to aggressively generating all control words in a single response. It was also originally applied to the task of keyword-to-phrase generation so we adapted it to dialogue generation by prompting the language model with the dialogue context and also replaced a bidirectional RNN model with our transformer-based model.

7 Results

7.1 Aggregated Results

We begin by presenting a top-level overview of our main baselines and methods because each evaluation metric captures a different aspect of performance. Table 1 includes averaged scores across tasks, parameters, and/or metrics for the main results in Tables 2 and 3 and Figure 4. These include results of our two proposed automatic metrics of long-term success rate and control word F1-score (Section 4) as well as human-evaluated quality metrics (Section 7.4). In subsequent sections, we will examine each of these results more closely.

The key insight in these aggregated results is that while FOP-based methods are not always the best-performing system for each metric, they are consistently the most reliable. Specifically, CGMH has high success rate, but lowest F1 and human scores. Prompt, on the other hand has the highest human evaluation scores but the worst success rate. This is not too surprising. It is, after all, an unmodified language model, so it should be fluent and on topic when viewed by a human. However, given its extremely low success rate, it is not viable for long-form controlled generation. In contrast, FOP-based methods are either the top 1 or 2 performing system across all summary statistics.

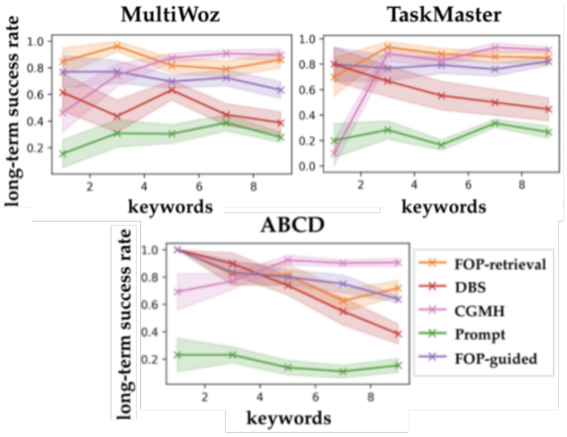


Figure 4: Long-term success rate computed on simulated roll-outs for MultiWoz, TaskMaster, and ABCD. Details on hyperparameters are in Appendix A.3.

7.2 Long-term Success Rate

The first analysis involves comparing all methods on long-term success rate, which measures the percentage of control words in generated simulated roll-outs. To do this, we train a separate user model with the training dataset. We perform a roll-out per test example with 10 generated system responses and 10 generated user responses and compute the percentage of control words in the generated system responses. When counting the number of generated words, we compare word stems.

Figure 4 shows the performance of all approaches when varying the number of control words. Both of our approach variants (FOP-retrieval and FOP-guided) have higher success rates than Prompt and DBS. Prompt is the method with the lowest performance because including the control words at the beginning without any re-training doesn't provide the model with sufficient information to generate the control words. DBS does well when there is only a few control words but struggles as the number of control words increases. This is because DBS is not able to filter out words that are irrelevant at the current time step and instead simply tries to generate the words one by one. This method is also unable to handle words when not in the exact order it should appear.

FOP-retrieval, in some cases, has higher performance than FOP-guided because it will get all keywords in the retrieved response correct. FOP-guided can choose to ignore these keywords if the LM overrides it. So, we would expect FOP-retrieval to do better on this metric, compared to

Methods	MultiWoz 2.3			TaskMaster-3			ABCD			Overall
	p	r	$f1$	p	r	$f1$	p	r	$f1$	avg($f1$)
$\mathcal{W}_{first}$	0.25	0.18	0.21	0.22	0.19	0.2	0.29	0.24	0.27	0.23
Fine-tuned	**0.64**	**0.23**	**0.34**	**0.82**	0.34	0.48	0.68	0.13	0.22	0.35
Prompt	0.45	0.18	0.25	0.81	0.36	0.49	**0.69**	0.18	0.29	0.34
DBS	0.4	0.2	0.27	0.43	0.27	0.33	0.39	0.17	0.24	0.28
CGMH	0.27	0.18	0.21	0.17	0.03	0.05	0.27	0.22	0.24	0.17
FOP-retrieval	0.38	0.18	0.25	0.68	0.38	0.49	0.65	0.33	0.44	0.39
FOP-guided	0.36	0.18	0.24	0.62	**0.48**	**0.54**	0.6	**0.36**	**0.45**	**0.41**

Table 2: Precision, recall, and F1-score for all methods on Multiwoz, TaskMaster, and ABCD. These metrics capture whether the approaches generate control words at the right time by using the control words in the ground truth response as a proxy. The last column is the macro $f1$-score average across all datasets.

Methods	FL	CC	RL	CO	DV
DBS	4.60*	3.65†	**3.80**	2.90	3.10†
CGMH	1.70†	1.24†	1.52†	1.12†	1.82†
FOP-retrieval	**4.81**	**4.77**	3.63	2.82	4.35
FOP-guided	4.36†	4.53*	3.77	**3.12**	**4.47**
Prompt	4.87	4.98	4.30	4.22	3.42
True	4.88	4.90	4.83	4.92	4.80

Table 3: Human evaluation of simulated roll-outs. **FL**: fluency; **CC**: control-consistency; **RL**: relevance; **CO**: Coherence; **DV**: diversity. * and † indicate significant differences from the best result in that column (bolded, excluding True and Prompt) with p-value < 0.05 and < 0.001 respectively, using Welch's t-test. Annotators rated fluency, control-consistency, and relevance per response, while coherence and diversity were annotated per conversation. All metrics are on a scale of 1 to 5.

FOP-guided. We also include an ablation experiment in Appendix A.1.1 to analyze the effect of removing the sliding window in FOP-guided. CGMH seems to do well on long-term success rate, but human evaluation (Section 7.4) results reveal that the generated responses are not very fluent. This method is one that can game previous evaluation metrics, as it tends to condense many or all control words into one utterance. Thus, these approaches are better evaluated through the next set of metrics: precision, recall, and F1-score.

7.3 Control Word P/R/F1

We now measure how well the approaches generate control words at the right time using precision, recall, and F1-score. Table 2 compares these metrics on all datasets. We see that, on average across all datasets, FOP-guided gets higher F1-scores compared with baseline methods. This is because by retrieving similar futures, we are able to guide the language model towards generating control words at appropriate points in the conversation. FOP-guided does worse on MultiWoz because the dataset contains more domains and has much more variety in the conversations. This diversity makes it hard for retrieval-based methods to successfully find similar conversations to guide generation.

The naive approach $\mathcal{W}_{\text{first}}$ gets low recall and precision since it only outputs the control words at the first utterance. Similar to $\mathcal{W}_{\text{first}}$, CGMH gets low F1-scores because it generates many control words early in the conversation rather than at a natural time. DBS also does not do well on these evaluation metrics as it is highly affected by the order of control words, while our method is able to retrieve similar futures to generate appropriate words at the current time step. Finally, Prompt does well on precision but not on recall as it's not explicitly guided to generate the control words.

7.4 Human Evaluation

Finally, we rate all methods on human evaluation. We follow recent work on good evaluation practices for text generation approaches (Karpinska et al., 2021). Further details are in Appendix A.4.

Fluency: Is the response fluent and grammatical?
Control consistency: *When* control words appear in the response, are they appropriately used?
Relevance: Is the response a natural reply to the previous utterance in the conversation?
Coherence: Are all of the system responses in the conversation coherent with respect to each other?
Diversity: Is there diversity in the system responses of the conversation?

Two raters annotated each example, and agreement was measured using Krippendorff's alpha for each of the 5 metrics (0.84, 0.74, 0.82, 0.76, 0.67). We present results in Table 3 for all five approaches as well as for the ground truth conversation. We focus on comparisons between DBS, CGMH, and the FOP methods, as these were the methods that performed comparably on control metrics (at least 40% on long-term success rate) and thus are reasonable baselines for long-term control.

CGMH consistently gets low scores across all metrics. Compared to DBS, FOP-guided performs similarly on fluency, relevance, and coherence but much better on control-consistency and diversity, which could be because retrieval helps decide naturally what to say throughout the conversation. FOP-guided is at least as good as FOP-retrieval on relevance, coherence, and diversity, while only slightly worse on fluency and control-consistency. This is because FOP-guided uses the context and retrieved sentence to *generate* a response, while FOP-retrieval selects an already fluent historical response. Overall, human evaluation results highlight that both of our proposed methods generate realistic, coherent text, while also generating a high percentage of control words.

8 Conclusion

In this paper, we propose the problem of constrained dialogue generation, which involves controlling dialogue responses such that a set of control words appear at some point in the future of the conversation. We propose a new set of metrics as well as a novel method that leverages retrieval of relevant conversations to control future generated responses. We show on three datasets that our method outperforms several constrained text generation baselines on quantitative metrics as well as human evaluation. As far as we are aware, this is the first work to address the problem of long-term control for dialogue generation.

9 Acknowledgments

We thank S.R.K Branavan and Derek Chen for their insightful feedback. We thank Tianyi Zhang for his starting code that we built upon in this work. We also want to thank Ethan Elenberg, Felix Wu, Clemens Rosenbaum, Sam Altschul, David Sontag, and the rest of the ASAPP research team for all of their feedback in making this work stronger.

References

Ashutosh Baheti, Alan Ritter, Jiwei Li, and Bill Dolan. 2018. Generating more interesting responses in neural conversation models with distributional constraints. *arXiv preprint arXiv:1809.01215*.

Paweł Budzianowski and Ivan Vulić. 2019. Hello, it's gpt-2–how can i help you? towards the use of pretrained language models for task-oriented dialogue systems. *arXiv preprint arXiv:1907.05774*.

Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Ben Goodrich, Daniel Duckworth, Semih Yavuz, Amit Dubey, Kyu-Young Kim, and Andy Cedilnik. 2019. Taskmaster-1: Toward a realistic and diverse dialog dataset. 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 4516–4525, Hong Kong, China. Association for Computational Linguistics.

Hengyi Cai, Hongshen Chen, Yonghao Song, Xiaofang Zhao, and Dawei Yin. 2020. Exemplar guided neural dialogue generation. In *International Joint Conference on Artificial Intelligence (IJCAI)*.

Yu Cao, Wei Bi, Meng Fang, and Dacheng Tao. 2020. Pretrained language models for dialogue generation with multiple input sources. *arXiv preprint arXiv:2010.07576*.

Derek Chen, Howard Chen, Yi Yang, Alexander Lin, and Zhou Yu. 2021. Action-based conversations dataset: A corpus for building more in-depth task-oriented dialogue systems. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3002–3017, Online. Association for Computational Linguistics.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Yuntian Deng, Anton Bakhtin, Myle Ott, Arthur Szlam, and Marc'Aurelio Ranzato. 2020. Residual energy-based models for text generation. *arXiv preprint arXiv:2004.11714*.

Angela Fan, Claire Gardent, Chloe Braud, and Antoine Bordes. 2020. Augmenting transformers with knn-based composite memory for dialogue. *arXiv preprint arXiv:2004.12744*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Asma Ghandeharioun, Judy Hanwen Shen, Natasha Jaques, Craig Ferguson, Noah Jones, Agata Lapedriza, and Rosalind Picard. 2019. Approximating interactive human evaluation with self-play for open-domain dialog systems. *arXiv preprint arXiv:1906.09308*.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017,* *System Demonstrations*, pages 43–48, Vancouver, Canada. Association for Computational Linguistics.

Aditya Grover, Jiaming Song, Ashish Kapoor, Kenneth Tran, Alekh Agarwal, Eric J Horvitz, and Stefano Ermon. 2019. Bias correction of learned generative models using likelihood-free importance weighting. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jiatao Gu, Kyunghyun Cho, and Victor OK Li. 2017. Trainable greedy decoding for neural machine translation. *arXiv preprint arXiv:1702.02429*.

Prakhar Gupta, Jeffrey P Bigham, Yulia Tsvetkov, and Amy Pavel. 2020. Controlling dialogue generation with semantic exemplars. *arXiv preprint arXiv:2008.09075*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Donghoon Ham, Jeong-Gwan Lee, Youngsoo Jang, and Kee-Eung Kim. 2020. End-to-end neural pipeline for goal-oriented dialogue systems using gpt-2. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 583–592.

Ting Han, Ximing Liu, Ryuichi Takanobu, Yixin Lian, Chongxuan Huang, Wei Peng, and Minlie Huang. 2020. Multiwoz 2.3: A multi-domain task-oriented dataset enhanced with annotation corrections and co-reference annotation. *arXiv preprint arXiv:2010.05594*.

Chris Hokamp and Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. *arXiv preprint arXiv:1704.07138*.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Marzena Karpinska, Nader Akoury, and Mohit Iyyer. 2021. The perils of using mechanical turk to evaluate open-ended text generation. *arXiv preprint arXiv:2109.06835*.

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

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Nearest neighbor machine translation. *arXiv preprint arXiv:2010.00710*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Phillip Lippe, Pengjie Ren, Hinda Haned, Bart Voorn, and Maarten de Rijke. 2020. Diversifying task-oriented dialogue response generation with prototype guided paraphrasing. *arXiv preprint arXiv:2008.03391*.

Andrea Madotto, Etsuko Ishii, Zhaojiang Lin, Sumanth Dathathri, and Pascale Fung. 2020. Plug-and-play conversational models. *arXiv preprint arXiv:2010.04344*.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. *arXiv preprint arXiv:2110.08118*.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6834–6842.

Damian Pascual, Beni Egressy, Florian Bolli, and Roger Wattenhofer. 2020. Directed beam search: Plug-and-play lexically constrained language generation. *arXiv preprint arXiv:2012.15416*.

Baolin Peng, Chunyuan Li, Jinchao Li, Shahin Shayandeh, Lars Liden, and Jianfeng Gao. 2020. Soloist: Few-shot task-oriented dialog with a single pre-trained auto-regressive model. *arXiv preprint arXiv:2005.05298*.

Nanyun Peng, Marjan Ghazvininejad, Jonathan May, and Kevin Knight. 2018. Towards controllable story generation. In *Proceedings of the First Workshop on Storytelling*, pages 43–49.

Matt Post and David Vilar. 2018. Fast lexically constrained decoding with dynamic beam allocation for neural machine translation. *arXiv preprint arXiv:1804.06609*.

Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2018. Semantically equivalent adversarial rules for debugging nlp models. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 856–865.

Iulian Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 30.

Lei Sha. 2020. Gradient-guided unsupervised lexically constrained text generation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8692–8703.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Jianheng Tang, Tiancheng Zhao, Chenyan Xiong, Xiaodan Liang, Eric Xing, and Zhiting Hu. 2019. Target-guided open-domain conversation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5624–5634, Florence, Italy. Association for Computational Linguistics.

Eric Wallace, Shi Feng, Nikhil Kandpal, Matt Gardner, and Sameer Singh. 2019. Universal adversarial triggers for attacking and analyzing nlp. *arXiv preprint arXiv:1908.07125*.

Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. 2019. Transfertransfo: A transfer learning approach for neural network based conversational agents. *arXiv preprint arXiv:1901.08149*.

Zeqiu Wu, Michel Galley, Chris Brockett, Yizhe Zhang, Xiang Gao, Chris Quirk, Rik Koncel-Kedziorski, Jianfeng Gao, Hannaneh Hajishirzi, Mari Ostendorf, et al. 2020. A controllable model of grounded response generation. *arXiv preprint arXiv:2005.00613*.

Can Xu, Wei Wu, Chongyang Tao, Huang Hu, Matt Schuerman, and Ying Wang. 2019. Neural response generation with meta-words. *arXiv preprint arXiv:1906.06050*.

Semih Yavuz, Abhinav Rastogi, Guan-Lin Chao, and Dilek Hakkani-Tur. 2019. Deepcopy: Grounded response generation with hierarchical pointer networks. *arXiv preprint arXiv:1908.10731*.

A Appendix

A.1 Additional results

A.1.1 Ablation of window in FOP-guided

We ran ablation experiments comparing FOP-guided with a version without the sliding window. Table 4 includes the results for all of the baselines on the most difficult setting for ABCD (9 control words).

Methods	Long-term success rate
Prompt	0.15
DBS	0.38
CGMH	0.91
FOP-retrieval	0.72
FOP-guided	0.69
FOP-guided (no-window)	0.56

Table 4: Ablation experiment for the most difficult setting in ABCD (9 control words). FOP-guided without a sliding window performs worse on long-term success rate.

Our approach FOP-guided gets more than 10% more control words in simulated rollouts, compared with FOP-guided without the window approach, which highlights the usefulness of the sliding window component. We also compare the two FOP-guided variants when varying the number of control words and see that FOP-guided consistently performs better (Figure 5).

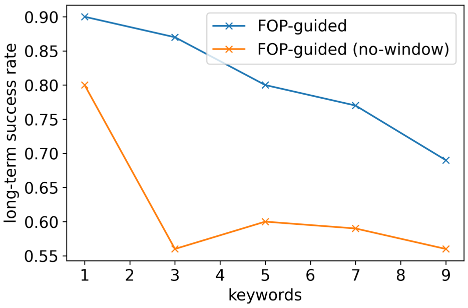


Figure 5: Long-term success rate on ABCD, comparing FOP-guided and FOP-guided without a sliding window.

A.2 Example simulations on ABCD

In Tables 5, 6, 7, 8, and 9, we show some example simulations on the ABCD dataset using a trained agent model for each of the methods.

A.3 Experiment details

We did a hyperparameter search over the following lambda values $\{0, 5, 10, 15, 20, 25\}$ for all datasets. On both ABCD and MultiWoz, the best hyperparameter for FOP-guided was $\lambda = 15$ and for DBS, it was $\lambda = 20$. For TaskMaster, the best hyperparameter for FOP-guided was $\lambda = 10$ and for DBS, it was $\lambda = 15$. CGMH was run with the recommended hyperparameters from the authors.

For all datasets, we used the number of candidate generations for FOP-guided as $N = 10$ and the window size for logit modification as $q = 4$. The number of examples used for multiple splits of each dataset is as follows: For the ABCD dataset, we used 8034 conversations for training and 1004 conversations each for dev and test splits. In the Multiwoz dataset, we used 8438, 1000, 1000 as train, dev and test splits respectively. Finally, for the Taskmaster-3 dataset, we used 16629, 3564, 3564 as train, dev and test datasets respectively.

We used the GPT2-medium model from the hugging-face repository as the pre-trained language model for all of our experiments. This model contains 345M parameters.

For all our experiments, we used a p3.2xlarge EC2 instance. This instance has one GPU with 16GB capacity and 61GB of RAM. Out of all of our experiments, simulated long-term success rate experiments took the most amount of GPU hours to run. Altogether it took somewhere between 24-36 GPU hours to complete all the experiments.

A.4 Human evaluation setting details

We recruited four trained annotators to evaluate generated conversations on the following five metrics, each on a scale of 1 to 5. We split up the examples across the four annotators such that each example was judged by two annotators. We included the ground truth conversation as an additional baseline to act as an upper bound. To ensure the ratings would be high-quality, we provided a rubric, included below, for each metric with examples for different ratings, did an initial pilot for a few sample conversations, and provided a reference sheet to help calibrate the ratings across annotators.

A.4.1 Rubric

Evaluate generated conversations on a few metrics, each on a scale of 1 to 5:

[utterance-level] Fluency: Is this response fluent and grammatical?

- 1: Generated responses do not make any sense, English-wise and grammar-wise, which could include misspelled words, no transition words, limited punctuation, skipped words, etc (e.g., “the figh help order”)
- 3: Generated responses have some good English so you can make out what is being said but it’s not well-formed sentences (e.g., “will you help order”)
- 5: Generated responses have perfect English and perfect grammar. Customers can use lower-case text as less-formal style so first-letter capitalization is not necessary (e.g., “can you help me refund my order?”)

[utterance-level] Relevance: Is this response a natural reply to the previous utterance in the conversation?

- 1: The generated response is not at all relevant to the conversation context/history (e.g., when asked for account id: “I can’t get my promo code”)
- 3: The generated response is somewhat relevant to the conversation context/history but not the best fit (e.g., when asked for account id: “No”)
- 5: The generated response is perfectly relevant and a great response to the conversation context/history (e.g., when asked for account id: “Account ID: 3425435”)

[utterance-level] Control-consistency: If control words appear in this response, are they appropriately used?

- 1: When used, the control words (which are uppercased) make no sense in the generated responses. They are fully forced into the responses (e.g., “TODAY account id: 435650”)
- 3: When used, the control words (which are uppercased) make some sense in the generated responses but are not super smooth (e.g., “I need help with my order, can you help TODAY?“)
- 5: When used, the control words (which are uppercased) are perfectly and naturally used in the generated responses (e.g., “TODAY, I want to buy a shirt. Can you help me?”)

[conversation-level] Diversity: Is there diversity in the customer responses of the conversation?

- 1: Almost all of the responses are repetitive and have no diversity (e.g., “ok” “ok” “ok” “thanks”)
- 3: Some of the generated responses provide diversity while many do not (e.g., “I want to buy a shirt” “can you help me with this?” “ok” “thanks”)
- 5: All of the generated responses are diverse and provide a variety of interesting words through the conversation. The customer can still say ok and thanks but it shouldn’t happen all the time and has to be appropriate for that point in the conversation (e.g., “I want to buy a shirt” “can you help me with this?” “ok sure” “thank you very much for your help!”)

[conversation-level] Coherence: Are all of the customer responses in the conversation coherent with respect to each other?

- 1: All of the responses have very little relation when seen together (e.g., “I want to buy a shirt”, “can you help me with my promo code”, “my subscription expired”)
- 3: Some of the generated responses in a conversation make sense together and some don’t (e.g., “I want to buy a shirt” “I have a promo code for buying this shirt” “my subscription expired”)
- 5: All generated responses make complete sense together (e.g., “I want to buy a shirt”, “I have a promo code for buying this shirt”, “thank you for helping me with this purchase!”)

FOP-guided	
Control words to generate: valid credit make minute try card purchase trying worked	
Speaker	**Utterance**
AGENT	Hello, thank you for contacting AcmeCorp, how may I help you today?
CUSTOMER	Hi I was **trying** to **make** a **purchase** but for some reason my **credit card** keeps getting rejected
AGENT	I would happy to help.
CUSTOMER	I got an email from you guys about a great sale and it listed this code and it said it was **invalid**
AGENT	Lets try a few things. First can you check the expiration date on your card.
CUSTOMER	No its **valid** thats why I am confused.
AGENT	Can you try to re-enter the card information one more time?
CUSTOMER	Ok just a **minute** please
AGENT	Alright, let me know if that helps.
CUSTOMER	Ok I did its s ordering right now. That **worked** and it says it is correct.
AGENT	Great, is there anything else I can help you with today? Is there anything else?
CUSTOMER	That is all I needed thanks
AGENT	Your welcome. Thank you for contacting us. Have a great day.
CUSTOMER	you too

Table 5: Example simulation on the ABCD dataset using the FOP-guided method

FOP-retrieval	
Control words to generate: valid credit make minute try card purchase trying worked	
Speaker	**Utterance**
AGENT	Hello, thank you for contacting AcmeCorp, how may I help you today?
CUSTOMER	Hi! I was **trying** to **make** a **purchase** but for some reason my **credit card** keeps getting rejected
AGENT	I would happy to help. Lets see if we can troubleshoot this issue.
CUSTOMER	ok let me know what i should **try**
AGENT	Lets try to make sure your card is not expired.
CUSTOMER	no. its **valid**. Thats why I am confused
AGENT	First can you try to log out and log back in.
CUSTOMER	Ok let me give that a shot.
AGENT	Okay, next can you check the expiration date on your card.
CUSTOMER	Ok hang on one **minute** let me **try** that
AGENT	No problem.
CUSTOMER	Tried that. It didn't **work** either
AGENT	Can you try to check the expiration on your card?
CUSTOMER	Let me **try** that
AGENT	Alright.
CUSTOMER	Well that kinda helped. I do not know why

Table 6: Example simulation on the ABCD dataset using the FOP-retrieval method

DBS

Control words to generate: valid credit make minute try card purchase trying worked

Speaker	**Utterance**
AGENT	Hello, thank you for contacting AcmeCorp, how may I help you today?
CUSTOMER	I would like to know if there is any reasonable way for **valid**ating an order?
AGENT	I would happy to look into this for you.
CUSTOMER	I already got a **credit** for a return and would like to correct it.
AGENT	May I have your full name or account id please?
CUSTOMER	Crystal Minh
AGENT	Thank you Crystal.
CUSTOMER	Can you **make** sure that this is correct?
AGENT	One moment please.
CUSTOMER	ok
AGENT	What is your username, email address, and Order ID?
CUSTOMER	crystalm **minute**
AGENT	Thanks Crystal. I need to verify your purchase.
CUSTOMER	crystalm **minute**@email.com
AGENT	Your refund is in progress and you initiated it online.
CUSTOMER	Okay. I will **try** it out now.

Table 7: Example simulation on the ABCD dataset using the DBS method

CGMH

Control words to generate: valid credit make minute try card purchase trying worked

Speaker	**Utterance**
AGENT	Hello, thank you for contacting AcmeCorp, how may I help you today?
CUSTOMER	My**valid credit card make** me **purchase minute** we **try** to **credit card** to **purchase**. Have been **trying** this **worked**
AGENT	I would happy to look into this for you.
CUSTOMER	I was wondering about this method of**valid**
AGENT	May I have your full name or account id please?
CUSTOMER	sure its a**valid**
AGENT	Thank you may I please have your full address?
CUSTOMER	9 Guess**valid**
AGENT	What method of payment would you like to use?

Table 8: Example simulation on the ABCD dataset using the CGMH method

Prompt	
Control words to generate: valid credit make minute try card purchase trying worked	
Speaker	**Utterance**
AGENT	Hello, thank you for contacting AcmeCorp, how may I help you today?
CUSTOMER	Hi I'm having a problem with my account
AGENT	I would happy to help.
CUSTOMER	The **credit card** keeps getting rejected
AGENT	Lets see if we can try a few things.
CUSTOMER	ok
AGENT	Can you try entering the information again?
CUSTOMER	Okay I did but it still doesn't **work**
AGENT	Next can you try logging out and back in?
CUSTOMER	Ok hold on
AGENT	Let me know if that helps.
CUSTOMER	hey that **worked**!

Table 9: Example simulation on the ABCD dataset using the Prompt method