SPADE: Structured Prompting Augmentation for Dialogue Enhancement in Machine-Generated Text Detection

Haoyi Li*
The University of Melbourne
haoyil4@student.unimelb.edu.au

Angela Yifei Yuan*
The University of Melbourne
angela.yuan@student.unimelb.edu.au

Soyeon Caren Han†
The University of Melbourne
Caren.Han@unimelb.edu.au

Christopher Leckie
The University of Melbourne
caleckie@unimelb.edu.au

Abstract

The increasing capability of large language models (LLMs) to generate synthetic content has heightened concerns about their misuse, driving the development of Machine-Generated Text (MGT) detection models. However, these detectors face significant challenges due to the lack of high-quality synthetic datasets for training. To address this issue, we propose SPADE, a structured framework for detecting synthetic dialogues using prompt-based positive and negative samples. Our proposed methods yield 14 new dialogue datasets, which we benchmark against eight MGT detection models. The results demonstrate improved generalization performance when utilizing a mixed dataset produced by proposed augmentation frameworks, offering a practical approach to enhancing LLM application security. Considering that real-world agents lack knowledge of future opponent utterances, we simulate online dialogue detection and examine the relationship between chat history length and detection accuracy. Our open-source datasets, code and prompts can be downloaded from https://github.com/AngieYYF/SPADE-customer-service-dialogue.

1 Introduction

Large language models (LLMs) are increasingly deployed in conversational systems, but their accessibility also enables adversaries to launch automated attacks. For instance, in online customer service chatrooms, attackers may use LLMs to launch prompt injection attacks that spread misinformation, or flood the system with realistic but excessive queries, leading to denial-of-service outcomes (OWASP Foundation, 2023; Zhan et al., 2024). These scenarios highlight the pressing need for high-performance Machine-Generated Text (MGT) detection in dialogue settings. However, existing detectors often fail due to the scarcity of high-quality datasets with dynamic dialogue contexts, where traditional data collection methods are time-consuming and expensive, limiting scalability.

Significant research has been conducted on MGT detection (Kirchenbauer et al., 2023; Lu et al., 2024; Bahad et al., 2024; Koike et al., 2024), focusing on long-form texts such as Reddit (Mitchell et al., 2023), news articles (Li et al., 2024; Wang et al., 2023), Wikipedia entries (Wahle et al., 2022), and student essays (Koike et al., 2024; Wahle et al., 2022). However, these types of texts differ fundamentally from dialogues, which are shorter, turn-based, and involve dynamic interactions between two parties that evolve as the conversation progresses. Traditional detection methods, which are designed for static, longer passages, struggle to handle the fluid and interactive nature of dialogues. This challenge is further exacerbated by the lack of high-quality, domain-specific dialogue datasets, which makes it difficult to develop robust MGT detection methods for conversational environments. The scarcity of suitable dialogue data has been a longstanding issue, and recent methods still have not fully addressed this problem. Collecting real-life dialogues from systems or LLM users is not only expensive but also impractical at scale. To overcome these limitations, data augmentation has emerged as a viable, cost-effective alternative (Sennrich et al., 2016; Kojima et al., 2022; Mao et al.; Labruna et al., 2023). However, maintaining fluency, coherence, and consistency with user goals across multiple interaction stages remains a challenge. Moreover, relying solely on a single augmentation method can limit model generalization, leading to poor performance when encountering out-of-distribution data (Hays et al., 2023).

In this paper, we propose five novel data augmentation frameworks, specifically designed for synthetic user dialogue generation. Note that the

* Both authors contributed equally to this research.
† Corresponding author

Proceedings of the The First Workshop on LLM Security (LLMSEC), pages 142–167

development of LLM-based chatbot detection models faces several key challenges: (1) scarcity of high-quality training data, (2) high costs associated with collecting real-life dialogue datasets, (3) maintaining coherence and fluency in augmented synthetic dialogues, and (4) inefficiencies in performing detection only after dialogue completion. Hence, the proposed frameworks employ a structured prompting approach to generate 14 sets of dialogues, which significantly reduce the costs associated with traditional data collection methods. Our frameworks, Partial-Chatbot and Full-Chatbot, are tailored to the interactive and dynamic nature of dialogue. Through automated and manual quality assurance, we ensure that the generated dialogues are fluent and closely aligned with user goals. Additionally, the frameworks support the simulation of online conversational environments, facilitating offline and real-time detection. The datasets are benchmarked against eight MGT detection models, demonstrating improved generalization performance when trained on a mixture of datasets created using our augmentation techniques. Upon simulating real-world settings, where agents are unaware of future user utterances, we observe a positive correlation between the volume of chat history and detection performance. The proposed datasets and methods enhance MGT detection in dialogues, particularly in cases with limited or incomplete chat history.

The contributions of this paper are:

1. We introduce novel, training-free data augmentation frameworks specifically designed for synthetic user dialogue generation. These frameworks produce 14 new dialogue datasets applicable across various domains, addressing the scarcity of high-quality dialogue data for MGT detection.

2. We refine and enhance domain-specific datasets, ensuring that the dialogues are coherent and aligned with specified user goals, offering a template for other domain-specific applications.

3. We benchmark the performance of these datasets across eight baseline models, demonstrating improved generalizability through the combination of diverse data augmentation methods.

4. We simulate real-time conversations and show that detection accuracy improves as chat history increases, reinforcing the importance of progressive detection.

To the best of our knowledge, this is the first work to introduce training-free dialogue data augmentation frameworks applicable to offline and online environments, advancing the detection of MGT across diverse dialogue settings.

2 Related Work

2.1 Dialogue Datasets

There are several open-sourced multi-domain dialogue datasets, such as MultiWOZ (Eric et al., 2020), SGD (Rastogi et al., 2020), CrossWOZ (Zhu et al., 2020), and EmoWOZ (Feng et al., 2022), which include customer service dialogues. These datasets only feature dialogues with human users, limiting their effectiveness for detection aimed at identifying synthetic users. While recent work (Zheng et al., 2023) has introduced datasets containing LLM-based synthetic users in customer service scenarios, it still falls short of addressing the critical need for extensive dialogue datasets specifically containing synthetic user utterances. Our research addresses this limitation by introducing cost-effective and training-free data augmentation frameworks that generate high-quality synthetic user dialogues.

2.2 Data Augmentation

Acquiring high-quality labelled training datasets is a costly and challenging task, leading to the development of various data augmentation methods to address data scarcity. Paraphrasing was initially conducted in early studies using back translation (Sennrich et al., 2016). With advancements in deep learning, researchers have developed specialized paraphrasing models, such as DIPPER (Krishna et al., 2024) and BART (Lewis et al., 2020; Okur et al., 2022). Goal-to-dialogue generation creates synthetic dialogues by prompting LLMs to output entire dialogues given user goals and instructions (Labruna et al., 2023). Similarly, end-to-end conversation generation assigns roles to 2 LLMs and asks them to complete dialogues interchangeably (Labruna et al., 2023; Abdullin et al., 2023; Abbasiantaeb et al., 2024). Although these two methods seem easy to implement, they have widely recognized drawbacks. Different LLMs require varying prompt structures, and logic and coherence issues can compromise the overall quality of

the dialogue. Our new data augmentation framework overcomes these challenges by maintaining essential conversational features and employing well-structured prompts tailored to widely used LLMs.

2.3 MGT Detection

MGT detection is a classification task where a model aims to classify a given text into categories such as human versus any subsets of LLMs. Detection approaches can generally be divided into three categories: statistical methods, fine-tuned pretrained models, and feature-based methods. Statistical methods rely on the different distributions of word choices between humans and language models. Some statistics and proposed models include cross-perplexity (Hans et al., 2024), entropy (Lavergne et al., 2008; Gehrmann et al., 2019), and log probability (Mitchell et al., 2023; Bao et al.), which had outstanding performance in zero-shot MGT detection. Fine-tuning pre-trained transformer models such as BERT and RoBERTa (Wang et al., 2023; Bahad et al., 2024; Guo et al., 2023), which study semantic features for MGT classification tasks, also have impressive performance. Feature-based models rely on difference in semantic and lexical features between human-written text and MGT (Mindner et al., 2023). The extracted features serve as input to common machine learning (ML) models for classification. We have chosen to evaluate our datasets using statistical-based, feature-based, and pretrained LLM-based methods, in order to compare the performance of detectors across different data augmentation frameworks and to evaluate the ability of data augmentation methods to enhance model generalization.

3 Data Augmentation Framework

This section details the proposed training-free dialogue data augmentation frameworks designed to generate high-quality synthetic dialogues, from bona fide human-generated dialogues. These frameworks fall into two main categories: Partial-Chatbot Data Augmentation and Full-Chatbot Data Augmentation.

Figure 1 outlines the abstract construction process of each framework. Appendix A.1 provides example dialogues and complete prompts.

3.1 Partial-Chatbot Data Augmentation

The Partial-Chatbot Data Augmentation frameworks generate dialogues partially authored by LLMs, while the remaining dialogue segments retain human-generated utterances.

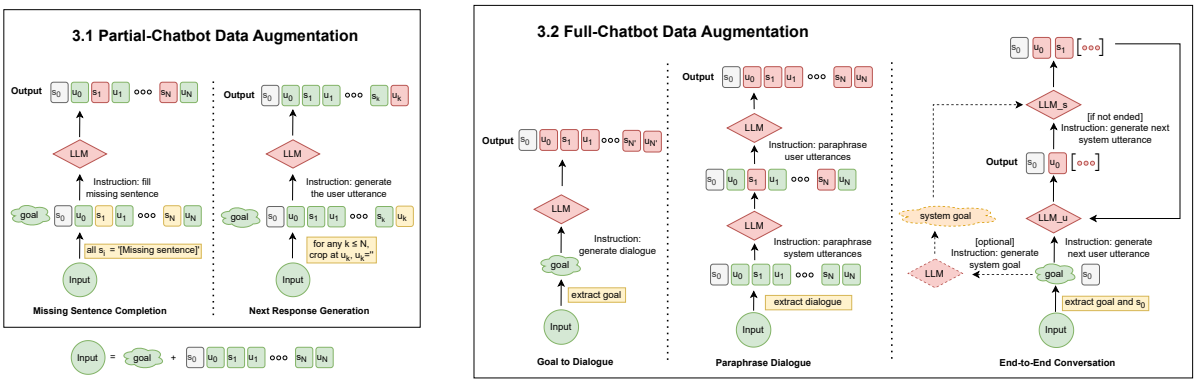
Missing Sentence Completion: The Missing Sentence Completion, denoted as f_{MS}, generates Partial-Chatbot dialogues by filling in the missing sentences for one of the participants in the conversation. All system utterances in bona fide dialogues are replaced by synthetic text, to be used as negative samples when positive samples contain synthetic system utterances. This controls the consistency of whether the system is a chatbot or human across positive and negative samples with synthetic and human users respectively. For each original dialogue d_i, $f_{MS}(d_i) = L(q(d_i), t_{MS})$, where $q(d_i)$ replaces all system utterances $d_{ij}^{(s)}$ with "[missing sentence]", and t_{MS} is a prompt engineered for LLM L to replace missing sentences. A comprehensive structure for prompt t_{MS} is provided in Appendix A.1.

Next Response Generation: The Next Response Generation framework, denoted as f_R, produces user responses based on incomplete dialogue history, ensuring consistency by maintaining original system utterances. This framework only generates user responses, which serve as positive samples for MGT detection. $f_R(d_i) = \{L(\phi(d_i^k), g_i, t_R) | k \leq N_i\}$ where the original dialogue d_i with N_i turns is cropped to produce incomplete chat histories d_i^k with exactly k turns, and $\phi(d_i^k)$ replaces the last user utterance $d_{ik}^{(u)}$ with an empty string. t_R is a prompt engineered for L to generate the user response $d_{ik}^{\prime(u)}$ according to the original goal g_i. This approach not only ensures dialogue coherence and goal alignment but also eliminates any reliance on synthetic system responses for detection, thereby enhancing applicability.

3.2 Full-Chatbot Data Augmentation

The Full-Chatbot Data Augmentation frameworks generate dialogues in which LLMs produce both system and user utterances.

Goal to Dialogue (G2D): The G2D framework, denoted as $f_G(g_i) = L(g_i, t_G)$, generates a Full-Chatbot dialogue based on a user goal g_i. Unlike traditional few-shot learning methods that require the selection of demonstrations for each goal, G2D employs a structured prompt t_G with comprehensive instructions for the LLM. This approach reduces input token overhead and achieves high goal-dialogue alignment without external training. The prompt



(a) Partial-Chatbot Data Augmentation

(b) Full-Chatbot Data Augmentation

Figure 1: Data augmentation frameworks, where the input is a dialogue alternating between a customer service system (s_i) and a human user (u_i) with a user goal ($goal$), and outputs a Partial-Chatbot or a Full-Chatbot dialogue. The initial system response s_0 can be omitted or set to a standard starting line common to all dialogues. LLM instructions shown here are incomplete.

t_G is tailored according to the following components: 1) Task Summary: A brief description of the dialogue objective. 2) Example Dialogue: A sample conversation demonstrating the expected interaction flow. 3) Goal-Specific Instructions for User: Detailed guidance on how the synthetic user should respond based on g_i. 4) Slot and Domain Knowledge for System: Contextual information is required for the system to provide coherent responses. 5) Conversation Termination Conditions: Criteria for ending the dialogue to ensure it remains goal-oriented. 6) Sensitive Information Masking: Instructions to anonymize sensitive details, such as replacing the exact reference number "AX12387" with "[ref]". This structured prompting enables the generation of realistic dialogues that align closely with the user's goal, increasing the diversity and variability of synthetic samples. Dialogues generated using G2D can serve as positive samples compared to those produced by the Missing Sentence Completion framework. Complete prompting templates are provided in Appendix A.1.

Paraphrase Dialogue (Par.): The Par. framework (f_P) uses an iterative paraphrasing strategy to convert Full-Human dialogues into Full-Chatbot dialogues while preserving the conversational structure and logical flow. The process involves two stages: (i) $d_i^1 = L_s(d_i, t_{P,s})$ is a dialogue with all system utterances $d_{ij}^{(s)}$ replaced, and (ii) $d_i^2 = L_u(d_i^1, t_{P,u})$ has all user utterances $d_{ij}^{1(u)}$ in stage 1 output replaced. This two-stage approach produces two distinct dialogue sets: d_i^1 as negative samples (synthetic system responses only) and d_i^2 as positive samples (fully synthetic dialogues). While this method offers limited flexibility in user simulation due to its dependence on the original dialogue's structure, it enhances the cohesiveness of the system's utterances. Example prompting structures are in Appendix A.1.

End-to-End Conversation (E2E Convo.): The E2E Convo. framework (f_L) generates fully synthetic dialogues by assigning distinct roles (system and user) to two instances of LLMs. The LLMs interact to create a complete dialogue. The prompt structure for E2E Convo. includes: 1) Task Summary: Overview of the dialogue scenario and expected outcomes. 2) Example Dialogue: A sample conversation to illustrate the interaction. 3) Role-Specific Instructions: Detailed guidance for both user and system LLMs. 4) Conversation Termination Conditions: Specifications for when to conclude the interaction. 5) Sensitive Information Masking: Instructions to mask identifiable information, such as replacing "AX12387" with "[ref]". 6) Chat History Context: Previously exchanged dialogue turns to maintain context. The generated dialogues can serve as positive samples for training detection models. In contrast, dialogues generated by the Missing Sentence Completion framework can serve as negative samples. The example prompting structure can be found in Appendix A.1.

4 Dialogue Data Construction

This section outlines the preprocessing of the Full-Human dataset, synthetic data generation, and quality assurance.

4.1 Data Source

The MultiWOZ 2.1 dataset (Eric et al., 2020) contains customer service dialogues like hotel booking, collected using a Wizard-of-Oz setup where two participants act as the user and system. We use this dataset with Covlab3 (Zhu et al., 2023) labeled user goals as our baseline for applying data augmentation frameworks. However, goal-dialogue mismatches led to repetitive responses, such as repeatedly asking, "Does this hotel have WiFi?" across different contexts. This was due to discrepancies between the dialogues and their annotated goals, including missing or incorrect goals. To resolve this, we conducted a two-step refinement (i) Llama 70B (Touvron et al., 2023) automatically verified goal achievement, and (ii) we manually revised goals to ensure alignment without changing dialogue content. Incomplete dialogues were removed, resulting in a final set of 616 out of 623 refined hotel dialogues.

4.2 Data Collection

We employ two widely used LLMs to generate the synthetic user datasets, GPT-3.5 (OpenAI, 2023) and Llama 70B (Touvron et al., 2023). We executed the framework defined in Section 3 to generate Partial-Chatbot and Full-Chatbot synthetic dialogues, utilizing the fine-tuned MultiWOZ 2.1 dataset defined in Section 4.1. As shown in Table 1, 14 new datasets are created according to our training-free data augmentation frameworks. We produced 6 Partial- and 8 Full-Chatbot dialogue datasets.

During the dialogue generation process, we found that LLM-generated dialogues include errors such as meaningless information, dialogues in the wrong format, and repeated utterances. To eliminate these errors, we regenerate the erroneous dialogues until we obtain correct results. The regeneration takes 15 rounds on average for each data augmentation method. The entire generation process for all 14 dialogue datasets cost approximately 10 AUD using the API. The quality of generated responses are assessed using both automatic and manual metrics. We automated the validation of several structural aspects (e.g., interleaving of user-system turns, absence of repetition) and manually review content quality. Re-generation stops when outputs pass these checks. To further test the robustness of our prompts, we conduct an exchanged prompts experiment for the employed LLMs, detailed in Appendix A.4.

Table 1: 14 new datasets generated using different data augmentation frameworks proposed.

No.	Category	Dataset
1	Full-Chatbot	GPT Par. Full-Chatbot
2		Llama Par. Full-Chatbot
3		GPT G2D
4		Llama G2D
5		GPT-GPT E2E Convo.
6		Llama-Llama E2E Convo.
7		GPT-Llama E2E Convo.
8		Llama-GPT E2E Convo.
9	Partial-Chatbot	GPT Par. Chatbot-Human
10		Llama Par. Chatbot-Human
11		GPT Missing Sentence Completion
12		Llama Missing Sentence Completion
13		GPT Next Response Generation
14		Llama Next Response Generation

Table 2: UniEval-Dialog quality assurance results of generated datasets.

Dataset	GPT-3.5	Llama 70B
Missing Sentence Completion	97.24%	98.88%
Next Response Generation	95.79%	96.74%
Par. Chatbot-Human	98.55%	97.43%
Par. Full-Chatbot	99.84%	99.52%
G2D	100.0%	97.73%
E2E Convo.	99.75%	99.12%

4.3 Quality Assurance

As the flexibility given to LLMs may cause coherence and fluency issues or misalignment between the goal and the augmented synthetic dialogues. We evaluate the quality of our dataset. For Partial-Chatbot synthetic dialogue, measurements focus on the coherence and fluency of the generated responses about the given chat history. For Full-Chatbot synthetic dialogues, after evaluating the dialogues according to the dimensions mentioned above, we additionally conduct automated and manual quality assurance on the degree of matching between the dialogues and the original goal to further assess the quality of the generated dialogues against the mismatch issues present in Section 4.1.

To conduct automated quality assurance on Partial-Chatbot and Full-Chatbot dialogues, we use a pre-trained model called UniEval-dialog (Zhong et al., 2022), which measures the generated dialogue responses regarding naturalness, coherence, engagement, groundedness, and understandability. The model outputs 'yes' or 'no.' We calculate the 'yes' rate to illustrate the quality of our generated dialogues.

Table 2 shows how often the generated dialogues were rated as coherent and fluent using the UniEval-

dialog model. All generated dialogue datasets achieved scores above 95%, demonstrating consistently high quality. The dialogues generated by the three Full-Chatbot data augmentation frameworks achieved scores over 97%, showing a higher quality compared to Partial-Chatbot dialogues.

To ensure each Full-Chatbot dialogue matches the provided goal, we conduct automatic and human survey-based degrees of match experiment. The results show that our generated dialogues achieve an average goal dialogue match score similar to the original human dialogues. Details of the experiments can be found in Appendix A.3.

In summary, all synthetic dialogue datasets generated from our training-free data augmentation frameworks have reached a standard that resolves the challenge for dialogue augmentation mentioned in Section 1.

5 Offline Dialogue Detection

This section focuses on offline dialogue detection, where a single-stage MGT detection model is applied to all user responses in a dialogue simultaneously. We evaluated the performance of our augmented datasets using eight models for MGT detection.

5.1 Feature-based Detection

Based on the features summarized in (Mindner et al., 2023), we implemented a selection of features relevant to dialogues, along with utterance counts and derived metrics, including 7 categories: Sentiment, Errors, Readability, Statistic, List Lookup, Document, Text Vector, Derived Features. Feature vectors with 910 dimensions were extracted from user utterances only. After scaling and F-test feature selection, these vectors were used as inputs for traditional ML models like Logistic Regression (LogR) (Berkson, 1944), Support Vector Machine (SVM) (Cristianini and Ricci, 2008), Random Forest (RF) (Breiman, 2001), Multilayer Perceptron (MLP) (Taud and Mas, 2018), and XGBoost (Chen and Guestrin, 2016).

5.2 Statistical-based Detection

Inspired by the entropy-based MGT detection method from Liu (Gehrmann et al., 2019), for each dialogue $d = \{w_0, \cdots, w_n\}$ consisting of n word tokens across n_{sen} user utterances, we calculate entropy as follows:

$$S = -\frac{\Sigma_{i=0}^{n} P(w_i) log_2(P(w_i))}{n_{sen}}$$

$P(w_i) = \frac{n_i}{N}$, where n_i represents the frequency of word w_i in the dataset, and N is the total number of tokens in the dataset. In addition to entropy, we include Binoculars (Hans et al., 2024) as another statistical model based on cross-perplexity allowing evaluation without task-specific fine-tuning.

5.3 PLM-based Detection

We adopted the PLM (Pretrained Language Model)-based detection model from Seq-RoBERTa (Wang et al., 2023), where dialogue-level predictions are made using hard voting across token-level predictions. Padding labels were applied to non-user tokens to exclude them from model training and final prediction. For the Next Response Generation dataset, meaningful labels were assigned only to the last user response, with all other tokens using padding labels.

5.4 Experiments and Results

We conducted experiments using all eight models from three detection method categories, testing across 14 datasets each containing 616 dialogues, generated using Partial-Chatbot and Full-Chatbot data augmentation frameworks (Section 3). The models were evaluated on both multiclass classification (distinguishing "Human,"GPT", and "Llama") and binary classification ("human" and "AI"). The evaluation metric used was the macro F1 score, where low scores indicate misclassifications for both chatbot and human detection. For the Next Response Generation dataset, each human response served as a negative sample, and each generated next response was used as a positive sample. For full dialogue datasets, negative and positive samples were defined as per Section 3. We randomly split 80% of each dataset by dialogue ID for training and reserved 20% for testing. Each model was trained and tested five times, and the mean F1 score was reported.

Table 3 presents the detailed results of our experiments for binary classification, and the results for multiclass classification can be found in Appendix A.2. Several key observations can be drawn. First, the detection performance on the Par. dataset was consistently lower across all models compared to the G2D and E2E Convo. datasets. This suggests that paraphrasing introduces greater complexity for

Table 3: Macro-F1 of detection models. The highest score of each detection task is in bold.

		Statistical		PLM		Feature			
Dataset	**Detection**	**Entropy**	**Binoculars**	**RoBERTa**	**MLP**	**XGboost**	**LogR**	**SVM**	**RF**
Par.	Binary	0.6740	0.7451	0.8942	0.9510	0.8754	**0.9592**	0.9558	0.9455
G2D	Binary	0.6617	0.8227	0.9913	**0.9914**	0.9432	0.9857	0.9878	0.9699
E2E Convo.	Binary	0.8846	0.8227	0.9939	**0.9976**	0.9816	0.9949	0.9949	0.9899
Next Response Generation	Binary	0.7176	0.6342	0.9372	0.9414	0.8780	**0.9537**	0.9436	0.9384

MGT detection, making it harder for models to identify synthetic dialogues. For instance, in binary classification, the MLP model achieved an F1 score of 0.95 on the Par. dataset, but a perfect score of 0.99 for both the G2D and E2E Convo. datasets. This performance gap indicates that paraphrased dialogues, by altering sentence structures while maintaining semantic meaning, are more difficult for models to detect as machine-generated, unlike the synthetic dialogues generated by G2D or E2E Convo., where the structure remains more predictable. Second, datasets containing single sentences, like Next Response Generation, exhibited consistently lower performance compared to full dialogue datasets. This highlights the importance of longer context in improving detection accuracy, as models can leverage richer linguistic features and patterns. For example, the MLP model's F1 score for Next Response Generation was 0.9414, significantly lower than the 0.99 achieved for both G2D and E2E Convo. datasets. This suggests that with more conversational turns and context, models can better differentiate between human and AI-generated dialogues by identifying language and interaction flow inconsistencies. Based on these findings, the best-performing models for our task are PLM-based and feature-based methods, especially RoBERTa, MLP, and LogR, which are employed in subsequent experiments to further explore MGT detection performance.

5.5 Cross Dataset Detection

To evaluate generalizability, we tested the top three models (MLP, LogR, and RoBERTa), on cross-dataset binary classification between human and AI-generated dialogues. We use Par., G2D and E2E Convo. datasets separately as testing datasets. To form a training dataset, we randomly sample 1246 dialogues with equally distributed human and synthetic data from different combinations of datasets other than the testing dataset. For RoBERTa, we fine-tuned the model for four epochs on each training dataset. For feature-based models MLP and LogR, we use the trained vectorizer from the training dataset to derive the TFIDF features for the dialogues in the testing dataset.

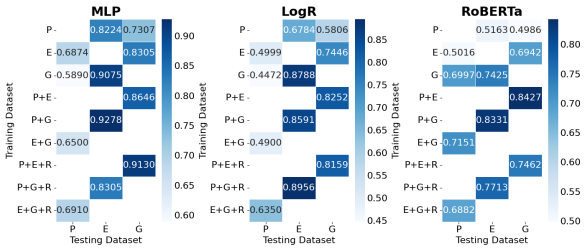


Figure 2: Performance for Cross Dataset Detection with LogR, MLP, and RoBERTa trained on the column-indexed dataset and tested on the row-indexed dataset. *P* represent Par. dataset, *E* represent E2E Convo. dataset, *G* represent G2D dataset, and *R* represent Next Response Generation dataset.

Figure 2 depicts the F1 scores for 12 different combinations of training and testing datasets, demonstrating that combining datasets from various data augmentation frameworks improves model generalizability. MLP models trained solely on the Par. dataset achieved an F1 score of 0.7307 on the G2D test set, whereas MLPs trained on both the Par. and E2E Convo. datasets performed significantly better, achieving an F1 score of 0.8646. Both models were trained on the same number of samples drawn from different sources. This underscores that training with aggregated datasets from diverse augmentation methods results in better generalization to out-of-distribution data. However, combining complete dialogues with single utterances in training datasets does not continually improve performance. When the Next Response Generation dataset was added to the training set, RoBERTa's performance dropped from above 0.80 to below 0.80—when tested on E2E Convo. and G2D datasets. This suggests that mixing data types can introduce noise and hinder the model's ability to detect complete synthetic dialogues. To provide further proof our conclusion on generalisability, we conducted additional experiments on other out-of-distribution datasets and different generators. Details about the setup of the experiment and the results can be found in the Appendix A.8.

Table 4: Macro-F1 on exact number of utterances.

Dataset	#Utterance	MLP	LogR	RoBERTa
E2E Convo.	1	0.7446	**0.9063**	0.7095
	2	0.8142	0.9538	**0.9545**
	3	0.8807	0.9683	**0.9825**
	4	0.8828	0.9809	**0.9949**
G2D.	1	**0.8505**	0.8080	0.7560
	2	0.8779	0.8386	**0.9012**
	3	**0.9301**	0.9031	0.9228
	4	**0.9618**	0.9543	0.9456
Par.	1	**0.6303**	0.6070	0.6293
	2	0.7525	**0.7528**	0.7174
	3	0.7836	**0.8504**	0.7969
	4	0.8013	**0.8767**	0.8356

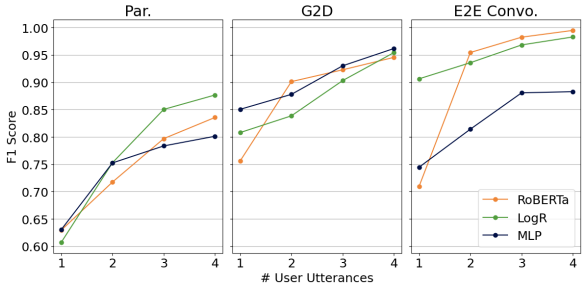


Figure 3: The line graph illustrates a positive relationship between detection rate and the number of user utterances.

6 Online Dialogue Detection

In an online dialogue setting, both the customer service system and the user can access the chat history but lack knowledge of future utterances. This dynamic introduces a new challenge: detecting chatbot users based on incomplete chat history, with updates occurring as new user utterances are progressively received. Unlike common MGT detection, which receives a passage as input, a new challenge is to detect the chatbot user based on the incomplete chat history and update the detection as user utterances are progressively received. The proposed E2E Convo. framework addresses this challenge by emulating an online environment with turn-based interaction between a system and a user. To conduct online dialogue detection on the generated dataset, we use the high-performing models in the offline dialogue detection setting (Section 5.4, including MLP, LogR, and RobERTa).

Experiments were conducted on the E2E Convo., Par., and G2D datasets to explore performance trends across different prompting structures. Models were initially trained on complete dialogues, and during detection, the input dialogues were progressively cropped to varying lengths while maintaining the same binary classification labels as the full dialogues. We tested the models on dialogues cropped to $k \in \{1, 2, 3, 4\}$ user utterances, as dialogues with less than k user utterances are excluded from testing datasets, and many dialogues terminate after 4 utterances.

On both feature-based models (MLP and LogR) and a PLM-based model (RoBERTa), Figure 3 reveals a positive correlation between detection accuracy and the number of user utterances across all datasets. As shown in Table 4, on the E2E Convo. dataset, MLP and RoBERTa showed F1 scores below 0.8 when only the first user utterance was available. In contrast, LogR achieved an F1 score of 0.9063 with a single utterance, which increased to 0.9809 when four utterances were available. This performance trend highlights the reduced detection accuracy with limited context, which delays the identification of AI users and increases resource consumption. Compared to full dialogue detection (Section 5), the reduced accuracy in online detection with partial conversations underscores the need for future work on continuous chatbot detection. Two methods to enhance the detection of incomplete dialogues are presented in Appendix A.7, and a qualitative analysis of online detection performance is provided in Appendix A.6.

7 Conclusion

The scarcity of high-quality datasets for MGT detection in customer service remains challenging. This paper introduced five structured data augmentation frameworks to reduce the costs of traditional dialogue collection, using prompting techniques to efficiently generate synthetic dialogues. We derived 14 new datasets and evaluating these across eight MGT detection models, we found that training on a mix of these datasets significantly improves generalization. Simulated online dialogue detection showed that longer chat histories enhance detection accuracy. Given the rising threats of misused LLMs in real-world systems, SPADE contributes a practical and extensible methodology for improving LLM safety.

Limitations

We present our data augmentation framework, which generates new datasets focused on customer service dialogues. A significant challenge is enabling continuous learning for the detector, as new utterances are incrementally introduced. Retraining the model every time when a new utterance is received is inefficient due to resource constraints. Our experiments revealed that expanding datasets

with varied chat history scenarios can effectively improve the early detection capabilities of current models. This approach helps the detector generalize across different interaction patterns and user behaviors. However, continuous learning strategies that allow models to update automatically to new utterances without the need for full retraining are an important area for future research. We believe that exploring these strategies will further enhance the efficiency and scalability of synthetic dialogue detection systems.

Ethics Statement

We follow the Code of Ethics. No private data or non-public information is used in this work. For manual quality assurance, we invited 15 volunteers, including 5 bachelor students, 8 master students and 2 PhD students. The human survey does not involve any personally sensitive information.

Acknowledgments

This work was supported in part by the Australian Research Council Centre of Excellence for Automated Decision-Making and Society, and by the Australian Internet Observatory, which is co-funded by the Australian Research Data Commons (ARDC) through the HASS and Indigenous Research Data Commons.

References

Zahra Abbasiantaeb, Yifei Yuan, Evangelos Kanoulas, and Mohammad Aliannejadi. 2024. Let the llms talk: Simulating human-to-human conversational qa via zero-shot llm-to-llm interactions. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, WSDM '24, page 8–17, New York, NY, USA. Association for Computing Machinery.

Yelaman Abdullin, Diego Molla, Bahadorreza Ofoghi, John Yearwood, and Qingyang Li. 2023. Synthetic dialogue dataset generation using LLM agents. In *Proceedings of the Third Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*, pages 181–191, Singapore. Association for Computational Linguistics.

Sankalp Bahad, Yash Bhaskar, and Parameswari Krishnamurthy. 2024. Fine-tuning language models for AI vs human generated text detection. In *Proceedings of the 18th International Workshop on Semantic Evaluation (SemEval-2024)*, pages 918–921, Mexico City, Mexico. Association for Computational Linguistics.

Guangsheng Bao, Yanbin Zhao, Zhiyang Teng, Linyi Yang, and Yue Zhang. Fast-detectgpt: Efficient zero-shot detection of machine-generated text via conditional probability curvature. In *The Twelfth International Conference on Learning Representations*.

Joseph Berkson. 1944. Application of the logistic function to bio-assay. *Journal of the American statistical association*, 39(227):357–365.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '16, page 785–794, New York, NY, USA. Association for Computing Machinery.

Nello Cristianini and Elisa Ricci. 2008. *Support Vector Machines*, pages 928–932. Springer US, Boston, MA.

Mihail Eric, Rahul Goel, Shachi Paul, Adarsh Kumar, Abhishek Sethi, Anuj Kumar Goyal, Peter Ku, Sanchit Agarwal, Shuyang Gao, and Dilek Hakkani-Tür. 2020. Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *12th International Conference on Language Resources and Evaluation, LREC 2020*, pages 422–428. European Language Resources Association (ELRA).

Shutong Feng, Nurul Lubis, Christian Geishauser, Hsien-chin Lin, Michael Heck, Carel van Niekerk, and Milica Gasic. 2022. EmoWOZ: A large-scale corpus and labelling scheme for emotion recognition in task-oriented dialogue systems. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4096–4113, Marseille, France. European Language Resources Association.

Sebastian Gehrmann, Hendrik Strobelt, and Alexander Rush. 2019. GLTR: Statistical detection and visualization of generated text. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 111–116, Florence, Italy. Association for Computational Linguistics.

Biyang Guo, Xin Zhang, Ziyuan Wang, Minqi Jiang, Jinran Nie, Yuxuan Ding, Jianwei Yue, and Yupeng Wu. 2023. How close is chatgpt to human experts? comparison corpus, evaluation, and detection. *Preprint*, arXiv:2301.07597.

Abhimanyu Hans, Avi Schwarzschild, Valeriia Cherepanova, Hamid Kazemi, Aniruddha Saha, Micah Goldblum, Jonas Geiping, and Tom Goldstein. 2024. Spotting llms with binoculars: zero-shot detection of machine-generated text. In *Proceedings of the 41st International Conference on Machine Learning*, pages 17519–17537.

Chris Hays, Zachary Schutzman, Manish Raghavan, Erin Walk, and Philipp Zimmer. 2023. Simplistic

collection and labeling practices limit the utility of benchmark datasets for twitter bot detection. In *Proceedings of the ACM Web Conference 2023*, WWW '23. ACM.

John Kirchenbauer, Jonas Geiping, Yuxin Wen, Jonathan Katz, Ian Miers, and Tom Goldstein. 2023. A watermark for large language models. In *International Conference on Machine Learning*, pages 17061–17084. PMLR.

Ryuto Koike, Masahiro Kaneko, and Naoaki Okazaki. 2024. Outfox: Llm-generated essay detection through in-context learning with adversarially generated examples. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(19):21258–21266.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Kalpesh Krishna, Yixiao Song, Marzena Karpinska, John Wieting, and Mohit Iyyer. 2024. Paraphrasing evades detectors of ai-generated text, but retrieval is an effective defense. *Advances in Neural Information Processing Systems*, 36.

Tiziano Labruna, Sofia Brenna, Andrea Zaninello, and Bernardo Magnini. 2023. Unraveling chatgpt: A critical analysis of ai-generated goal-oriented dialogues and annotations. In *International Conference of the Italian Association for Artificial Intelligence*, pages 151–171. Springer.

Thomas Lavergne, Tanguy Urvoy, and François Yvon. 2008. Detecting fake content with relative entropy scoring. In *Proceedings of the 2008 International Conference on Uncovering Plagiarism, Authorship and Social Software Misuse - Volume 377*, PAN'08, page 27–31, Aachen, DEU. CEUR-WS.org.

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

Yupeng Li, Haorui He, Jin Bai, and Dacheng Wen. 2024. Mcfend: A multi-source benchmark dataset for chinese fake news detection. In *Proceedings of the ACM Web Conference 2024*, volume 9 of *WWW '24*, page 4018–4027. ACM.

Yijian Lu, Aiwei Liu, Dianzhi Yu, Jingjing Li, and Irwin King. 2024. An entropy-based text watermarking detection method. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 11724–11735, Bangkok, Thailand. Association for Computational Linguistics.

Chengzhi Mao, Carl Vondrick, Hao Wang, and Junfeng Yang. Raidar: generative ai detection via rewriting. In *The Twelfth International Conference on Learning Representations*.

Lorenz Mindner, Tim Schlippe, and Kristina Schaaff. 2023. Classification of human- and ai-generated texts: Investigating features for chatgpt. In *Artificial Intelligence in Education Technologies: New Development and Innovative Practices*, pages 152–170, Singapore. Springer Nature Singapore.

Eric Mitchell, Yoonho Lee, Alexander Khazatsky, Christopher D Manning, and Chelsea Finn. 2023. DetectGPT: Zero-shot machine-generated text detection using probability curvature. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 24950–24962. PMLR.

Eda Okur, Saurav Sahay, and Lama Nachman. 2022. Data augmentation with paraphrase generation and entity extraction for multimodal dialogue system. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4114–4125.

OpenAI. 2023. Gpt-3.5: Openai's generative pre-trained transformer 3.5. `https://platform.openai.com`.

OWASP Foundation. 2023. Owasp top 10 for large language model applications. https://owasp.org/www-project-top-10-for-large-language-model-applications/(Accessed: 2024-09-02).

Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. 2020. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 8689–8696.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. 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 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

H. Taud and J.F. Mas. 2018. *Multilayer Perceptron (MLP)*, pages 451–455. Springer International Publishing, Cham.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, and 1

others. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, and 1 others. 2023. Llama: Open and efficient foundation language models. https://ai.facebook.com/blog/large-language-model-llama-meta-ai/.

Jan Philip Wahle, Terry Ruas, Frederic Kirstein, and Bela Gipp. 2022. How large language models are transforming machine-paraphrase plagiarism. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 952–963, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Pengyu Wang, Linyang Li, Ke Ren, Botian Jiang, Dong Zhang, and Xipeng Qiu. 2023. SeqXGPT: Sentence-level AI-generated text detection. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 1144–1156, Singapore. Association for Computational Linguistics.

Qiusi Zhan, Zhixiang Liang, Zifan Ying, and Daniel Kang. 2024. InjecAgent: Benchmarking indirect prompt injections in tool-integrated large language model agents. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 10471–10506, Bangkok, Thailand and virtual meeting. Association for Computational Linguistics.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric P Xing, and 1 others. 2023. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. *CoRR*.

Ming Zhong, Yang Liu, Da Yin, Yuning Mao, Yizhu Jiao, Pengfei Liu, Chenguang Zhu, Heng Ji, and Jiawei Han. 2022. Towards a unified multi-dimensional evaluator for text generation. In *2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022*.

Qi Zhu, Christian Geishauser, Hsien-Chin Lin, Carel van Niekerk, Baolin Peng, Zheng Zhang, Shutong Feng, Michael Heck, Nurul Lubis, Dazhen Wan, and 1 others. 2023. Convlab-3: A flexible dialogue system toolkit based on a unified data format. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 106–123.

Qi Zhu, Kaili Huang, Zheng Zhang, Xiaoyan Zhu, and Minlie Huang. 2020. CrossWOZ: A large-scale Chinese cross-domain task-oriented dialogue dataset. *Transactions of the Association for Computational Linguistics*, 8:281–295.

A Appendix

A.1 Prompt Example for Data Augmentation Framework

In this section, we provide examples of prompt settings based on a sample full-human dialogue for all components of the data augmentation framework, including eight tables that illustrate sample prompts for each of our proposed data augmentation frameworks.

Table 8 shows the goal and original Full-Human dialogue of our example. Table 9 and Table 10 show the Llama 70B structured prompt example for Missing Sentence Completion with a consist of Task, Slot and Domain Knowledge, Chain of thought and Chat history with missing sentences. Table 11 shows the example prompt for GPT-3.5 used in the Missing Sentence Completion framework. Table 12 provides an example prompt for Next Response Generation, which follows the same structure for both GPT-3.5 and Llama 70B. Table 13 shows the two staged prompt for Par. framework. Table 14 and Table 15 show the example prompt for G2D framework. Table 16 and Table 17 show an example prompt for GPT-3.5 acting as the user or system in the E2E Convo. framework. Table 18, Table 19 and Table 20 show an example prompt for Llama 70B acting as the user or system in the E2E Convo. framework.

A.2 Multiclass Classification

Table 5 presents the macro F1 scores of detection models on multiclass classification task, which classifies dialogues as generated by human or individual LLMs. Using the new datasets, the labels include "Human", "GPT", and "Llama". Zero-shot detection methods such as Binoculars (Hans et al., 2024) targets binary classification only, and is not applicable to the multiclass detection task.

By collapsing all synthetic dialogue sources into a single "AI" category, binary classification performance (reported in Table 3) consistently outperformed the multiclass classification performance across the same models. For instance, XGBoost, LogR, and SVM all showed higher F1 scores in binary classification than multiclass classification, where distinguishing between multiple LLMs proved more difficult.

A.3 Full-Chatbot Quality Assurance

For Full-Chatbot dialogues, we additionally measure the degree of match between the goal and the

Table 5: Macro-F1 of multiclass detection models. The highest score of each detection task is in bold.

		Statistical		PLM	Feature				
Dataset	**Detection**	**Entropy**	**Binoculars**	**RoBERTa**	**MLP**	**XGboost**	**LogR**	**SVM**	**RF**
Par.	Multi-class	0.5394	-	0.9003	**0.9483**	0.8862	0.9245	0.9352	0.9112
G2D	Multi-class	0.5985	-	0.9710	**0.9913**	0.9564	0.9835	0.9809	0.9754
E2E Convo.	Multi-class	0.6019	-	0.9708	**0.9851**	0.9561	0.9755	0.9756	0.9634
Next Response Generation	Multi-class	0.5096	-	0.8962	**0.9161**	0.8404	0.8990	0.8916	0.8591

generated dialogue using automated and human survey-based methods. This aims to ensure our generated dialogue does not suffer from the mismatch issues in the original Full-Human dataset as identified in Section 4.1.

A.3.1 Automated Quality Assurance

This automated goal-dialogue match assurance combined the use of the pre-trained dialogue state tracking model (DST) (Zhu et al., 2023) and the pre-trained sentence similarity model (Reimers and Gurevych, 2019). We input the dialogue and goal separately into the DST model and compare their domain and slot values results using the sentence similarity model, deriving a goal dialogue match score.

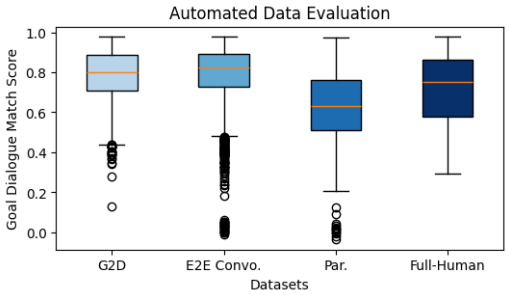


Figure 4: Comparison of goal-dialogue match scores from Automated Quality Assurance for dialogues produced by G2D, E2E Convo., Par., and Original Full-Human.

Figure 4 shows a boxplot to compare the goal dialogue match score. The result indicates that dialogues generated by G2D and E2E Convo. data augmentation methods have a high average goal dialogue match score of approximately 0.8, while the dialogues generated by the Par. framework show a slightly lower score. This may be due to the framework not providing a dialogue goal in the prompt. To further the findings, we conducted a human survey and asked participants to rank the synthetic dialogues according to the degree of match. Details of the experiments can be found in Appendix A.3.2. In general, our generated synthetic dialogues are of high quality, as the goal-dialogue match score is similar to that of the Full-Human dialogue dataset, averaging 0.8.

A.3.2 Manual Quality Assurance

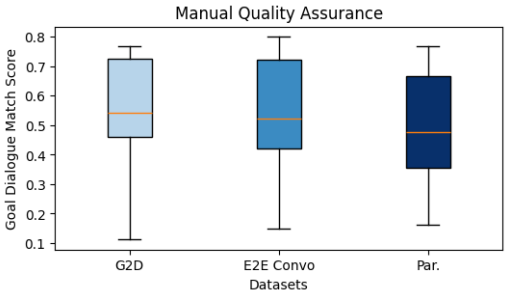


Figure 5: The boxplot comparison using goal-dialogue match scores from Manual Quality Assurance for dialogues generated from G2D, E2E Convo. and, Par..

In the process of Manual Quality Assurance we employ 15 volunteers to complete a survey. The human survey template is shown in Figure 7, with three main parts: the goal of the research, the task, and the steps. The survey includes 10 unique goals and 50 synthetic dialogues generated from different data augmentation frameworks. For each unique goal there are 5 dialogues generated using varies data augmentation methods. We inquiry volunteers to complete two tasks: (i) Decide if the given goal match the dialogue by answering 'Yes' or 'No'. (ii) Ranking 5 dialogues under the same goal according to the degrees of match. During the processing of survey the participants do not know which dialogue is generated from which data augmentation framework, and were ask to filling a google form to complete the survey.

To define the metric, we calculate the match rate for each dialogue written as:

$$\frac{C_{yes}}{N_p}$$

where C_{yes} denotes the number of 'yes' responses for the first task, N_p denotes number of participates.

The participants with different highest education levels at university level, which include 5 bachelor students, 8 master students and 2 phd students

Then, we calculate the weighted ranking score for each types of data augmentation framework using the formula:

$$\frac{\Sigma_{i=1}^{N_p} R_i}{5 \cdot N_p}$$

where R represents the ranking score for each dialogue. If participants believe that a dialogue best matches the goal, it is recorded as 5, which is the highest score a single dialogue can earn. We multiply the two results calculated above to get a final score to represent the degree of goal dialogue match.

Figure 5 illustrates the goal dialogue match scores for synthetic dialogues generated from our data augmentation frameworks. It shows that Paraphrase dialogues are more likely to be treated as synthetic dialogues with the lowest degree of match between the goal and the generated dialogue, compared to E2E Convo. or G2D dialogues. This supports our conclusion in Appendix A.3.1 that Paraphrase dialogues have relatively lower goal dialogue match scores.

A.4 Prompt Exchange Experiment

We conducted a prompt exchange experiment to compare the generation capabilities of GPT-3.5 and Llama 70B (Section 4.2). In our experiment, we generated synthetic dialogues using the augmentation frameworks defined in Section 3, by exchanging prompts between GPT-3.5 and Llama 70B for 20 randomly selected dialogues. For qualitative analysis, we used the end-to-end conversation data augmentation framework as an example. We use Llama 70B user prompt for GPT-3.5 user and conduct E2E Convo. with GPT-3.5 system with true GPT-3.5 system prompt and GPT-3.5 user with Llama 70B user prompt. The generation results show that when a prompt designed for Llama 70B is used with GPT-3.5, the conversation tends to exhibit more formatting issues, such as incorrect information masking and unexpected conversation stops. In contrast, using a GPT-3.5 prompt for Llama 70B leads to fewer formatting issues, but the model becomes more information-hungry and rigidly sticks to the goal, as the system version goal is used in the GPT-3.5 prompt.

For qualitative analysis, we take the goal from Table 8 as an example. The goal of the user is to book a hotel for 6 people for 3 nights. When using the GPT-3.5 prompt structure on Llama 70B, the output response from the synthetic system will provide information about the "6 people for 3 nights" before the user provides that information. Sample questions include, "I've found a [hotel name] with availability for 6 people for 3 nights; would you like me to confirm the booking?" Since Llama 70B was provided with a system version goal that was modified from the original user goal, it leads to Llama 70B rigidly sticking to the goal when using a GPT-3.5 prompt.

A.5 Qualitative Analysis for Offline Dialogue Detection

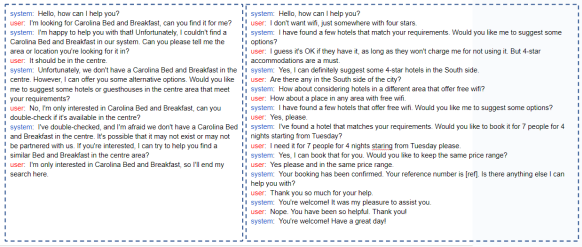


Figure 6: Left: Dialogue with fewer than 5 user utterances, which is easier to misclassification. Right: Dialogue with more than 5 user utterances, typically classified correctly.

In Section 5, we conduct experiments on synthetic user detection using different models. In this Appendix, we present a sample case of a typical misclassification scenario. From Figure 6, the left-hand side dialogue is a sample with fewer than 5 user utterances, while the right-hand side dialogue is a sample with more than 5 user utterances. After comparing the misclassified dialogues, we found that errors mostly occurred when the generated dialogues included fewer utterances. These dialogues were sometimes misclassified due to the limited information provided. On the other hand, dialogues with more than five user utterances were seldom misclassified. This suggests that having more chat history provides a better learning base for synthetic user dialogue detection, which also indicates the potential limitation of early detection with limited chat history.

A.6 Qualitative Analysis for Online Dialogue Detection

In this section we provide an example of misclassification when there is only one user utterance available. When LLM-users start a conversation, their opening sentence can be similar to human users, making the detection harder. For example, users attempting to find a particular hotel in the

E2E Convo. dataset is commonly misclassified by RoBERTa when only 1 utterance is available, but correctly classified afterwards. Here is a llama-user classified as human: "user: Hello, I'm looking for Bridge Guest House. Can you help me find it?" A possible reason is that such utterances contain minimal information, and is very similar to how humans start similar dialogues. We can identify opening sentences made by human users that exhibit small differences to the misclassified example: "user: Hello, I am looking for a hotel called the worth house. Can you help me find it?" This reinforces the limitation of early synthetic dialogue detection in an online environment, as introduced in Section A.5.

A.7 Online Dialogue Detection Enhancement

Table 6: Model Performance on Par., G2D, E2E Convo. with stacking or expanded dataset strategies. Performance on exactly the number of user utterances specified, and dialogues with fewer user utterances are excluded.

		Stack	Expanded Dataset	
Dataset	**#Utterance**	Stack	$ExpPro_{MLP}$	$ExpPro_{RoBERTa}$
Par.	1	0.6908	**0.7479**	0.6959
	2	0.8170	0.8286	**0.8612**
	3	0.8855	0.8085	**0.8919**
	4	**0.9231**	0.9130	0.8927
G2D	1	0.8865	0.7577	**0.9028**
	2	0.8997	0.6703	**0.9238**
	3	**0.9382**	0.9197	0.9263
	4	**0.9668**	0.9644	0.9533
E2E Convo.	1	0.9373	0.8028	**0.9792**
	2	0.9670	0.8542	**0.9901**
	3	0.9874	0.8667	**0.9944**
	4	0.9923	0.8373	**0.9996**

We found two potential frameworks that are previously used in a similar situation to enhance the detection performance on incomplete dialogues, which are stacking, and dataset expansion.

A.7.1 Stacking

Feature based models M_F and $RoBERTa$ model receive different input and features for detection, and it is hypothesized that more comprehensive detection can be made when the two approaches are combined via a stacking classifier M_S. The following procedure aims to train a system with enhanced performance on detection given n user utterances:

- Given training dataset D_T and labels y_T
- Train base classifiers M_F and $RoBERTa$ on D_T and y_T
- Randomly sample $D'_T \subseteq D_T$ and extract the first n user utterances from each dialogue to form a stacking dataset D_S with labels y_S
- M_F and $RoBERTa$ make inferences on D_S, constructing P_S which consists of $RoBERTa$ predictions (average of per-word probabilities in the utterances) and M_F logits
- Train M_S on P_S and y_S

Then to detect a dialogue d_i of n user utterances:

- Base classifiers M_F and $RoBERTa$ make inferences on d_i, giving p_i consisting of $RoBERTa$ predictions and M_F logits
- M_S makes inferences on p_i to produce final prediction

A.7.2 Dataset Expansion Approach

We expand the training dataset for progressive dialogue detection via $ExpPro$, specifically, for each complete dialogue d_i with N_i turns, crop the dialogue at first $k \in \mathbb{Z}^+ < N_i$ turns and include them into the training dataset.

The method is applied on the feature-based MLP and pre-trained RoBERTa, which are the top-performing classifiers.

A.7.3 Experiment and Results

We conducted our experiments using robust performance models MLP and RoBERTa, as indicated in Section 5, based on the two defined methods described above. We performed comparison experiments on E2E Convo. dataset with progressive dialogue, using a varying number of user utterances as input, having label of either 'human' or 'AI'.

In our experiments, we utilize user utterances ranging from 1 to 4, as more than half of the dialogues do not contain more than 4 user utterances, which would not represent the majority of cases. In Table 6, the F1 scores of each experiment are recorded. We found that RoBERTa, trained on expanded datasets that include all possible cases of chat history, demonstrated the best performance, showing a significant improvement compared to the original results trained on full dialogues and tested on progressive dialogues. The results indicate that common performance improvement methods are effective for progressive dialogue detection.

A.8 Out-of-Distribution Detection

To further demonstrate that integrating multiple augmentation frameworks enhances generalizability, experiments are conducted using subsets of two external datasets featuring customer service

Table 7: RoBERTa's Macro-F1 performance on SGD and CrossWOZ datasets augmented using G2D framework, when trained on different datasets based on MultiWOZ dataset.

Train Dataset	SGD	CrossWOZ
P	0.3333	0.3333
E	0.5016	0.5238
G	0.4286	0.3960
Single Average	0.4212	0.4177
P + E	0.6328	0.5572
P + G	0.7971	0.4956
E + G	0.3984	0.5696
Combination Average	0.6094	0.5408

line dialogues with human user: SGD (Rastogi et al., 2020) and CrossWOZ (Zhu et al., 2020). In addition, a different augmentor, Gemini 1.5 (Team et al., 2024), is employed to perform the G2D augmentation for synthetic user dialogue generation. A RoBERTa model is trained on the datasets based on MultiWoz and tested on the additional datasets. During the experiments, we control all training set sizes (1232 dialogues) when training RoBERTa on a single or a combination of frameworks, ensure consistent testing datasets across training scenarios, and ensure consistent datasets in terms of masked information across positive and negative samples (e.g. [hotel name]), which is a more challenging scenario.

The RoBERTa Macro-F1 results are presented in Table 7. Despite the performance decline caused by differences in the external testing dataset compared to our training dataset, the use of a different LLM, and the fact that the CrossWOZ dataset used is translated into English and typically features short dialogues (fewer than 5 turns), the results across both additional and our own datasets consistently demonstrate that integrating multiple frameworks enhances generalisability.

A.9 Data Augmentation Framework Demonstration

In this section, a demonstration of our proposed data augmentation frameworks is illustrated in Figure 8. The demonstration takes a dialogue as an example and shows the exact process of augmentation for the dialogue system, including all relevant queries in our framework.

Table 8: Example for Full-Human dialogue with goal.

Components	Prompt
Goal	The user is looking for a place to stay. The hotel should be in the cheap price range and should be in the type of hotel. The hotel should include free parking and should include free wifi. Once the user find the hotel the user want to book it for 6 people and 3 nights starting from tuesday. If the booking fails how about 2 nights. Make sure the uer get the reference number.
Chat History	user: am looking for a place to to stay that has cheap price range it should be in a type of hotel. \n system: Okay, do you have a specific area you want to stay in? \n user: no, i just need to make sure it's cheap. oh, and i need parking. \n system: I found 1 cheap hotel for you that includes parking. Do you like me to book it? \n

Table 9: Structured user prompt example for Missing Sentence Completion data augmentation for Llama 70B.

Components	Prompt
Task	Task: Replace each of the "[missing sentence]" in the dialogue.
Slot and Domain Knowledge	"internet": { "description": "whether the hotel has internet", "is_categorical": true, "possible_values": ["free", "no", "yes"] }
	"parking": { "description": "whether the hotel has parking", "is_categorical": true, "possible_values": ["free", "no", "yes"] }, "area": { "description": "area or place of the hotel", "is_categorical": true, "possible_values": ["centre", "east", "north", "south", "west"] } "stars": { "description": "star rating of the hotel", "is_categorical": true, "possible_values": ["0", "1", "2", "3", "4", "5"] } "price range": {"description": "price budget of the hotel", "is_categorical": true, "possible_values": ["expensive", "cheap", "moderate"] } "type": {"description": "what is the type of the hotel", "is_categorical": true, "possible_values": ["guesthouse", "hotel"] } "name": { "description": "name of the hotel", "is_categorical": false, "possible_values": [] } "book people": { "description": "number of people for the hotel booking", "is_categorical": false, "possible_values": [] } "book stay": { "description": "length of stay at the hotel", "is_categorical": false, "possible_values": [] } "book day": { "description": "day of the hotel booking", "is_categorical": true, "possible_values": ["monday", "tuesday", "wednesday", "thursday", "friday", "saturday", "sunday"] } "phone": { "description": "phone number of the hotel", "is_categorical": false, "possible_values": [] } "postcode": { "description": "postcode of the hotel", "is_categorical": false, "possible_values": [] } "address": { "description": "address of the hotel", "is_categorical": false, "possible_values": [] } "ref": { "description": "reference number of the hotel booking", "is_categorical": false, "possible_values": [] } "choice": { "description": "number of hotels that meet the requirement", "is_categorical": false, "possible_values": [] }
Chain of Thought	For each missing sentence, your response should be in format of: turn_id - impact of immediately preceding user sentence - impact of immediately following user sentence (note that the real system only have knowledge up to the missing sentence) - impact of overall previous and following context - one line replacing the missing sentence Afterall, print the completed entire dialogue. Here is a demonstration of the task where response is bounded by ==========:

Table 10: Structured user prompt example for Missing Sentence Completion data augmentation for Llama 70B (continuous Table 9).

Components	Prompt
Chain of Thought	user: I'm looking for a hotel to stay at in the centre, can you look this up for me? system: [missing sentence] user: Not in terms of that, but do they have free parking and have a 3 star rating? system: [missing sentence] user: Okay, I'd like to book a room at the Gonville Hotel for 4 nights. There will be 6 people and we will be arriving on Saturday. system: [missing sentence] user: Yes, what about 2 nights instead of 4? system: [missing sentence] user: No, that looks like everything. Thanks. Bye. system: [missing sentence] ========== 1. - the user asks the system to look up a hotel in the centre, system should respond to this query - "not in terms of that" seems to be responding to the system's suggestion which the user does not care about (e.g. pricerange, free wifi that are not required by user anywhere in the dialogue). the user asks if "they" have free parking and 3 star rating, which means the system should provid some hotel suggestions, but NOT parking and star information! - later in the chat the user mentioned Gonville hotel which the system has likely suggested to them. If suggesting particular hotels, it is likely one of them. - There are three hotels in the center of town. Do you prefer something moderate or expensive? 2. - the user asked if the suggested hotels has free parking and is 3 star which the system must respond to - the user replies "okay" to the system's suggestion and provided details of their booking at Gonville Hotel. The system has likely suggested Gonville Hotel and asked if the user wish to make a booking - NA - The Gonville hotel has 3 stars and parking, and the University Arms hotel has 4 stars and parking. They are both expensive. Would you like more details? 3. - booking details at a particular hotel has been provided by the user, the system can attempt to make a booking - the user says 'yes' to system and then shortend the stay to 2 nights. Which means the system was potentially unable to make booking as initially required and suggested to shorten the days. - Book day, book stay, book people must be provided to make a booking. In case of failed booking, system can suggest to change length or time of stay, or suggest another hotel that satisfy all requirement, or ask user to relax a previously stated requirement (in this case 3 star, free parking, area) - I'm sorry, there are no rooms available for that length of stay. Could you shorten your stay or book a different day possibly? 4. - the user agreed to shorten the stay - the user says 'no' to system and claims that they are all set and ended the conversation. The system has likely successfully make a booking and asked if user need anything else. - NA - Sure, that worked. You have booked 2 nights and your reference number is RU89U6V8. Can I be of further help today? 5. - the user expressed appreciation and ended the conversation - no more user sentence. - user has ended conversation. system should end conversation - You're welcome. Enjoy your stay! Completed dialogue: user: I'm looking for a hotel to stay at in the centre, can you look this up for me? system: There are three hotels in the center of town. Do you prefer something moderate or expensive? user: Not in terms of that, but do they have free parking and have a 3 star rating? system: The Gonville hotel has 3 stars and parking, and the University Arms hotel has 4 stars and parking. They are both expensive. Would you like more details? user: Okay, I'd like to book a room at the Gonville Hotel for 4 nights. There will be 6 people and we will be arriving on Saturday. system: I'm sorry, there are no rooms available for that length of stay. Could you shorten your stay or book a different day possibly? user: Yes, what about 2 nights instead of 4? system: Sure, that worked. You have booked 2 nights and your reference number is RU89U6V8. Can I be of further help today? user: No, that looks like everything. Thanks. Bye. system: You're welcome. Enjoy your stay! ==========
Chat history	Here is the dialogue of your task: user: am looking for a place to to stay that has cheap price range it should be in a type of hotel. \n System: [missing sentence] \n user: no, i just need to make sure it's cheap. oh, and i need parking. \n System: [missing sentence] \n

Table 11: Structured user prompt example for Missing Sentence Completion data augmentation for GPT3.5.

Components	Prompt
Goal	goal: The user is looking for a place to stay. The hotel should be in the cheap price range and should be in the type of hotel. The hotel should include free parking and should include free wifi. Once the user find the hotel the user want to book it for 6 people and 3 nights starting from tuesday. If the booking fails how about 2 nights. Make sure the uer get the reference number. \n
Chat History	dialogue: user: am looking for a place to to stay that has cheap price range it should be in a type of hotel. \n System: [missing sentence] \n user: no, i just need to make sure it's cheap. oh, and i need parking. \n System: [missing sentence] \n
Task	Replace all the "[missing sentence]" in the dialogue. please output the entire dialogue.

Table 12: Structured user prompt example for Next Response Generation data augmentation.

Components	Prompt
Task	Task: Generate the next user response according to the given goal and chat history. Your response must start with 'user:'! \n
Goal	Goal: The user is looking for a place to stay. The hotel should be in the cheap price range and should be in the type of hotel. The hotel should include free parking and should include free wifi. Once the user find the hotel the user want to book it for 6 people and 3 nights starting from tuesday. If the booking fails how about 2 nights. Make sure the uer get the reference number. \n
Chat History	Chat history: user: am looking for a place to to stay that has cheap price range it should be in a type of hotel. \n system: Okay, do you have a specific area you want to stay in? \n user: no, i just need to make sure it's cheap. oh, and i need parking. \n system: I found 1 cheap hotel for you that includes parking. Do you like me to book it? \n

Table 13: Structured user prompt example for Par. data augmentation.

Stages	Components	Prompt
Stage 1	Task Summary	A customer and a server line assistant are in dialogue. Replace each existing system response with a response you would have said if you were the system. Ensure the new responses logically follow the preceding dialogue and lead naturally into the unchanged user responses. The output should remain the same format as the dialogue!\n
	Dialogue	user: am looking for a place to to stay that has cheap price range it should be in a type of hotel. \n system: Okay, do you have a specific area you want to stay in? \n user: no, i just need to make sure it's cheap. oh, and i need parking. \n system: I found 1 cheap hotel for you that includes parking. Do you like me to book it? \n
Stage 2	Dialogue	here is the chat history: user: am looking for a place to to stay that has cheap price range it should be in a type of hotel. \n system: Sure thing! Are you looking for a specific area or just anywhere with affordable prices? \n user: no, i just need to make sure it's cheap. oh, and i need parking. \n system: I've found a hotel that fits your criteria - it's budget-friendly and offers parking. Would you like me to proceed with the booking? \n
	Task Summary	please rewrite each sentence starting with 'user:'. do not copy and paste, you should rephrase it! you should add the exactly unchanged system response to form the output.

Table 14: Structured user prompt example for G2D data augmentation part 1.

Components	Prompt
Task Summary	Task: create a dialogue between a user and a system. \n
Example Dialogue	Here is a demonstration for the dialogue you are going to generate: \n user: Hello, I'm looking for a guesthouse on the westside to stay. \n system: I am sorry I have no guest houses in the west. Can I look one up in another area for you? \n user: Are there any other places like that that also have free parking and free wifi in that area? \n system: I have found one that is cheap with 4 stars, and one that is moderately priced with 3 stars? Do either of those sound appealing to you? \n user: The cheap one with the better rating sounds appealing. \n system: Great! That's Finches Bed and Breakfast. It's located at 144 Thornton Road cb30nd. Would you like to book a stay? \n user: Please book it for Friday,thank you. \n system: Can you please tell me how many people will be staying and for how many nights? \n user: Yes, 3 people for 5 nights starting on monday. \n system: Booking was unsuccessful. Want to try another hotel? \n user: Yes, in the same price range please. \n system: Worth House in the North is 4 stars and cheap. Would you like to try it? \n user: The amount of stars do not matter as long as it is in the same price range. Does Worth House in the North include free parking and internet? \n system: Okay well I was able to book you at The Worth House and your reference number is 86XVFEUY. \n user: Thanks, that's all I need. Have a nice day. \n system: You are welcome. Also for your reference the Worth House does have free parking and wifi. May I help you with anything else? \n user: No thanks. Thanks again for your help. \n system: Enjoy your stay! \n
Goal-Specific Instructions for User	For the dialog that you have to generate in this Section, the instructions for the "user" are the following: The user is looking for a place to stay. The hotel should be in the cheap price range and should be in the type of hotel. The hotel should include free parking and should include free wifi. Once the user find the hotel the user want to book it for 6 people and 3 nights starting from tuesday. If the booking fails how about 2 nights. Make sure the uer get the reference number. Every user message should be followed by a system message. Be polite and don't forget to say goodbye.
Slot and Domain Knowledge for System	For the dialog that you have to generate in this section, the instructions for the "system" are the following: "internet": { "description": "whether the hotel has internet", "is_categorical": true, "possible_values": ["free", "no", "yes"] } "parking": { "description": "whether the hotel has parking", "is_categorical": true, "possible_values": ["free", "no", "yes"] }, "area": { "description": "area or place of the hotel", "is_categorical": true, "possible_values": ["centre", "east", "north", "south", "west"] } "stars": { "description": "star rating of the hotel", "is_categorical": true, "possible_values": ["0", "1", "2", "3", "4", "5"] } "price range": {"description": "price budget of the hotel", "is_categorical": true, "possible_values": ["expensive", "cheap", "moderate"] } "type": {"description": "what is the type of the hotel", "is_categorical": true, "possible_values": ["guesthouse", "hotel"] } "name": { "description": "name of the hotel", "is_categorical": false, "possible_values": [] } "book people": { "description": "number of people for the hotel booking", "is_categorical": false, "possible_values": [] } "book stay": { "description": "length of stay at the hotel", "is_categorical": false, "possible_values": [] } "book day": { "description": "day of the hotel booking", "is_categorical": true, "possible_values": ["monday", "tuesday", "wednesday", "thursday", "friday", "saturday", "sunday"] } "phone": { "description": "phone number of the hotel", "is_categorical": false, "possible_values": [] } "postcode": { "description": "postcode of the hotel", "is_categorical": false, "possible_values": [] } "address": { "description": "address of the hotel", "is_categorical": false, "possible_values": [] } "ref": { "description": "reference number of the hotel booking", "is_categorical": false, "possible_values": [] } "choice": { "description": "number of hotels that meet the requirement", "is_categorical": false, "possible_values": [] } Domain of knowledge needed (include everything is not mandatory): (parking, area, star rating, price range, type of hotel, name of hotel, book people number, length of stay, book day, phone number, postcode, address of hotel, reference number, number of hotel meet requirement)

Table 15: Structured user prompt example for G2D data augmentation part 2.

Components	Prompt
Conversation Termination Conditions	please generate a dialogue according to the instructions. If you achieve your goal, express your thanks and generate **[END]** token. If you think the assistant can not help you or the conversation falls into a infinite loop, generate **[STOP]** token.
Sensitive Information Masking	Please mask the following information in the generated dialogue: (name of hotel as [hotel name], phone number as [phone number], postcode as [postcode], address of hotel as [address], reference number as [ref]).

Table 16: Structured user prompt example for E2E Convo. data augmentation using GPT3.5.

Components	Prompt
Task Summary	Task: act as a user communicating with a system.
Example Dialogue	Here is a demonstration for the dialogue you are going to generate: \n user: Hello, I'm looking for a guesthouse on the westside to stay. \n system: I am sorry I have no guest houses in the west. Can I look one up in another area for you? \n user: Are there any other places like that that also have free parking and free wifi in that area? \n system: I have found one that is cheap with 4 stars, and one that is moderately priced with 3 stars? Do either of those sound appealing to you? \n user: The cheap one with the better rating sounds appealing. \n system: Great! That's Finches Bed and Breakfast. It's located at 144 Thornton Road cb30nd. Would you like to book a stay? \n user: Please book it for Friday,thank you. \n system: Can you please tell me how many people will be staying and for how many nights? \n user: Yes, 3 people for 5 nights starting on monday. \n system: Booking was unsuccessful. Want to try another hotel? \n user: Yes, in the same price range please. \n system: Worth House in the North is 4 stars and cheap. Would you like to try it? \n user: The amount of stars do not matter as long as it is in the same price range. Does Worth House in the North include free parking and internet? \n system: Okay well I was able to book you at The Worth House and your reference number is 86XVFEUY. \n user: Thanks, that's all I need. Have a nice day. \n system: You are welcome. Also for your reference the Worth House does have free parking and wifi. May I help you with anything else? \n user: No thanks. Thanks again for your help. \n system: Enjoy your stay! \n
Role-Specific Instructions (User)	For the dialog that you have to generate in this Section, the instructions for the "user" are the following: The user is looking for a place to stay. The hotel should be in the cheap price range and should be in the type of hotel. The hotel should include free parking and should include free wifi. Once the user find the hotel the user want to book it for 6 people and 3 nights starting from tuesday. If the booking fails how about 2 nights. Make sure the user get the reference number. Every user message should be followed by a system message. Be polite and don't forget to say goodbye. \n
Role-Specific Instructions (System)	I will be the system. \n
Conversation Termination Conditions	please generate a dialogue according to the goal. If you achieve your goal (booking sucessful or find the hotel), express your thanks and generate **[END]** token. If you think the assistant can not help you or the conversation falls into a infinite loop, generate **[STOP]** token. \n
Sensitive Information Masking	please mask the following information in the generated dialogue: (name of hotel as [hotel name], phone number as [phone number], postcode as [postcode], address of hotel as [address], reference number as [ref]). \n The output user response should be in the format of "user:...". \n It should be only one sentence.

Table 17: Structured system prompt example for E2E Convo. data augmentation using GPT3.5.

Components	**Prompt**
Task Summary	Task: act as a system communicating with a user.
Example Dialogue	Here is a demonstration for the dialogue you are going to generate: \n user: Hello, I'm looking for a guesthouse on the westside to stay. \n system: I am sorry I have no guest houses in the west. Can I look one up in another area for you? \n user: Are there any other places like that that also have free parking and free wifi in that area? \n system: I have found one that is cheap with 4 stars, and one that is moderately priced with 3 stars? Do either of those sound appealing to you? \n user: The cheap one with the better rating sounds appealing. \n system: Great! That's Finches Bed and Breakfast. It's located at 144 Thornton Road cb30nd. Would you like to book a stay? \n user: Please book it for Friday,thank you. \n system: Can you please tell me how many people will be staying and for how many nights? \n user: Yes, 3 people for 5 nights starting on monday. \n system: Booking was unsuccessful. Want to try another hotel? \n user: Yes, in the same price range please. \n system: Worth House in the North is 4 stars and cheap. Would you like to try it? \n user: The amount of stars do not matter as long as it is in the same price range. Does Worth House in the North include free parking and internet? \n system: Okay well I was able to book you at The Worth House and your reference number is 86XVFEUY. \n user: Thanks, that's all I need. Have a nice day. \n system: You are welcome. Also for your reference the Worth House does have free parking and wifi. May I help you with anything else? \n user: No thanks. Thanks again for your help. \n system: Enjoy your stay! \n
Role-Specific Instructions (System)	for this dialogue, you are the system, here is the goal for system: Do not copy anything from the demonstration.please do not repeat yourself.Note that you should not make booking unless the goal explicitly mentioned a booking.you can only use the information provided in chat history.you can only generate one sentence each time. \n
system version Goal	The system needs to find a hotel in the cheap price range, with the type specified as "hotel." The hotel must offer free parking and free Wi-Fi. Once a suitable hotel is found, the system should proceed to book it for 6 people for 3 nights, starting from Tuesday. If the booking fails, the system should attempt to book for 2 nights. The system must ensure that the user receives a reference number for the booking.
Role-Specific Instructions (User)	I will be the user. \n
Conversation Termination Conditions	please generate a dialogue according to the goal. If you achieve your goal (booking sucessful or find the hotel), express your thanks and generate **[END]** token. If you think the assistant can not help you or the conversation falls into a infinite loop, generate **[STOP]** token. \n
Sensitive Information Masking	please mask the following information in the generated dialogue: (name of hotel as [hotel name], phone number as [phone number], postcode as [postcode], address of hotel as [address], reference number as [ref]). \n The output user response should be in the format of "user:...". \n It should be only one sentence.

Table 18: Structured user prompt example for E2E Convo. data augmentation using Llama 70B part 1.

Components	**Prompt**
Task Summary	Task: Simulate as an user with a particular goal and generate one response to a hotel service system. Response must start with "user:". After you achieved all your goals, end the conversation and generate "[END]" token. If you think the system cannot help you or the conversation falls into an infinite loop, generate a "[STOP]" token. The response must be one line only!
Slot and Domain Knowledge for System	The information you can ask for or provide include: "internet": { "description": "whether the hotel has internet", "is_categorical": true, "possible_values": ["free", "no", "yes"] } "parking": { "description": "whether the hotel has parking", "is_categorical": true, "possible_values": ["free", "no", "yes"] }, "area": { "description": "area or place of the hotel", "is_categorical": true, "possible_values": ["centre", "east", "north", "south", "west"] } "stars": { "description": "star rating of the hotel", "is_categorical": true, "possible_values": ["0", "1", "2", "3", "4", "5"] } "price range": {"description": "price budget of the hotel", "is_categorical": true, "possible_values": ["expensive", "cheap", "moderate"] } "type": {"description": "what is the type of the hotel", "is_categorical": true, "possible_values": ["guesthouse", "hotel"] } "name": { "description": "name of the hotel", "is_categorical": false, "possible_values": [] } "book people": { "description": "number of people for the hotel booking", "is_categorical": false, "possible_values": [] } "book stay": { "description": "length of stay at the hotel", "is_categorical": false, "possible_values": [] } "book day": { "description": "day of the hotel booking", "is_categorical": true, "possible_values": ["monday", "tuesday", "wednesday", "thursday", "friday", "saturday", "sunday"] } "phone": { "description": "phone number of the hotel", "is_categorical": false, "possible_values": [] } "postcode": { "description": "postcode of the hotel", "is_categorical": false, "possible_values": [] } "address": { "description": "address of the hotel", "is_categorical": false, "possible_values": [] } "ref": { "description": "reference number of the hotel booking", "is_categorical": false, "possible_values": [] } "choice": { "description": "number of hotels that meet the requirement", "is_categorical": false, "possible_values": [] } Information with "mask_token" specified must be replaced by corresponding token in your response, unless it is provided by the system or in your goal. Do not ask for or provide other information. You do not need to confirm details with the system unless it is ambiguous.
Example Dialogue	Here is a demonstration partial dialogue unrelated to your own goal:\n user: Hello, I'm looking for a guesthouse on the westside to stay. \n system: I am sorry I have no guest houses in the west. Can I look one up in another area for you? \n user: Are there any other places like that that also have free parking and free wifi in that area? \n system: I have found one that is cheap with 4 stars, and one that is moderately priced with 3 stars? Do either of those sound appealing to you? \n user: The cheap one with the better rating sounds appealing. \n system: Great! That's Finches Bed and Breakfast. It's located at 144 Thornton Road cb30nd. Would you like to book a stay? \n user: Please book it for Friday,thank you. \n system: Can you please tell me how many people will be staying and for how many nights? \n user: Yes, 3 people for 5 nights starting on monday. \n system: Booking was unsuccessful. Want to try another hotel? \n user: Yes, in the same price range please. \n system: Worth House in the North is 4 stars and cheap. Would you like to try it? \n user: The amount of stars do not matter as long as it is in the same price range. Does Worth House in the North include free parking and internet? \n system: Okay well I was able to book you at The Worth House and your reference number is 86XVFEUY. \n user: Thanks, that's all I need. Have a nice day. \n system: You are welcome. Also for your reference the Worth House does have free parking and wifi. May I help you with anything else? \n user: No thanks. Thanks again for your help. \n system: Enjoy your stay! \n Do not copy anything from the demonstration! \n

Table 19: Structured user prompt example for E2E Convo. data augmentation using Llama 70B part 2.

Components	**Prompt**
Role-Specific Instructions (User)	Here is your goal: \n The user is looking for a place to stay. The hotel should be in the cheap price range and should be in the type of hotel. The hotel should include free parking and should include free wifi. Once the user find the hotel the user want to book it for 6 people and 3 nights starting from tuesday. If the booking fails how about 2 nights. Make sure the user get the reference number. Note that you should not make booking unless the goal explicitly mentioned a booking. Do not ask for or provide information not specified in the goal. If you are looking for a specific hotel which cannot be found, and the goal does not specify alternative action, end the conversation.

Table 20: Structured prompt system example for E2E Convo. data augmentation using Llama 70B.

Components	**Prompt**
Task Summary	Task: Simulate as a hotel service system and generate one response to a user. Response must start with ”system:”. If and only if the user has no more queries or generated ”[END]”, end the conversation and generate ”[END]” token. If you think the conversation falls into an infinite loop, generate a ”[STOP]” token.
Slot and Domain Knowledge for System	The information you can ask for or provide include: ”internet”: { ”description”: ”whether the hotel has internet”, ”is_categorical”: true, ”possible_values”: [”free”, ”no”, ”yes”] } ”parking”: { ”description”: ”whether the hotel has parking”, ”is_categorical”: true, ”possible_values”: [”free”, ”no”, ”yes”] }, ”area”: { ”description”: ”area or place of the hotel”, ”is_categorical”: true, ”possible_values”: [”centre”, ”east”, ”north”, ”south”, ”west”] } ”stars”: { ”description”: ”star rating of the hotel”, ”is_categorical”: true, ”possible_values”: [”0”, ”1”, ”2”, ”3”, ”4”, ”5”] } ”price range”: {”description”: ”price budget of the hotel”, ”is_categorical”: true, ”possible_values”: [”expensive”, ”cheap”, ”moderate”] } ”type”: {”description”: ”what is the type of the hotel”, ”is_categorical”: true, ”possible_values”: [”guesthouse”, ”hotel”] } ”name”: { ”description”: ”name of the hotel”, ”is_categorical”: false, ”possible_values”: [] } ”book people”: { ”description”: ”number of people for the hotel booking”, ”is_categorical”: false, ”possible_values”: [] } ”book stay”: { ”description”: ”length of stay at the hotel”, ”is_categorical”: false, ”possible_values”: [] } ”book day”: { ”description”: ”day of the hotel booking”, ”is_categorical”: true, ”possible_values”: [”monday”, ”tuesday”, ”wednesday”, ”thursday”, ”friday”, ”saturday”, ”sunday”] } ”phone”: { ”description”: ”phone number of the hotel”, ”is_categorical”: false, ”possible_values”: [] } ”postcode”: { ”description”: ”postcode of the hotel”, ”is_categorical”: false, ”possible_values”: [] } ”address”: { ”description”: ”address of the hotel”, ”is_categorical”: false, ”possible_values”: [] } ”ref”: { ”description”: ”reference number of the hotel booking”, ”is_categorical”: false, ”possible_values”: [] } ”choice”: { ”description”: ”number of hotels that meet the requirement”, ”is_categorical”: false, ”possible_values”: [] } Information with “mask_token” specified must be replaced by corresponding token in your response, unless it is provided by the system or in your goal. Do not ask for or provide other information. You do not need to confirm details with the system unless it is ambiguous.
Example Dialogue	Here is a demonstration partial dialogue unrelated to your own goal:\n user: Hello, I’m looking for a guesthouse on the westside to stay. \n system: I am sorry I have no guest houses in the west. Can I look one up in another area for you? \n user: Are there any other places like that that also have free parking and free wifi in that area? \n system: I have found one that is cheap with 4 stars, and one that is moderately priced with 3 stars? Do either of those sound appealing to you? \n user: The cheap one with the better rating sounds appealing. \n system: Great! That’s Finches Bed and Breakfast. It’s located at 144 Thornton Road cb30nd. Would you like to book a stay? \n user: Please book it for Friday,thank you. \n system: Can you please tell me how many people will be staying and for how many nights? \n user: Yes, 3 people for 5 nights starting on monday. \n system: Booking was unsuccessful. Want to try another hotel? \n user: Yes, in the same price range please. \n system: Worth House in the North is 4 stars and cheap. Would you like to try it? \n user: The amount of stars do not matter as long as it is in the same price range. Does Worth House in the North include free parking and internet? \n system: Okay well I was able to book you at The Worth House and your reference number is 86XVFEUY. \n user: Thanks, that’s all I need. Have a nice day. \n system: You are welcome. Also for your reference the Worth House does have free parking and wifi. May I help you with anything else? \n user: No thanks. Thanks again for your help. \n system: Enjoy your stay! \n Do not copy anything from the demonstration! \n

Goal of this research

We are going to determine the degree of match between generated AI dialogues by using different data augmentation methods and the assigned goal.

To do this, we will create dialogue datasets generated from different data augmentation methods.

Please read some conversation texts and identify the degree of match.

Task

There are 10 unique goals. For each unique goal, there are five dialogues generated by large language models.

What you need to do is to answer 2 questions:

1. Does the dialogue match the goal? (Please select "yes" or "no".)
2. Please rank the five dialogues according to the degree of match.

Steps

Step 1: Read the goal and the 5 dialogues under the goal.

You are given a goal like the one below:

Goal: You are looking for a particular hotel called Autumn House. Once you find the hotel, you want to book it for 8 people for 2 nights starting from Monday. If the booking fails, consider Wednesday instead. Make sure to get the reference number.

You are given dialogues like the ones below. Each dialogue under the unique goal will have a name denoted as "Dialogue 1," "Dialogue 2," etc.

Dialogue 1:

- **User:** I am looking for a particular hotel. Its name is called Autumn House.
- **System:** I have found Autumn House. It is located at 710 Newmarket Road. Would you like the phone number?
- **User:** No, thanks. Would you book Autumn House for me starting on Monday, please?
- ...

Step 2: Judge if the goal matches the dialogue and circle the correct answer.

Before each dialogue, you will find a multiple-choice box like this:" [yes, no]." Please select the correct answer.

- **Yes:** The dialogue matches the goal.
- **No:** The dialogue does not match the goal.

[yes, no] Dialogue 1: ...

[yes, no] Dialogue 2: ...

[yes, no] Dialogue 3: ...

[yes, no] Dialogue 4: ...

[yes, no] Dialogue 5: ...

Step 3: Rank the 5 dialogues under the same goal according to the degree of match.

Rank: 1 > 2 > 3 > 4 > 5 (This means you think Dialogue 1 matches the goal best, and Dialogue 5 matches the goal the least.)

Figure 7: Human survey instruction template.

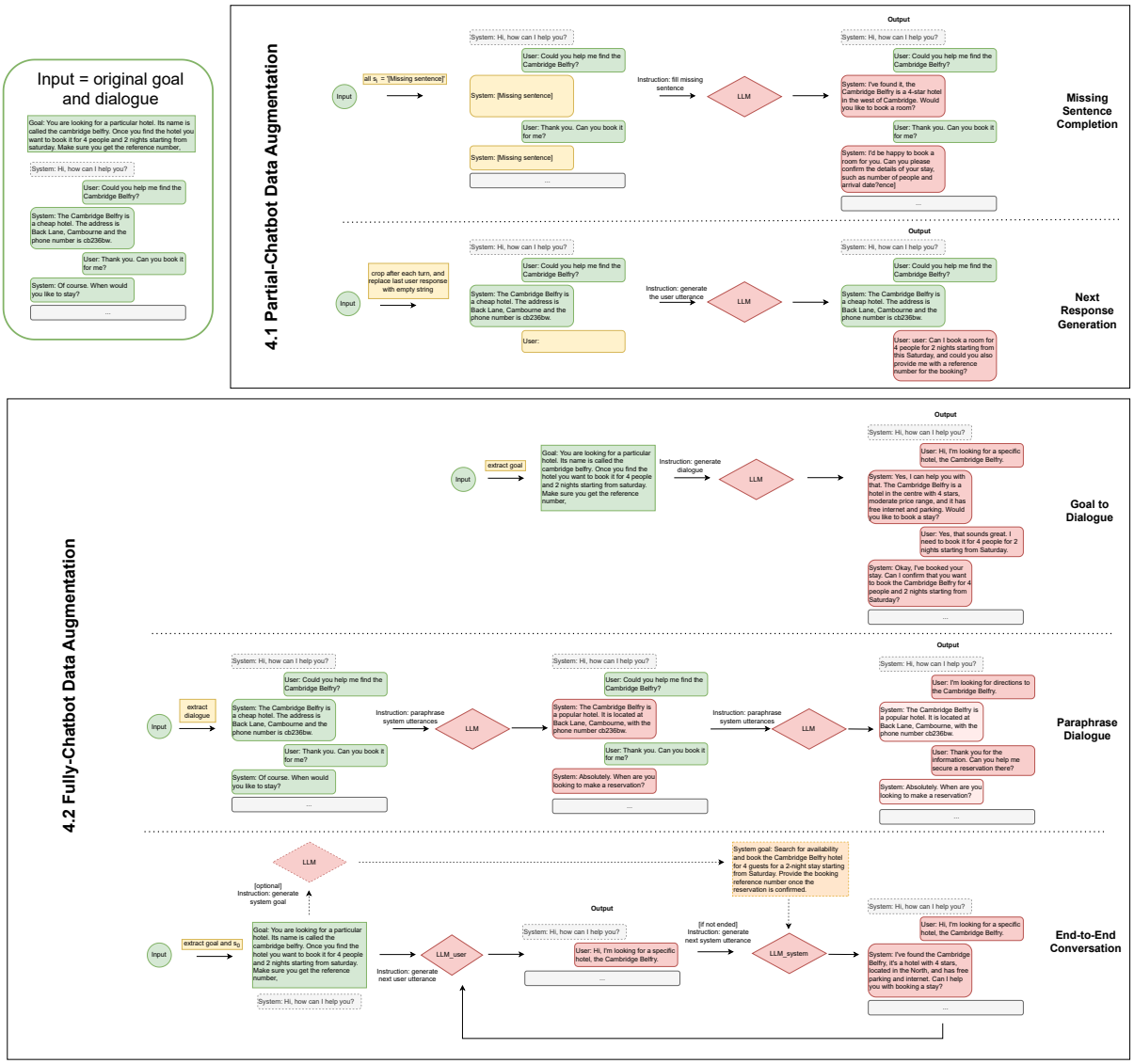


Figure 8: Demonstration of the data augmentation frameworks. The initial system response is set to a standard starting line common to all dialogues. The instructions provided to LLMs as depicted in the figure are incomplete.