Evaluating Task-Oriented Dialogue Consistency through Constraint Satisfaction

Anonymous ACL submission

Abstract

Task-oriented dialogues must maintain consistency both within the dialogue itself, ensuring logical coherence across turns, and with the conversational domain, accurately reflecting external knowledge. We propose to conceptualize dialogue consistency as a Constraint Satisfaction Problem (CSP), wherein variables represent segments of the dialogue referencing the conversational domain, and constraints among variables reflect dialogue properties, including linguistic, conversational, and domain-based aspects. To demonstrate the feasibility of the approach, we utilize a CSP solver to detect inconsistencies in dialogues re-lexicalized by an LLM. Our findings indicate that: (i) CSP is effective to detect dialogue inconsistencies; and (ii) consistent dialogue re-lexicalization is challenging for state-of-the-art LLMs, achieving only a 0.15 accuracy rate when compared to a CSP solver. Furthermore, through an ablation study, we reveal that constraints derived from domain knowledge pose the greatest difficulty in being respected. We argue that CSP captures core properties of dialogue consistency that have been poorly considered by approaches based on component pipelines.

1 Introduction

Task-oriented dialogue (TOD) systems (McTear, 2020; Louvan and Magnini, 2020; Balaraman et al., 2021) play a crucial role in human-computer interaction, facilitating seamless communication between users and machines to accomplish specific tasks. A peculiar characteristic of TODs is that they need to maintain consistency at two levels: (i) internally within the dialogue itself, ensuring that information in a turn is consistent with information in other turns, and (ii) consistency with the content of the conversational domain of the dialogue system. Internal consistency is responsible for the coherence of the dialogue, making it possible to maintain a meaningful exchange between

```
R1: N=Taberna A=centre F=spanish P=cheap
R2: N=Espana A=centre F=spanish P=moderate
R3: N=Beirut A=centre F=lebanese P=cheap
```

U1: I am looking for a restaurant serving Spanish food.

S1: There are three restaurants serving Spanish food, one is **cheap** and the other is **moderate** price range. Which price range would you prefer?

U2: I am looking for a **cheap** restaurant in any area that serves Spanish food.

S2: **Beirut** is **cheap** and serves Lebanese food. Would you like the location information?

Figure 1: An inconsistent task-oriented dialogue with a Knowledge Base. Red values indicate internal inconsistencies, purple values indicate external inconsistencies.

the participants. External consistency, on the other hand, allows the dialogue to correctly reflect domain knowledge. In this paper, we investigate how dialogue consistency in TOD can be effectively modeled such that possible violations (i.e., inconsistencies) can be automatically detected.

Figure 1, shows a fragment of a Knowledge Base (three restaurants in a city) and a short dialogue in which a user expresses preferences for restaurants serving Spanish food, and the system responds providing information about available options. There are two inconsistencies in this dialogue: first, at turn S1, the system mentions three restaurants serving Spanish food, which is not consistent with the domain knowledge, where there are two such restaurants (domain inconsistency). Second, at turn S2, the system introduces a Lebanese restaurant, while it would have been expected to mention a

Spanish restaurant (dialogue inconsistency). We assume that a well-formed TOD should not manifest any inconsistency of the type reported in our example. However, while relevant work on evaluating TODs has focused on single dialogue components (e.g., dialogue state tracking (Henderson et al., 2014)), consistency evaluation has received much less attention. The problem is even more urgent now that end-to-end approaches (Bang et al., 2023; Lai et al., 2023) are by-passing component evaluations. Automatic detection of dialogue inconsistencies is crucial when dialogues are generated by Large Language Models (LLMs), using few-shot or zero-shot approaches. While LLMs have the capacity to generate TODs without being fine-tuned on training data, it is well known that they are prone to hallucinations (Ji et al., 2022), which may affect dialogue consistency. Furthermore, in dynamic domains where the conversational context evolves over time (Labruna and Magnini, 2023, 2022), maintaining dialogue consistency becomes even more challenging. The possible presence of inconsistencies in TODs (Qin et al., 2021) raises the problem of detecting them, which is the topic of the paper.

The novel intuition of the paper is to consider dialogue consistency as a kind of *Constraint Satisfaction Problem* (*CSP*). We investigate how to assess the consistency of a TOD under the following working hypothesis: (i) first, dialogue consistency can be modeled with constraints that need to be respected by appropriate linguistic realizations; (ii) such constraints can be well represented to define a CSP, whose allowed solutions can be identified by a CSP solver; (iii) a TOD is consistent if its linguistic realizations belong to the set of solutions allowed by a CSP solver for that dialogue. In the paper, we discuss how dialogue constraints are defined, how they can be extracted and modeled as a CSP, and how to set up an experimental setting where we can empirically prove that a CSP solver can detect inconsistencies in a dialogue.

The contributions of the paper are the following: (i) we model TOD consistency as CSP: to the best of our knowledge, this is a fully original approach; (ii) we set up a reusable experimental setting where TOD consistency can be automatically evaluated against a CSP solver;[1] (iii) we show that current state-of-the-art LLMs still struggle to solve simple dialogue consistency tasks, which opens to further research in dialogue consistency.

[1] All resources are publicly available at https://github.com/mwozgpt/tod-csp

2 Dialogue Consistency as a Constraint Satisfaction Problem

In this section, we explore the conceptualization of dialogue consistency in the CSP framework. We first describe the fundamental component of a conversational domain (Section 2.1), then we elucidate the various constraints that contribute to dialogue coherence (Section 2.2), encompassing linguistic, dialogic, and domain-based considerations. We finally expound upon the formalization of dialogue constraints as CSPs (Section 2.3), delineating the process of modeling dialogue coherence as a constraint satisfaction task.

2.1 Conversational Domain

A conversational domain for a TOD refers to the specific topic that the dialogue revolves around, encompassing all the knowledge that is pertinent to the conversation. In this context, the conversational domain is typically represented by a domain ontology providing a schema of the concepts (e.g., RESTAURANT, HOTEL, MOVIE), a set of slots S (e.g., FOOD, AREA, PRICE) for the concepts, and the set of values that each slot can assume (e.g., EXPENSIVE, MODERATE, and CHEAP for the PRICE slot). Then, a domain KB comprises a collection of instances for the ontology concepts, each consisting of [slot,slot-value] pairs, adhering to the domain ontology schema.

2.2 Dialogue Consistency

A TOD can be conceptualized as a sequence of conversational turns between a user and a system aimed at achieving a specific goal. Within this framework, ensuring the consistency of the dialogue is crucial for effective communication between the user and the system. We consider three types of constraints, which need to be respected for a dialogue to be consistent: linguistic, dialogic and domain-based constraints.

Linguistic Constraints. They are necessary to respect general linguistic rules of language, including morpho-syntactic rules (e.g., genre and number agreement) and syntax-based rules (e.g., the correct use of a preposition). For instance, if we are given with the following masked utterance:

```
U: I am looking for a restaurant in <MASK>.
```

the choice of *center* as substitute to the mask token is valid, while *expensive* would not be suitable, because the preposition *in* is rarely used to introduce a price in English.

Dialogic Constraints. They maintain the semantic coherence across successive turns of the dialogue, ensuring that each utterance logically aligns with the preceding context, thereby facilitating a seamless flow of information. As an example, suppose the following masked dialogue turns:

```
U: I would like an Italian restaurant.
S: There is no <MASK> restaurant in the
   center.
```

Here both *Italian* and *cheap* would be eligible choices from a linguistic point of view, but only *Italian* would maintain the coherence with the previous turn in the dialogue.

Domain Constraints. They ensure alignment between the dialogue content and the domain knowledge, thereby maintaining the dialogue's alignment with relevant factual information. Consider, for instance, a KB with the following restaurants:

```
R1: N=Mario A=east F=italian P=expensive
R1: N=Napoli A=centre F=italian P=moderate
```

And the following piece of masked dialogue:

```
U: I am looking for an Italian restaurant
   in the centre.
S: We have <MASK> restaurants available for
   your preferences.
```

Then, the only admissible choice for the masked token would be *one*, as selecting any other number would introduce an inconsistency with the information provided in the KB.

2.3 Dialogue Consistency as CSP

A CSP (Kumar, 1992) imposes certain conditions on a finite set of variables through constraints. Each variable has a finite set of possible values, known as its domain, and constraints define which combinations of values are allowed for specific subsets of the variables. A constraint can be given either explicitly, by enumerating the tuples allowed, or implicitly, e.g., by an algebraic expression. The solution of a CSP is an instantiation of all the variables for which all the constraints are satisfied. A CSP is solvable if it has at least one solution, otherwise it is unsolvable or overconstrained.

The hypothesis of this paper is that the dialogue constraints outlined in Section 2.2 can be modeled as CSPs. Intuitively, variables are the portions of the dialogue that need to be constrained (i.e., the <MASK> tokens in our examples), while the range of possible values for the variables are expressed, either explicitly or implicitly, in a domain KB for that dialogue. The CSP task consists of selecting variable assignments that adhere to linguistic, dialogic, and domain constraints. To formalize this notion, consider a dialogue d_i for which n variables (i.e., masked tokens) $x_1, x_2, \ldots, x_n$ have been defined. Let D_i denote the domain of possible values for variable x_i; let $\mathcal{C}$ be the set of constraints (i.e., linguistic, dialogic, and domain constraints) over the dialogue d_i, and let c represent a single constraint in $\mathcal{C}$. The CSP task is to determine if there exists an assignment $A = \{(x_1, a_1), (x_2, a_2), \ldots, (x_n, a_n)\}$ with $a_i \in D_i$ for $1 \leq i \leq n$, such that A satisfies all constraints in $\mathcal{C}$. This problem can be formulated as follows:

$$\textit{Satisfies}(\{(x_1, a_1), (x_2, a_2), \ldots, (x_n, a_n)\}, C_j)$$

$$\forall C_j \in \mathcal{C}$$

where $\textit{Satisfies}(A, C_j)$ denotes the binary relationship between an assignment A and a constraint C_j, indicating whether the assignment satisfies the constraint.

3 Methodology

This section outlines the process of modeling a TOD as a CSP, and then to assess the dialogue consistency using a CSP solver. The assessment involves three key steps for a $[d, kb]$ pair, where d is a dialogue and kb is a Knowledge Base: (1) identification of variables within the dialogue d (Section 3.1); (2) extraction of dialogue constraints and construction of a CSP solver for the $[d, kb]$ pair (Section 3.2); and (3) application of the CSP solver to determine if the dialogue d represents a feasible solution with respect to the defined constraints (Section 3.3). These phases of the methodology are illustrated in Figure 2.

3.1 Identifying Dialogue Variables

At step 1 (see Figure 2), we consider a TOD d and a kb (i.e., a set of entities described by slot-value pairs) related to the conversational domain of the dialogue. We do not assume any particular dependency between d and kb: d could be either fully covered by kb (i.e., all mentions of slot values in d are present in kb), only partially covered, or not

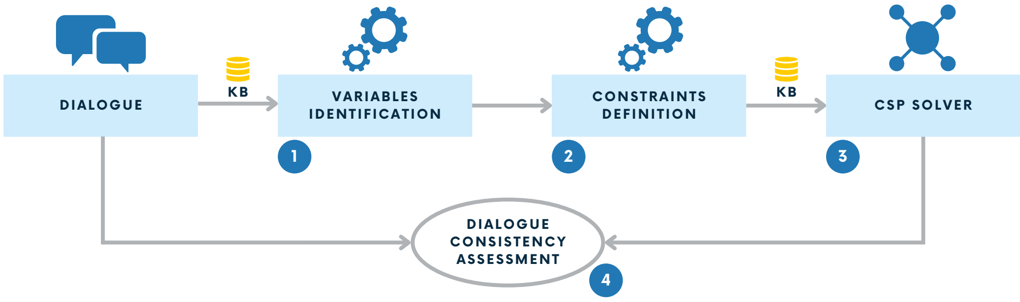

Figure 2: Overview of the CSP-based methodology applied to TOD consistency.

covered at all. We consider text portions in d referring to the conversational domain as potential CSP variables: a text portion referring to a slot value or mentioning amounts of instances in kb. The rationale is that both slot values and instance amounts are elements that better characterize a TOD and are responsible for its consistency. In our example in Figure 1, we will obtain the following variables with their assignments:

$[x_1 = \mathit{Spanish}]$, $[x_2 = \mathit{three}]$, $[x_3 = \mathit{Spanish}]$, $[x_4 = \mathit{cheap}]$ ... $[x_{10} = \mathit{Lebanese}]$.

3.2 Extracting Dialogue Constraints

We have now established a set $\mathcal{X}$ of variables $x_1, x_2, ..., x_n$, where each variable x_i can assume a value either from the slot values or from instance amounts described in kb. Moving to step 2 in Figure 2, we now extract the set of constraints $\mathcal{C}$ over the values that can be assigned to $\mathcal{X}$ variables. We consider the three categories of constraints introduced in Section 2.2: linguistic, dialogic, and domain-based constraints.

Extracting linguistic constraints. We model linguistic constraints as the need for a variable derived from a slot value to match the semantic type of its slot type. For instance, given the utterance *I am looking for a restaurant at* x_1, the value of the variable x_1 must belong to the AREA type. More precisely, $C1$ is defined as follows:

$$C1 : x_1 \in V$$

where V is the set of values belonging to the same slot type as the original value. Constraint $C1$, is meant to avoid that a variable can assume values that are semantically non valid. For instance, avoiding that x_1=NORTH can be assigned to a FOOD, as in *I am looking for a restaurant at* INDIAN, which is ungrammatical in English.

Extracting dialogic constraints. There are two dialogic constraints that we currently consider. $C2$ for ensuring that variables mentioning the same slot value in d are assigned to the same value. $C3$ for ensuring that variables with the same semantic type occurring in the same utterance are assigned to different values. Given the turn *U: I want an* x_1 *restaurant. S: There are 3 restaurant that serve* x_2, we define $C2$ as follows:

$$C_2 : x_1 = x_2$$

where the aim is to keep internal coherence across the dialogue turns. Given the utterance *We have* x_1, x_2, *or* x_3 *restaurants.*, we define $C3$ as:

$$C_3 : x_1 \neq x_2, \quad x_1 \neq x_3, \quad x_2 \neq x_3$$

which captures non redundancy at the utterance level.

Extracting domain-based constraints. There are three domain-based constraints that we currently consider. All of them are meant to guarantee consistency between the number of instances mentioned in d and the actual number of instances present in kb. We distinguish three cases: $C4$ covers the cases when an utterance in d states that there are no instances in kb; $C5$ covers the cases where it is stated that there is at least one instance; and $C6$ the cases where there are exactly n instances.

As for $C4$, consider an utterance indicating no results for a search: *There are no restaurants serving* x_1 *food*, assuming that there are no restaurants with [FOOD=x_1] in kb. For this utterance, $C4$ is defined as:

$$C4 : \neg\exists i \in KB \text{ with values } x_1$$

implying that the variable x_1 can not assume a value that is present in an instance of the KB.

As for $C5$, consider the utterance: *We have many* x_1 *restaurants at* x_2, where at least one restaurant with [FOOD=x_1] and [AREA=x_2] is supposed to exist in kb. For this utterance, $C5$ is defined as:

$$C5 : \exists i \in KB \text{ with values } x_1, x_2$$

imposing the existence of at least one instance with values x_1 and x_2.

Finally, for $C6$, consider the utterance *There are* x_1 *restaurants at* x_2. We define the constraint as:

$$C6 : |\{i \in KB \text{ with value } x_2\}| = x_1$$

to check that the number of instances with value x_2 is exactly equal to x_1.

3.3 Assessing Dialogue Consistency

Once all variables and constraints for a dialogue d are identified, a CSP solver computes possible solutions for the variables in d given kb (step 3 in Figure 2). If one of these solutions matches the variable assignments in d, we consider d consistent with respect to kb (step 4 in Figure 2). For example, in the dialogue and kb illustrated in Figure 1, the variable assignments do not match any CSP admissible solution. Specifically, variable assignment $[x_2 = three]$ violates $C6$, referring to an incorrect number of Spanish instances in kb, and variable $[x_{10} = Lebanese]$ violates $C2$, as it does not maintain coherence with the previous turns. If the CSP solver finds at least one solution, the variable assignments in the dialogue must match one of those solutions, ensuring all constraints are followed. On the other hand, if no solution is found with respect to kb, the variable assignments should be empty or contain values not in kb to ensure consistency. These aspects will be further explored in the experiments discussed in Section 4.

4 Experimental Setting

In this section, we present the experimental setup used to assess dialogue consistency through a CSP solver. We describe the general setting and the purposes of the experiments (Section 4.1), the dataset utilized (Section 4.2), the KBs associated to each dialogue (Section 4.3), the tools employed for constraint satisfaction (Section 4.4), the language model used for dialogue generation (Section 4.5), the baselines against which we compare our results (Section 4.6) and finally, the evaluation metrics that have been used (Section 4.7).

Dataset	**# dialogues**	**# variables**
ALL	131	768
0 SOL.	56	403
1 SOL.	16	58
2-10 SOL.	27	143
11-100 SOL.	18	95
101+ SOL.	14	69

Table 1: Dialogue distribution based on the number of solutions provided by the CSP solver.

4.1 Purposes and General Setting

The purpose of the experiments is to check the feasibility of the CSP-based approach described in Section 3 for detecting dialogue inconsistencies. Our focus is not on optimizing the performance of the CSP solver but rather on investigating critical aspects of the process in a realistic setting. Several steps are involved in this process:

1. Initially, we require dialogue-knowledge base (d-kb) pairs. As for dialogues d, we utilize MultiWoz (Han et al., 2020) dialogues, which are already annotated for dialogue state tracking, enabling precise identification of variables within the dialogue. From an annotated MultiWoz dialogue d, we derive a de-lexicalized version d_{delex}, where dialogue content is replaced with CSP variables.

2. Additionally, for each dialogue, we derive a knowledge base (kb) from the MultiWoz ontology, allowing variation in both the size and type of instances.

3. With d_{delex} and kb established, the next step involves generating variable assignments that can be assessed via a CSP solver. To produce dialogues with potential realistic inconsistencies, we employ a large language model (LLM). The LLM is tasked with re-lexicalizing the variables (i.e., substituting slot-values to CSP variables) in d_{delex}, considering the provided kb. The LLM prompt is illustrated in Appendix A. This re-lexicalization process aims to maximize correctness while adhering to all implicit dialogue constraints.

4. Finally, the re-lexicalized dialogue d_{relex} produced by the LLM serves as a variable assignment and is compared with the solutions of the

Constraint	# variables	% coverage
C1	**768**	**1.00**
C2	686	0.89
C3	108	0.14
C4	9	0.01
C5	394	0.51
C6	197	0.25

Table 2: Number of dialogue variables affected by constraints and their proportion.

CSP solver on the same d-kb pair to produce a consistency score.

4.2 MultiWOZ Dataset

The experimental data was sourced from the MultiWOZ 2.3 dataset (Han et al., 2020), a widely used benchmark for TOD systems comprising more than ten thousand conversations between a user and a system, covering various domains such as restaurants, hotels, or attractions. For our experiments, we focus on restaurant-related dialogues from the MultiWOZ dataset. In total we consider 131 dialogues with 768 total de-lexicalizations (i.e., CSP variables), as shown in the first row of Table 1. In addition, Table 1 categorizes the dataset into groups based on the number of solutions identified by MiniZinc (see Section 4.4) for each dialogue.

4.3 Knowledge Base

The kb employed in the experiments are sourced from the MultiWOZ database. Specifically, for each dialogue d in MultiWOZ, we selected a pertinent instance from the global MultiWOZ KB that aligns with the content of the dialogue. This ensures both relevance and coherence between the dialogue and the associated domain information. Additionally, to introduce variability in the composition of the dialogue kb, we randomly sampled a set of n instances from the global MultiWOZ KB, where n is a randomly generated number between 0 and 8. This approach ensures a diverse range of instances in the dialogue kb while constraining the total number of instances to a maximum of 9, facilitating efficient prompting of the kb to the LLM.

4.4 MiniZinc Constraint Solver

As for CSP solver, we use MiniZinc (Nethercote et al., 2007), an open-source constraint programming language specifically designed for modeling and solving constraint satisfaction problems. We employed MiniZinc to obtain solutions satisfying the dialogue constraints for ourevaluation purposes. MiniZinc provides a high-level modeling language that allows users to express problem constraints and objectives. It supports a wide range of constraint types, which make it suitable for modeling diverse problem domains. Among MiniZinc's suite of solvers, we leveraged Chuffed (Chu et al., 2018), a state-of-the-art solver known for its efficiency in solving CSPs through time optimization, especially advantageous for addressing complex and large-scale optimization problems.

4.5 GPT-3.5-Turbo Language Model

For dialogue re-lexicalization, we employed the GPT-3.5-Turbo language model, a member of the OpenAI GPT family (Achiam et al., 2023), specifically designed to perform well in conversational contexts. GPT-3.5-Turbo was prompted with both (d_{delex}) and its associated kb. This comprehensive input served to guide the model to produce dialogues that adhere to the implicit constraints, thereby ensuring dialogue coherence and adherence to the domain. We utilized GPT-3.5 for inference in zero-shot mode (see Appendix A), without any fine-tuning, leveraging the API version dated "2023-05-15" with a temperature setting of 0.9 to ensure balanced exploration and exploitation during dialogue generation.

4.6 Baselines

We introduce two dialogue re-lexicalization baselines, for a comparative analysis with GPT. The first baseline (RANDOM), produces a dialogue d_{relex} where variables in d_{delex} are randomly assigned to slot values present in the kb. The second baseline (MOST FREQUENT) produces a dialogue d_{relex} where variables in d_{delex} are assigned to the most frequent value observed in the kb. By contrasting our evaluation results with these baselines, we gain insights into the efficacy of our approach in capturing and assessing dialogue consistency.

4.7 Evaluation Metrics

Global Consistency Accuracy (GCA) and Variable Consistency Accuracy (VCA) are the two metrics used to evaluate the adherence of a dialogue to a specific set of constraints. Given a re-lexicalized dialogue d_{relex} where variables are assigned to values, GCA measures the overall accuracy of the assignments for each variable. The average GCA is

Dataset	GCA	VCA
RANDOM	0.01	0.06
MOST FREQUENT	0.01	0.10
GPT	**0.15**	**0.27**

Table 3: Global and variable consistency for dialogues re-lexicalized by GPT compared to the RANDOM and MOST FREQUENT baselines.

Dataset	GCA	VCA
0 SOL.	0.0	0.0
1 SOL.	0.31	0.48
2-10 SOL.	0.22	0.53
11-100 SOL.	0.22	0.55
101+ SOL.	**0.36**	**0.70**

Table 4: Assessment of global and variable consistency for re-lexicalized dialogues across solution groups.

calculated as the proportion of dialogues that fully comply with all defined constraints:

$$GCA = \frac{\sum_{i=1}^{N} \left(\prod_{j=1}^{M} \mathit{Satisfies}(A_i, C_j) \right)}{N}$$

where N is the total number of dialogues, and $\mathit{Satisfies}(A_i, C_j)$ is a binary indicator function that returns 1 if and only if all variable assignments in dialogue d_i comply with the constraint j, 0 otherwise. On the other hand, VCA assesses the assignment accuracy on individual variables within the dialogue. We compare the dialogue assignment to the solutions of the CSP solver and find the most similar solution; then, we count how many variable assignments coincide with the assignments of the most similar solution. We formally define VCA as follows:

$$VCA = \frac{\sum_{i=1}^{N} |\mathit{CorrectAssignments}(d_i)|}{M}$$

where N is the total number of dialogues, M is the total number of variables in the dialogues, and $\mathit{CorrectAssignments}(d_i)$ are the variable assignments in dialogue d_i that coincide with the assignments of the most similar solution provided by the CSP solver. GCA and VCA provide insights into the ability of the dialogue generation system to maintain coherence and fidelity to the underlying domain knowledge while generating responses. Higher values of GCA and VCA indicate better performance in terms of dialogue quality and consistency.

5 Results

Table 2 presents the impact of each constraint on the variables in the dataset, detailing the percentage of variables influenced by each constraint. This shows that $C1$ (i.e., assigned values need to respect the semantic type of the variable) applies to all variables in the dataset, while $C4$ (no instances in kb) applies only nine time in total. Table 3 compares the global and variable consistency in dialogues re-lexicalized by GPT with the RANDOM and MOST FREQUENT baselines. GPT dialogues exhibit significantly higher global and variable consistency compared to the baseline datasets. Table 4 assesses GCA and VCA for GPT dialogues across various CSP solution groups. Results show that dialogues with more solutions tend to have higher GCA and VCA scores, while the model is not able to recognize and address the 0 solution cases.

Table 5 presents the results of an ablation study, where we systematically remove each constraint one by one and analyse their impact on GCA and VCA for each configurations. Results show that the most critical constraint is $C6$ (i.e., exact match with number of kb instances). Additionally, we conducted experiments where groups of constraints were collectively removed to observe their influence on the dialogue generation process, confirming that domain-based constraints are more critical.

6 Discussion

The experiment results shed light on several key aspects of consistency assessment for TODs. First, comparing GPT and the two baselines (RANDOM and MOST FREQUENT) on re-lexicalized dialogues, we note the better quality achieved by the GPT model (see Table 3), both in term of GCA and VCA. GPT can effectively re-lexicalize dialogues that more closely adhere to the defined constraints. Furthermore, the assessment of global and variable consistency across different solution groups reveals interesting patterns (see Table 4). Dialogues with a higher number of solutions tend to exhibit higher levels of consistency, indicating that the model performs better when presented with more options to fulfill constraints. At the other extreme, the model is not able to address cases where no feasible solution exists, as it always provides an attempt of assignment for the variables. This finding

Constraint	GCA	VCA
ALL EXCEPT C1	0.15	0.31
ALL EXCEPT C2	0.15	0.27
ALL EXCEPT C3	0.15	0.29
ALL EXCEPT C4	0.16	0.30
ALL EXCEPT C5	0.15	0.32
ALL EXCEPT C6	**0.21**	**0.48**
ALL EXCEPT DIALOGIC	0.15	0.30
ALL EXCEPT DOMAIN	**0.23**	**0.56**

Table 5: Ablation study: global and variable consistency under different constraint configurations.

emphasizes the importance of considering the richness and diversity of CSP solutions, as they have a strong impact on the quality and consistency of re-lexicalized dialogues. Additionally, analysing the distribution of constraints on the dialogue variables, reveals significant variations (see Table 2), with certain constraints exerting a stronger influence than others. The ablation study provides valuable insights into the impact of the different constraints on dialogue re-lexicalization. Excluding domain constraints, in particular, leads to significantly higher GCA and VCA scores, indicating the critical role of domain-specific knowledge in shaping dialogue coherence and relevance (see Table 5). This suggests that recent LLMs may not effectively leverage the provided kb, highlighting an area for potential improvement in future iterations of language model training and dialogue re-lexicalization techniques. Our experiments have shown that modeling and assessing dialogue consistency through CSP is both feasible and challenging. We were able to highlights both strengths and weaknesses of dialogue generation and to discern which constraints are met and which are not, gaining insight into the specific features and challenges inherent in this process.

7 Related Work

TOD systems have been extensively investigated in NLP. (Allen et al., 2001). Recent research has explored the use of neural network architectures for dialogue state tracking (Wu et al., 2020; Zhao et al., 2021) and policy learning (Su et al., 2016; Liu and Lane, 2017). Several metrics have been proposed to assess the performance of TOD systems, including task completion rates, user satisfaction scores, and objective measures for system components, such as precision, recall, and F1-score (Chen et al., 2017; Santhanam and Shaikh, 2019; Deriu et al., 2021). Recent studies have emphasized the importance of holistic evaluation frameworks that consider multiple aspects of dialogue quality (Zhang et al., 2021).

Maintaining consistency and coherence in dialogues is essential for effective communication between users and dialogue systems. Previous research has investigated various approaches to ensure dialogue coherence, including coherence modeling (Cervone et al., 2018), and coherence-based response generation (Cervone and Riccardi, 2020), aiming to enhance the naturalness and fluency of generated dialogues. Finally, several studies have explored the application of CSPs to language. These include early attempts to ensure coherence in generated text (Kibble and Power, 2004), model preposition lexicalization using constraints (Moriceau and Saint-Dizier, 2004), guide lexical choices through constraints (McKeown et al., 1997), and treat context-sensitive utterance generation as a CSP (Popescu et al., 2009).

8 Conclusion

In this paper, we have introduced a novel approach to assess dialogue consistency in the context of TODs using a metric based on Constraint Satisfaction. In our approach, variables represent de-lexicalized segments of the dialogue and constraints reflect linguistic, conversational, and domain-based properties of TODs. Our experiments have demonstrated the feasibility of this approach, enabling us to effectively identify and quantify inconsistencies present in the dialogues. An interesting side-effect of our investigation is the observation that state-of-the-art LLMs often introduce numerous inconsistencies when tasked with re-lexicalizing dialogues. These inconsistencies primarily concern domain knowledge adherence, resulting in an overall accuracy of only 0.15 at the dialogue level. Our study highlights the potential of CSP-based methodologies in evaluating dialogue consistency and identifying areas for improvement in automated dialogue generation systems. Future research should further explore the application of CSP in this domain and investigate strategies to enhance the coherence of LLM-generated dialogues, particularly in applications with strong domain knowledge requirements.

9 Limitations

Our study is subject to several limitations that warrant consideration. Firstly, the process of defining constraints for dialogue consistency assessment is complex and multifaceted. While we have delineated several constraints in this study, the TOD landscape is vast, and additional constraints may need to be identified and incorporated to capture a broader range of dialogue scenarios accurately. Each constraint is formulated based on our current understanding of the phenomena, acknowledging that further investigations may uncover additional constraints. Additionally, we also consider implementation feasibility, as certain constraints may require more extensive implementation efforts to detect. Moreover, the selection and prioritization of constraints inherently involve subjective judgment, and achieving consensus on the most relevant constraints for a given dialogue domain may pose a challenge.

Secondly, while we employed a state-of-the-art Large Language Model (LLM) for dialogue generation and consistency assessment, the performance of alternative language models remains unexplored. Investigating the effectiveness of various LLM architectures, pre-training strategies, or fine-tuning approaches could provide valuable insights into their suitability for TOD tasks.

Furthermore, while our methodology endeavors to be as generalizable as possible, it is important to acknowledge that nuances in dialogue structures and domain-specific knowledge may exist across different datasets, and there may still be aspects of dialogue consistency that our approach may not fully capture. Exploring additional datasets spanning diverse domains and languages could offer a more comprehensive understanding of dialogue consistency challenges and the efficacy of our proposed methodology.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. arXiv preprint arXiv:2303.08774.

James Allen, George Ferguson, and Amanda Stent. 2001. An architecture for more realistic conversational systems. In Proceedings of the 6th international conference on Intelligent user interfaces, pages 1–8.

Vevake Balaraman, Seyedmostafa Sheikhalishahi, and Bernardo Magnini. 2021. Recent neural methods on dialogue state tracking for task-oriented dialogue systems: A survey. In Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2021, Singapore and Online, July 29-31, 2021, pages 239–251. Association for Computational Linguistics.

Yejin Bang, Samuel Cahyawijaya, Nayeon Lee, Wenliang Dai, Dan Su, Bryan Wilie, Holy Lovenia, Ziwei Ji, Tiezheng Yu, Willy Chung, et al. 2023. A multitask, multilingual, multimodal evaluation of chatgpt on reasoning, hallucination, and interactivity. arXiv preprint arXiv:2302.04023.

Alessandra Cervone and Giuseppe Riccardi. 2020. Is this dialogue coherent? learning from dialogue acts and entities. arXiv preprint arXiv:2006.10157.

Alessandra Cervone, Evgeny Stepanov, and Giuseppe Riccardi. 2018. Coherence models for dialogue. arXiv preprint arXiv:1806.08044.

Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. 2017. A survey on dialogue systems: Recent advances and new frontiers. Acm Sigkdd Explorations Newsletter, 19(2):25–35.

Geoffrey Chu, Peter J. Stuckey, Anthony Schutt, Thorsten Ehlers, Graeme Gange, and Keith Francis. 2018. Chuffed, a lazy clause generation solver. https://github.com/chuffed/chuffed.

Jan Deriu, Alvaro Rodrigo, Arantxa Otegi, Guillermo Echegoyen, Sophie Rosset, Eneko Agirre, and Mark Cieliebak. 2021. Survey on evaluation methods for dialogue systems. Artificial Intelligence Review, 54:755–810.

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

Matthew Henderson, Blaise Thomson, and Jason D. Williams. 2014. The second dialog state tracking challenge. In Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL), pages 263–272, Philadelphia, PA, U.S.A. Association for Computational Linguistics.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Yejin Bang, Andrea Madotto, and Pascale Fung. 2022. Survey of hallucination in natural language generation. ACM Computing Surveys.

Rodger Kibble and Richard Power. 2004. Optimizing referential coherence in text generation. Computational Linguistics, 30(4):401–416.

Vipin Kumar. 1992. Algorithms for constraint-satisfaction problems: A survey. AI magazine, 13(1):32–32.

Tiziano Labruna and Bernardo Magnini. 2022. Fine-tuning bert for generative dialogue domain adaptation. In Text, Speech, and Dialogue, pages 490–501.

Tiziano Labruna and Bernardo Magnini. 2023. Addressing domain changes in task-oriented conversational agents through dialogue adaptation. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop, pages 149–158.

Tuan M Lai, Giuseppe Castellucci, Saar Kuzi, Heng Ji, and Oleg Rokhlenko. 2023. External knowledge acquisition for end-to-end document-oriented dialog systems. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, pages 3633–3647.

Bing Liu and Ian Lane. 2017. Iterative policy learning in end-to-end trainable task-oriented neural dialog models. In 2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), pages 482–489. IEEE.

Samuel Louvan and Bernardo Magnini. 2020. Recent neural methods on slot filling and intent classification for task-oriented dialogue systems: A survey. In Proceedings of the 28th International Conference on Computational Linguistics, pages 480–496, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Kathleen McKeown, Michael Elhadad, and Jacques Robin. 1997. Floating constraints in lexical choice.

Michael McTear. 2020. Conversational ai: Dialogue systems, conversational agents, and chatbots. Synthesis Lectures on Human Language Technologies, 13(3):1–251.

Véronique Moriceau and Patrick Saint-Dizier. 2004. A constraint-based model for preposition choice in natural language generation. Constraint Solving and Language Processing, page 124.

Nicholas Nethercote, Peter J. Stuckey, Rowan Becket, Simon Brand, Greg J. Duck, and Guido Tack. 2007. Minizinc: Towards a standard cp modelling language. In CP 2007, volume 4741 of LNCS, pages 529–543. Springer.

Vladimir Popescu, Jean Caelen, and Corneliu Burileanu. 2009. A constraint satisfaction approach to context-sensitive utterance generation in multi-party dialogue systems. International Journal of Speech Technology, 12:95–112.

Libo Qin, Tianbao Xie, Shijue Huang, Qiguang Chen, Xiao Xu, and Wanxiang Che. 2021. Don't be contradicted with anything! ci-tod: Towards benchmarking consistency for task-oriented dialogue system. arXiv preprint arXiv:2109.11292.

Sashank Santhanam and Samira Shaikh. 2019. Towards best experiment design for evaluating dialogue system output. arXiv preprint arXiv:1909.10122.

Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. On-line active reward learning for policy optimisation in spoken dialogue systems. arXiv preprint arXiv:1605.07669.

Peng Wu, Bowei Zou, Ridong Jiang, and AiTi Aw. 2020. Gcdst: A graph-based and copy-augmented multi-domain dialogue state tracking. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 1063–1073.

Chen Zhang, Grandee Lee, Luis Fernando D'Haro, and Haizhou Li. 2021. D-score: Holistic dialogue evaluation without reference. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29:2502–2516.

Jeffrey Zhao, Mahdis Mahdieh, Ye Zhang, Yuan Cao, and Yonghui Wu. 2021. Effective sequence-to-sequence dialogue state tracking. arXiv preprint arXiv:2108.13990.

A Appendix A: GPT prompt

Below is an instruction that outlines a task, along with a Knowledge Base containing domain-specific information to be utilized, and a dialogue for you to work on. Return a response that effectively fulfills the task.

Instruction:
Fill in the [MASK] placeholders in the dialogue based on the information provided in the Knowledge Base. Provide the updated dialogue exactly as it was given, but with the placeholders replaced by the appropriate values for each turn in the dialogue. If a turn does not contain any placeholders, leave the sentence unchanged. Turns should start with either User or System. Be aware of leaving blank spaces before punctuation as in the original (e.g. Hi , instead of Hi,)

Knowledge Base:
Restaurant #1 - Area: centre, Food: british, Price: moderate
Restaurant #2 - Area: west, Food: european, Price: expensive

Dialogue:

USER: I 'm looking for a restaurant serving **[MASK]** food in any area .

SYSTEM: There are no **[MASK]** restaurants in the area .

USER: Well can I get the phone number to a **[MASK]** restaurant ?
SYSTEM: Restaurant one seven serves **[MASK]** food . Their phone number is 01223 355166 . Can I help you with anything else ?
USER: That 's it , goodbye
SYSTEM: your welcome good bye

Response:

USER: I 'm looking for a restaurant serving **european** food in any area .
SYSTEM: There are no **european** restaurants in the area .
USER: Well can I get the phone number to a **british** restaurant ?
SYSTEM: Restaurant one seven serves **british** food . Their phone number is 01223 355166 . Can I help you with anything else ?
USER: That 's it , goodbye
SYSTEM: your welcome good bye