Unsupervised Extraction of Dialogue Policies from Conversations

Makesh Narsimhan Sreedhar, Traian Rebedea and **Christopher Parisien**
NVIDIA
Santa Clara, CA
{makeshn, trebedea, cparisien}@nvidia.com

Abstract

Dialogue policies play a crucial role in developing task-oriented dialogue systems, yet their development and maintenance are challenging and typically require substantial effort from experts in dialogue modeling. While in many situations, large amounts of conversational data are available for the task at hand, people lack an effective solution able to extract dialogue policies from this data. In this paper, we address this gap by first illustrating how Large Language Models (LLMs) can be instrumental in extracting dialogue policies from datasets, through the conversion of conversations into a unified intermediate representation consisting of canonical forms. We then propose a novel method for generating dialogue policies utilizing a controllable and interpretable graph-based methodology. By combining canonical forms across conversations into a flow network, we find that running graph traversal algorithms helps in extracting dialogue flows. These flows are a better representation of the underlying interactions than flows extracted by prompting LLMs. Our technique focuses on giving conversation designers greater control, offering a productivity tool to improve the process of developing dialogue policies.[1]

1 Introduction

Chatbots and virtual assistants have emerged as powerful tools for guiding users or automating specific tasks across different domains, from facilitating restaurant reservations (Budzianowski et al., 2018) to handling product returns on e-commerce platforms (Chen et al., 2021).

Most task-oriented dialogue systems (TODS) nowadays use two key components: a Natural Language Understanding (NLU) engine and a Dialogue Manager (Bocklisch et al., 2017). The role of the NLU engine is to perform intent detection and slot extraction, essential for understanding the user's requests. Concurrently, the Dialogue Manager leverages the current dialogue state, alongside the intent and slots identified from the latest user message, to determine the subsequent bot action or response. In most cases, both the NLU and Dialogue Manager rely on expert human intervention, typically involving a mix of conversation designers and data scientists. The NLU component requires a predefined set of user intents and slots while the Dialogue Manager necessitates dialogue policies that dictate the bot responses.

In the development of task-oriented assistants, it is common to have access to a corpus of preexisting conversations. Recent research has shown considerable interest in harnessing these conversational corpora to construct TODS. Extracting intents directly from these dialogues has demonstrated significant potential in augmenting the efficiency of conversation designers in modeling the NLU component (Chatterjee and Sengupta, 2020; Kumar et al., 2022; Du et al., 2023). However, the task of deriving dialogue policies from the same set of conversations presents a more complex challenge and requires a nuanced understanding of conversational dynamics and objectives. Only a limited number of studies have ventured into this domain, exploring methodologies for automatic dialogue policy extraction (Richetti et al., 2017; Vakulenko et al., 2019; Ferreira, 2023).

In this paper, we introduce a novel hybrid methodology (§3) that combines Large Language Models (LLMs) with graph-based algorithms for the automated extraction of dialogue policies from a corpus of task-specific conversations. To that end, we first translate the turns in each dialogue into canonical forms (Sreedhar and Parisien, 2022) using an LLM. The canonical forms are then clustered together to smooth out minor variations, following which we construct a graph modelling the entire corpus of conversations. This graph is akin

[1] Data and code can be found at https://github.com/makeshn/flows_from_conversations.

Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing, pages 19029–19045
November 12-16, 2024 ©2024 Association for Computational Linguistics

to a *flow network*, where nodes represent canonical forms of dialogue turns and edges signify the progression and connection between different turns. Finally, we apply path-finding algorithms to this graph to extract dialogue policies.

The proposed approach combines sequences of user and assistant canonical forms, that can be seen as *dialogue trajectories*, into a more complex ***dialogue policy***. Thus, the extracted policies can handle digressions that are expressed using a branching logic determined by the intent of a user message. As all trajectories in our policies, including digressions, are composed of sequences of (user and bot) canonical forms, we can express the dialogue policy extraction from a corpus of conversations as a compositional task that combines translation and multi-document summarization. This enables using automatic metrics (e.g. BLEU, BERTSCORE) for evaluating the quality of the generated policies (§4). We also show that these metrics correlate very well with human evaluation (§5).

Our findings indicate superior performance of our hybrid graph and LLM-based approach over techniques that rely solely on prompting LLMs for policy generation. In addition to better quantitative performance, the graph-based methodology provides enhanced controllability, interpretability, and robustness. These qualities render it a practical and effective tool for aiding conversation designers, in contrast to the more opaque, black-box nature of prompt-based LLM approaches.

Our main contributions are as follows:

- Demonstrating the feasibility of extracting dialogue policies expressed as sequences of user and assistant canonical forms from a corpus of conversations focused on a specific task.
- Modelling conversations with a flow network graph derived from the sequences of canonical forms provides an efficient method for policy extraction. The evaluation of dialogue policies computed using our hybrid graph and LLM approach demonstrates superior performance compared to prompt-based methods.
- Providing a controllable and highly interpretable practical solution to be used by conversation designers in real-world scenarios.
- Contributing to the field by releasing the extracted dialogue policies for tasks in two popular datasets for TODS: SGD (Rastogi et al., 2020) and ABCD (Chen et al., 2021).

2 Background

Task-Oriented Dialogue. Most tools for building task-oriented chatbots and virtual assistants use two different components: NLU and a Dialogue Manager (Liu et al., 2021). These range from commercial solutions (e.g. Google DialogFlow (Google, 2024) or Oracle Digital Assistant (Bors et al., 2020)) to open-source tools like Rasa (Bocklisch et al., 2017) or research-focused platforms such as ConvLab (Lee et al., 2019).

Dialogue policies can be modeled as sequences of user intents and bot actions, for example using *stories* in Rasa (Bocklisch et al., 2017) or *Colang flows* in NeMo Guardrails (Rebedea et al., 2023). Our work is valuable for this modeling: the extracted dialogue policies can serve as starting points for conversation designers to refine.

Traditionally, the development of task-oriented dialogue systems (TODS) required manual effort from experts. However, recent tools and methods aim to reduce this effort by leveraging large datasets for automatic intent discovery, with some addressing the challenge of dialogue policy generation. Even end-to-end neural TODS (Hosseini-Asl et al., 2020) that embed intents and policies in model weights can use the extracted human-readable dialogue policies to enhance the explainability of the underlying opaque systems.

Canonical Forms. NLU has typically used discriminative components for intent classification and slot labeling. With advances in generative text models (Radford et al., 2018; Brown et al., 2020), NLU can now be remodeled as a generative engine for intents and slots. Canonical forms (Sreedhar and Parisien, 2022) encode the intent of conversation turns in a concise, standard form. Unlike the closed set of expert-defined intent classes, canonical forms are generated by models and are task-independent, offering a flexible way to encode dialogue policies (Rebedea et al., 2023).

Intent Discovery. Intent mining has lately been an active topic not only in conversations (Chatterjee and Sengupta, 2020), but also in web queries (Vedula et al., 2020). Most of the works employ various clustering algorithms (DBSCAN (Chatterjee and Sengupta, 2020), k-means (Du et al., 2023), iterative (Benayas et al., 2023)), with different text embeddings. Recent works propose using contrastive learning for training specific embeddings for this task (Du et al.,

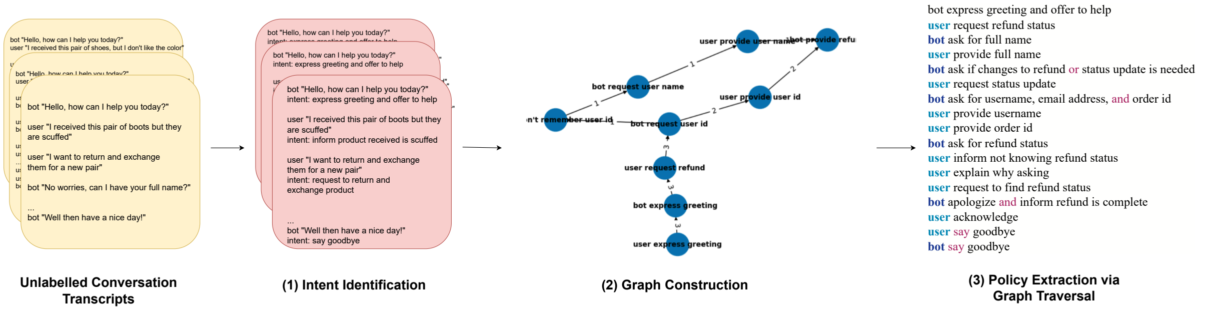

bot "Hello, how can I help you today?"
user "I received this pair of shoes, but I don't like the color"
bot "Hello, how can I help you today?"
user "I received this pair of boots but they are scuffed"
user "I want to return and exchange them for a new pair"
bot "No worries, can I have your full name?"
...
bot "Well then have a nice day!"
Unlabelled Conversation Transcripts
bot "Hello, how can I help you today?"
intent: express greeting and offer to help
user "I received this pair of boots but they are scuffed"
intent: inform product received is scuffed
user "I want to return and exchange them for a new pair"
intent: request to return and exchange product
...
bot "Well then have a nice day!"
intent: say goodbye
(1) Intent Identification
user provide user name
bot provide refu
bot request user name
user provide user id
n't remember user id
bot request user id
user request refund
bot express greeting
user express greeting
1
1
1
2
2
3
3
3
(2) Graph Construction
bot express greeting and offer to help
user request refund status
bot ask for full name
user provide full name
bot ask if changes to refund or status update is needed
user request status update
bot ask for username, email address, and order id
user provide username
user provide order id
bot ask for refund status
user inform not knowing refund status
user explain why asking
user request to find refund status
bot apologize and inform refund is complete
user acknowledge
user say goodbye
bot say goodbye
(3) Policy Extraction via Graph Traversal

Figure 1: The three stages of the proposed solution for extracting dialogue flows: 1) Label user and bot turns in the conversations with canonical forms (§3.1); 2) Construct an interaction graph between user and bot canonical forms (§3.2); 3) Use graph traversal to extract dialogue flows as sequences of canonical forms (§3.3).

2023; Kumar et al., 2022) or using a dual-stage clustering (Du et al., 2023). Similar to our proposed intent discovery approach, some methods generate intent names as well (Vedula et al., 2020; Benayas et al., 2023). Our intent discovery method combines an LLM p-tuned for generating canonical forms with an extra clustering step and is applied for both user and bot intents.

Dialog Flow Extraction. There are just a handful of works tackling the generation of dialogue policies. Earlier works employed a type of process mining that required either using a taxonomy of speech acts (Vakulenko et al., 2019) or other predefined classes (Richetti et al., 2017) for each turn in the conversation. Ferreira (2023) is the most similar to our proposed method, as it considers a graph-based approach to identify frequent sequences of turn types, but it employs a specific taxonomy of dialogue acts to label the turns. One important advantage of our approach is that it does not need any human intervention.

Unsupervised dialogue structure discovery (Lu et al., 2022; Shi et al., 2019) is similar to dialogue flow extraction. However, there are important differences: they mainly aim to discriminate conversations in different tasks from a dataset and the latent structures used to encode the state of a conversation cannot be easily used by the Dialogue Manager of a TODS. Another task that has some similarities is workflow discovery which aims at predicting API calls given a task-oriented conversation, but it was only explored in a low-data regime, not fully unsupervised (Hattami et al., 2022).

3 Method

To extract dialogue policies (flows) from conversational data, we propose a pipeline comprising three key stages: intent identification, graph construction utilizing the identified intents, and the application of graph traversal algorithms for the extraction of dialogue flows. The functionality is depicted in Fig. 1: the input is a corpus of conversations on a given task and the output is the dialogue policy as a combination of sequences of canonical forms.

3.1 Intent Identification

We begin with a corpus of task-specific conversations, such as customer interactions regarding product returns in an e-commerce setting. These conversations are structured as a series of exchanges between an user and a human agent, composed of n dyads. The typical format of a conversation is an alternating sequence of user and agent turns, represented as $[u_1, a_1, ..., u_n, a_n]$.

The primary objective at this initial stage of the pipeline is to assign an intent label to each turn in the conversation, effectively mapping: $\text{turn}_i \rightarrow \text{intent}(\text{turn}_i)$ This enables us to analyze conversations at a higher level of abstraction rather than operating at the level of individual turns. The abstracted conversation can thus be depicted using the corresponding intents, $[\text{intent}(u_1), \text{intent}(a_1)..., \text{intent}(a_n)]$.

Not only that intent labels are not available in an unsupervised context, but the intents provided as part of TOD datasets are not easily transferable across domains. We adopt the usage of *canonical forms* (Sreedhar and Parisien, 2022) for inferring the intents of conversation turns. This approach offers a practical alternative, enabling intent identification without relying on predefined label sets.

Canonical Forms. Intent labels traditionally tend to be terse, and this often hinders the generalization of models to new domains. Canonical forms are concise, yet descriptive phrases that can capture the essence of utterances in the conversation (see Fig. 2). More complex examples are shown in Appendix B highlighting that canonical forms can also capture slots in addition to the intent (e.g. `bot ask for city`, `user provide city`).

Weak Supervision. We leverage the impressive generalization capabilities of language models to extract canonical forms across a wide range of domains. Starting from a small set of 200 conversations from two tasks in the ABCD dataset (product returns and shipping inquiries), canonical forms for each turn are obtained using `text-davinci-003`, OpenAI's instruction-tuned LLM (Ouyang et al., 2022). Using this weakly supervised data, a smaller LLM(§A.2) is p-tuned (Liu et al., 2022) to predict the canonical form for a particular turn given the conversation history, i.e. it learns the mapping: $[u_1, a_1, u_2, ..., u_i] \rightarrow \text{intent}(u_i)$. The trained model is then used to annotate utterances with canonical forms across different domains. It is employed for all our experiments, showing its generalization not only to the other tasks in the ABCD dataset, but also to a different domain (SGD).

Aligning a separate model allows for obtaining more consistent and cheaper annotations than using OpenAI models. Moreover, as the p-tuning dataset is small we also plan to obtain human annotations and release a commercially viable model for generating canonical forms. More details about the p-tuned model and prompting used for obtaining weak labels are shown in Appendix A.

Intent Normalization. Using a generative approach to obtain the canonical form for utterances introduces variability in how similar intents are described (see Fig. 2). In this stage, we want to group canonical forms that represent identical intents, identify a representative form within each group, and subsequently re-annotate the conversations with the representative forms for each group.

We extract the canonical forms not only for user turns, but also for agent responses. For the normalization stage, we use agglomerative clustering independently for the set of canonical forms (user and agent). The embeddings for the canonical forms are computed using the `MiniLM-L6` model (Reimers and Gurevych, 2019). The representative canonical form for each cluster is chosen based on frequency. All other canonical forms within each cluster are then substituted with this representative canonical form. This procedure yields a collection of conversations labelled with a consistent and unified set of canonical forms. Additional implementation details are in Appendix A.

3.2 Graph Construction

The conversations with the unified canonical forms allow us to construct a graph denoting how each conversation proceeds. Let us consider a conversation with canonical forms $[\bar{u}_1, \bar{a}_1, \bar{u}_2, ...\bar{u}_n, \bar{a}_n]$ where $\bar{u}_i$ and $\bar{a}_i$ denote the canonical forms for user turn u_i and agent turn a_i respectively. This allows us to construct a linear path that denotes how the conversation progressed:

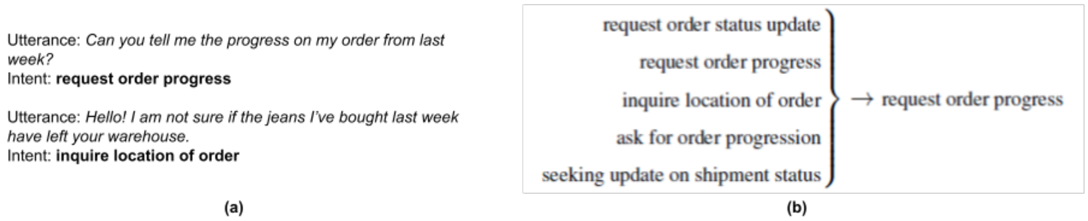
$$\bar{u}_1 \rightarrow \bar{a}_1 \rightarrow \cdots \rightarrow \bar{u}_n \rightarrow \bar{a}_n$$

We construct an *interaction graph* by merging all such linear paths for all conversations given a specific task. The canonical forms corresponding to the user and agent turns are the **nodes** of the graph. A directed **edge** connects each canonical form to the next in the sequence within the conversation. The frequency of a particular transition between two canonical forms (such as $\bar{u}_i \rightarrow \bar{a}_i$ or $\bar{a}_i \rightarrow \bar{u}_{i+1}$), determines the weight of the corresponding edge, i.e. this weight represents the number of occurrences of that transition across all conversations. This results in a weighted directed graph that effectively captures the dynamics of dialogue progression across multiple conversations.

3.3 Policy Extraction via Graph Traversal

Given the constructed interaction graph, we can extract various *dialogue flows* using graph algorithms. Assuming that a dialogue flow can be represented as a path from a source node to a destination, we can employ various graph traversal algorithms for this stage. Our intuition is that the dialogue policy for the *happy path* of a task should be the most commonly traversed path. Since the weights of the edges are a proxy for the number of conversations in which that transition occurs, we can consider an algorithm where the objective is to maximize the minimum capacity along a path.

Fattest-Path Dijkstra. Given a graph depicting a transportation network (graph whose weights are considered as transportation capacities), this is a variation of Dijkstra's algorithm where we want to find a path between the source and the target such that the minimum weight of any edge in the path is as large as possible (Cormen et al., 2022).


Utterance: *Can you tell me the progress on my order from last week?*
Intent: **request order progress**

Utterance: *Hello! I am not sure if the jeans I've bought last week have left your warehouse.*
Intent: **inquire location of order**

(a)
request order status update
request order progress
inquire location of order
ask for order progression
seeking update on shipment status
→ request order progress
(b)

Figure 2: Intent identification: (a) Extraction of canonical forms from conversation turns using an LLM, (b) Intent normalization via clustering.

Let $G = (N, E)$ be a graph with nodes N and edges E, each edge e having a weight $w(e)$. Let $P = \langle s = n_0, n_1, ..., n_k = t\rangle$ be a path from source s to target t. The bottleneck for path P, denoted by $F(P)$, is defined as:

$$F(P) = \min_{0 \le i < k} \{w(n_i, n_{i+1})\}$$

The goal is to find the path P^* with the largest bottleneck out of all possible paths from s to t, $\mathcal{P}(s,t)$:

$$P^* = \arg\max_{P \in \mathcal{P}(s,t)} \{F(P)\}$$

The source node is chosen as the most commonly occurring canonical form for the first turn observed across all conversations, while the target node is the most frequent final turn canonical form. We then apply the algorithm and extract the fattest-width path as our initial version of the dialogue flow, df.

3.3.1 Dialogue Flow Digressions

Extracting only the widest path presents us with an incomplete view of the dialogue progression (the *"happy"* or *main path*). We need to find alternative paths arising from nodes on the widest path to capture a more complete dialogue policy.

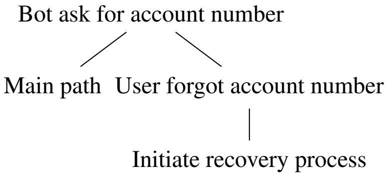

Bot ask for account number
Main path
User forgot account number
Initiate recovery process

Figure 3: Digression from the main path in a graph.

To fully comprehend the necessity of examining digressions, let us consider the example in Fig. 3. Consider a scenario where after the bot asks for the user's account number, the main dialogue path continues with the step `user provides account number`. However, a potential deviation might occur if the user does not recall the account number. This deviation leads to an alternate path, starting with the canonical form "user forgot account number" and branching into an account recovery subflow. Digressions help us enhance the structure and flow of the dialogue policy.

Identifying Digressions. For each node in the main dialogue flow, n_i, we examine all nodes n_j that are directly connected to n_i, i.e. $n_i \rightarrow n_j$.

To identify potential digressions, we employ a similarity-based thresholding method. If the similarity measure between the canonical form of node n_j and the next node on the main dialogue flow n_{i+1} falls below a specified threshold ϵ, we mark n_j as a digression candidate:

$$\text{sim}(n_{i+1}, n_j) < \epsilon \implies \text{digression candidate}$$

Then we determine the widest path from each digression candidate node to the final node n_{end} in the dialogue flow. This procedure yields a set of potential alternative paths, denoted as P_{alt}.

Finally, we compute the similarity between each alternative path and the main dialogue trajectory. Paths that exhibit a high similarity to the main dialogue trajectory are discarded, the remaining paths whose similarity to the main flow is below a threshold κ are considered digressions. For our experiments, we determined to use $\kappa = 0.8$ for selecting digressions by employing a manual evaluation on a small set of extracted digressions with different threshold levels. To compute the similarity between two paths (main and digression) we concatenate the canonical forms between start and end nodes on each path to compute its embedding. An example of a dialogue trajectory can be found in Table 1.

4 Experimental Settings

4.1 Datasets

We consider two widely used task-oriented dialogue datasets: Schema Guided Dialogue (SGD) (Budzianowski et al., 2018) and Action-Based Conversations Dataset (ABCD) (Chen et al.,

Domain	Dialogue Flow with Digression
GetWeather	user request weather information
	bot ask for city
	user provide city
	bot provide weather forecast
	when user ask for humidity
	bot provide humidity
	user request music
	bot recommend song
	user acknowledge recommendation
	bot ask if song should be played
	user confirm request to play song
	bot inform song is playing
	user thank
	bot ask if any other help is needed
	user thank and say goodbye
	bot say goodbye

Table 1: Examples of extracted dialogue trajectory with digressions.

Dataset	#Turns(Avg)	Graph	`gpt-4-turbo`
SGD	19.10	6.10 (7.98)	5.59 (7.66)
ABCD	11.40	4.61 (5.24)	4.04 (5.00)

Table 2: Comparison of mean LCS length between extracted (Graph, `gpt-4-turbo`) policies and conversations. Exact match LCS is outside brackets, similarity-based LCS is inside.

2021). A relevant aspect for our dialogue policy extraction task is that the conversations were generated using different approaches. SGD dialogues were generated by crowd-sourced paraphrasing of a set of dialogue sketches created automatically driven by a state-machine dialogue policy. Meanwhile, ABCD contains more realistic conversations between a client and a customer support agent (both non-experts) with the agent following a dialogue script resembling real-world customer support scenarios.

Schema Guided Dialogue (SGD). A comprehensive dataset containing 20 domains (or tasks) and 20k annotated conversations. These domains encompass a diverse range of user interactions relevant to an assistant use case, such as setting up calendars, looking for events, and making travel arrangements including different bookings.

Action-Based Conversations Dataset (ABCD). This dataset is designed to facilitate the development of more realistic customer service dialogue systems, primarily in the e-commerce setting. It contains over 10k human-to-human dialogues, which include the agent taking a specific sequence of actions to accomplish various tasks. The tasks span multiple domains, including managing account details, inquiring about the status of shipping, and handling processes related to initiating and monitoring refunds.

4.2 Baselines

Given the limited previous work, we consider the following alternatives for evaluating and comparing the efficacy of the proposed approach (§3).

Graph Traversal - Longest Path. This algorithm identifies the longest path in a directed graph, which, in this context, represents the longest sequence of dialogue turns in a conversation.

$$L_{max} = \max \{\text{len}(p) : p \in \mathcal{P}(s,t)\}$$

Graph Traversal - Maximum Weighted Path. This method computes the path in a graph that has the highest cumulative weight. In our case, the computed *happy* (main) path would maximize the number of conversations modelled by summing the frequency of each transition on that path.

$$P_{max} = \max \left\{ \sum_{e \in p} w(e) : p \in \mathcal{P}(s,t) \right\}$$

Prompting-Based Alternatives. Utilizing the set of conversations annotated with canonical forms (Sec. §3.1) as input, we prompt LLMs to generate the most suitable dialogue policy. We use OpenAI `gpt-4-turbo` and `gpt-3.5-turbo`, additional details are in Appendix C.

4.3 Evaluation

The dialogue policies generated by our method can be expressed in natural language as a sequence of subsequent canonical forms, similar to a conversation. Thus, our automatic evaluation strategy compares a conversation, as a sequence of canonical forms, with the dialogue policy. Additional evaluation details are in Appendix D (automatic) and E (manual, e.g. annotator instructions and interface).

4.3.1 Automatic Evaluation

Text Similarity Metrics. Extracting dialogue flows from conversations falls at the intersection of two well-defined language tasks. It can be framed as a translation problem, wherein the goal is to transform unstructured conversational data into a structured dialogue flow format. Additionally, we can view it as a multi-document summarization task (Ma et al., 2022) involving the distillation of multiple conversations into a concise dialogue flow representation. As the objective is to quantify the ability of the dialogue policy to model the conversations in the corpus, we use standard *text generation metrics* (BLEU, ROUGE, METEOR, and BERTSCORE) (Celikyilmaz et al., 2020) to assess the dialogue flow coverage and quality. To achieve this, we use the canonical forms representation for both conversations and the dialogue policy.

Structure-Preserving Metric. To evaluate how well a dialogue policy respects the structure of conversations and the sequential ordering of canonical forms, we utilize the *Longest Common Subsequence (LCS)*. The longest subsequence common to two sequences can be non-contiguous, but it respects the ordering of elements in each sequence. Let $C = \{c_1, c_2, \ldots, c_m\}$ represent the sequence of utterances in a conversation, and let $P = \{p_1, p_2, \ldots, p_n\}$ represent the sequence of actions in an extracted dialogue policy - both encoded as canonical forms. The LCS metric, denoted as $L(C, P)$, quantifies the number of utterances from conversation C that can be handled by policy P in exactly the same order, thereby providing a measure of how well the policy reflects the structure of the conversation.

We compute LCS using two methods: exact match and similarity-based match. Exact match extracts subsequences that have the same canonical forms both in the conversation and the policy, while the similarity-based match uses embedding similarity for matching canonical forms considering two elements a match if their similarity score exceeds a given threshold. This allows a more flexible matching that can correct some of the errors introduced by the intent identification stage.

4.3.2 Human Evaluation

For an in-depth assessment, we selected five domains from the SGD dataset: the best two for the graph-based method, the best two for the strongest baseline (`gpt-4-turbo`), and one domain where the performance gap was minimal (see Fig. 4). From each domain, we sampled 10 conversations and paired them with the dialogue flows extracted by each method (graph, `gpt-4-turbo`). Human annotators were then tasked with mapping each step in the dialogue flow to a corresponding turn in the actual conversation - more details are in Appendix E. In addition to mapping canonical forms in the dialogue flow to the corresponding turn in the conversation, annotators were also asked to rate how relevant the canonical form was to that turn. A score of 1 implied that the canonical form described the user utterance comprehensively, and a score of 0.5 meant that certain details in the utterance were not captured by the canonical form.

Precision and Recall of Policies. Through this detailed evaluation, we were able to determine the precision and recall of canonical forms used in the dialogue trajectory. Precision captures how many of the identified canonical forms correctly describe conversation turns and recall measures how well the canonical forms cover the actual turns in the conversation. This process allows us to validate the efficacy of the automatic metrics used in evaluating the extracted dialogue flows Let us assume that we have a sample conversation that goes like

User: "What's the weather like today?"
Bot: "The weather is sunny with a high of 75 degrees."
User: "Will it rain tomorrow?"
Bot: "No, it is expected to be clear all day tomorrow."
User: "What about this weekend?"
Bot: "It might rain on Saturday, but Sunday should be sunny."

The predicted dialogue trajectory for this conversation is:

```
user ask about weather
bot provide weather
user ask about weather tomorrow
bot provide weather
user ask about weather weekend
bot provide weather
bot provide weather
```

The human annotator is tasked with mapping turns in the conversation with the appropriate intent/canonical form from the dialogue trajectory.

User: "What's the weather like today?"
→ *user ask about weather*

Bot: "The weather is sunny with a high of 75 degrees." → *bot provide weather*

User: "Will it rain tomorrow?"
→ *user ask about weather tomorrow*

Bot: "No, it is expected to be clear all day tomorrow." → *bot provide weather*

User: "What about this weekend?"
→ *user ask about weather weekend*

Bot: "It might rain on Saturday, but Sunday should be sunny." → *bot provide weather*

Once we have this mapping, we see that the user canonical forms encode the state (similar to intent and slots in a standard NLU), while the bot canonical forms measure how well the predicted responses from the extracted policy match the bot responses in the conversation under evaluation. We then evaluate the performance in terms of micro-precision and micro-recall of the graph-based and prompt-based approaches on the "user" canonical forms and the "bot" canonical forms of the extracted policies.

5 Results and Analysis

This section provides a quantitative comparison of graph-based methods and prompting-based techniques. Additionally, a qualitative analysis, examining the variances in extracted dialogue flows, the effect of incorporating digressions into these flows, and the degree of flexibility and control provided by the graph-based approach is also presented.

5.1 Automatic Metrics

Text Similarity Metrics. Table 3 presents a comparative analysis of various graph-based and prompting-based methods using text-similarity metrics. In the graph-based category, the Longest-Path and Max-Weighted-Sum methods demonstrate similar performance, with minor variations in scores. The Fattest-width Dijkstra method significantly outperforms both methods in all metrics for both datasets (+8/+12 BLEU, +8/+11 ROUGE), indicating a more effective approach in extracting dialogue flows.

When compared with prompting-based approaches with LLMs, Fattest-width Dijkstra shows improved performance (+1.5 BLEU, +2 ROUGE) over GPT-4 as well. While an optimal prompt might lead to marginally improved scores, the trend suggests that the proposed graph-based method demonstrates a more consistent and effective way to extract dialogue flows.

Structure-Preserving Metric. Table 2 compares the mean LCS length between policies extracted by the graph and `gpt-4-turbo` methods. The policies extracted using the graph-based method consistently achieve higher LCS scores, indicating better alignment with the conversations. These results can be interpreted that about a third of all interactions can be correctly handled by the dialogue policies.

Relation to Intent Identification. All metrics used for automatic evaluation, including LCS, are dependent on the unsupervised intent identification (§3.1). Therefore it is important to determine the quality of this step.

To evaluate the accuracy of intent identification, we manually annotate conversations with canonical forms and compare these annotations with the canonical forms predicted by the p-tuned LLM. Semantic similarity between the predicted and ground-truth canonical forms is measured using MiniLM-v6. If the similarity score exceeds 0.8, the prediction is considered correct; otherwise, it is marked as incorrect. Using this approach, we achieve a user intent accuracy of 70% and a bot intent accuracy of 87% with the p-tuned LLM. When replacing the model with `text-davinci-003` for canonical form prediction, user intent accuracy improves to 86%, and bot intent accuracy increases to 93%. More recent models, such as LLaMa-3.1-70B-Instruct (Llama Team, 2024), further enhance performance, achieving 94% accuracy in user intent and 97% in bot intent identification.

5.2 Human Evaluation

The annotations from the human evaluation allow us to compute precision and recall metrics to evaluate the extracted dialogue flows. In our context, higher precision indicates that a higher number of canonical forms from the dialogue flow are utilized to describe turns in the conversations. A higher recall implies that a greater number of turns in the conversation are accurately covered by the canonical forms from the dialogue flow.

Table 4 shows that the dialogue flows extracted using the graph-based approach exhibit significantly higher precision compared to the flows from `gpt-4-turbo`. This suggests that the graph method is less noisy and more representative of the main flow of the interaction. The recall of the graph-based approach is marginally better than `gpt-4-turbo` indicating similar efficacy in capturing conversation turns. Table 7 in Appendix §4.3.2 shows a breakdown of performance for user and bot canonical forms.

		SGD				ABCD			
		BLEU	METEOR	ROUGE-L	BERTSCORE	BLEU	METEOR	ROUGE-L	BERTSCORE
Graph-based (***ours***)	Longest Path	19.16	42.94	40.76	45.04	18.39	33.12	38.45	36.46
	Max Weighted Sum	19.29	42.55	40.79	44.75	19.22	35.49	39.59	37.66
	Fattest-Width Dijkstra	27.87	54.27	48.87	52.78	30.08	48.26	50.10	46.65
	+ with 1 Digression	**28.54**	**54.31**	**49.23**	**52.97**	**30.93**	**48.44**	**50.40**	**47.83**
Prompting-based	gpt-3.5-turbo	25.52	49.77	47.54	51.37	27.64	43.54	47.88	44.31
	gpt-4-turbo	26.33	52.76	48.19	52.30	28.54	45.02	48.07	44.99

Table 3: Comparison of dialogue flow extraction methods using automatic metrics. Fattest path Dijkstra exhibits superior performance over other graph algorithms and surpasses `gpt-4-turbo` in prompting-based approach across SGD and ABCD datasets, while adding digressions provides even larger improvement.

Method	Precision	Recall
Graph	73.06	65.62
`gpt-4-turbo`	68.72	64.64

Table 4: Precision and Recall between extracted dialogue policies and human annotated conversations.

Domain	Graph	`gpt-4-turbo`
BookAppointment	0.59	0.51
SearchHotel	0.60	0.53
ReserveRestaurant	0.55	0.57
GetEventDates	0.43	0.52
PlayMedia	0.37	0.49

Table 5: Comparison of Graph and `gpt-4-turbo` scores for the 5 domains in SGD used for human evaluation.

The average scores for each domain are shown in Table 5. We observe that the trends of the manually annotated scores are consistent with automatic evaluations (see Fig. 4). Domains where the graph method outperforms `gpt-4-turbo` in the automatic evaluation, such as *'BookAppointment'* and *'SearchHotel'*, are reflected similarly by human annotators. This indicates a strong correlation between automatic metrics and human ratings. The lower scores for the graph method in specific areas can be attributed to the lack of digressions in the main dialogue flows used for human evaluation. Future improvements addressing this aspect could enhance the efficacy of the graph method.

5.3 Considerations for Developers

The proposed graph-based methodology offers several advantages over prompting-based techniques, particularly in terms of control and flexibility for conversation designers.

Controllability. Graph-based methods provide superior control, allowing designers to specify dialogue flow length, identify digressions, and decide which digressions to include. Prompt-based methods lack this fine-tuned control and interpretability, making precise modifications challenging. Graph-based methods allow control over dialogue flow length, allowing developers to balance precision and recall effectively.

Adding Digressions. Integrating digressions into dialogue flows enhances understanding of conversational dynamics. As shown in Table 3, adding a single digression improves all metrics by about 1 point. Graph-based methods facilitate precise identification and mapping of digressions, offering a clear visual representation of dialogue progression, which is beneficial for conversation modelling.

Robustness. Prompting-based approaches can be brittle and influenced by the order of conversation presentation, leading to inconsistent results. Graph-based methods produce deterministic outputs, ensuring predictable and consistent results regardless of input order, which is crucial for reliable conversation design.

6 Conclusion

Generating dialogue policies from a dataset of conversations can significantly reduce the effort required by conversation designers and domain experts to develop TODS. We propose a novel hybrid LLM and graph-based method to extract dialogue policies without relying on a predefined set of dialogue acts.

Our results are significant for three reasons. First, we demonstrate that dialogue policies can be computed using network flow in a graph of all possible conversations for a given task. Second, modeling conversations as sequences of canonical forms enhances explainability and controllability. Third, incorporating digressions as high-flow paths in the graph allows conversation designers to control the granularity of dialogue policies.

7 Limitations and Risks

The dialogue policies generated with the proposed approach are not perfect and should not be used to implement any TODS without careful inspection by a conversation designer or domain expert. Moreover, we acknowledge that in most cases the extracted dialogue flows will be iteratively improved by human experts. Therefore, our method is mainly intended to serve as a productivity tool. As the generated policies are expressed as sequences of canonical forms expressed in English it provides a good degree of explainability for the generated dialogue policies. At the same time, the mechanism for identifying digressions helps control the granularity and coverage of the dialogue policies and can be used by experts to analyze existing datasets.

A further limitation of our research is that we have not fully investigated the impact of various intent identification methods. For example, the clustering algorithm and sentence embeddings used by the intent normalization stage might influence the performance of our graph-based method. At the same time, other intent extraction methods described in Section §2 should also be compared to our proposed method. All these will go into future work and experiments.

At last, while in our work we have shown that the automatic evaluation using text generation metrics (e.g. BLEU, BERTSCORE) are correlated very well with the human evaluation on 5 different conversational tasks, this may not be the case on other conversation datasets. Therefore, we encourage developers that want to use this approach for evaluating the performance of the extracted dialogue flows to check first that the automatic metrics are well correlated with (at least a small) human annotated dataset that measures overlap between policies and conversations with domain experts.

The main risks of our approach is that the generated dialogue policies might contain canonical forms that are irrelevant or even malicious, but are extracted somehow from the corpus of conversations offered as input. However, we consider this should not be the case as the extracted policies should always be investigated and curated by a conversation designer.

8 Broader Impact

Upon acceptance for publication, we aim to release both the code and the generated dialogue policies for the ABCD and SGD datasets. In accordance with OpenAI terms of usage, this data would be available only for research purposes and would not be commercially usable. We also aim to release a set of conversations that are annotated with Llama3-70B-Instruct to enable better annotation quality. We consider that the existence of such a tool for extracting dialogue policies would benefit companies and developers that have access to datasets of task-oriented conversations.

References

Alberto Benayas, Miguel Angel Sicilia, and Marçal Mora-Cantallops. 2023. Automated creation of an intent model for conversational agents. *Applied Artificial Intelligence*, 37(1):2164401.

Tom Bocklisch, Joey Faulkner, Nick Pawlowski, and Alan Nichol. 2017. Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*.

Luc Bors, Ardhendu Samajdwer, and Mascha Van Oosterhout. 2020. Oracle digital assistant. *A Guide to Enterprise-Grade Chatbots. New York*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. 2018. MultiWOZ - a large-scale multi-domain Wizard-of-Oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026, Brussels, Belgium. Association for Computational Linguistics.

Asli Celikyilmaz, Elizabeth Clark, and Jianfeng Gao. 2020. Evaluation of text generation: A survey. *arXiv preprint arXiv:2006.14799*.

Ajay Chatterjee and Shubhashis Sengupta. 2020. Intent mining from past conversations for conversational agent. *arXiv preprint arXiv:2005.11014*.

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

Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2022. *Introduction to algorithms*. MIT press.

Bingzhu Du, Nan Su, Yuchi Zhang, and Yongliang Wang. 2023. A two-stage progressive intent clustering for task-oriented dialogue. In *Proceedings of The Eleventh Dialog System Technology Challenge*, pages 48–56, Prague, Czech Republic. Association for Computational Linguistics.

Patrícia Ferreira. 2023. Automatic dialog flow extraction and guidance. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 112–122, Dubrovnik, Croatia. Association for Computational Linguistics.

Google. 2024. DialogFlow Documentation | Google Cloud. Online at `https://cloud.google.com/dialogflow/doc`. Accessed: 2024-02-05.

Amine El Hattami, Stefania Raimondo, Issam Laradji, David Vázquez, Pau Rodriguez, and Chris Pal. 2022. Workflow discovery from dialogues in the low data regime. *arXiv preprint arXiv:2205.11690*.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. *Advances in Neural Information Processing Systems*, 33:20179–20191.

Rajat Kumar, Mayur Patidar, Vaibhav Varshney, Lovekesh Vig, and Gautam Shroff. 2022. Intent detection and discovery from user logs via deep semi-supervised contrastive clustering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1836–1853, Seattle, United States. Association for Computational Linguistics.

Chen-Yu Lee, Chun-Liang Li, Chu Wang, Renshen Wang, Yasuhisa Fujii, Siyang Qin, Ashok Popat, and Tomas Pfister. 2021. Rope: reading order equivariant positional encoding for graph-based document information extraction. *arXiv preprint arXiv:2106.10786*.

Sungjin Lee, Qi Zhu, Ryuichi Takanobu, Zheng Zhang, Yaoqin Zhang, Xiang Li, Jinchao Li, Baolin Peng, Xiujun Li, Minlie Huang, and Jianfeng Gao. 2019. ConvLab: Multi-domain end-to-end dialog system platform. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 64–69, Florence, Italy. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Xingkun Liu, Arash Eshghi, Pawel Swietojanski, and Verena Rieser. 2021. Benchmarking natural language understanding services for building conversational agents. In *Increasing Naturalness and Flexibility in Spoken Dialogue Interaction: 10th International Workshop on Spoken Dialogue Systems*, pages 165–183. Springer.

AI @ Meta Llama Team. 2024. The llama 3 herd of models.

Bo-Ru Lu, Yushi Hu, Hao Cheng, Noah A. Smith, and Mari Ostendorf. 2022. Unsupervised learning of hierarchical conversation structure. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 5657–5670, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Congbo Ma, Wei Emma Zhang, Mingyu Guo, Hu Wang, and Quan Z Sheng. 2022. Multi-document summarization via deep learning techniques: A survey. *ACM Computing Surveys*, 55(5):1–37.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training.

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.

Traian Rebedea, Razvan Dinu, Makesh Narsimhan Sreedhar, Christopher Parisien, and Jonathan Cohen. 2023. Nemo guardrails: A toolkit for controllable and safe llm applications with programmable rails. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 431–445.

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.

Pedro Henrique Piccoli Richetti, João Carlos de AR Gonçalves, Fernanda Araujo Baião, and Flávia Maria Santoro. 2017. Analysis of knowledge-intensive processes focused on the communication perspective. In *International Conference on Business Process Management*, pages 269–285. Springer.

Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Weiyan Shi, Tiancheng Zhao, and Zhou Yu. 2019. Unsupervised dialog structure learning. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 1797–1807, Minneapolis, Minnesota. Association for Computational Linguistics.

Makesh Narsimhan Sreedhar and Christopher Parisien. 2022. Prompt learning for domain adaptation in task-oriented dialogue. In *Proceedings of the Towards Semi-Supervised and Reinforced Task-Oriented Dialog Systems (SereTOD)*, pages 24–30, Abu Dhabi, Beijing (Hybrid). Association for Computational Linguistics.

Svitlana Vakulenko, Kate Revoredo, Claudio Di Ciccio, and Maarten de Rijke. 2019. Qrfa: A data-driven model of information-seeking dialogues. In *Advances in Information Retrieval: 41st European Conference on IR Research, ECIR 2019, Cologne, Germany, April 14–18, 2019, Proceedings, Part I 41*, pages 541–557. Springer.

Nikhita Vedula, Nedim Lipka, Pranav Maneriker, and Srinivasan Parthasarathy. 2020. Open intent extraction from natural language interactions. In *Proceedings of The Web Conference 2020*, pages 2009–2020.

A Implementation Details for Intent Identification

A.1 Prompt Template for Creating Weak Labels.

As mentioned in §3.1, we have used `text-davinci-003` from OpenAI to create an initial set of weak labels for 200 conversations from the ABCD dataset on 2 different tasks. For this, we have used the following prompt:

```
Your task is to annotate conversational
utterances with intents expressed as
canonical forms. Canonical forms are
short summaries representing the intent
of the utterance - it is neither too
verbose nor too short. Here is an
example to show you how the task is to
be performed.

{example}

Annotate the following conversation in a
similar manner. if similar intents are
detected, make sure to use the same
canonical forms as in the example given.
for other ones, use the ones in the
example above as reference and craft
them. Each turn of the conversation
should be annotated with the
corresponding canonical forms.

{conv}

Output the annotated conversation with
canonical forms.
```

The usage of `text-davinci-003` was based on its performance in generating canonical forms used for dialogue rails in NeMo Guardrails (Rebedea et al., 2023). While the model has been deprecated at the end of 2023, initial experiments show that the new model, `gpt-3.5-turbo-instruct` achieves a similar performance for this task. For all runs, we have used greedy decoding with temperature equal to 0.

A.2 P-Tuned LLM for Intent Identification with Weak Labels

For this study, we make use of our in-house 43 billion parameter model as the base LM. The 43B model is a decoder-only GPT architecture LLM that has been trained on 1.1 trillion tokens. It has 48 layers and uses a vocabulary size of 256 thousand, RoPE positional embeddings (Lee et al., 2021) and SwiGLU activation (Shazeer, 2020) without dropout. It was aligned using a combination of publicly available and proprietary alignment data. For p-tuning, we used a batch size of 8, learning rate of `1e-4`, number of virtual tokens as 30 and trained for 50 epochs with early stopping. The best performance was obtained at epoch 20. The training data is structured such that the model is trained to predict the canonical form for a particular turn given the conversation history up to that point. The training data consists of 850 samples and the validation data consists of 300 samples.

A.3 Intent Normalization

For SGD, we use agglomerative clustering with a clustering threshold of 0.9 and Euclidean distance as the metric. Similarly, for ABCD dataset the clustering threshold is set at 0.7. We select these clustering thresholds after running a hyperparameter search over a range of clustering threshold values (0.5-1.0).

B Examples of Generated Dialogue Policies

In Table 8 we show the dialogue policies extracted with the proposed graph-based method for 4 domains from the SGD dataset. For each domain, we can compare the main (happy) path with a flow containing one additional digression added to the main path. To tackle the branching of a flow (e.g. digression vs. main path) we are using some simple syntactic features supported by Colang flows (Rebedea et al., 2023), i.e. the special keyword `when`. This works by traversing the digression path only when the specific user intent in the `when` condition is met, otherwise continuing with the main path.

In addition to the dialogue flows, each domain also has a sample conversation (out of several hundreds) used to generate the flows.

C Details for Prompting-Based Dialogue Policy Generation

The following methodology has been used to generate the dialogue policy using prompting given the corpus of conversations for a task.

After several iterations, we have used the following prompt which provides good results for the task of generating a dialogue flow from a set of conversations modelled using canonical forms.

```
Here is a list of dialogue flows that
denote how conversations usually proceed
between a user and a bot. Your task is
to create a dialogue flow that best
represents the conversation flow given
all the dialogue flows below.

{conversations_with_canonical_forms}

What is the most commonly traversed path
in this set of conversations? Output it
following a similar format as the
conversations above. Only display the
output path. Do not add any comments or
other text.
```

Due to context length limitations, we utilize a batch of 100 conversations as input for the LLM (and the graph method). Following this, we extract the dialogue flow from these conversations.

D Automatic Evaluation Metrics by Domain

The evaluation is always conducted using a distinct set of conversations that were not included in the batch of 100 conversations used for extracting the dialogue flows. For example, in a domain with 300 conversations, the dialogue flow is extracted from the 100 conversations at a time and evaluated against the remaining 200. This helps ensure a fair evaluation and mitigates the risk of overfitting.

In Figs. 4 and 5 we provide the BLEU and ROUGE scores for each task in the SGD and ABCD datasets for the top two performing methods: Graph and GPT-4. In each graph, the tasks are ordered from left to right based on the value of the difference in performance on that metric between the proposed graph-based method and GPT-4 prompting. We can easily see that the Graph method is out-performing GPT-4 in more than 60% of the tasks for both datasets.

E Manual Evaluation

For manual evaluation, we use 8 annotators. For this task, we selected volunteers instead of relying on crowd-workers. Each volunteer has at least a MSc in Computer Science or related domain, being at least knowledgeable in NLP. Each annotator was tasked with reviewing 25 pairs of conversation and associated dialogue flow. More, annotators received a balanced distribution between flows generated by the Graph-based method and by GPT-4. The annotators are asked to map the canonical forms in the dialogue flow to the corresponding conversation turn, as well as assign a score to the canonical form (4.3). The annotation UI is shown in Fig. 6. The annotators are not made aware of which method is used to extract the dialogue flow to prevent any potential bias.

Before starting the annotations, one of the authors of the paper provided about 10 pairs of conversation and dialogue flow as samples annotations and also a short guide of about 3-4 pages on the annotation process. The guide included an explanation of the task, the annotation UI, and had a short lost of Q&A. The time required per annotator was about 2 hours and the annotators were paid for this task.

In order to have consistent manual annotations, each pair of conversation and dialogue flow was labelled by two different annotators. We have obtained a substantial inter-rater agreement, Cohen's κ=0.71, considering a binary classification task for the canonical forms in the dialogue flow (matched or not matched by a turn in the current conversation).

Metric	Minimum	Maximum	Average	Standard Deviation
BLEU	25.6	26.5	26.1	0.3
ROUGE-L	47.7	48.4	47.9	0.26

Table 6: Variance in BLEU and ROUGE-L metrics across 5 runs of the prompting-based method using gpt-4-turbo.

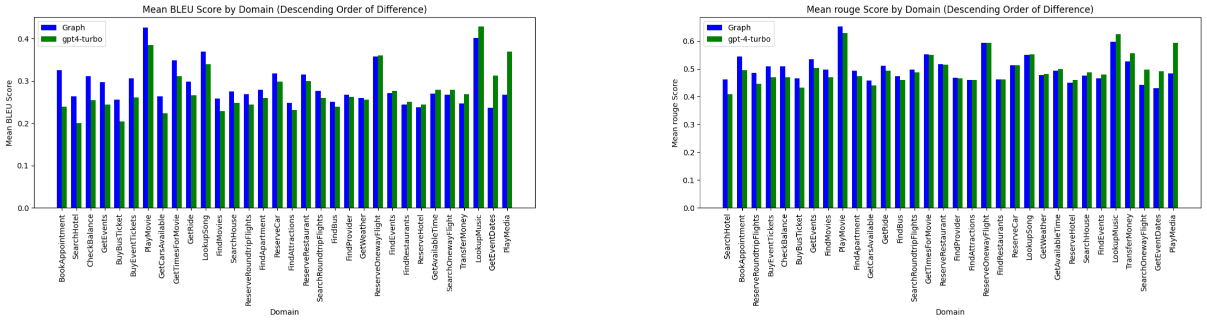
F Variance in LLM output

To evaluate the effect of altering the sequence of conversations, we prompted gpt-4-turbo to extract the dialogue flows for all domains in the SGD dataset. This process was repeated five times for each domain, with the same set of conversations in the prompt, but with their order randomized in each iteration.

Table 6 indicates a relatively narrow range in both BLEU and ROUGE scores indicating consistent performance. However, determinism is preferred as it ensures reproducibility and reliability in the results.

	User - Precision	User - Recall	Bot - Precision	Bot - Recall
Graph	76.5	68.3	78.8	68.4
gpt-4-turbo	71.4	66.8	73.3	67.5

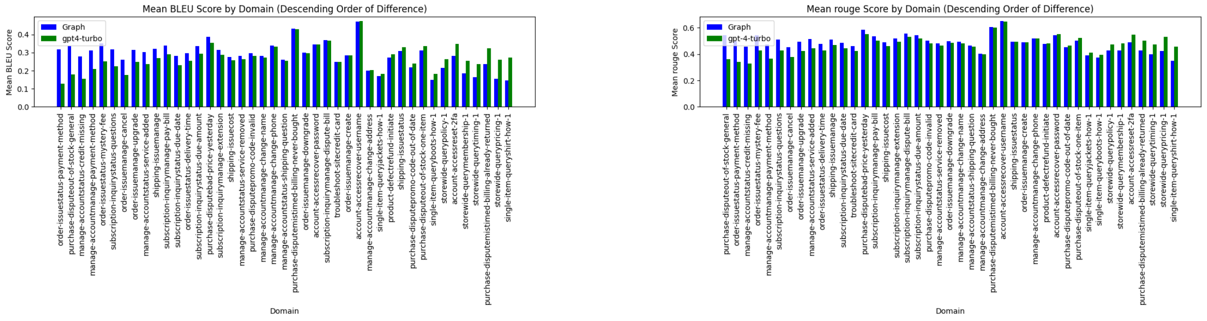
Table 7: Performance metrics of the graph-based approach and gpt-4-turbo model in terms of precision and recall for user and bot canonical forms when compared with human annotations.



(a) BLEU

(b) ROUGE-L

Figure 4: Performance obtained by the proposed graph-based method and the top performing baseline (`gpt-4-turbo`) for the SGD dataset. The results are ordered from left to right on the value of the difference in performance between graph-based and GPT-4.



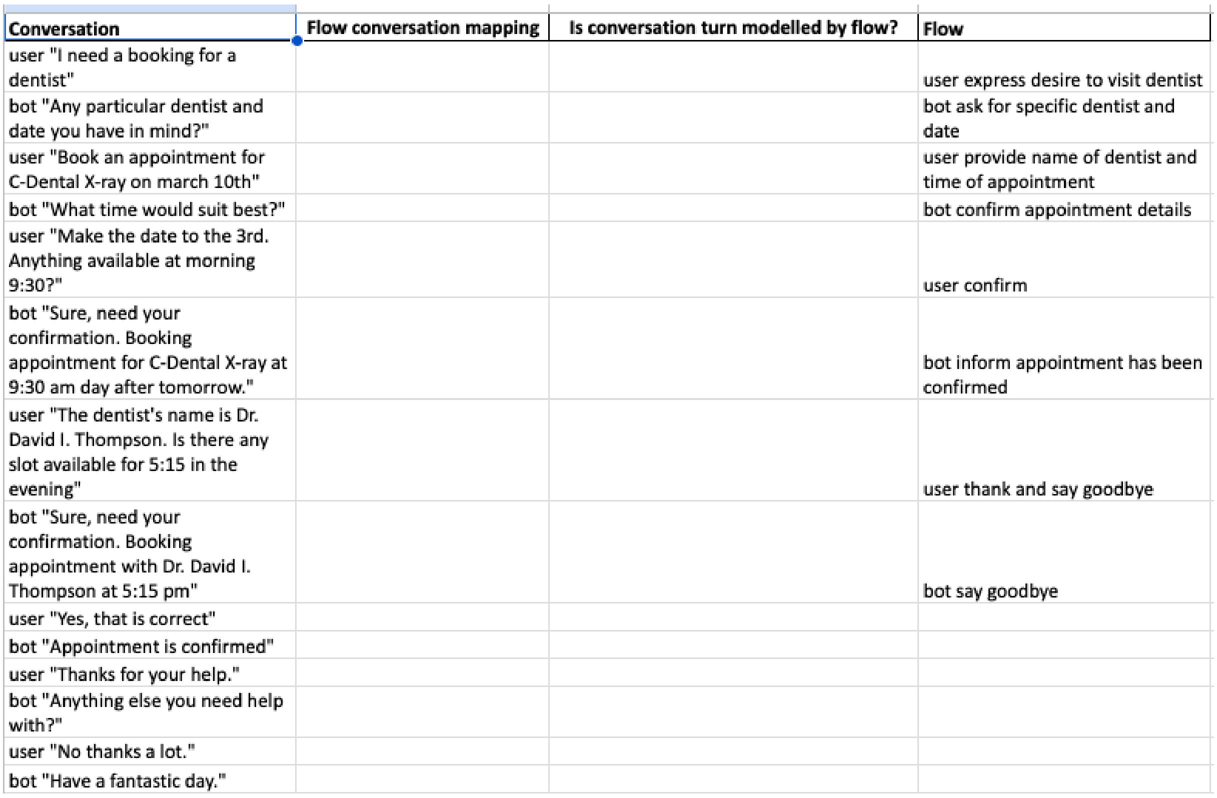
(a) BLEU

(b) ROUGE-L

Figure 5: Performance obtained by the proposed graph-based method and the top performing baseline (`gpt-4-turbo`) for the ABCD dataset. The results are ordered from left to right on the value of the difference in performance between graph-based and GPT-4.

Domain	Dialogue Flow	Dialogue Flow with Digression	Conversation
PlayMedia	user request to play song bot ask for song to play user provide song name bot confirm request to play song user confirm bot inform movie is playing user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user request to play song bot ask for song to play user provide song name bot confirm request to play song user confirm bot inform movie is playing ***when*** user request to make transfer bot ask for amount to transfer user provide amount to transfer bot confirm transfer request user confirm bot inform movie is playing user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user "I wish to play the song. Broadcast it on my display. I love Kacey Musgraves song." bot "Which song do you need?" user "I need the song Slow Burn." bot "Kindly ensure that, Slow Burn is will play on TV." user "Yes. it is perfect." bot "Song is started." user "Thank you so much." bot "Do you need any other help?" user "I wish to make a transfer amount from my checking account to savings account of Jerry." bot "How much to be transferred?" user "Please transfer 1110 bucks." bot "Kindly ensure that, $1,110 is transferred from my checking account to savings account of Jerry." user "Yes. It is correct." bot "The transfer is made successfully." user "I wish to play the movie on internet now." bot "What movie do you wish to watch?" user "Play Mary Magdalene with subtitles." bot "Kindly ensure that, playing Mary Magdalene without subtitles." user "Yes. What type of genre it is? Who is the director?" bot "Movie started. It is a Drama movie directed by Garth Davis." user "Thank you so much." bot "Do you need any other help?" user "No. Thank you so much." bot "Have a fantastic day."
GetWeather	user request weather information bot ask for city user provide city bot provide weather forecast user request music bot recommend song user acknowledge recommend song bot ask if song should be played user confirm request to play song bot confirm request to play song user confirm bot inform song is playing user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user request weather information bot ask for city user provide city bot provide weather forecast ***when*** user ask for humidity bot provide humidity user request music bot recommend song user acknowledge recommend song bot ask if song should be played user confirm request to play song bot inform song is playing user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user "I need the weather forecast for the 7th." bot "What city would you like to check the weather?" user "Please check Ciudad de Mexico" bot "For that day there is a 6 percent chaning of rain and the average temperature should be 93 degrees Fahrenheit." user "What will the humidity be? Will it be windy?" bot "The humidity will be around 24 percent and the average wind speed 5 miles per hour." user "Thanks. Can you sind some songs for me?" bot "I found 10 songs. Do you like Stitches by Shawn Mendes on the album Handwritten?" user "That sounds great. Please play the song." bot "Please confirm: play Stitches on the TV." user "That is correct. What genre is that?" bot "The song is now playing. It's a pop song." user "Thanks." bot "Can I assist you with something else?" user "No thanks." bot "Enjoy your day."
LookupSong	user request to find a song bot suggest song user confirm song recommendation bot ask if user wants to play song user request to play song on kitchen speaker bot confirm request to play song on kitchen speaker user confirm bot inform song is playing user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user request to find a song bot suggest song ***when*** user ask for genre bot provide genre user confirm song recommendation bot ask if user wants to play song user request to play song on kitchen speaker bot confirm request to play song on kitchen speaker user confirm bot inform song is playing user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user "I want to hear a song from nine years back." bot "How about Alone by Alan Walker from the album Different World?" user "What genre of music is that?" bot "It's labeled as a Pop song." user "Okay, that one will work. Please play it on the Kitchen speaker." bot "To confirm: You want to play Alone on the Kitchen speaker." user "Sorry, no. Please play it on the speaker in the bedroom." bot "Okay. So, I'll play the song Alone on the Bedroom speaker." user "That's right. Please go ahead." bot "Sure. Enjoy the music!" user "Thanks. That's all for now." bot "Have a wonderful day!"
FindMovies	user express desire to watch a movie bot ask for location user provide location bot suggest movies user confirm bot ask if user wants to buy tickets user decline to buy tickets bot ask if any other help is needed user thank and say goodbye bot say goodbye	user express desire to watch a movie bot ask for location user provide location bot suggest movies ***when*** user ask for showtime bot provide showtime user acknowledge provide showtime user confirm bot ask if user wants to buy tickets user decline to buy tickets bot ask if any other help is needed user thank and say goodbye bot say goodbye	user "Look for a movie for me to watch." bot "Where are you located?" user "Find me movies in Union City." bot "There are 10 movies there. What do you think about Breakthrough, Captain Marvel or After?" user "Captain Marvel sounds great. At what time could I watch it?" bot "What date do you want to see it on?" user "I'd like to watch it on the 7th of March." bot "I found 1 showtime for Captain Marvel. There is one at 9 pm in Century 25 Union Landing & XD." user "Sounds good." bot "Do you want to buy tickets?" user "No, I do not want to buy them right now." bot "Do you need any more help?" user "No, thanks." bot "Enjoy your day."
GetRide	user request for cab bot offer shared ride user provide number of passengers and request shared ride bot ask for destination user provide destination bot ask for shared ride user confirm shared ride bot confirm request for shared ride user confirm bot inform ride has been booked user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user request for cab ***when*** bot ask for number of seats user provide number of seats bot ask for ride type user provide preferred ride type bot confirm request for ride bot offer shared ride user provide number of passengers and request shared ride bot ask for destination user provide destination bot ask for shared ride user confirm shared ride bot confirm request for shared ride user confirm bot inform ride has been booked user thank bot ask if any other help is needed user thank and say goodbye bot say goodbye	user "I need to go to 19369 Stevens Creek Boulevard Suite 130 for which I need a cab with Luxury ride. Can you find one?" bot "Can you tell me the number of seats that you need?" user "Yes, I need only one." bot "Please confirm to book a Luxury cab with 1 seat to 19369 Stevens Creek Boulevard Suite 130." user "can you check for a Cab with pool ride for 4 persons?" bot "please confirm to book a Cab with Pool ride with 4 seats." user "Yes, that is okay for me." bot "Your booking was confirmed and the Cab is on the way to your location." user "Can you tell me the exact time ***when*** the Cab will reach my place?" bot "will be there in 3 minutes." user "Yes, that is great, thanks for your work." bot "Anything else you need?" user "No, thanks, that's all I need." bot "Have a good day."

Table 8: Several examples of extracted dialogue policies for different domains in SGD dataset using the proposed graph-based method, with and without digressions, accompanied by a sample conversation from each domain.



Conversation	Flow conversation mapping	Is conversation turn modelled by flow?	Flow
user "I need a booking for a dentist"			user express desire to visit dentist
bot "Any particular dentist and date you have in mind?"			bot ask for specific dentist and date
user "Book an appointment for C-Dental X-ray on march 10th"			user provide name of dentist and time of appointment
bot "What time would suit best?"			bot confirm appointment details
user "Make the date to the 3rd. Anything available at morning 9:30?"			user confirm
bot "Sure, need your confirmation. Booking appointment for C-Dental X-ray at 9:30 am day after tomorrow."			bot inform appointment has been confirmed
user "The dentist's name is Dr. David I. Thompson. Is there any slot available for 5:15 in the evening"			user thank and say goodbye
bot "Sure, need your confirmation. Booking appointment with Dr. David I. Thompson at 5:15 pm"			bot say goodbye
user "Yes, that is correct"			
bot "Appointment is confirmed"			
user "Thanks for your help."			
bot "Anything else you need help with?"			
user "No thanks a lot."			
bot "Have a fantastic day."			

Figure 6: Annotation UI for the Manual Evaluation