A Computational Approach to Modeling Conversational Systems: Analyzing Large-Scale Quasi-Patterned Dialogue Flows

Mohamed Achref Ben Ammar
Software Engineering and Mathematics Department
INSAT
Tunis, Tunisia
Callem.AI
Paris, France
achref.benammar@ieee.org

***Abstract*—The analysis of conversational dynamics has become increasingly important with the rise of large language model-based conversational systems. As these systems interact with users across diverse contexts, understanding and representing the underlying patterns in conversations are critical for ensuring consistency, reliability, and dependability. In this work, we present a novel computational framework for constructing conversational graphs that effectively capture the flow and patterns within sets of conversations that do not follow strict conversational structures but nevertheless exhibit common underlying conversational flow patterns—referred to as quasi-patterned sets of conversations. Our approach combines advanced embedding techniques, clustering, and large language models to extract intents and transitions, leading to a clear, interpretable graph representation. Through comparative analysis of various graph simplification methods, we demonstrate that our Filter & Reconnect method produces the most readable and insightful graphs, allowing for the visualization of complex conversational flows with minimal noise. This work offers a solution for analyzing large-scale dialogue datasets, with practical applications for enhancing automated conversational systems.**

***Index Terms*—Dialogue systems modeling, conversation pattern recognition, quasi-patterned dialogue, natural language processing, dialogue structure representation, stochastic modeling, data-driven conversational insights, and conversation flow analysis.**

I. Introduction

With the rise of large language model (LLM)-based conversational systems, the analysis of conversational dynamics has gained increasing importance. Understanding how conversations flow, identifying underlying intents, and modeling dialogue structures are critical for improving the effectiveness of these systems in real-world applications. Recent advancements in natural language processing (NLP) [1] have facilitated progress in conversational systems modeling techniques such as intent extraction [2] and dialogue act modeling [3]. While dialogue act modeling classifies utterances into communicative functions using models like Long Short-Term Memory (LSTM) networks [4], intent extraction identifies the underlying intent of an utterance through models like biLSTM-CRF [5]. However, these techniques primarily focus on analyzing conversations in isolation rather than exploring recurring patterns across sets of conversations.

In this work, we propose a computational approach to constructing conversational graphs; each node in a conversational graph represents an intent, while directed edges denote transitions between them, weighted by the frequency or probability of such transitions.

Our approach focuses on quasi-patterned sets of conversations, which are collections of dialogues that, while not adhering to strict conversational structures, still display common underlying patterns in the flow of intents. These conversations may vary in specifics, but they tend to follow similar intent transitions and themes, reflecting consistent conversational dynamics across multiple interactions.

The primary steps of our approach include utterance embedding using Sentence-BERT (SBERT) [6], clustering [7], and intent extraction through LLMs [8]. Following these steps, we construct a transition matrix and generate the conversational graph. These processes will be thoroughly detailed in the following sections of this paper.

The paper is structured as follows: Section II reviews related work, Section III describes the dataset and its selection rationale, Section IV outlines our methods and evaluation, Section V presents the results and discussion, Section VI concludes the study, and Section VII addresses limitations.

II. Related Work

The analysis of conversational systems has been a growing area of research, particularly with the proliferation of natural language processing (NLP) techniques and the development of large language models (LLMs). Various approaches have been proposed to model dialogue flows and extract meaningful patterns from conversational data. This section provides an overview of the major works pertaining to the topic of conversational systems modeling, from intent extraction and dialogue act modeling to the applications of conversational systems modeling in customer support, highlighting the gap our framework seeks to address.

A. Intent Extraction And Dialogue Act Modeling

Intent extraction aims to identify the underlying intention or goal behind user utterances. Techniques for intent extraction have evolved from rule-based approaches [9] to machine learning-based models, which leverage semantic embeddings and contextual features to improve accuracy [2]. Pre-trained language models, such as BERT [10], have further enhanced intent extraction capabilities by enabling models to capture rich contextual information. However, these models often focus on individual utterances without considering their place in the broader dialogue flow.

Dialogue act modeling seeks to classify utterances into communicative functions such as questions, requests, or confirmations. This line of research has produced notable results in labeling conversational data using supervised learning techniques [3], often focusing on specific domains such as customer service or task-oriented dialogue systems. While dialogue act models capture the functional role of each utterance, they tend to operate on individual conversation segments rather than holistically across sets of conversations.

Both intent extraction and dialogue act modeling are vital to our work, as they enable the identification of key components within dialogues. However, they fall short in addressing the higher-level patterns found in large-scale, quasi-patterned conversations. Our framework builds on these foundational approaches but moves beyond isolated utterances to explore the relationships and transitions between intents across conversations.

B. Applications in Customer Support and Automated Systems

The practical implications of conversational systems modeling are particularly relevant for customer support systems. Prior research has demonstrated the effectiveness of NLP techniques in automating responses and improving user experience in customer service settings [11]. By analyzing large-scale, quasi-patterned customer support conversations, our framework provides a means of improving conversational agents by identifying common paths, detecting bottlenecks, and suggesting optimizations based on real conversational data.

Our work contributes to this area by offering a solution for analyzing unstructured, real-world dialogues, thus enabling more adaptive and responsive automated systems.

III. Dataset

The dataset chosen for this study is the ABCD (Action-Based Conversations Dataset) [13], which contains customer support conversations between agents and customers. This dataset is particularly well-suited for our research because customer support interactions often loosely follow guided paths. Customers generally have recurring types of inquiries, leading to recurring patterns in the agent's flow of responses. While these conversations do not follow a strict dialogue structure, the patterns observed in the agents' actions and responses across multiple conversations provide sufficient regularity to classify this dataset as a quasi-patterned set of conversations.

The distribution of utterance lengths in the dataset is shown in Figure 1. As seen from the histogram, most utterances are relatively short, containing fewer than 15 words, with a smaller number extending to 30 or more words. This distribution aligns with typical customer support interactions, where shorter, more direct exchanges are common.

Furthermore, the utterance length distribution by role in Figure 2 highlights that agents tend to use shorter utterances, with most containing fewer than 5 words. Customers, on the other hand, tend to have slightly longer utterances, which more often exceed 10 words. This distinction between agent and customer utterances reflects the typical nature of customer service dialogues, where agents provide concise responses and customers may explain their issues in more detail.

This makes it an ideal dataset for constructing and analyzing conversational graphs that aim to capture the quasi-patterned nature of such sets of interactions.

In this context, several key terminologies are used throughout the dataset:

- **Agent**: Refers to the customer support agent handling the interaction.
- **Customer**: Refers to the customer making the inquiry or seeking support.
- **Action**: Represents a task or operation performed by the agent, such as accessing a database, searching through an FAQ page, or retrieving customer information. These actions are an integral part of the dialogue and are often triggered by specific customer requests or queries.

To provide a quantitative overview of the dataset, the general statistics metrics are summarized in Table I.

IV. Methodology

In this section, we present the methodology of our approach. Figure 3 presents a general overview of our pipeline. The process begins by extracting utterances from the dataset, then each utterance is embedded by a text-embedding model [6],

Fig. 1: Distribution of utterance lengths (in words) in the dataset.

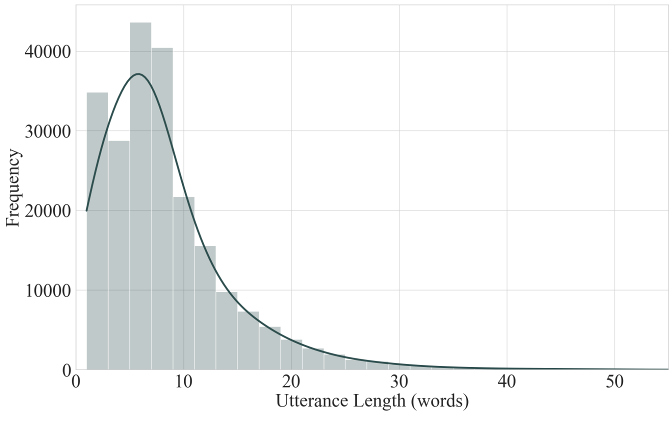
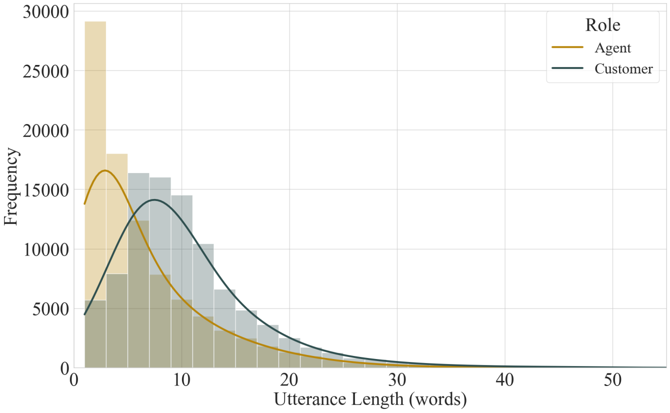


Fig. 2: Distribution of utterance lengths (in words) by role.



thus transforming them into vectors that represent the semantic meaning of each utterance.

These vectors are then clustered to identify the key utterance intents contained in the quasi-patterned set of conversations using an LLM. After that, we construct a Markov Chain, which will be filtered and processed to discard the noise and irrelevant intent transitions from the graph, and thus we create a conversational graph that can be analyzed to extract conversational flow patterns.

A. Vector Embedding Generation

The first step involves generating vector embeddings for each utterance using a pre-trained text embedding model [6]. Formally, let $U = \{u_1, u_2, \ldots, u_N\}$ be the set of utterances. A pre-trained embedding model $f : U \to \mathbb{R}^d$ maps each utterance u_i to a vector $\mathbf{x}_i = f(u_i)$. By doing so, we capture semantic and contextual properties of the utterances in a continuous metric space. For this work, we utilized the all-MiniLM-L12-v2 model [6], which converts textual data into high-dimensional vectors $\mathbf{x}_i \in \mathbb{R}^d$ that represent the semantic meaning of the utterances. These embeddings form the foundation for clustering, as they encapsulate both syntactic and semantic similarities [7]. By using embeddings that capture the contextual meaning of utterances, we ensure that the clusters formed represent coherent thematic groups.

B. Clustering and Intent Extraction

Once the vector embeddings $\{\mathbf{x}_i\}_{i=1}^N$ of the utterances are generated, we use K-means++ [12] for clustering. Let N_c be the number of clusters. K-means aims to find $\{\boldsymbol{\mu}_c\}_{c=1}^{N_c}$ that minimize:

$$\min_{\{\boldsymbol{\mu}_c\}} \sum_{i=1}^{N} \min_{1 \le c \le N_c} \|\mathbf{x}_i - \boldsymbol{\mu}_c\|^2.$$

K-means++ strategically initializes centroids $\boldsymbol{\mu}_c \in \mathbb{R}^d$ to be well-spread, leading to faster convergence and more accurate clusters. This reduces the likelihood of poor clustering due to random initialization, while maintaining the simplicity and scalability of K-means. The algorithm is efficient for large datasets, making it ideal for grouping utterances into meaningful thematic categories with minimal computational cost.

TABLE I: General Dataset Statistics for ABCD Dataset

General Dataset Statistics	Value
Average dialogue length (iterations)	22
Average dialogue length (characters)	904
Average dialogue length (words)	175.17
Maximum dialogue length (words)	632
Minimum dialogue length (words)	35
Median dialogue length (words)	166.00
Variance of dialogue length (words)	3683.11
Standard deviation of dialogue length (words)	60.69

After clustering, we proceed with intent extraction from the clusters. To achieve this, we select the *top_n* vectors, with $n = 15$, closest to the centroid of each cluster and extract the common intent from the corresponding utterances. Let $I_c = \{i : u_i \text{ is assigned to cluster } c\}$ be the set of indices of utterances in cluster c. We then sort the utterances in I_c by their distance to the centroid $\boldsymbol{\mu}_c$, i.e., we find a permutation $(i_1, i_2, \ldots, i_{|I_c|})$ of I_c such that:

$$\|\mathbf{x}_{i_1} - \boldsymbol{\mu}_c\| \le \|\mathbf{x}_{i_2} - \boldsymbol{\mu}_c\| \le \cdots \le \|\mathbf{x}_{i_{|I_c|}} - \boldsymbol{\mu}_c\|.$$

We then define:

$$S_c = \{u_{i_j} : 1 \le j \le n\},$$

which is the set of n utterances in cluster c whose embeddings are closest to $\boldsymbol{\mu}_c$. For intent extraction, we leverage the gemini-1.5-flash model [14] to summarize the core intent of each cluster, enabling us to label the groups with meaningful thematic descriptions.

C. Markov Chain Construction

Following clustering and intent extraction, we build a Markov chain [15] to model the transition probabilities between clusters based on the conversations' flow. A Markov chain is a stochastic model that describes transitions from one state to another according to defined probabilities. In this context, the states correspond to the different clusters, and the transitions between them represent the flow of intents throughout the conversations.

Let $S = \{s_1, s_2, \ldots, s_N\}$ denote the sequence of cluster IDs assigned to the N utterances in the order they appear in the conversations. We construct a transition matrix $T \in \mathbb{R}^{N_c \times N_c}$, where N_c is the number of unique clusters. The matrix element $T_{i,j}$ is defined as:

$$T_{i,j} = \frac{\text{count}(s_t = i, s_{t+1} = j)}{\sum_{j=1}^{N_c} \text{count}(s_t = i, s_{t+1} = j)},$$

where $\text{count}(s_t = i, s_{t+1} = j)$ is the frequency of transitions from cluster i to cluster j observed in the dataset. Each row in the transition matrix T sums to 1, indicating a probability distribution of transitions from one cluster to the next. Therefore, $T_{i,j}$ expresses the probability of moving from cluster i to cluster j.

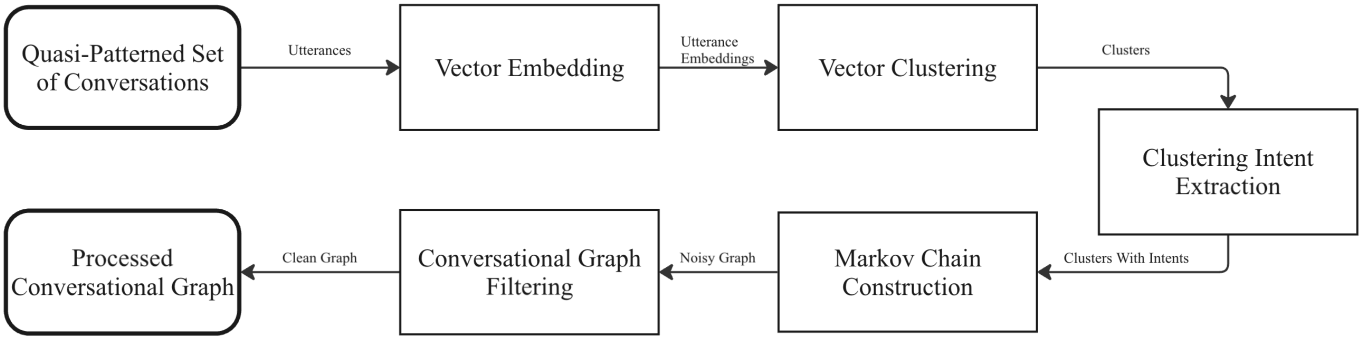

Quasi-Patterned Set of Conversations
Utterances
Vector Embedding
Utterance Embeddings
Vector Clustering
Clusters
Clustering Intent Extraction
Clusters With Intents
Markov Chain Construction
Noisy Graph
Conversational Graph Filtering
Clean Graph
Processed Conversational Graph

Fig. 3: Flowchart showing the process of conversational graph construction.

The transition matrix T encapsulates the probabilistic flow of intents between clusters, representing the natural progression of themes and topics in the dialogues. This Markov chain model maps how conversational elements evolve, providing insights into the underlying recurring patterns of the conversations; it will be considered as our base conversational graph for the next steps.

D. Conversational Graph Processing

After constructing the Markov chain, we refine the conversational graph using three different processing techniques to best fit the analysis of quasi-patterned conversations. We represent the conversational graph as $G = (V, E)$, where $V = \{1, \ldots, N_c\}$ is the set of cluster nodes and E consists of directed edges weighted by the transition probabilities $T_{i,j}$. Each technique simplifies the graph in different ways, aiming to improve readability and interpretability while maintaining meaningful conversational paths. These techniques are: Threshold Filtering, Top-K Filtering, and Filter & Reconnect.

1) Threshold Technique: This method simplifies the graph by removing edges with weights below a certain threshold τ:

$$E_\tau = \{(i, j) \in E : T_{i,j} \geq \tau\}.$$

A higher threshold τ results in a sparser graph with fewer, more significant transitions, while a lower τ retains more transitions but may introduce noise.

2) Top-K Filtering Technique: This approach retains the top K highest-weighted edges for each node while also applying a minimum weight threshold τ. For each node i, let π_i be a permutation of its neighbors sorted in descending order by $T_{i,j}$. Define:

$$E_{K,\tau} = \{(i, j) : T_{i,j} \geq \tau \text{ and } j \in \{\pi_i(1), \ldots, \pi_i(K)\}\}.$$

This ensures that the most likely transitions are retained, providing a focused representation of the conversational graph.

3) Filter & Reconnect Method: This method constructs an acyclic conversational graph by filtering and simplifying transitions. First, edges with weights below a threshold τ and self-transitions are removed. For each node, only the top K strongest incoming edges are retained. Next, any cycles in the graph are identified, and the weakest edges within them are removed to ensure the graph remains acyclic. Finally, any subgraphs that were disconnected during this process are reconnected to the main graph by restoring the strongest transitions, ensuring that the final structure remains interpretable and tree-like.

E. Evaluation

We evaluate our approach by determining the readability and interpretability of the final graphs produced by our approach. For readability, we evaluate it by determining the number of cycles in that graph; the more cycles a conversational graph has, the less readable it becomes and the harder it is to understand our quasi-patterned set of conversations. And for interpretability, we evaluate the graph by determining how much that graph looks like a tree. That is because quasi-patterned sets of conversations generally follow a tree structure, with branches being determined by the different paths conversations can take as they unfold. For this, we calculate:

- **Branching Factor**: The average number of edges per vertex in the graph.
- **δ-hyperbolicity**: Measures how close a graph is to being Gromov-hyperbolic [16], meaning it quantifies how much the graph's metric resembles a hyperbolic space [17] by analyzing the slimness of triangles formed by shortest paths in the graph, which correlates with tree-like structures because trees are the simplest examples of hyperbolic spaces. A graph with low delta-hyperbolicity has properties similar to a tree, where the distance between nodes can often be efficiently described by paths that are close to each other, much like the branching nature of trees.

V. Results and Discussions

In this section, we present the outcomes of applying three distinct graph simplification techniques—Threshold Filtering, Top-K Filtering, and Filter & Reconnect—on the conversational dataset. Each method is evaluated based on key performance metrics: delta-hyperbolicity, branching factor, and the number of cycles in the resulting graphs. We also provide visual representations of the generated conversational graphs to facilitate comparative analysis and insight.

A. Threshold Filtering Results

The Threshold Filtering method, while effective at removing low-probability transitions, does not perform well in generating interpretable conversational graphs. The primary issue is the significant presence of noise, which results in a high number of cycles (56 cycles). As shown in Figure 4, the threshold filtering approach performs poorly in terms of the delta-hyperbolicity metric (2.50), indicating that the resulting graph deviates substantially from a tree-like structure. This renders the graph less readable and interpretable for understanding the conversational flow.

B. Top-K Filtering Results

This method significantly reduces the complexity of the graph by retaining only the most important transitions. It excels in terms of delta-hyperbolicity, achieving a perfect score of 0, which indicates that the graph closely resembles a tree. However, this method also introduces a major limitation: it results in disconnected subgraphs. As shown in Figure 5, although the number of cycles is reduced to 2, the disconnected nature of the graph hinders its ability to effectively capture the natural flow of conversation between intents and topics.

C. Filter & Reconnect Results

This method produces the most interpretable and coherent conversational graph. As shown in Figure 6, this method ensures a tree-like structure by eliminating all cycles (0 cycles) while preserving meaningful branching of intents. The resulting graph showcases a clear conversational flow, making it highly readable and interpretable. In addition, the delta-hyperbolicity metric remains at 0, indicating that the graph closely approximates a tree structure, which is ideal for representing the flowing nature of conversation intents.

Fig. 4: Conversational graphs generated using threshold filtering for (τ = 0.1).

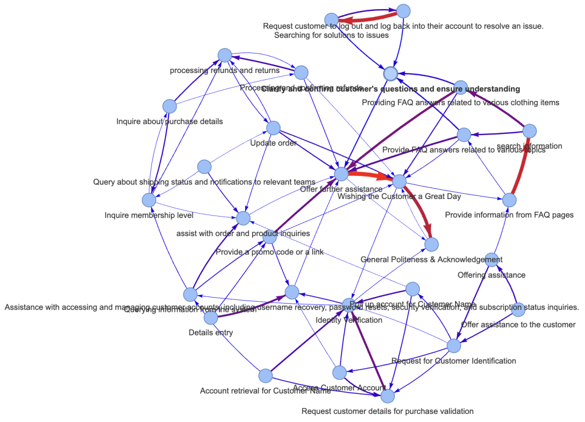

Request customer to log out and log back into their account to resolve an issue.
Searching for solutions to issues
processing refunds and returns
Processing and confirming refunds
Clarify and confirm customer's questions and ensure understanding
Providing FAQ answers related to various clothing items
Inquire about purchase details
Update order
Provide FAQ answers related to various topics
search information
Query about shipping status and notifications to relevant teams
Offer further assistance
Wishing the Customer a Great Day
Inquire membership level
Provide information from FAQ pages
assist with order and product inquiries
Provide a promo code or a link
General Politeness & Acknowledgement
Offering assistance
Assistance with accessing and managing customer accounts, including username recovery, password resets, security verification, and subscription status inquiries.
Querying information from the system
Pull up account for Customer Name
Identity Verification
Details entry
Offer assistance to the customer
Request for Customer Identification
Account retrieval for Customer Name
Access Customer Account
Request customer details for purchase validation

Fig. 5: Conversational graphs generated using Top-K filtering for (top-k = 1 and τ = 0.1).

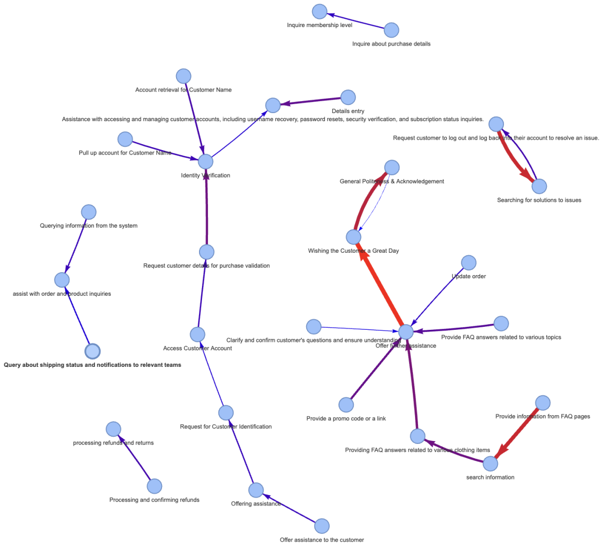
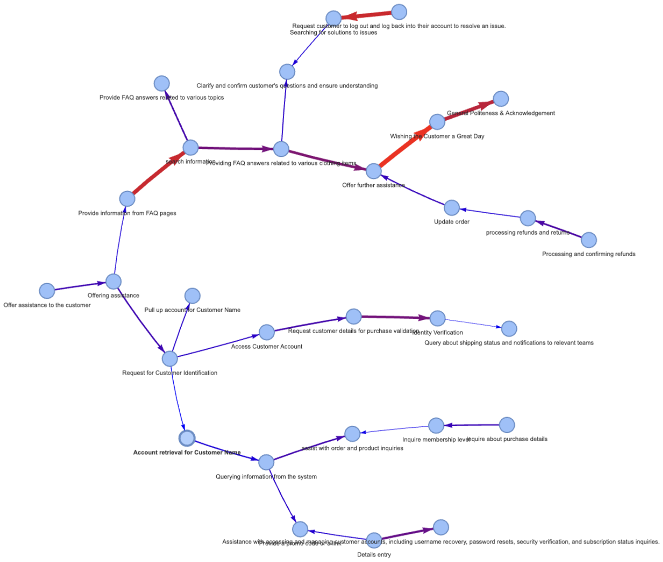

Inquire membership level
Inquire about purchase details
Account retrieval for Customer Name
Details entry
Assistance with accessing and managing customer accounts, including username recovery, password resets, security verification, and subscription status inquiries.
Request customer to log out and log back into their account to resolve an issue.
Pull up account for Customer Name
Identity Verification
General Politeness & Acknowledgement
Searching for solutions to issues
Querying information from the system
Wishing the Customer a Great Day
Request customer details for purchase validation
Update order
assist with order and product inquiries
Clarify and confirm customer's questions and ensure understanding
Offer further assistance
Provide FAQ answers related to various topics
Access Customer Account
Query about shipping status and notifications to relevant teams
Provide a promo code or a link
Provide information from FAQ pages
Request for Customer Identification
processing refunds and returns
Providing FAQ answers related to various clothing items
search information
Processing and confirming refunds
Offering assistance
Offer assistance to the customer

Fig. 6: Conversational graph generated using the Filter & Reconnect method for (top-k = 1 and τ = 0.1).


Request customer to log out and log back into their account to resolve an issue.
Searching for solutions to issues
Clarify and confirm customer's questions and ensure understanding
Provide FAQ answers related to various topics
General Politeness & Acknowledgement
Wishing the Customer a Great Day
search information
Providing FAQ answers related to various clothing items
Offer further assistance
Provide information from FAQ pages
Update order
processing refunds and returns
Processing and confirming refunds
Offering assistance
Offer assistance to the customer
Pull up account for Customer Name
Request customer details for purchase validation
Identity Verification
Query about shipping status and notifications to relevant teams
Access Customer Account
Request for Customer Identification
Account retrieval for Customer Name
assist with order and product inquiries
Inquire membership level
Inquire about purchase details
Querying information from the system
Assistance with accessing and managing customer accounts, including username recovery, password resets, security verification, and subscription status inquiries.
Details entry

D. Comparative Analysis

Table II summarizes the performance of the graph simplification methods across key metrics: δ-hyperbolicity, branching factor, and the number of cycles. As the table demonstrates, the Filter & Reconnect method outperforms the other techniques, yielding a graph with no cycles, a reasonable branching factor of 0.97, and a delta-hyperbolicity of 0, indicating that it most closely approximates an ideal tree structure. In contrast, the Threshold Filtering method suffers from excessive noise, resulting in a high number of cycles and poor hyperbolicity, while the Top-K Filtering method, despite achieving perfect hyperbolicity, produces a graph that is disconnected and less informative, as seen in Figures 4, 5, and 6.

Overall, the Filter & Reconnect method is the most effective for producing readable and interpretable conversational graphs,

TABLE II: Evaluation of Best-Performing Combination for Each Method With a Sarch Space of k $= \{1, 2, 3, 4\}$ and $\tau \in [0, 1]$

Method	δ-hyperbolicity	Branching Factor	Number Of Cycles
Threshold Filtering	2.50	**2.41**	56
Top-K Filtering	**0.00**	0.86	2
Filter & Reconnect	**0.00**	0.97	**0**

making it the recommended approach for quasi-patterned sets of conversations. While the Top-K Filtering method has certain strengths, such as perfect hyperbolicity, its disconnected structure limits its utility. Finally, the Threshold Filtering method, though simple, suffers from noise and high cyclicality, making it less suited for generating interpretable conversational graphs.

VI. Conclusion

In this study, we presented a novel computational approach to modeling quasi-patterned conversational flows using conversational graphs. By applying advanced text embedding techniques and clustering methods, we effectively extracted conversational intents and transitions to build interpretable graph representations. Through the comparative analysis of multiple graph simplification methods—Threshold Filtering, Top-K Filtering, and Filter & Reconnect—we demonstrated that the Filter & Reconnect method yielded the most readable and insightful graphs. This method preserves meaningful transitions between conversational intents while eliminating noise, resulting in a tree-like structure that aligns well with the quasi-patterned nature of the dataset.

Our approach offers a solution for analyzing large-scale dialogue datasets, providing valuable insights into conversational dynamics that can be leveraged to enhance automated systems, such as customer support agents and dialogue management systems. The successful application of our method to the ABCD dataset highlights its potential to uncover underlying conversational structures in other large-scale, loosely structured datasets, opening the door to further developments in conversational system optimization.

VII. Limitations

A. Parameter Sensitivity in Filtering Methods

While the Filter & Reconnect method proves effective in reducing noise, it requires careful tuning of parameters such as the threshold and top-k edges, which might not generalize across all datasets without manual intervention.

B. Assumption of Quasi-patterned Conversations

Another limitation is the assumption that conversations follow quasi-patterned flows, which might not be true for more free-form or highly dynamic dialogues. For such cases, more adaptive techniques that can account for greater variability in conversational paths may be necessary.

Acknowledgment

The author extends acknowledgment to Callem AI and, in particular, to Hamza Ghouili, CEO of Callem AI, for providing the resources and support necessary for the completion of this research. The author also expresses gratitude to INSAT for fostering an academic environment conducive to the development of this work.

References

[1] J. G. Carrasco Ramírez, ”Natural Language Processing Advancements: Breaking Barriers in Human-Computer Interaction,” *Journal of Artificial Intelligence and General Systems (JAIGS)*, vol. 3, no. 1, pp. 31-39, Feb. 2024.

[2] S. Schuster, S. Gupta, R. Shah, and M. Lewis, ”Cross-lingual Transfer Learning for Multilingual Task Oriented Dialog,” in *Proc. of the 2019 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, MN, pp. 3795–3805, 2019. doi: 10.18653/v1/N19-1380.

[3] H. Khanpour, N. Guntakandla, and R. Nielsen, ”Dialogue Act Classification in Domain-Independent Conversations Using a Deep Recurrent Neural Network,” in *Proc. of COLING 2016, the 26th International Conf. on Computational Linguistics: Technical Papers*, Osaka, Japan, pp. 2012–2021, 2016. Available: https://aclanthology.org/C16-1189.

[4] S. Hochreiter, ”Long Short-Term Memory,” Neural Computation, 1997.

[5] Z. Huang, W. Xu, and K. Yu, ”Bidirectional LSTM-CRF Models for Sequence Tagging,” arXiv preprint arXiv:1508.01991, 2015.

[6] N. Reimers and I. Gurevych, ”Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks,” arXiv, 2019. Available: http://arxiv.org/abs/1908.10084.

[7] S. Padmasundari and S. Bangalore, ”Intent Discovery Through Unsupervised Semantic Text Clustering,” in *Proc. of Interspeech*, vol. 2018, pp. 606–610, 2018.

[8] S. Minaee, T. Mikolov, N. Nikzad, M. Chenaghlu, R. Socher, X. Amatriain, and J. Gao, ”Large Language Models: A Survey,” *arXiv preprint arXiv:2402.06196*, 2024. [Online]. Available: https://arxiv.org/abs/2402.06196

[9] A. De and S. K. Kopparapu, ”A rule-based short query intent identification system,” in 2010 International Conference on Signal and Image Processing, 2010, pp. 212–216.

[10] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, ”BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, MN, USA, 2019, pp. 4171–4186. doi: 10.18653/v1/N19-1423.

[11] A. Ram, R. Prasad, C. Khatri, A. Venkatesh, P. G. Nagaraju, and J. W. F. Liu, ”Conversational AI: The Science Behind the Alexa Prize,” in *ArXiv Preprint arXiv:1801.03604*, 2018. Available: https://arxiv.org/abs/1801.03604.

[12] D. Arthur and S. Vassilvitskii, ”K-Means++: The Advantages of Careful Seeding,” in *Proc. of the Annual ACM-SIAM Symp. on Discrete Algorithms*, pp. 1027–1035, 2007. doi: 10.1145/1283383.1283494.

[13] D. Chen, H. Chen, Y. Yang, A. Lin, and Z. Yu, ”Action-Based Conversations Dataset: A Corpus for Building More In-Depth Task-Oriented Dialogue Systems,” in *Proc. of the 2021 Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Online, pp. 3002–3017, 2021. doi: 10.18653/v1/2021.naacl-main.239.

[14] M. Reid, N. Savinov, D. Teplyashin, D. Lepikhin, T. Lillicrap, J.-B. Alayrac, R. Soricut, A. Lazaridou, O. Firat, J. Schrittwieser, et al., ”Gemini 1.5: Unlocking Multimodal Understanding Across Millions of Tokens of Context,” arXiv, 2024. Available: https://arxiv.org/abs/2403.05530.

[15] J. R. Norris, *Markov Chains*, 2nd ed. Cambridge, UK: Cambridge University Press, 1998
.

[16] S. Bermudo, J. M. Rodríguez, J. M. Sigarreta, and J.-M. Vilaire, ”Gromov hyperbolic graphs,” Discrete Mathematics, vol. 313, no. 15, pp. 1575–1585, 2013.

[17] W. Chen, W. Fang, G. Hu, and M. W. Mahoney, ”On the Hyperbolicity of Small-World and Treelike Random Graphs,” *Internet Mathematics*, vol. 9, no. 4, pp. 434–491, 2013. doi: 10.1080/15427951.2013.828336. Available: https://doi.org/10.1080/15427951.2013.828336.