End-to-End Gujarati Task-Oriented Dialogue Management using Reinforcement Learning

RACHANA PARIKH[1], DR. HIREN JOSHI[2]
[1, 2] *Gujarat University*

Abstract*— Nowadays, there's an increased demand for dialogue systems in local languages due to the ongoing need for continuous support in specific service domains. Ra-ther than relying solely on human resources, dialogue systems offer a viable solu-tion. Dialogue management plays a pivotal role in determining the most effective actions for the system at each stage. In this study, we introduce a task-oriented dialogue system for Gujarati language, leveraging reinforcement learning. This system comprises three key components: natural language understanding (NLU), Dialogue Management (DM), and Natural Language Generation (NLG). Our model seamlessly interacts with databases, extracting valuable information. Rein-forcement learning is employed specifically for the DM, employing an enhanced Deep Q-learning Network (DQN) strategy to bolster the agent's resilience against environmental noise. Additionally, we propose a unified model for the NLU module, demonstrating its effectiveness through experiments conducted on Guja-rati dialogue datasets. The results showcase the superior performance of our model over the conventional rule-based multi-turn dialogue system for Gujarati dialogues.

Index Terms*— Task-Oriented Dialogue, Reinforcement Learning, Deep Q-learning (DQN), End-to-End dialogue management model

I. INTRODUCTION

Over the past decade, goal-oriented dialogue systems have stood out as integral parts of modern virtual personal assistants, enabling users to interact naturally for more efficient task completion. Traditional systems follow a complex modularized structure involving Language Understanding (LU), Dialogue Management (DM), and Natural Language generation (NLG) [1]. Recent strides in deep learning have prompted the application of neural models in dialogue systems. A network-based end-to-end trainable task-oriented dialogue system treats the learning process as mapping dialogue histories to system responses using an encoder-decoder model [2]. However, this supervised learning approach necessitates extensive training data and might struggle to establish a robust policy due to limited exploration of dialogue control in the training dataset. There's a shift towards employing end-to-end Reinforcement Learning (RL) in dialogue state tracking and policy learning for DM [3].

Dialogue management differs significantly from discrete action domains like game playing [4], where an agent might have a narrow set of moves, whereas a DM offers a broader spectrum of dialogue acts with distinct semantics. While game episodes can span hundreds of steps, task-oriented dialogues typically involve fewer turns, where each system action can significantly influence the dialogue's direction or duration. Consequently, errors by the DM are more impactful and temporally localized compared to these domains.

What distinguishes dialogue management is its collaborative nature, unlike the more individual-focused domains mentioned earlier. The interaction between a user and an assistant resembles a cooperative game, where both players strive to achieve a goal. The user seeks information or action, while the dialogue system accesses a database or service to fulfill the user's objective. Communication occurs through dialogue moves (referred to as dialog acts), and users are often willing to offer feedback, explicit or implicit, if it enhances the system's performance [5].

II. RELATED WORK

Before the recent surge of neural network-based approaches in dialogue systems, researchers have been exploring the potential of reinforcement learning for this task. They put forward a novel dialogue system designed within a Markov Decision Process framework, which facilitates the application of reinforcement learning to train dialogue policies [6].

Studies in the field of task-oriented dialogue have demonstrated the efficacy of a hybrid learning method employed for training task-oriented dialogue systems through live user interactions. This research led to the creation of a neural network-based task-oriented dialogue agent, specifically tailored for end-to-end dialogue. Notably, the proposed learning methodology enabled the end-to-end dialogue agent to effectively learn from its mistakes by leveraging imitation learning from user interactions. Moreover, the subsequent application of reinforcement learning, coupled with user feedback post-imitation learning, notably enhanced the agent's capability to successfully accomplish tasks [7].

Traditional task-oriented dialogue management systems are often encumbered by the requirement for domain-specific handcrafting, which substantially impedes scalability to new domains. In contrast, end-to-end dialogue management systems, where all components are trained from dialogue transcripts, circumvent this constraint. In a bid to address this limitation, researchers designed an unsupervised dialogue-manager specifically catering to goal-oriented dialogue applications. The context of Corporate Information Organization (CIO) domain underscores the tasks, involving assessment of user-initiated dialogues to comprehend the nature of the tickets, issuance of database (DB) and API calls, and utilizing the outputs of such calls to assist users. This research demonstrated the viability of an unsupervised dialog state-tracking and management system rooted in jointly trained LSTM-RNNs, effectively handling nuanced dialog management scenarios [8].

In an endeavor focused on Indian languages, particularly Hindi, researchers introduced a universal Deep Reinforcement Learning framework capable of synthesizing dialogue managers adaptable to a variety of intents within a domain. This innovative approach dissects conversations between agents and users into hierarchies, effectively segregating subtasks pertinent to different intents. Hierarchical Reinforcement Learning, especially utilizing options, served as the mechanism for learning policies across different hierarchies operating at distinct time steps, successfully addressing user queries. Notably, the designed dialogue management module was trained to be reusable across languages with minimal supervision, showcasing its versatility across the "Restaurant" domain in English and Hindi [9].

As the development of task-oriented dialogue systems rapidly progresses, the necessity for labeled dialogue corpora has become increasingly apparent. In addressing this need, a Hindi Dialogue Restaurant Search (HDRS) corpus was released, serving as a benchmark for comparing various state-of-the-art dialogue state tracking (DST) models [10].

To comprehensively understand the research inquiries underlying the Spoken Language Understanding (SLU) and Dialogue State Tracking (DST) modules in the context of Indic languages, particularly Hindi, researchers curated the Hindi Dialogue Restaurant Search (HDRS) corpus. This corpus served as the foundation for comparing various state-of-the-art SLU and DST models. Moreover, for the dialogue manager (DM), extensive exploration into deep-learning Reinforcement Learning (RL) methods, including actor-critic algorithms with experience replay, was conducted [11].

There's a scarcity of research on Indic Gujarati task-oriented dialogue management systems, indicating a notable gap in the existing literature.

III. PROPOSED FRAMEWORK

The system's overall architecture, depicted in Fig. 1, comprises three core components: the Natural Language Understanding (NLU), Dialogue Management (DM), and Natural Language Generation (NLG) modules. Specifically, the DM module encompasses a state tracker and policy learning. The policy learning facet employs Deep Q-Network (DQN) algorithms. To illustrate, consider the sentence 'મને નજીકના રેસ્ટોરન્ટનો રસ્તો બતાવો' In this scenario, the NLU module processes the sentence to derive the semantic frame or user action. The NLU module adeptly recognizes the user's intent and slots within the query, denoting the outcome as Intent: inform; Slot—Value: (poi_type = રેસ્ટોરન્ટ; distance = નજીક). This union of intent and slot-value forms the user action. Subsequently, this user action enters the state tracker segment, which employs a series of 0/1 vectors to denote whether slots have been identified, effectively

representing the dialogue state. Based on this dialogue state, the DQN engages in policy learning, eventually selecting the agent action comfirm(poi_type = રેસ્ટોરન્ટનો); request(poi). This agent action is then processed by the NLG module, generating a response such as 'તમે કઈ રેસ્ટોરન્ટનો તમે પસંદ કરશો ' Throughout the conversation, the accumulation of semantic parsing in utterances significantly bolsters the DM's capacity to track dialogue states robustly, enabling the system to provide relevant and goal-oriented responses to assist users.

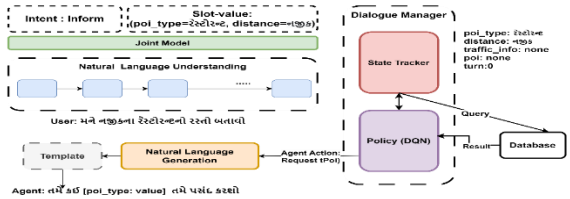

Fig. 1. Demonstration of the complete neural Gujarati Language dialogue system: employing reinforcement learning to train all elements seamlessly from user utterances.

a. Natural Language Understanding Module

The visual representation in Figure 2 delineates the architecture of the Natural Language Understanding (NLU) module, crafted to extract semantic information from user utterances. Its primary goal revolves around identifying intent and grasping semantic constituents, achieved through the processes of intent detection and slot filling. Consider a sample sentence like મને નજીકના રેસ્ટોરન્ટનો રસ્તો બતાવો' sourced from the dataset, where each word aligns with a designated slot label, while the entire sentence embodies a specific intent. This encapsulates the core essence of Language Understanding (LU), where the fundamental objective lies in categorizing the domain of a user's query and discerning domain-specific intentions, concurrently populating slots to formulate a comprehensive semantic frame.

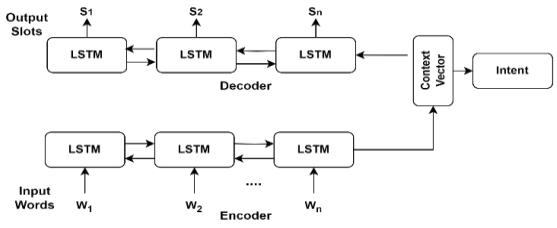

Fig. 2. An NLU model for Gujarati Dialogue

In this framework, slot tags are systematically represented leveraging the IOB (in-out-begin) format, showcased in the visual representation within Table 1. This format delineates the input word sequence.

Table 1. Intent and slot classification

Sentence	મને	નજીકના	રેસ્ટોરન્ટનો	રસ્તો	બતાવો
Slot	O	b-distance	b-poi_type	O	O
Intent	Inform				

$$\overline{a} = w_{1,}w_{2,} \ldots, w_n < EOS > \quad (1)$$

$$\overline{b} = s_{1,}s_{2,} \ldots, s_n, i_m \quad (2)$$

Here, $\overline{a}$ signifies the sequence of input word w1..wn, while $\overline{b}$ encompasses the related slots s1…sn and the overarching sentence intent. The core mechanism of the LU component is powered by a singular Bidirectional Long Short-Term Memory (BiLSTM) framework, effectively conducting both intent prediction and slot filling concurrently through equation (3):

$\overline{b}$ = BiLSTM($\overline{a}$) (3)

The core objective of LU involves maximizing the conditional probability of the slots and intent $\overline{b}$ given the word sequence $\overline{a}$. This is mathematically formulated as[12][13]:

$$p(\overline{b} \mid \overline{a}) = (\prod_1^n p(s_i \mid w_1, \ldots, w_i)\, p(i_m \mid \overline{b}) \quad (4)$$

Training of the BiLSTM model's weights is achieved using backpropagation, optimizing the conditional likelihood of the training set labels. The resulting tag set amalgamates IOB-format slot tags and intent tags, enabling supervised training using the available dialogue actions and utterance pairs within the labeled dataset.

b. Dialogue Management Module

Introduction to DM Agent and its Role:

The DM agent is a crucial part of a dialogue system, essentially acting as the main connection point across different parts of the system. In Figure 3, we see the internal structure of this DM agent. It is responsible for receiving the output from the Natural Language Understanding (NLU) module, which essentially provides the meaning behind what a user says or asks.

This DM agent consists of four major components: the DQN agent, Dialogue State Tracker (ST), user (or user simulator), and EMC (Error Model Controller). Each of these plays a specific role in managing and processing information during a conversation.[14][15]

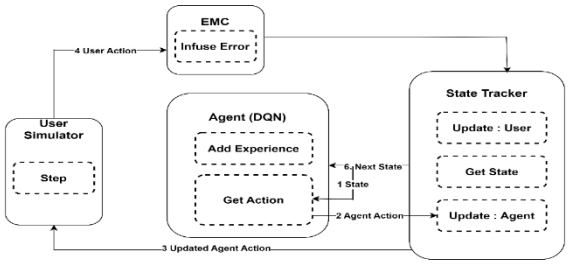

Fig. 3. Conversational System Dialogue Management Agent workflow

Understanding the Conversation Flow:
Let's go through the steps involved in a single round of conversation within this system. First off, we need to identify the current state of the conversation, which might be the last known state or an initial state if it's the beginning of a new conversation. This state is then given as input to the agent's 'get action' method.

The agent responds by providing an action based on the current state, and this action is then sent to the dialogue ST for an update. The ST keeps track of the ongoing conversation and also adjusts the agent's action based on information retrieved from a database.

Now, this updated action from the agent is forwarded to the user, or in some cases, a simulated user. The user then crafts a response based on rules and provides feedback, like whether the conversation is going well or not. This response from the user is then tampered with errors by the EMC. The tampered response is sent back to the ST, which saves this information but doesn't significantly change the user's action.

Finally, the ST processes all this information and determines the next state of the conversation. This completes one cycle or 'experience tuple' in the conversation, and the information gathered during this cycle is stored in the agent's memory.

Two Stages of Classic DM: Dialogue State Tracking and Policy Learning:
Here's how they work: Dialogue State Tracking: In earlier systems, tracking the state of a conversation was based on predefined rules or heuristics. However, more recent approaches treat this as a supervised learning problem, where the output from the NLU module helps update the dialog states. These states constantly change and evolve during a conversation and are represented using various vectors, combining different elements like user and agent actions, available database information, and historical conversation turns. The simulated user generates a series of actions like a real user throughout the conversation. The user's objective includes both informative slots (constraints denoted as C) and request slots (labeled as R). Typically, the user's objective remains consistent unless the system can't find information for the requested slots in the database. At each time step t, the user simulator creates the subsequent user action, $a_{u,t}$, based on the current state, $s_{u,t}$, and the previous agent action, a_{t-i}, transitioning to the next user state, $s_{u,t+1}$.[16]Policy Learning: This stage involves making decisions or selecting the best responses during a conversation. In this setup, the dialogue system is seen as a Markov Decision Process (MDP), which includes states, actions, rewards, and policies.[17] States represent the current situation in the conversation, actions are the possible moves the system can make (like responding or asking for information), rewards are the feedback received after each action, and policies dictate the system's behavior based on the current state.[18][19]

Role of User Simulator in Reinforcement Learning: In reinforcement learning setups for dialogue systems, a user simulator is essential. It's designed to mimic a real user's behavior and interact naturally with the dialogue system. During a conversation, this simulator generates actions like a real user would. The user's goal typically includes providing information (informable slots) and requesting information (requestable slots) from the system. During training, placeholder values are used in responses, which are later replaced by actual information during testing.

The DM agent is like the conductor of an orchestra, coordinating different elements within the dialogue system. It takes in information, processes it, and responds accordingly, all while learning and improving its responses over time through reinforcement learning techniques. The dialogue system aims to understand users better, track ongoing

conversations effectively, and learn from interactions to provide more accurate and useful responses.

c. Natural Language Generation Module

As previously mentioned, the user's action follows the policy learning module. Subsequently, the NLG (Natural Language Generation) module generates human-like texts based on these actions. To enhance the quality of generation, we've adopted a method that combines both template-based and model-based approaches, especially due to limited labeled data availability.

d. End-to-End Reinforcement Learning for Dialogue Management

To master our system's interactive policy, we employ reinforcement learning (RL) in training the DM in an end-to-end manner, enabling fine-tuning of each neural network component. The policy is embodied as a Deep Q-Network (DQN), accepting the state, s_t, from the state tracker as input and generating Q (s_t, a; θ) for all actions a. The Q-learning update rule, used in the Deep Q-Network (DQN), is represented by the following formula:[20]

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1},a) - Q(s_t,a_t) \quad (5)$$

During training, we incorporate €-greedy exploration and an experience replay buffer with a dynamically adaptable size. In each simulation epoch, we simulate X dialogues (where X = 100) and store these state transition tuples (s_t, a_t, r_t, s_{t+1}) in the experience replay buffer for subsequent training. In a simulation epoch, the current DQN undergoes multiple updates, dependent on the batch size and the existing size of the experience replay buffer. Once the simulation epoch concludes, the target network is substituted with the current DQN, with the target DQN updated only once per simulation epoch.[21]

The experience replay strategy holds significant importance in RL training. In our buffer update approach, we aggregate all experience tuples from the simulation and empty the pool until the current RL agent achieves a success rate threshold (equivalent to a rule-based agent's performance). Following this, we replenish the buffer using experience tuples from the current RL agent. This strategy stems from the understanding that the initial DQN performance might not generate effective experience replay tuples, hence we wait until the current RL agent reaches a specific success rate (like that of a rule-based agent) before refreshing the experience replay pool.[22]

IV. EXPERIMENT

We deploy the dialogue system in car assistant scenarios focused on navigation. Our trials occur on a Gujarati dialogue dataset. Throughout the conversation, the system persistently extracts the meaning from user utterances, and the dialogue state evolves until the user's goal is met or the maximum number of dialogue turns is reached. Post-dialogue, the system receives a binary outcome (success or failure) based on the overall conversation. A successful conversation necessitates two criteria: (1) reception of all necessary attributes and setting navigation (2) non-matching attributes.

a. Dataset

Dataset from Stanford University's public car assistant has been extracted for our experimentation, originally curated by Stanford University and annotated by field experts. [22] Given that the dataset is in English, we translated it into Gujarati, supplementing it with an additional 1600 dialogues. The initial dataset encompasses 3,031 multi-turn conversations spanning schedule (Sch.), weather (Wea.), and navigation (Nav.) domains. We specifically focused on Navigation dialogues, comprising 1256 instances, resulting in a total of 2856 dialogues utilized. On average, this dataset features 3 turns per conversation. Within this dataset, certain slots serve as informative slots, setting constraints for database searches, while others are requestable slots, allowing users to query actual values from the database.

b. Evaluation Metrics

In our experiments, within the NLU Module, we have specifically designated assessment criteria. For slot filling, the F1-score, while for intent detection, we relied on accuracy. We evaluate the NLU using different architectures such as LSTM, BiLSTM, and CNN, calculating diverse evaluation metrics.

Regarding the DM Agent, we assessed it using evaluation metrics: success rate, average reward. The success rate indicates the percentage of successful dialogues where the user's objective is met. Average

reward signifies the mean reward obtained during reinforcement learning across each dialogue. These two metrics collectively portray the overall performance of the agent.

c. Experimental Setting

The maximum number of dialogue turns is capped at 8. A successful dialogue results in a +16 reward, while a failure yields a -8 reward. To encourage shorter dialogues, a reward of -1 step is deducted for each turn. The dataset is split into 80% for training and 20% for testing. The ε (epsilon) of the ε-greedy strategy is set at 0.1 to ensure effective exploration of the action space, and the discount factor (γ) in the Bellman equation is 0.9. The buffer size (D) is 2000 with a batch size of 30. The learning rate is fixed at 0.001. Each simulation epoch involves 100 dialogue sessions, and prior to training, the buffer is pre-filled. In the chart, the blue line illustrates the performance of the rule-based agent, while the orange line indicates the performance of the single-DQN agent during training. Notably, the NLG module adopts a template-based approach.

d. Results and Analysis

NLU Module: Table 2 illustrates our model's performance, indicating favorable outcomes across two dimensions: slot filling (F1) and intent detection (Acc).

Table 2. Metrics evaluating intent and slot classification performance.

Model	Intent Accuracy	LSTM F1-Score	BiLSTM F1-Score
Encoder-Decoder joint model	91.8	88.6	89.3
Encoder-Decoder Attention joint model	92.3	89.2	89.9

Table 3 illustrates a successful dialogue instance between the agent and user in the proposed model. In this scenario, the user's objective was to find the closest restaurant, specifying the poi_type as "રેસ્ટોરન્ટ" (“restaurant”) and the distance as " નજીક" (“nearest”). The database provided two options: સ્ટારબક્સ (Starbucks) and ચાઇનાટાઉન (Chinatown), which the agent presented to the user. The user opted for સ્ટારબક્સ (Starbucks) and requested its address, prompting the agent to promptly display the address sourced from the database.

Table 3. An instance demonstrating a sequence of actions within a dialogue conversation.

Speaker	Intent	Request Slot	Inform Slot
User	request	poi	Poi_type: રેસ્ટોરન્ટન, distance: નજીક
Agent	inform	-	Poi: સ્ટારબક્સ, poi: ચાઇનાટાઉન
User	Request	address	Poi: સ્ટારબક્સ
Agent	inform	-	Address: 329 અલ કેમિનો રિયલ

DM Agent: The Figure. 4 demonstrates a significant performance gap favoring RL-based methods over traditional rule-based approaches in terms of success rate. Hence, the experimental findings underscore the superiority and effectiveness of our proposed DQN agent. In Gujarati dialogues, the rule-based model achieved a 38% success rate, while the reinforcement-based DQN model achieved a 64% success rate.

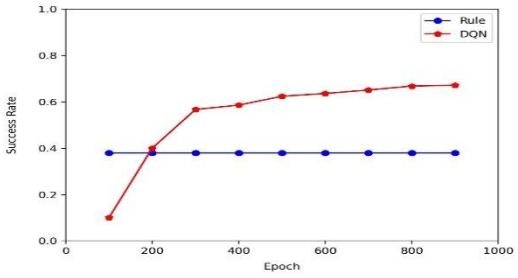

Fig. 4. The success rate achieved by the agent throughout the learning phase.

In Figure 5, it's evident that Reinforcement Learning-based DQN methods outperformed traditional rule-based approaches significantly in terms of average reward.

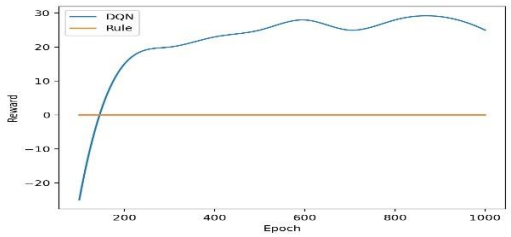

Fig. 5. The average reward obtained by the agent during the learning process.

In Figure 6, we conducted further assessment of the rule based and DQN by engaging real human users. The double DQN agent underwent training using real user data. In each conversation session, one of these agents engaged with a user, presenting them with a predefined user goal sampled from our dataset. After each session, users were asked to rate the dialogue's naturalness and coherence on a scale from 1 (worst) to 5 (best). Typically, we utilized 100 dialogue sessions for our experiments. The graph on the left illustrates the success rates of these agents when interacting with real users.

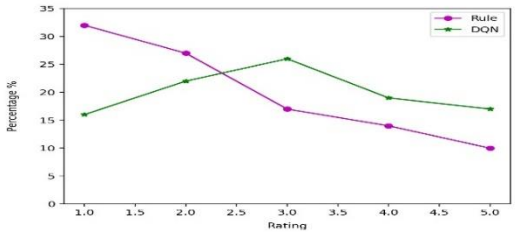

Fig. 6. Displays the Rating Distribution across Various Models.

CONCLUSION

This study introduces a comprehensive learning structure designed for task-oriented neural dialogues in Gujarati. Our findings indicate that reinforcement learning models in Gujarati conversations surpass rule-based counterparts, offering enhanced adaptability for genuine, task-driven interactions. This RL-based Gujarati dialogue framework serves as a steppingstone for future endeavors in the domain. Our experimentation concentrated on the car assistant dataset's Navigation domain, and our upcoming research aims to extend this to multi-domain Gujarati dialogue management. Additionally, our focus will involve refining strategies for integrating external information into the database for more effective system performance.

REFERENCES

[1] Zue, V., Seneff, S., & Glass, J. R. (2000). JUPlTER: a telephone-based conversational interface for weather information. IEEE Transactions on Speech and Audio Processing, 8(1), 85-96. DOI: 10.1109/89.817460.

[2] Dhingra, B., Li, L., Li, X., Gao, J., Chen, Y. N., Ahmed, F., & Deng, L. (2017). Towards End-to-End Reinforcement Learning of Dialogue Agents for Information Access. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 484–495). Vancouver, Canada: Association for Computational Linguistics.

[3] Bordes, A., Boureau, Y. L., & Weston, J. (2016). Learning end-to-end goal-oriented dialog. arXiv preprint arXiv:1605.07683.

[4] Mnih, V., et al. (2013). Playing atari with deep reinforcement learning. arXiv preprint arXiv:1312.5602.

[5] Knox, W. B., & Stone, P. (2012). Reinforcement learning from simultaneous human and mdp reward. In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1 (pp. 475–482). International Foundation for Autonomous Agents and Multiagent Systems.

[6] Singh, S., Kearns, M., Litman, D., & Walker, M. (1999). Reinforcement learning for spoken dialogue systems. In Proceedings of the 12th International Conference on Neural Information Processing Systems, NIPS'99 (pp. 956–962). Cambridge, MA, USA: MIT Press.

[7] Liu, B., Tur, G., Hakkani-Tur, D., Shah, P., & Heck, L. (2018). Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. arXiv preprint arXiv:1804.06512.

[8] GM, S., & Sengupta, S. (2019). Unsupervised Multi-task Learning Dialogue Management. In Proceedings of the ACM India Joint International Conference on Data Science and Management of Data (CODS-COMAD '19) (pp. 196–202).

Association for Computing Machinery. DOI: 10.1145/3297001.3297026.

[9] Saha, T., Gupta, D., Saha, S., & Bhattacharyya, P. (2021). A Unified Dialogue Management Strategy for Multi-intent Dialogue Conversations in Multiple Languages. ACM Trans. Asian Low-Resour. Lang. Inf. Process., 20(6), Article 99. DOI: 10.1145/3461763.

[10] Malviya, S., Mishra, R., Barnwal, S. K., & Tiwary, U. S. (2021). HDRS: Hindi Dialogue Restaurant Search Corpus for Dialogue State Tracking in Task-Oriented Environment. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 29, 2517-2528. DOI: 10.1109/TASLP.2021.3065833.

[11] Malviya, S. (2021). Design and Development of Spoken Dialogue System in Indic Languages. In Proceedings of the 18th International Conference on Natural Language Processing (ICON) (pp. 654–657). National Institute of Technology Silchar, Silchar, India: NLP Association of India (NLPAI).

[12] Parikh, R. B., & Joshi, H. (2021). Gujarati Task Oriented Dialogue Slot Tagging Using Deep Neural Network Models. In R. Sharan & A. Kumar (Eds.), Soft Computing and its Engineering Applications: Second International Conference, icSoftComp 2020, Changa, Anand, India, December 11–12, 2020, Proceedings 2 (pp. 27-37). Springer Singapore.

[13] Chen, Y. N., Hakanni-T¨ur, D., Tur, G., Celikyilmaz, A., Gao, J., & Deng, L. (2016). Syntax or semantics? knowledge-guided joint semantic frame parsing. In Proceedings of the 6th IEEE Workshop on Spoken Language Technology (pp. 348–355).

[14] Schatzmann, J., Thomson, B., Weilhammer, K. (2007). Agenda-based user simulation for bootstrapping a POMDP dialogue system. In: Human Language Technologies: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers, 149-152. Association for Computational Linguistics (2007).

[15] Su, P. H., Gasic, M., Mrksic, N. (2016). Continuously learning neural dialogue management. arXiv preprint arXiv:1606.02689.

[16] Young, S., Gai, M., Thomson, B. (2013). POMDP-based statistical spoken dialog systems: A review. Proceedings of the IEEE, 101(5), 1160-1179.

[17] Zue, V., Seneff, S., Glass, J. R. (2000). JUPlTER: A telephone-based conversational interface for weather information. IEEE Transactions on Speech and Audio Processing, 8(1), 85-96.

[18] Williams, J., Raux, A., Ramachandran, D. (2013). The dialog state tracking challenge. In: Proceedings of the SIGDIAL 2013 Conference, 404-413.

[19] Parikh, R. B., & Joshi, D. H. (2020). Gujarati speech recognition–A review. 549, 6.

[20] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518, 529–533.

[21] Schaul, T., Quan, J., Antonoglou, I., Silver, D. (2015). Prioritized experience replay. arXiv:1511.05952.

[22] Shalyminov, I., Lee, S., Eshghi, A., & Lemon, O. (2019). Data-efficient goal-oriented conversation with dialogue knowledge transfer networks. arXiv preprint arXiv:1910.01302.

[23] Li, X., Chen, Y. N., Li, L., et al. (2017). End-to-end task-completion neural dialogue systems. arXiv preprint arXiv:1703.01008.