POLYTHECNIC UNIVERSITY OF TURIN

Master's Degree in Data Science and Engineering

Master's Degree Thesis

Hierarchical Attention for Conversational Agents

Supervisors

Prof. Giuseppe RIZZO

Dr. Fabio CAFFARO

Candidate

Khudayar FARMANLI

July 2024

Summary

The argument of the thesis focuses on Knowledge-Enhanced Conversational Agents. In particular, it focuses on the implementation of a specific type of Recurrent Neural Network, Long Short-Term Memory (LSTM), to leverage the temporal dependencies of dialogue turns for extracting knowledge from a knowledge base. The thesis investigates the use of transformers for encoding multimodal language content and exploits the hierarchical structure of the knowledge base by creating three downstream tasks. These tasks are aimed at recognizing the domain, the entities involved, and the documents referenced in the user's request. The experiments are conducted with DSTC11 Track 5, which is a de facto standard for developing conversational agents.

Acknowledgements

I would like to thank my family, who supported me both mentally and financially throughout my entire education process. I also owe a great deal of gratitude to my supervisors, Prof. Giuseppe Rizzo and Dr. Fabio Caffaro, for guiding me throughout my exhaustive thesis journey. They provided immense help and demonstrated great patience, even for the most trivial questions I presented.

I dedicate this work to the memory of my beloved grandmother Hadiyya, who recently passed away. Her unwavering love, wisdom, and encouragement had a profound and positive impact on my life. She instilled in me the values of perseverance and dedication, which have been instrumental in my academic journey. Her memory continues to inspire me, and I am deeply grateful for the time we shared.

Table of Contents

List of Tables

List of Figures

Acronyms

CA
: conversational agent

NLP
: natural language processing

AI
: artificial intelligence

NLU
: natural language understanding

DSTC
: dialogue system technology challenge

KCA
: knowledge-enhanced conversational agents

ML
: machine learning

TOD
: task-oriented dialogue

API
: application programming interface

DB
: database

FAQ

frequently asked questions

SK-TOD

subjective-knowledge-based TOD

KB

knowledge base

BoW

bag of words

TF-IDF

term frequency-inverse document frequency

RNNs

recurrent neural networks

LSTM

long short-term memory

IR

information retrieval

PRR

passage re-ranking

DKR

dense knowledge retrieval

HDKR

hierarchical dense knowledge retrieval

TP

true positive

FP

false positive

FN

false negative

EM

exact match accuracy

Chapter 1

Introduction

Conversational Agents (CAs) are artificial-intelligence powered systems designed to engage in natural language conversation with users. The term “conversation agent” is frequently used interchangeably with the terms “intelligent personal assistant” [1], “chatbot” [2], or “conversational agent” [3]. In a world that is even more digital every passing year, CAs have gained eminence as tools that simulate human-like interactions, providing information, assistance, or entertainment through chat-based interfaces. Leveraging various techniques from natural language processing (NLP), machine learning, and dialogue management CAs aim to understand the user request, generate proper responses, and deliver personalized experiences. From customer support to virtual companions, CAs are employed in a wide variety of applications, transforming the way we interact with technology and providing a seamless bridge between humans and machines.

Initial efforts and first developments about CA had been done in previous century. Schöbel et al. consider the improvements done in this field in five different waves [4] (see Fig. 1.1). In 1950, one of the foundational contributors of theoretical computer science and artificial intelligence Alan Turing, in his well-known paper [5] proposed his test to assess whether a machine can think or has a conscience. This paper was the introduction of the ongoing discussion around the “humanness” of machines and the technical feasibility and future of machines. Less than 20 years later, first chatterbot (in modern times, chatbot) ELIZA was introduced based on natural language communication by Joseph Weizenbaum [6] and it served as foundation for subsequent chatbot and conversational AI research, inspiring the development of more sophisticated systems with advanced NLP techniques. This phenomenon also caused emergence of so-called “ELIZA effect” [7] which is the tendency of individuals to attribute human-like understanding or intelligence to computer programs or chatbots. The achievement of a computer program that can resemble a conversation with a human under specific conditions that are predetermined represent first wave of research on CAs.

The second wave of CA research, for the first time, made use of NLP and statistical methods and took process a lot further. Thus, this stage of development is considered as "exploration" phase. Moreover, with this wave, methods such as pattern recognition and the first simple AI solutions entered CA research. Due to this, specialized languages and considerably more sophisticated CAs entered the market. The chatbot A.L.I.C.E. (release 1995) and the special language it was programmed with, the so-called artificial intelligence markup language [8] is the most prominent example of this period.

In the third wave, improvements gained significant momentum in every part of entire topic of CA research based on technological advancements made in the 2000s and the mid-2000s so, the period is called "kick-off". Of course, this development revealed itself in emergence of further technologies and implementations like IBM Watson in 2006. This made IBM the first big-tech company to release a sophisticated product in CAs domain and it led other big-tech companies to give attention in the following years.

The groundwork for the fourth wave of CA research was built with the revolution of the mobile phone market (i.e., the release of the Apple iPhone in 2007) and general consumer electronics that followed during the 2010s and the technological improvements that came with it. Especially, in the 2010s, significant improvements particularly in AI and NLP led CAs to become mainstream and occupy more attention. The shift from text-based to voice-based CAs made them more reachable to the broader population, given that they can be operated without the need to type in the text. Thus, the popularity increased tremendously, and it showed itself in research and, financial investment into these agents, not only by researchers but also from big companies like Google and Amazon who realized the massive potential. In 2011 Apple Siri was launched alongside the iPhone 4S, introducing voice-activated virtual assistants to a wide audience and it attained huge popularity. Then in 2014, Amazon Alexa was released integrated with the Amazon Echo smart speaker and it was followed by Google Assistant in 2016, leveraging Google's knowledge graph and machine learning capabilities.

The fifth wave of CA covers the last few years of research and the near future of it. One of the key focuses is the pursuit of "true" or "general" AI through automation or autonomous CAs. Technologies like Google's LaMDA [9] and OpenAI's ChatGPT have emerged as prominent advancements, enhancing natural language interactions and enabling a wide range of tasks. Especially, by the release of beta version of OpenAI's ChatGPT in November 2022, many experts called this AI technology the most disruptive of all in that year and the future [10]. This technology enables a wide variety of tasks during a conversational interaction, such as writing essays, helping with coding and, many other creative tasks. All these developments aim to make CAs more human-like, blurring the line between human and machine. Additionally, the fifth wave emphasized improved understanding of natural language, including sarcasm, through advanced deep learning and Natural Language Understanding (NLU) methods. Besides of text-based CAs and voice-based CAs, embodied CAs become also popular by integrating embodiment and virtual agents, such as avatars, and this further enhances the anthropomorphic qualities of CAs. These advancements pave the way for highly customizable CAs with unprecedented social presence. Ultimately, the fifth wave emphasizes adaptability, personalization, and individualization of human-computer interactions, creating seamless and integrated experiences in various aspects of life.

Conversational agents, while achieving promising results, also have many limitations such as contextual understanding, domain expertise, common sense reasoning, handling ambiguity, emotional intelligence, and ethical considerations. They struggle with accurately interpreting the context of conversations, especially in unfamiliar domains. Their responses may lack depth and relevance outside their specialized knowledge areas. CAs often struggle with understanding ambiguous queries and fail to exhibit emotional intelligence in recognizing and responding to user emotions. Ethical concerns include privacy, data security, and biases. Overcoming these limitations requires advancements in NLU, domain adaptation, common-sense reasoning, emotional modeling, and adherence to ethical guidelines. Of course, there are more than we mentioned and overcoming these limitations requires continued research and development in areas such as NLU, context modeling, knowledge representation, and ethical guidelines to enhance the capabilities and reliability of CAs.

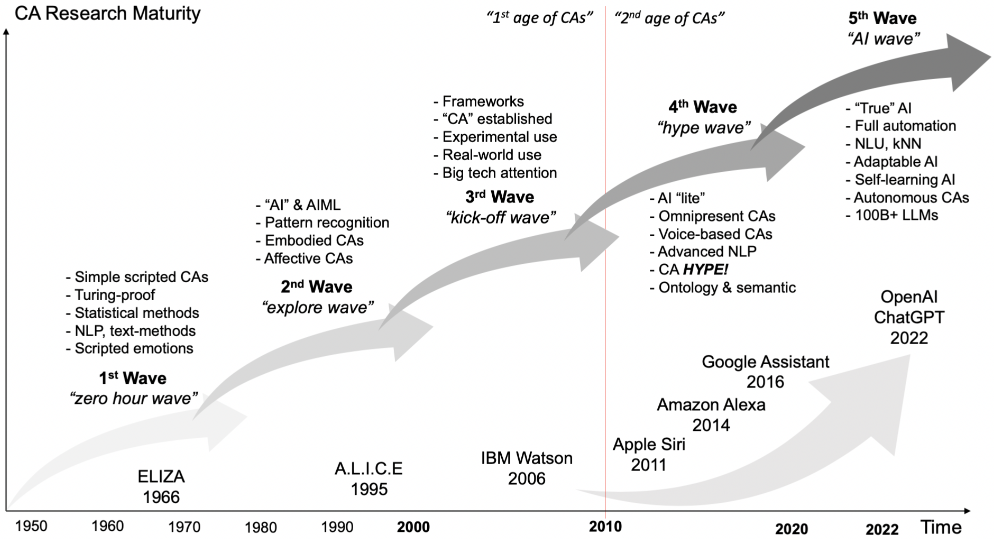


Figure 1.1: The five waves of Conversational Agent Research [4]

In the context of the fifth track of the Dialogue System Technology Challenge 11 (DSTC11), this study demonstrates the application of the attention mechanism to leverage the temporal dependencies of dialogue turns between the user and the system. Additionally, it is combined with the exploitation of the hierarchical structure of the knowledge base during information retrieval to enhance the effectiveness and efficiency of KCAs. The information provided in the subsequent chapters is summarized below:

- **Chapter 2**: This section provides information about Knowledge-enhanced Conversational Agents and DSTC challenges, with an in-depth discussion of the specific task at hand.
- **Chapter 3**: In this section, the related work on text data manipulation techniques and information retrieval methods is discussed.
- **Chapter 4**: Here, the adopted methodology is framed with an evaluation of the method alongside its predecessors to provide an in-depth understanding.
- **Chapter 5**: In this section, the experiments conducted are presented, detailing the experimental setup and the results yielded by the methods.
- **Chapter 6**: Conclusion.

Chapter 2

Task

Our work in this experiment concentrates on the task introduced by Amazon in the eleventh Dialogue System Technology Challenge (DSTC), specifically focusing on Track 5: Task-oriented Conversational Modeling with Subjective Knowledge [11]. The goal of this challenge is to improve Task-oriented Dialogue (TOD) Systems, which traditionally rely on domain APIs and structured knowledge bases to answer user requests. The aim is to incorporate a new, unstructured knowledge base that includes subjective knowledge, thereby enhancing the system's ability to provide comprehensive responses. For this reason, this chapter will provide a detailed overview of the task at hand.

In section 2.1, KCAs are detailed, highlighting their distinctive features and characteristics to enhance understanding of their objectives. The following section delves into DSTC11 - Track 5. It begins with an overview of the DSTC challenges, setting the stage for a discussion on the precursors to the current task, thereby providing a comprehensive background. This section then progresses to present detailed information about the current challenge, including its specific characteristics, the dataset utilized, and the evaluation metrics employed. Exploring the evolution of the topic from general information about KCAs, through DSTC challenges, to the specific features of DSTC11 Track 5, provides a deeper understanding by tracing how the topic has developed over time.

2.1 Knowledge-enhanced Conversational Agents

In recent years, the development of Knowledge-enhanced Conversational Agents (KCAs) has marked a significant advancement in the field of artificial intelligence. These sophisticated systems leverage extensive knowledge bases [12] to provide accurate, contextually relevant responses, setting a new standard for user interaction.

Unlike their predecessors, which relied on simple machine learning (ML) models or predefined rules, KCAs integrate structured information from diverse sources, such as databases and knowledge graphs, thereby providing factual and up-to-date responses. This integration allows for interactions that are not only more informative but also more engaging.

The NLU capabilities of these systems are particularly deserving attention. Advanced algorithms enable the agents to comprehend complex queries, facilitating a more natural conversational flow. Furthermore, some agents possess dynamic learning abilities, allowing them to update their knowledge bases with new information accumulated from ongoing interactions [13], thus continually improving their performance.

A key feature of knowledge-enhanced conversational agents is their *multi-domain* expertise. By accessing domain-specific knowledge bases, these agents can handle inquiries across various fields, from everyday topics like weather and news to more specialized areas such as healthcare [14] and finance. This versatility enhances their utility across a broad spectrum of applications. The ability to offer personalized recommendations through understanding user context and preferences significantly enhances user satisfaction and engagement. *Semantic reasoning*, another critical aspect of KCAs, allows them to infer answers to implicitly stated questions, providing an interactive experience that closely mimics human conversation [15].

In the advancement of KCAs, the overarching objective is multifaceted, aiming to refine their comprehension capabilities, broaden the scope of their knowledge repositories, and tailor interactions to individual user profiles. Integral to their development is the emphasis on safeguarding user privacy, enabling multimodal communication channels, adeptly navigating conversational misunderstandings, and scaling operations to accommodate a growing user base. These enhancements collectively drive the evolution of conversational AI, setting a new standard for human-computer interaction.

2.1.1 Characteristics and Typology of KCAs

Knowledge-enhanced Conversational Agents, also called knowledge-based conversational agents [16], incorporate external knowledge sources to improve their understanding, generation, and overall interaction capabilities. These agents leverage various forms of knowledge, including databases, knowledge graphs and text corpora to provide more accurate, relevant and informative responses to the user. Depending on the perspective from which they are analyzed, these agents can be categorized into several key groups:

Based on Goal of knowledge used:

- ***Task-Oriented*** agents are designed to accomplish specific tasks or help users achieve particular objectives through conversation. They are usually domain-specific, focusing on a narrow set of functions but executing them efficiently. The goal is to provide quick and accurate responses to user queries or commands related to the agent's particular area of expertise. For example, Personal Assistants which help to manage personal tasks [17] such as setting reminders, scheduling appointments, sending messages, or providing navigation instructions.

- ***Non-Task-Oriented (Social)*** agents often referred to as chatbots or social bots, are designed for open-ended conversation and social interaction. These agents aim to mimic human-like conversational abilities, providing companionship or entertainment rather than performing specific tasks. They are usually open-domain, and capable of discussing a wide range of topics to keep the user engaged.

- ***Hybrid Systems*** that combine elements of both task-oriented and non-task-oriented conversational agents. These hybrids aim to provide a more versatile user experience by being capable of performing specific tasks while also engaging in more open-ended, social interactions. For example, Google Assistant is hybrid because it combines task-oriented functionalities like managing personal tasks and controlling smart devices with non-task-oriented conversational abilities for engaging social interactions and entertainment.

Based on Method of Knowledge Integration:

- ***Static Integration*** agents incorporate fixed set of knowledge at the design time, which doesn't change during interactions.

- ***Dynamic Integration*** agents continuously update or draw knowledge in real-time from various sources like the web, databases, or through user request.

- ***Hybrid Integration*** agents combine both static and dynamic knowledge sources to enhance versatility and depth.

Based on the Nature of the Conversational Interface

- ***Text-based*** agents interact with users through text input and output, convenient for chat applications and text messaging services.

- ***Voice-based*** agents utilize speech recognition and synthesis to enable spoken language interactions, useful for hands-free applications and accessibility.

- ***Multimodal*** agents combine heterogeneous forms of input and output, including text, voice, and even visual elements, to provide a richer interaction experience to the user.

Based on Application Domain

- ***Single-domain*** agents focuses on a specific area or function, providing deep but narrow expertise. For instance, an agent solely dedicated to scheduling medical appointments is a single-domain agent.

- ***Multi-domain*** agents on the contrary operate across multiple, distinct areas (domains) within a broader context, capable of handling a variety of tasks or topics within that context. For instance, an arbitrary agent called Travel Planner which covers information related to flights, hotel bookings, local attraction suggestions etc. would be considered multi-domain agent.

- ***Open-domain*** agents are designed to engage in conversations on a wide range of topics without being restricted to a specific domain or set of tasks. These agents aim to simulate human-like interactions, capable of discussing virtually any subject matter the user wishes to explore. The goal is to maintain a coherent and contextually relevant dialogue over a broad spectrum of topics, mirroring the conversational flexibility of humans. Systems like ChatGPT [18] are examples of open-domain conversational agents capable of engaging users in discussions on a multitude of subjects, from science and technology to art and philosophy.

Based on Interaction Complexity

- ***Single-Turn*** agents handle one query and provide one response per interaction cycle, without maintaining the context or flow of conversation beyond that single exchange. Each query is treated as an isolated event, with no memory of previous interactions within the same session. Single-turn agents are typically used for straightforward tasks, such as answering specific questions or executing simple commands, where the interaction does not require follow-up questions or clarification.

- ***Multi-Turn*** agents are capable of engaging in conversations that involve multiple exchanges between the user and the agent, maintaining context and remembering the flow of conversation across several turns. This ability allows them to handle more complex interactions, such as resolving ambiguities, asking for clarifications, and progressively building upon the user's inputs to reach a conclusion or fulfill a request. Multi-turn conversations mimic more natural human dialogue, where the discussion evolves and the context from earlier parts of the conversation influences responses later on.

2.1.2 Architecture of KCAs

While there are various types of KCAs differing in several aspects, a general architecture that encompasses all of them can still be outlined. Fig. 2.1 depicts the general architecture of KCAs with its key components [19]. The components are:

- **User Interface**: Through this component user interacts with the system. It could be a chat or a voice interface where the users input their requests and receive system responses.

- **NLU**: This module is for processing the user's input and understanding the intent (what the user wants to achieve) and the context (the relation of the input to the dialogue history) of the conversation.

- **Dialogue Manager**: This module is the central part of the architecture that communicates between the NLU, the response generator, and data sources. It controls the flow of the conversation by deciding what action to take based on the user's intent and the context.

- **Response Generator**: Once the dialogue manager has decided on a course of action, the response generator creates a natural language response to the user's query. It is responsible for translating the system's action into something understandable and informative for the user.

- **Data Sources**: These are the unstructured or structured, internal or external knowledge sources from which the system retrieves information to ground its responses.

- **Action execution/Information retrieval**: These are tasks the system may perform to address the user's request. Action execution is about taking the user's intent, which has been understood and processed, and turning it into a real-world action or result. Information retrieval is the process of finding the relevant data from the knowledge base.

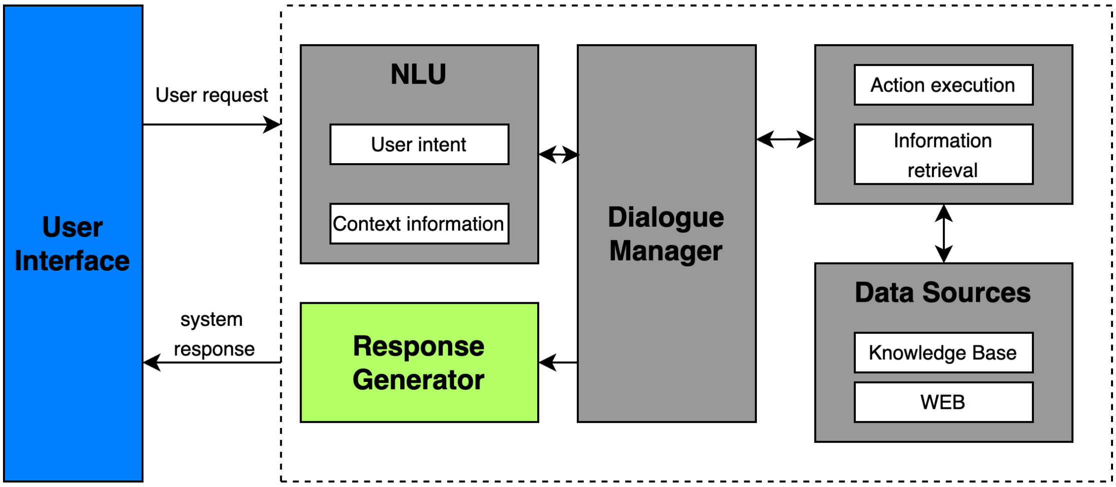


Figure 2.1: Modular architecture of KCAs

2.2 The Dialogue System Technology Challenge

The Dialog System Technology Challenges (DSTC) also formerly known as The Dialogue State Tracking Challenges are annually held, series of competitions aimed at speeding up the advancement of conversational AI technologies and encouraging new work that advances the state-of-the-art methods. Initiated in 2013, these challenges bring together researchers and practitioners from around the world to address pressing problems in dialogue system development, including NLU, dialogue management, and response generation. To reflect the evolving needs and emerging trends within the field of conversational AI, each challenge iteration focuses on specific themes. Through these years, DSTC has expanded its scope from dialogue state tracking to include tasks like end-to-end dialogue modeling, knowledge-grounded conversations, and the integration of subjective knowledge into CAs.

From DSTC1 to DSTC5, the challenges progressively focused on enhancing dialogue systems, starting with dialogue state tracking (also called “belief tracking”) [20] only in a restaurant domain and gradually expanding to more complex scenarios. Each challenge introduced new elements, DSTC3 tested adaptability to unseen domains [21], DSTC4 explored multi-domain dialogues [22], and DSTC5 added cross-lingual capabilities [23]. This evolution demonstrated a continuous push towards improving the robustness, versatility, and real-world applicability of conversational AI technologies.

From DSTC6 onwards, the challenges further expanded the scope (unlike previous editions, multiple tracks proposed) and complexity of conversational AI research. DSTC6 introduced end-to-end dialogue systems and multimodal dialogue capabilities, marking a shift towards more integrated and versatile conversational models [24]. With DSTC7, the focus had widened to include end-to-end conversation modeling and response generation, emphasizing the generation of more contextually relevant and coherent responses [25]. DSTC8 continued this trajectory by exploring multi-domain dialogues and knowledge-grounded conversations, aiming to boost the capability of dialogue systems to provide more informative and accurate responses based on external knowledge sources [26]. DSTC9 built upon these themes, introducing tracks related to interactive evaluation, domain adaptation, and beyond domain APIs, pushing the envelope in terms of system interactivity and adaptability [27]. DSTC10 pushed the boundaries of conversational AI further through a range of innovative challenges, from enhancing dialogues with internet memes to grounding conversations in external knowledge, and advancing multimodal AI. It also focused on improving reasoning in scene-aware dialogues and developing better evaluation tools for dialogue systems, reflecting a broad effort to make AI interactions more engaging, context-aware, and secure [28].

In DSTC11, the focus was on integrating subjective knowledge into task-oriented conversations and improving cross-lingual and cross-domain dialogue state tracking. This challenge underscored the shift towards creating conversational AI systems that are not only adaptable across languages and domains but also capable of personalized interactions, reflecting a deepened commitment to enhancing user experience. The proposed tracks are (1) Ambiguous Candidate Identification and Coreference Resolution for Immersive Multimodal Conversations; (2) Intent Induction from Conversations for Task-Oriented Dialogue; (3) Speech-Aware Dialog Systems Technology Challenge; (4) Robust and Multilingual Automatic Evaluation Metrics for Open-Domain Dialogue Systems; (5) Task-oriented Conversational Modeling with Subjective Knowledge.

We focus on "Knowledge Selection" sub-task of Track 5 which aims to enhance dialogue systems by integrating subjective user preferences and knowledge into task-oriented conversations, and to create more personalized and context-aware interactions.

2.2.1 DSTC9 - track 1 and DSTC10 - track 2

As already mentioned, each year, the DSTCs evolve by introducing more complex and diverse tasks, progressively enhancing conversational AI's capabilities in understanding, adaptability, and interaction with humans and external knowledge sources. Before diving into DSTC11 - track 5, it would be better to introduce how it came to this point. "Beyond Domain APIs: Task-oriented Conversational Modeling with Unstructured Knowledge Access" track of DSTC9 and "Knowledge-grounded Task-oriented Dialogue Modeling on Spoken Conversations" track of DSTC 10 are predecessors of our concerned topic that we should particularly mention.

In DSTC9 - track 1, the aim was to address a significant challenge in conversational AI: enabling task-oriented dialogue (TOD) systems to go beyond the limitations of predefined domain-specific APIs and structured databases (DBs). This track searched ways to explore and develop methodologies that allow dialogue systems to dynamically access, interpret, and utilize unstructured knowledge sources such as text from websites, documents, or open-domain corpora to enhance their conversational competence. For this reason, the turns that could be handled by the existing task-oriented conversational models with no extra knowledge requirement disregarded, and the focus put on the turns that require knowledge access. This task addressed with the following three subtaks: 1) *Knowledge-seeking Turn Detection*, 2)*Knowledge Selection*, and 3) *Knowledge-grounded Response Generation* (see Fig. 2.2).

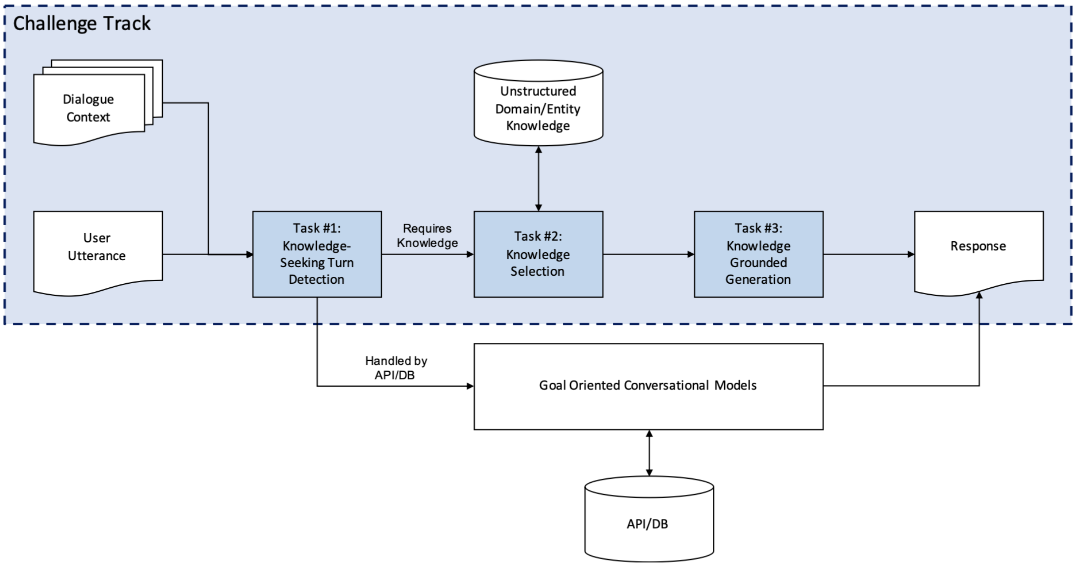

Challenge Track
Dialogue Context
User Utterance
Unstructured Domain/Entity Knowledge
Task #1: Knowledge-Seeking Turn Detection
Requires Knowledge
Task #2: Knowledge Selection
Task #3: Knowledge Grounded Generation
Response
Handled by API/DB
Goal Oriented Conversational Models
API/DB

Figure 2.2: Architecture of the task that focuses on the knowledge access branch in the shaded box [29]

At the end of challenge task, top 12 teams in the overall objective score were selected and their best entries manually evaluated by humans in terms of *Accuracy* and *Appropriateness*. Then the Spearman's rank correlation coefficient (Spearman 1961) was calculated between the ranked lists of all the entries in every pair of objective and human evaluation metrics. As a result, it has been revealed that the *Knowledge Selection* (Task2) metrics has stronger correlations than the other subtasks' (Task1-Detection, Task3-Generation) metrics to the final ranking (see Fig. 2.3) [30]. This denotes how the knowledge selection is a key task to improve end-to-end performance and it helped in later works to researchers to have a deeper view of the task.

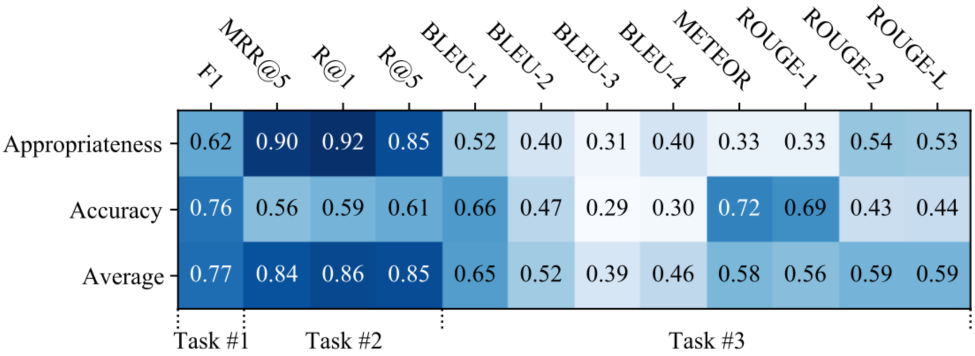


Figure 2.3: Correlations between the objective and human evaluation metrics

In DSTC10 - track 2, two sub-tracks were proposed: 1) multi-domain dialogue state tracking and 2) task-oriented conversational modeling with unstructured knowledge access [31]. Second sub-track was direct extension of the DSTC9 - track 1. The novelty introduced in this challenge was related to spoken conversations. Unlike before, where emphasize put on written conversations, introduction of additional spoken conversations within dataset was also introduction of additional problems to solve and tackle with. Because obviously, the manner of speaking and writing differs, even when the context, intent, and meaning of the conversations are the same. Spoken conversations are prone to have extra noises from disfluencies[1] or barge-ins[2] which are usually not the case in processing written texts. Moreover, errors from speech recognition introduce further complexities in the practical development of spoken dialogue systems and researchers addressed these problems.

2.2.2 DSTC11 - track 5

TOD systems focus on developing CAs that assist users in reaching specific objectives, such as making hotel or restaurant reservations. Traditionally, these systems have concentrated on delivering information and executing tasks based on user requests, limited to the capabilities of predefined DBs [32] or APIs [33]. While being able to help simple user queries like booking hotel, reserving seat in restaurant etc. they fall short to answer follow-up questions user may have, such as "whether

[1] Interruptions or hesitations, such as "um," "uh," repetitions, or corrections.

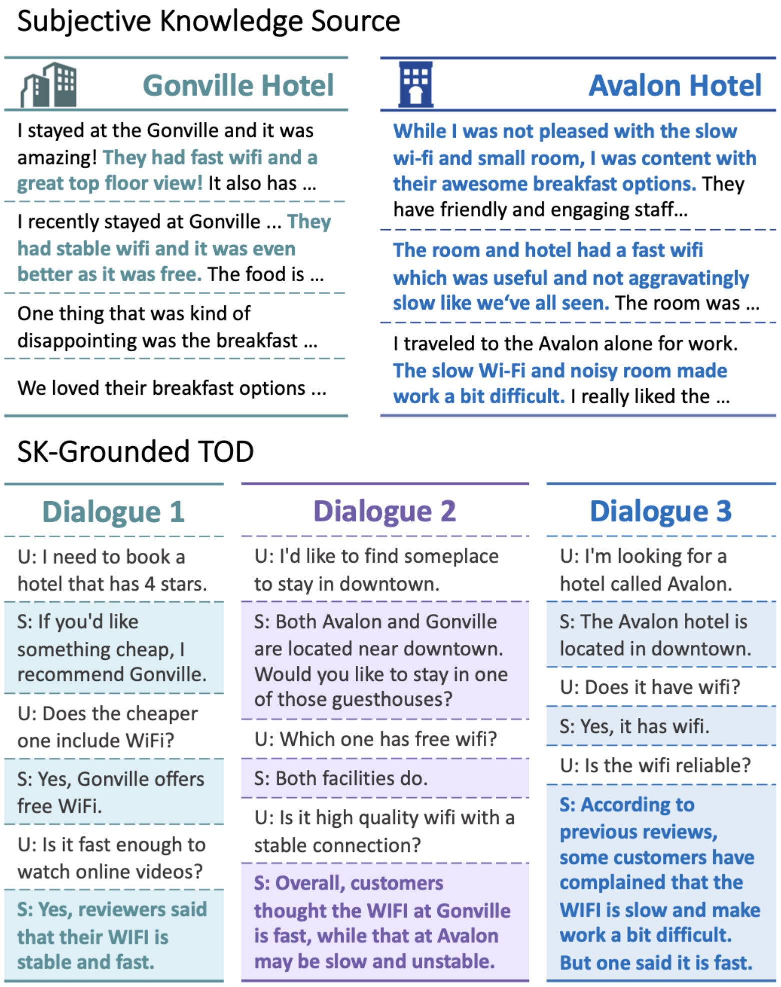
[2] When a speaker interrupts another speaker before they have finished talking.

they are allowed to bring pets" or "what the cancellation policy is". To address this issue, DSTC9 - track 1 proposed exploiting information coming from FAQs by accessing external unstructured knowledge sources. This was effective strategy but it had some gaps too. For instance, when the system gets subjective user requests like "Is the WIFI reliable?" or "Does the restaurant have a good atmosphere?" agent is not able to answer these questions only by using FAQs. In DSTC11 - track 5 for addressing this issue, integration of subjective knowledge sources was introduced as a major target by Zhao et al. [11] to research community. This novel subjective-knowledge-based TOD (SK-TOD) also introduced first corresponding dataset with knowledge-seeking dialogue contexts and manually annotated responses which grounded in subjective knowledge.

As every novelty this one also introduced some intrinsic challenges to tackle with. Unlike its predecessors where factual information (FAQs) about particular entity was required to answer the user request, here there can be cases where we should refer more than one entity to provide satisfying response. Because subjective insights, such as the experiences, opinions, and preferences of other customers which we refer to ground the system response are qualitative concepts. So, the cases where we should compare two or more places for quality of particular facility are inevitable. Also, in cases where even only information about particular entity is required, the system has to ground final response with all the related reviews retrieved, since some customers write positive comments about facility and some of them do not like, and mention it in a negative way. So, the final response should contain general information to let the user to understand what is going on. In fig. 2.4, we see how responses were grounded with subjective knowledge. Above we see customer reviews of corresponding hotels. In below part, dialogue sessions between the system (denoted by S) and three different users (denoted by U) were given, where the last user utterances are knowledge-seeking turns with subjective knowledge asking WIFI quality of the hotel(s). Last system responses which highlighted are system responses grounded with subjective knowledge where response with comparison of two entities (second) and response which covers both positive and negative reviews (third) are shown.

Task Description

In the field of human-computer interaction, particularly in conversational systems or chatbots, understanding the flow and history of a conversation is crucial for generating relevant and coherent responses. This understanding is encapsulated in what is known as the dialogue context. The dialogue context is a structured representation of the conversation that captures both the user's inputs and the system's responses up to the current point in time. It serves as a foundation for the



Subjective Knowledge Source

Gonville Hotel

I stayed at the Gonville and it was amazing! They had fast wifi and a great top floor view! It also has ...

I recently stayed at Gonville ... They had stable wifi and it was even better as it was free. The food is ...

One thing that was kind of disappointing was the breakfast ...

We loved their breakfast options ...

Avalon Hotel

While I was not pleased with the slow wi-fi and small room, I was content with their awesome breakfast options. They have friendly and engaging staff...

The room and hotel had a fast wifi which was useful and not aggravatingly slow like we've all seen. The room was ...

I traveled to the Avalon alone for work. The slow Wi-Fi and noisy room made work a bit difficult. I really liked the ...

SK-Grounded TOD

Dialogue 1

U: I need to book a hotel that has 4 stars.

S: If you'd like something cheap, I recommend Gonville.

U: Does the cheaper one include WiFi?

S: Yes, Gonville offers free WiFi.

U: Is it fast enough to watch online videos?

S: Yes, reviewers said that their WIFI is stable and fast.

Dialogue 2

U: I'd like to find someplace to stay in downtown.

S: Both Avalon and Gonville are located near downtown. Would you like to stay in one of those guesthouses?

U: Which one has free wifi?

S: Both facilities do.

U: Is it high quality wifi with a stable connection?

S: Overall, customers thought the WIFI at Gonville is fast, while that at Avalon may be slow and unstable.

Dialogue 3

U: I'm looking for a hotel called Avalon.

S: The Avalon hotel is located in downtown.

U: Does it have wifi?

S: Yes, it has wifi.

U: Is the wifi reliable?

S: According to previous reviews, some customers have complained that the WIFI is slow and make work a bit difficult. But one said it is fast.

Figure 2.4: Examples of the SK-TOD task [11]

system to interpret the user's latest utterance[3] and decide on the most appropriate response.

The dialogue context $C = [U_1, S_1, U_2, S_2, \ldots, U_t]$ is given between the user and the system. Each user utterance U_i is followed by system response S_i, aside from the last user utterance U_t. The conversation involves one or more entities, denoted as $E = \{e_1, e_2, \ldots, e_m\}$. Beside this we have subjective knowledge source $B = [(e_1, R_1), (e_2, R_2), \ldots]$ which contains all the entities and their corresponding customer reviews. Each entity e has multiple reviews (also called knowledge snippets) $R = \{R_1, R_2, R_3, \ldots\}$ and each review can be divided into segments

[3] Spoken or written statement that conveys a complete idea or thought by a speaker or writer

$[K_1, K_2, \ldots]$, such as paragraphs, sentences, sub-sentences.

The architecture of SK-TOD is similar to precedent challenges' architecture. There is only a small difference in Knowledge Selection phase which is separated into two steps ($EntityTracking + KnowledgeSelection$) unlike before. In Figure 2.5, the pipeline architecture is shown with all four sequential sub-tasks. Each sub-task has its unique goal:

- **Knowledge-seeking Turn Detection**: The goal here is to define if the last user utterance U_t in the given dialogue requires external knowledge access or not. It is regarded as a binary classification problem, where the dialogue context C is used as an input, and the output is a binary indicator.

- **Entity Tracking**: In this step, the goal is to determine a subset of relevant entities from the pool $E = \{e_1, e_2, \ldots, e_m\}$. This step is added to decrease computational cost since it reduces number of candidates to look in knowledge selection phase.

- **Knowledge Selection**: The goal here is to select relevant knowledge snippets for the user's request. Given the dialogue context C and the set of candidate knowledge snippets K of the previously determined relevant entity or entities E, $K^+ \subseteq K$ subset of relevant knowledge snippets selected. There is no exact number K^+ to retrieve relevant knowledge snippets like previous challenges.

- **Response Generation**: Final step, where the goal is to create system response utterance S_t. Given the dialogue context C and the selected relevant knowledge snippets K^+, the response is generated.

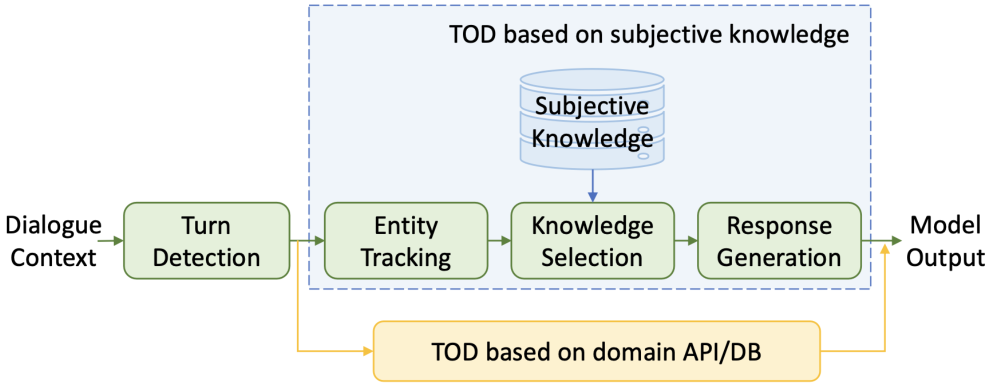


Figure 2.5: Architecture of the SK-TOD [11]

Dataset

In DSTC11 - track 5, organizers introduced an augmented version of MultiWOZ 2.1 [34] dataset for the participants. The dataset includes newly introduced subjective knowledge-seeking turns and it was collected via Amazon Mechanical Turk[4] by English-speaking crowd workers from the USA, CA, and GB, who underwent pre-qualification tests and whose work was manually validated for quality. Data collection included review generation with specified personas, aspects, and sentiments based on common hotel and restaurant review criteria.

In total, 19,695 instances with subjective user requests and subjective knowledge-grounded responses were collected. Besides this, another 18,363 dialogues without subjective user requests were sampled from the original MultiWOZ dataset to support the knowledge-seeking turn detection task. After creating a mixture of both, the final dataset was split into 75% *training* set, 10.8% *validation* set and 14.2% *test* set. In the validation and test sets, there are unseen instances, meaning their aspects are not included in the training set. This is to evaluate the model's ability to generalize to arbitrary aspects. Also, as previously mentioned, this challenge involves numerous cases where dialogue contexts are associated with multiple ground-truth entity labels, adding to the task's complexity. The number of instances with multiple entities for each set, along with all the previously mentioned statistics, are detailed in Table 2.1.

	Train	**Val**	**Test**
Full data	28,431	4,173	5,475
Knowledge-seeking data	14,768	2,129	2,798
Seen instances	14,768	1,471	1,547
Unseen instances	0	658	1,252
Multi-entity instances	412	199	436

Table 2.1: General statistics of DSTC11 - track 5

[4] Crowdsourcing platform that connects businesses and developers with a global workforce to complete tasks that require human intelligence

The challenge dataset is organized as follows: It comprises a Dialogue dataset, which features conversations between users and a system (see Fig. A.3), and a Labels dataset, containing ground-truth labels for the corresponding instances in the dialogue dataset (see Fig. A.5). Additionally, there's a Knowledge dataset (knowledge base), which consists of reviews and FAQs related to each specific domain/entity (see Fig A.2). Here, *domain* refers to categories like "hotel" and "restaurant," and *entity* denotes individual names, such as a specific hotel (e.g., BRIDGE GUEST HOUSE) or a restaurant (e.g., ALEXANDER BED AND BREAKFAST). Each entity is accompanied by its unique collection of review documents and FAQs. The review documents not only contain textual sentences but also include metadata providing further details on each review, covering aspects such as traveler type (e.g., Couples, Business Travelers), notable dishes (e.g., Beer-Braised Chicken Stew), and drinks (e.g., Beer, Ale). The FAQs are stored as question-and-answer pairs. Figure 2.6 illustrates an example of a knowledge-seeking dialogue context between a user and a system, shown in the upper section, with its corresponding ground-truth label displayed in the lower section. In this example, only one knowledge snippet is identified as the ground-truth label for the Knowledge Selection task. Generally, there is more than one ground-truth knowledge snippet. This particular example has been selected solely for visualization purposes. All the information in the knowledge section refers to the corresponding knowledge snippet in the knowledge base (KB).

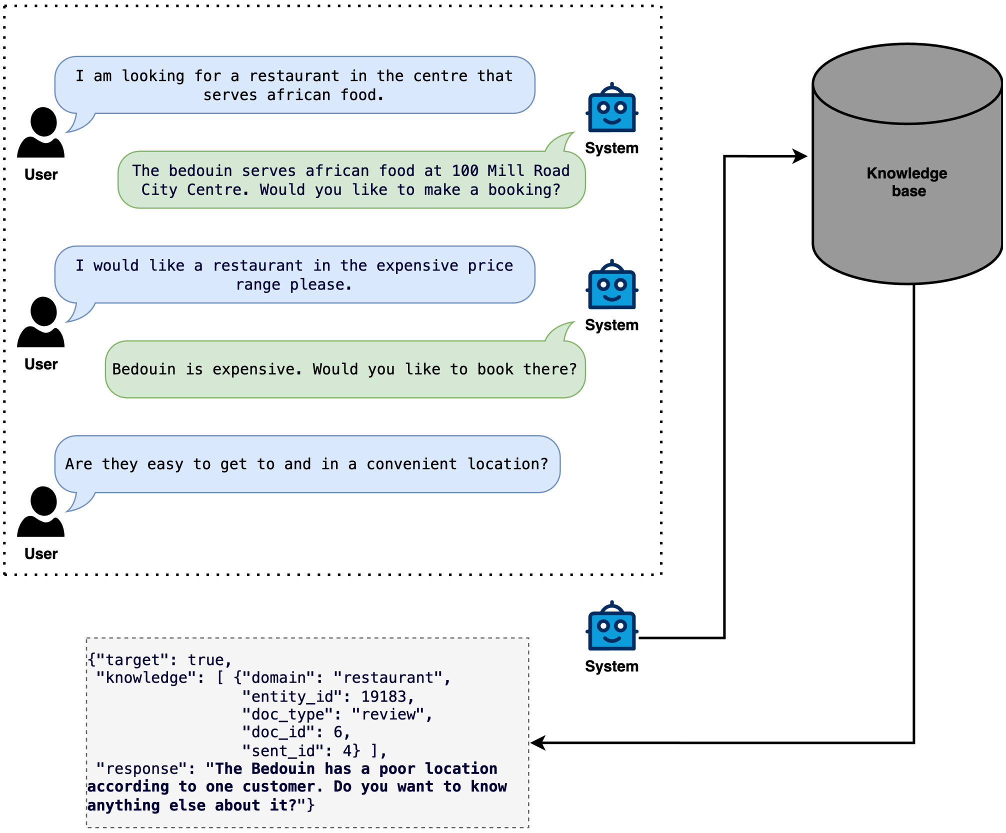
Table 2.2 provides statistical data on domains, showing that there are only two distinct domains across all examples. It details their distribution in the training and validation sets and lists the number of unique entities within each domain.

	Hotel	**Restaurant**
Dialogues(Train)	7,859	6,909
Dialogues(Val)	1,436	693
Entities	33	110

Table 2.2: Statistics for Domain

Evaluation metrics

In this challenge, each participating team is permitted to submit up to five system outputs, each containing the results for all three tasks on the unlabeled test instances. Evaluation is done in two phases. Firstly, each submission is evaluated using task-specific objective metrics (see Table 2.3) by comparing to the ground-truth labels and responses. Then, based on the overall objective score, the finalists are selected and manually evaluated.


I am looking for a restaurant in the centre that serves african food.
User
System
The bedouin serves african food at 100 Mill Road City Centre. Would you like to make a booking?
I would like a restaurant in the expensive price range please.
User
System
Bedouin is expensive. Would you like to book there?
Are they easy to get to and in a convenient location?
User
Knowledge base
System

```
{"target": true,
 "knowledge": [ {"domain": "restaurant",
                "entity_id": 19183,
                "doc_type": "review",
                "doc_id": 6,
                "sent_id": 4} ],
 "response": "The Bedouin has a poor location
according to one customer. Do you want to know
anything else about it?"}
```

Figure 2.6: Example of knowledge-seeking dialogue and its corresponding label instance

Tasks	Evaluation metrics
Knowledge-seeking turn detection	$Precision/Recall/F_1score$
Knowledge selection	$Precision/Recall/F_1score/Accuracy$
Response generation	$BLEU/ROUGE/METEOR$

Table 2.3: Task-specific objective metrics for DSTC11- track5

Given the interconnected nature of tasks within the pipelined framework, the final scores for knowledge selection and response generation are computed by taking into account the initial step of knowledge-seeking turn detection, specifically its recall and precision metrics, as follows:

$$S_p(X) = \frac{\sum_{x_i \in X}(s(x_i) \cdot f_1(x_i) \cdot \tilde{f}_1(x_i))}{\sum_{x_i \in X} \tilde{f}_1(x_i)}, \tag{2.1}$$

$$S_r(X) = \frac{\sum_{x_i \in X} \left(s(x_i) \cdot f_1(x_i) \cdot \tilde{f}_1(x_i) \right)}{\sum_{x_i \in X} f_1(x_i)}, \tag{2.2}$$

$$S_f(X) = \frac{2 \cdot S_p(X) \cdot S_r(X)}{S_p(X) + S_r(X)}, \tag{2.3}$$

where $f_1(x)$ and $\tilde{f}_1(x)$ represent the reference and the predicted outcomes for the task of knowledge-seeking turn detection and, $s(x)$ denotes the score associated with either knowledge selection or the response generation, based on a target metric for an individual instance $x \in X$.

Then a set of multiple scores across different tasks and metrics are aggregated into a single overall objective score to define the finalists. This score is mean reciprocal rank and it is computed as follows:

$$S_{\text{overall}}(e) = \frac{1}{|\mathcal{M}|} \sum_{i=1}^{|\mathcal{M}|} \frac{1}{\text{rank}_i(e)}, \tag{2.4}$$

where $rank_i(e)$ is the submission entry e ranking in the $i - th$ metric with respect to all the other submissions and, M is the number of metrics that have been considered.

After defining finalists, manual evaluation phase is done by the following two crowd-sourcing tasks:

- **Appropriateness**: whether the response is fluent and naturally connected to the dialogue context.
- **Accuracy**: whether the sentiment proportion provided by the response is accordant with that of the subjective knowledge.

Chapter 3

Related work

In this chapter, related work is discussed. It is started with an explanation of how text data is handled in machine learning. The techniques used are examined, and their development is demonstrated by providing how each technique was proposed to address some issues related to those existing at that period of time. Then, the most famous information retrieval techniques are discussed, especially in the context of DSTC challenges. Their strengths and weaknesses are precisely demonstrated to lay the base for the choice of future methodology.

3.1 Generation of Sentence Embeddings

The focus in NLP is on the interactions between computers and human language and, in particular, on how to program computers to process this textual data efficiently. As is well known, machines inherently lack the capability to process language data in its raw form. Consequently, it is imperative to represent textual information numerically before any form of processing or analysis can be undertaken. For this reason, various techniques are applied to obtain *embeddings* of words or sentences, transforming textual data into a numerical representation that machines can process.

Sentence embeddings are a common method in NLP that involves mapping sentences into *fixed-length vectors* of real numbers in a high-dimensional space, moving beyond the focus on individual words, as seen in word embeddings. Sentence embeddings can be obtained either through sparse vectorization methods like Bag-of-Words or TF-IDF, which create high-dimensional, sparse representations, or through dense vectorization methods like Recurrent Neural Networks or Transformer-based models, which generate continuous, low-dimensional embeddings capturing semantic meaning in dense vector spaces. The second approach is designed to encapsulate the semantic content of sentences, incorporating not just the specific words present, but also the broader context and subtle nuances of their arrangement. Such embeddings are crucial to a variety of NLP applications that require understanding the meaning of texts at a higher level than individual words, such as Semantic Textual Similarity, Information Retrieval, and Question Answering. By transforming sentences into vectors that carry their semantic meaning, sentence embeddings facilitate the processing and analysis of text at a nuanced level, crucial for the sophisticated understanding and interaction that modern NLP tasks require.

3.1.1 Sparse Vector Representation

Sparse vector representation methods are fundamental and pioneering techniques in NLP and in general ML, used to convert text data into numerical vectors. These methods encode text information into vectors with mostly zero elements, making it compact and efficient for storage and processing. Despite its simplicity, sparse vector representation provides essential benefits, they are straightforward to implement and understand, adaptable to apply various NLP tasks and computationally efficient. Techniques like the Bag of Words model and TF-IDF rely on sparse vectors, laying the groundwork for subsequent advancements in text representation.

The Bag of Words (BoW)

BoW method represents text data by breaking down sentences into individual words or phrases and counting their occurrences, disregarding grammar and word order but maintaining multiplicity. Simply, the idea is to transform textual data into numerical features that can be used for various computational tasks. It treats every word as equally significant, without considering the word's relevance across the corpus. This can sometimes lead to overemphasizing common words[1] that might not be useful for some tasks such as classification and, search.

For a corpus C containing D documents $\{d_1, d_2, \ldots, d_D\}$ and a vocabulary V consisting of N unique words $\{w_1, w_2, \ldots, w_N\}$ extracted from the corpus, the BoW representation of a document d_i is a vector $\mathbf{v}_i = [v_{i1}, v_{i2}, \ldots, v_{iN}]$, where each element v_{ij} corresponds to the frequency of the word w_j in the document d_i.

The frequency of a word w_j in a document d_i can be represented as:

$$v_{ij} = \mathrm{freq}(w_j, d_i) \tag{3.1}$$

where v_{ij} is the number of times w_j appears in d_i.

[1] The words that have higher frequency across documents

Term Frequency-Inverse Document Frequency (TF-IDF)

The TF-IDF method provides an improvement to a limitation that is inherently introduced by the BoW method. As already stated, the BoW method simply counts word occurrences and it often mistakenly assigns high importance to common articles like 'the', despite their minimal contribution to a sentence's meaning. TF-IDF solves this problem by weighing the term frequencies (TF) with the inverse document frequency (IDF), thereby reducing the weight of words that occur too frequently across documents and are thus less informative. In summary, TF-IDF provides a more nuanced and balanced representation of text data, emphasizing words that are particularly characteristic of a document.

The objective of the TF is to measure how frequently a term (word) occurs in a document. Since each document's length differs, it is probable that a term would appear more frequently in longer documents than in shorter ones. Hence, the term frequency is often divided by the document length (the total number of terms in the document) as a way of normalization:

$$TF(w_j, d_i) = \frac{\text{freq}(w_j, d_i)}{\sum_{k=1}^{N} \text{freq}(w_k, d_i)} \tag{3.2}$$

where $TF(w_j, d_i)$ is the term frequency of word w_j in document d_i.

The objective of IDF is to measure how important the term is. While computing TF, all terms are considered equally important. However, it is known that certain terms, such as "is", "for", and "this", may appear a lot of times but have little or no importance. Therefore we need to weigh down the frequent terms while scaling up the rare ones, by calculating the following equation:

$$IDF(w_j, C) = \log\left(\frac{D}{1 + |\{d_i \in C : w_j \in d_i\}|}\right) \tag{3.3}$$

where $|\{d_i \in C : w_j \in d_i\}|$ is the number of documents where the word w_j appears (i.e., df_j), and D is the total number of documents in the corpus. The addition of 1 in the denominator is to avoid division by zero for terms that appear in all documents.

Finally, the TF-IDF score is calculated by multiplying these two values:

$$TF\text{-}IDF(w_j, d_i, C) = TF(w_j, d_i) \times IDF(w_j, C) \tag{3.4}$$

Disadvantages

These methods have some drawbacks because of their innate structure. Below the most significant ones are described:

- **High dimensionality**: Since the size of the vector representation increases proportional to the size of the vocabulary, having large dataset equals to having high-dimensional vectors and it leads to inefficiency in terms of storage and processing time.
- **Sparsity**: Since each document typically contains only a small subset of the vocabulary, most of the elements in the vectors are zeros and this also leads to inefficiency in terms of memory usage and may not be optimized for some ML algorithms, which perform better with dense input features.
- **Loss of order**: Since the models disregard the syntax and structure of the words within the sentence, it leads to a loss of contextual information.
- **Semantic understanding**: Inherently, these models do not capture the semantic relationships between words, which limits their ability to understand the text meaningfully.

3.1.2 Dense Vector Representation

While we have discussed the pros and cons of sparse vector representation methods, they are not as widely used as before, except in certain specific applications. Sentence embeddings, generated by deep learning models such as Recurrent Neural Networks and Transformer-based models, quickly became more prevalent following their introduction as a tool. These dense vector representations transform sentences into continuous, high-dimensional vectors where semantically similar sentences are mapped to proximate points in the vector space. Unlike sparse representations, which result in high-dimensional vectors with many zeros, dense vectors are typically lower-dimensional and fully populated, meaning every dimension has a meaningful value.

Recurrent Neural Networks (RNNs)

RNNs [35] are a special kind of deep learning models and they are used to detect patterns in sequential data. They have a form of "memory" that retains information from previous computations, enabling them to predict subsequent elements in a sequence. This feature allows RNNs to perform tasks that require an understanding of temporal dynamics where the context evolves over time, such as language modeling, speech recognition, and time-series analysis. In particular for text data, these models handle phrases, sentences, and entire documents by processing them word by word or sentence by sentence based on target, capturing the sequential information, thus generating representations that consider order and the context of words or sentences.

We can show the core functionality of RNNs with simple yet powerful set of equations. At each time-step t, the hidden state h_t of the network is computed based on the current input x_t and the previous hidden state h_{t-1}:

$$h_t = \sigma(W_{hh}h_{t-1} + W_{xh}x_t + b_h) \tag{3.5}$$

where W_{hh} and W_{xh} are weight matrices, b_h is a bias vector, and σ represents a non-linear activation function, such as tanh or ReLU. The output at each time-step, y_t, is then calculated from the current hidden state:

$$y_t = W_{hy}h_t + b_y \tag{3.6}$$

Since having mentioned how RNNs function and what kind of advantages they bring, now it is also important mention the disadvantages come with them such as vanishing and exploding gradients. These issues arise as the errors get back-propagated through each time-step and can either shrink exponentially, becoming negligible (vanishing), or increase exponentially, becoming too large to manage (exploding), which makes training RNNs on long sequences a challenging task. Moreover, RNNs can struggle with long-term dependencies due to their inherent design of sequential processing, meaning that they might not successfully retain information from earlier time-steps in long sequences and, this sequential nature also restricts parallel processing capabilities, impacting training efficiency on modern hardware.

Long Short-Term Memory Units (LSTMs)

LSTM network [36] was introduced as a sophisticated solution to improve the limitations of RNNs with its special gating mechanism that regulates the flow of information. This architecture allows LSTM to effectively retain important information across long sequences while filtering out unnecessary data, solving the problem of vanishing gradients and enhancing the model's capability to capture long-term dependencies within the data.

The effectiveness of LSTM lies in its unique cell structure, which consists of several key components: the cell state and three types of gates: forget gate, input gate, and output gate. These components work together to control the flow of information like an orchestra.

1. **Forget Gate** (f_t): This gate determines which information the LSTM should discard from the cell state. It checks the previous hidden state h_{t-1} and the

current input x_t, passing them through a Sigmoid function σ, which outputs numbers between 0 (forget) and 1 (keep).

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{3.7}$$

2. **Input Gate** (i_t) **and Candidate Cell State** ($\tilde{C_t}$): Together, these components define which new information will be added to the cell state. The input gate determines the values to update, and the candidate cell state creates a vector of new candidate values that can be added to the state.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{3.8}$$

$$\tilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{3.9}$$

3. **Cell State Update**: The old cell state C_{t-1} is updated to the new cell state C_t. This update is a combination of forgetting the things marked by the forget gate and incorporating the new candidate values adjusted by how much we decide to update each state value.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C_t} \tag{3.10}$$

4. **Output Gate** (o_t) **and Hidden State** (h_t): The output gate defines what the next hidden state h_t should be. The hidden state includes information about previous inputs and is used for predictions. The output gate checks the previous hidden state and the current input and determines which parts of the cell state to output. Then, we filter the cell state through tanh (to push the values to be between -1 and 1) and multiply it by the output of the output gate, so the output is only the parts we decided to.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{3.11}$$

$$h_t = o_t * \tanh(C_t) \tag{3.12}$$

Even if LSTM provides all these benefits over RNNs, it also comes with its tradeoffs. LSTM has high computational complexity which makes it more challenging to train and demands significant computational resources. Nevertheless, its proficiency in capturing profound temporal patterns in data makes it invaluable for various applications such as language translation and predicting stock market trends.

Transformers

The Transformer model architecture had been introduced by Vaswani et al. to the deep learning community in 2017 with the paper called "Attention Is All You Need" [37] and this change revolutionized the field of NLP. Transformers, like RNN and LSTM models, also belong to the category of deep learning models specifically designed for processing sequential data, but unlike them, instead of processing data sequentially, these models use a mechanism called self-attention to weigh the importance of different parts of the input data. This allows them to process all words or sentences in the sequence simultaneously and, it leads to more efficient training on modern hardware. Moreover, this mechanism captures relationships between words better than previous models regardless of their distance in the text. In summary, transformers are more flexible and easy to generalize on a wide variety of deep learning tasks.

In the original paper, the transformer model was composed of an encoder and a decoder, each consisting of a stack of identical layers (see Fig. 3.1).

- **Encoder**: The input of sequence of symbol representations $(x_1, \ldots, x_n)$ is mapped into a sequence of continuous representations $z = (z_1, \ldots, z_n)$ by the encoder. Each encoder layer consists of two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. Residual connections around each of these sub-layers, followed by layer normalization, help to facilitate deep network training.

- **Decoder**: Given z, the decoder is responsible for producing an output sequence $(y_1, \ldots, y_n)$ of symbols. Each decoder layer contains the two sub-layers present in the encoder, with an additional multi-head attention layer that focuses on the encoder's output. This architecture allows the decoder to focus on relevant parts of the input sequence, facilitating tasks like translation where the alignment between input and output elements is crucial.

Later adaptations of transformers have been tailored for specific tasks that may use only the encoder or decoder part. Encoder-only transformer models, like BERT (Bidirectional Encoder Representations from Transformers) [38], are excellent at understanding and analyzing text, making them a perfect fit for tasks like sentiment analysis and question answering. They operate by processing input sequences to generate comprehensive embeddings that represent the sequence's context. Conversely, decoder-only models, like GPT (Generative Pretrained Transformer) [39], excel in generating coherent and contextually relevant text based on a given input, making them perfect fit for applications in text generation and CAs. Both types of models leverage the transformer architecture's strengths, with encoder-only models focusing on input interpretation and decoder-only models on output generation.

The main drawbacks of transformer models include their need for significant computational resources, particularly for larger models, and the complexity of the attention mechanism, which makes them challenging to interpret and understand.

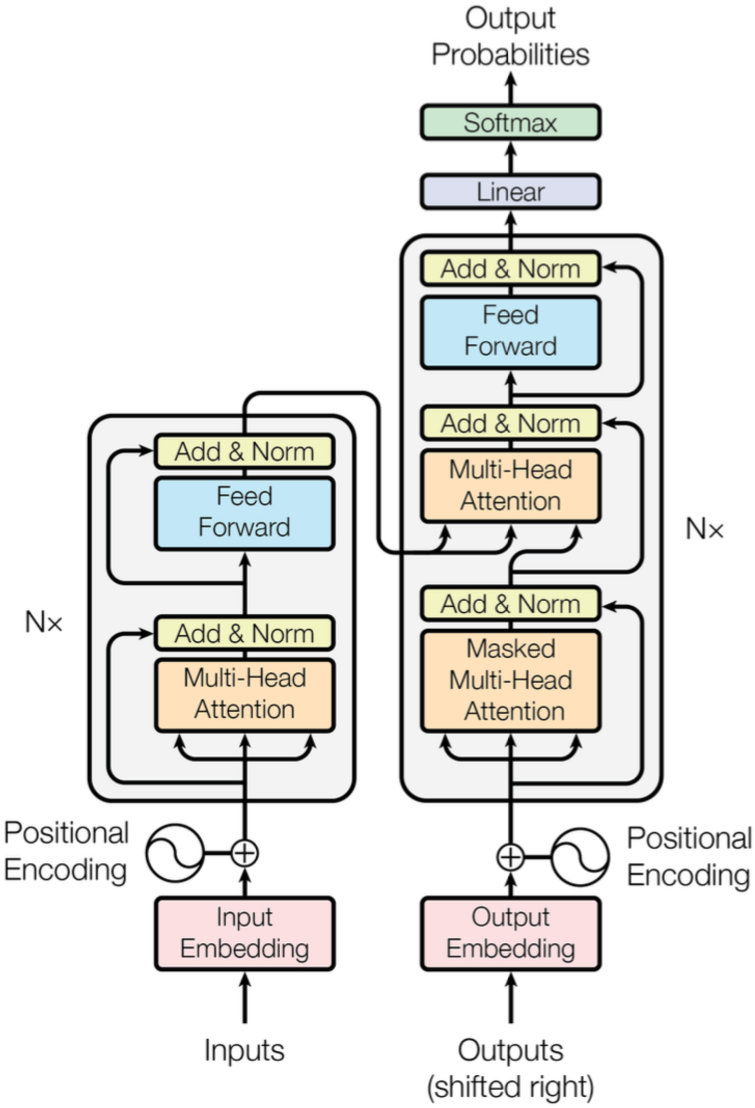


Figure 3.1: The Transformer model architecture [37]

3.2 Information retrieval techniques

Information retrieval (IR) is the discipline of extracting information relevant to a particular need from a large collection of data. This domain integrates various aspects of computer science, library science, and information science to manage, search, and organize information in a way that makes it easily accessible to users. The main objective of IR techniques is to find, understand, and provide information that meets a user's request, covering everything from simple keyword searches to complicated natural language queries. There are various fields of IR applications such as Web Search Engines, Social Media and Content Platforms and Question Answering Systems. Each field requires particular approaches and in DSTC Task-oriented Conversational Modeling challenges two methods have got special attention for Knowledge Selection sub-task, Passage Re-Ranking and Dense Knowledge Retrieval. Picking proper IR technique to apply is crucial in terms of accuracy of retrieval process and computational cost.

3.2.1 Passage Re-Ranking

Passage Re-Ranking (PRR) [40] is a method, where for an input query, candidate passages (knowledge snippets) in database are re-ranked based on their relevance. PRR integrates advanced models such as BERT into the retrieval process to refine search results, leading to outcomes that are more accurate and contextually relevant compared to traditional methods. For instance, in case of BERT, the query and the passage are concatenated with $[CLS]^2 + \textit{query} + [SEP]^3 + \textit{passage}$ structure and fed to the model to calculate relevance score. In general, the process can be outlined simply as follows:

- **Feature extraction**: Given a final user utterance U_t and knowledge base $K = (k_1, \ldots, k_n)$ consisting of knowledge snippets, each (U_t, k_i) pair is fed to the model and relevance score s_i is calculated.
- **Re-ranking**: Based on the scores, knowledge snippets are re-ranked.
- **Retrieval**: $K^+ \subseteq K$ relevant knowledge snippets are retrieved based on defined strategy which can be top n documents or all knowledge snippets above defined relevance threshold.

[2] Special marker added at the beginning of input sequences in transformer models, used to aggregate a representation of the entire sequence for classification tasks.

[3] Special marker used in transformer models like BERT to indicate the separation between sentences or the end of a sentence in a sequence

The choice of loss function for training the model is essential for a method like PRR. Binary Cross-entropy loss is one of the most prevalent function for fine-tuning PRR models. Given the query, this loss function optimizes the model to correctly classify passages as either relevant or irrelevant (positive or negative).

$$L = -\sum_{j \in J_{\text{pos}}} \log(s_j) - \sum_{j \in J_{\text{neg}}} \log(1 - s_j) \tag{3.13}$$

- The first part of the equation $-\sum_{j \in J_{\text{pos}}} \log(s_j)$ is for penalizing the model when it assigns low scores for positive passages where J_{pos} is the set of all positive instances and s_j is the model's predicted relevance score for each positive instance in the set. The logarithm function amplifies the penalty when s_j deviates from the actual label (which is 1 for positives), pushing the model to increase these scores.

- The second part of the equation $-\sum_{j \in J_{\text{neg}}} \log(1 - s_j)$ does the opposite. It penalizes the model when it assigns high scores to negative passages. Here J_{neg} is the set of negative instances. The loss function pushes the model to lower the scores for negative passages, the penalty increases when the model incorrectly assigns high relevance score (s_j) to a passage that is actually not relevant.

Although PRR is a very effective method for informational retrieval process, it is not so efficient. Since the inference is conducted online, the re-ranking process must also be performed in real-time for each query, making it a computationally expensive process. Furthermore, given that time complexity increases linearly with the size of the knowledge base, its practicality for real-world scenarios may be questionable.

3.2.2 Dense Knowledge Retrieval

Dense Knowledge Retrieval (DKR) [41], also known as Dense Passage Retrieval [42] is another sophisticated method in the field of information retrieval which uses the power of advanced models, particularly in the context of question answering and various other NLP tasks. Traditional retrieval methods, such as TF-IDF or BM25 [43], rely on sparse vector representations, leading to the loss of various features which we extensively discussed in previous sections. In contrast, DKR uses dense vector representations for both queries and documents, enabling more nuanced and semantically rich matching between them. The DKR method also mitigates the issue of time complexity during the inference phase, a primary limitation associated with the PRR method. It does so by indexing all the knowledge snippets in a low-dimensional and continuous space one single time offline, such that it can retrieve efficiently the relevant snippets to the input request for the reader at run-time.

- **Offline phase**: Embedding space is created by indexing all the knowledge snippets in knowledge base $K = (k_1, \ldots, k_n)$.
- **Online phase**: When the system gets user query U_t, the query is converted into dense vector representation. The same model is used as was used for the knowledge snippets. Then the system calculates the dot product or cosine similarity between the query embedding and knowledge embeddings to find the most relevant ones to the query embedding.

The choice of loss function for training the model to generate effective embeddings for both queries and knowledge snippets is crucial in DKR method. The goal is the same with PRR method, ensuring that the embeddings of queries are close to the embeddings of their relevant knowledge snippets (positives) while being distant from the embeddings of irrelevant ones (negatives). Triplet loss [44] is a widely adopted function for fine-tuning models by simultaneously feeding an anchor, a positive, and a negative sample during training. Here α distance margin is used to ensure the positive is at least α away from the negative example (see fig. 3.2). The loss function is described below:

$$L(a, p, n) = \max\Big(d(a_i, p_i) - d(a_i, n_i) + \text{margin}, 0\Big) \tag{3.14}$$

where a, p, n are anchor, positive and negative respectively, d is distance function.

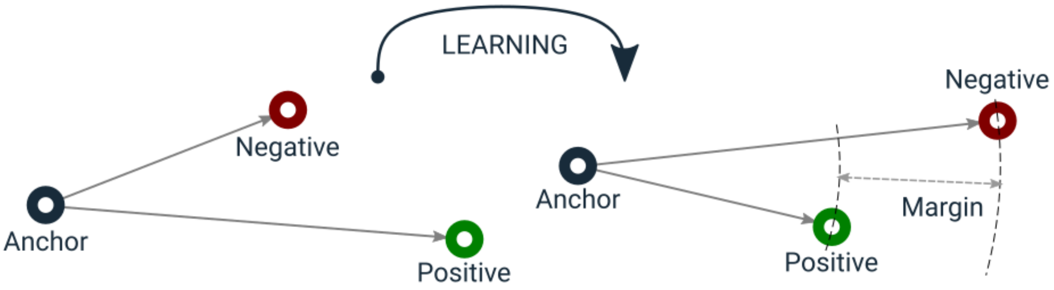


Figure 3.2: A visual representation illustrates how Triplet Loss brings the positive example closer to the anchor and pushes the negative example α margin away, based on the Euclidean distance function.

Like most methods, DKR comes with its trade-offs. While it reduces inference time through offline embedding space generation and enables fast retrieval from large corpora, the compromise lies in accuracy. The point of the issue is that in the embedding space, knowledge snippets with similar sentence meanings receive similar embeddings. Consequently, during inference, there's a high likelihood of retrieving incorrect knowledge snippets without considering their actual entities. For instance, a user query regarding the wifi quality of a particular hotel may receive high similarity scores for knowledge snippets of other hotels mentioning the same topic, which fails to satisfy the user's request. Thus, although the DKR method is well-suited for open-domain information retrieval tasks, its effectiveness may be limited in domain-specific contexts due to this disadvantage.

Chapter 4

Methodology

In the context of IR tasks, it is extremely important to retrieve relevant information to satisfy users' requests. That's why researchers put emphasis on accuracy in the first place. However, inference time is also a vital aspect of these tasks. Without efficient inference time, methods developed for high-accuracy tasks may not be practical for real-world scenarios. Researchers often face trade-offs between accuracy and efficiency, making it crucial to find a balanced approach that delivers highly accurate results within a reasonable timeframe.

In this chapter, the methodology adopted is discussed, providing a deep understanding of the reasons behind the choice of methods. This explanation begins with an overview of how predecessor methods have been applied in similar information retrieval processes. It then proceeds to highlight the developments in terms of accuracy and efficiency that these applications have undergone, thereby framing the work within an evolutionary context.

4.1 Effective and Efficient Information Retrieval

4.1.1 Hierarchical PRR and DKR

The DSTC9 Track 1 challenge was fruitful, yielding a variety of useful methods proposed for the KCA task. The research conducted by Thulke et al. [41] was notable for its application of various techniques aimed at enhancing the efficiency of the retrieval process without compromising accuracy. They introduced two approaches to enhance task efficiency: Hierarchical Selection and Dense Knowledge Retrieval.

In the hierarchical selection method, they focused on improving inference time by dividing retrieval task into three parts. Because they detected that one of the main issues related baseline PRR method was its computational complexity. Considering the knowledge base K with all its knowledge snippets, the computational complexity of the PRR task is given by:

$$\mathcal{O}(|K|) = \mathcal{O}\left(|D| \cdot \frac{|E|}{|D|} \cdot \frac{|K|}{|E|}\right), \tag{4.1}$$

where D is total number of domains and, E is total number of entities. So, the total number of computations required is equal to the total number of knowledge snippets. Moreover, use of larger encoder models for more accurate results will increase the computation time of a single operation and consequently the PRR method's feasibility is questionable for real-world scenarios. For this reason, the hierarchical selection method proposes a strategy to decrease inference time by eliminating most of the unrelated knowledge snippets from the computation process. It achieves this by dividing the retrieval process into three sub-parts. By leveraging the metadata available in the knowledge base, the relevant domain is initially identified based on the user query. Consequently, all knowledge snippets belonging to other domains are automatically excluded. Subsequently, the relevant entity within that domain is determined. In the final stage, only the knowledge snippets associated with that entity are encoded and compared to the user query to identify the relevant ones. Below the complexity of the task is given:

$$\mathcal{O}\left(|D| + \frac{|E|}{|D|} + \frac{|K|}{|E|}\right), \tag{4.2}$$

As can be seen, the complexity has decreased drastically in comparison to the baseline PRR method applied in the challenge [29]. In this case, we only need to consider the knowledge snippets that satisfy $K \in E \in D$ to be passed through the encoder model.

Although the hierarchical selection method offers significant improvements, its efficiency in real-world scenarios is not guaranteed either. This is because knowledge bases for retrieval tasks tend to have a tremendous amount of knowledge snippets, even for a single entity within a domain. For this reason, they have also tested the Dense Knowledge Retrieval method, which promises even greater efficiency. This approach utilizes dense representations of sentences for information retrieval tasks, derived from pre-trained transformer models that capture deeper semantic meanings. A proposed Siamese network architecture includes two components: a dialogue context encoder that operates during inference and a knowledge snippets encoder that runs offline. This structure allows for the application of an appropriate ranking function, framing the task as a metric learning problem. RoBERTa [45] models had been applied for the encoders.

Since the embeddings of the knowledge snippets can be pre-computed, only the embedding of the current dialogue context needs to be generated during runtime. If the total number of these snippets is relatively small, i.e. in the thousands as in case of the DSTC9 Track 1, the k nearest neighbor search to find the closest embedding is quite negligible compared to the inference time of the transformer model like in PRR method. Thus, the complexity of inference for this method is given by:

$$\mathcal{O}(1). \tag{4.3}$$

Even when dealing with a very large number of knowledge snippets, there are efficient means of search [46].

Task	**Model**	**R@1**	**Runtime (sec.)**
Selection	Baseline (PRR)	62.0	276.53
	Hierarchical (PRR)	**89.9**	13.79
	DKR	84.4	**0.04**

Table 4.1: Results on DSTC9 Track 1 test data

The results of these experiments were as expected. Table 4.1 partially presents results for the selection sub-task. The hierarchical selection model achieved the best result for R@1[1], which is the highest indication of the final response's relevance to the user request. As shown in the table, this model also drastically reduced inference time compared to the baseline model. However, the best performance was achieved by the DKR method. In conclusion, both methods yielded better results than the baseline method, but each has its trade-offs when compared to the other.

4.1.2 Hierarchical Dense Knowledge Retrieval

The experiments conducted by Thulke et al. [41] offered comprehensive insights into enhancing the accuracy and efficiency of KCAs within the scope of the DSTC9 Track 1 challenge. Given that Hierarchical Selection offers better accuracy and DKR yields faster inference times, Caffaro, in his master's thesis, proposed a new method called Hierarchical Dense Knowledge Retrieval (HDKR) [47], which combines the strengths of both approaches. So, both offline embedding generation

[1] This metric checks if the most relevant item appears as the first recommendation or retrieval

and online information retrieval processes are done in a hierarchical manner by exploiting intrinsic structure of knowledge base K. The whole process can be framed as below:

1. ***Dense Domain Retrieval***: A transformer model is trained to bring closer the vector representations of dialogue context U and corresponding domain $d_U \in D$ to ensure that the dialogue contexts and their relevant domains are proximate in the embedding space. This model serves as a *domain encoder* and offline, it is used to get the embedding vectors of all domains presented in knowledge base K. At inference, the domain encoder computes only the embedding of the new dialogue context, and based on similarity score the most relevant domain is retrieved.

2. ***Dense Entity Retrieval***: In this phase, given the dialogue context U and its domain d_U, a transformer model is trained to bring the vector representations of the dialogue context and its corresponding entity $e_U \in E$. This model functions as an *entity encoder* and offline, it is used to get the embedding vectors of all entities presented in K. During inference, similar to the domain case, the entity encoder computes the embedding of a new dialogue context, then compares it to pre-computed entity embeddings. It identifies the most relevant entity by evaluating their similarity scores.

3. ***Dense Knowledge Retrieval***: This is the final phase, designed to retrieve relevant knowledge snippets in response to a user's request. Given the dialogue context U, its domain d_U and the entity e_U , a transformer model is trained to create embedding space where the vector representations of the dialogue context and its corresponding knowledge snippets $K^+ \in K$ are proximate. This model, the *knowledge encoder*, calculates the embedding vectors of all knowledge snippets contained in K offline. During inference, it calculates the embedding of a new dialogue context to identify the top n relevant knowledge snippets based on similarity scores. The final retrieved knowledge snippets are then used for Response Generation sub-task, this is why, accuracy of Knowledge Selection task is critical.

The results provided in the work [47] proves the effectiveness of the proposed method. As presented in Table 4.2, the R@1 metric improved with the implementation of HDKR compared to the original DKR method proposed by Thulke et al. [41]. Moreover, while DKR provides tremendous improvement on inference time, by ignoring unrelated domains and their respective entities while similarity comparison, HDKR provides additional decrease on inference time.

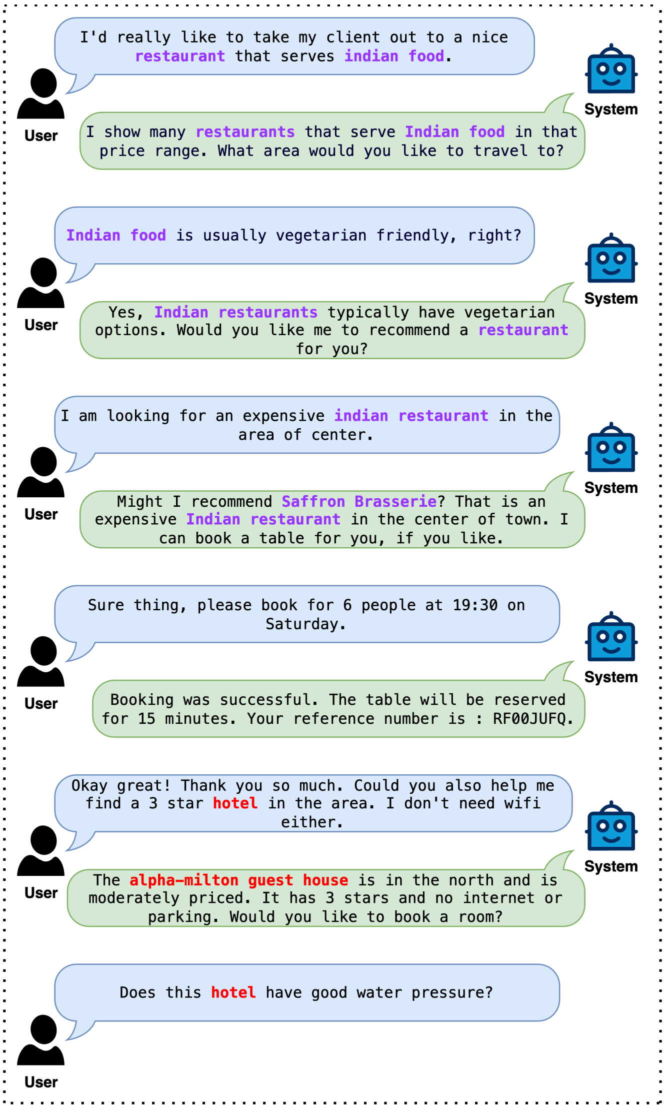
Task	**Model**	**R@1**	**Runtime**
Selection	HDKR	**87.4**	**0.028**
	DKR	83.8	0.040

Table 4.2: Results (originally provided) on DSTC9 Track 1 test data

Like any method, HDKR is not without its trade-offs. One notable drawback is that its hierarchical information retrieval process, due to its cascading architecture, is prone to error propagation. Incorrectly identifying a domain or entity can lead to the retrieval of entirely incorrect knowledge snippets, resulting in inaccurate response generation.

4.1.3 Importance of Granularity in Dialogue Context

The relationship between efficiency and accuracy is discussed till now by mentioning related research done for developing highly effective KCAs. Each proposed method has significantly improved upon the weaknesses of its predecessors. The overall performance of HDKR is remarkable, but its propensity for error propagation deserves attention. The root of the problem lies in the representation of dialogue context vectors. This method involves feeding new dialogue contexts into pre-trained models to generate embedding vectors, which are then compared with pre-computed embeddings of domains, entities, or knowledge snippets. The cascading architecture of the retrieval method make sure that subsequent stages of process yield wrong results and consequently, wrong response generated. The issue often stems from overlooking the granularity in understanding dialogue context. The problem arises because the model receives the dialogue context in its entirety, yet it hasn't been trained to track the dialogue's development through to the final user utterance. Without emphasizing the key parts that define the final target of the dialogue context, it's highly likely that errors will occur. For instance, Fig. 4.1 serves as an excellent example of how the dialogue context can evolve up by switching the focus to the last user request, which seeks subjective knowledge. Initially, the user inquires about the restaurant domain, and as the conversation progresses, they narrow down to a specific choice within that domain. However, the conversation doesn't end there. The user then shifts the focus to the hotel domain, and before the final utterance, the system suggests a specific entity (a particular hotel name). This dialogue context demonstrates shifts in both domain and entity, making it challenging for a model to accurately identify the target without considering the context's granularity in detail.



I'd really like to take my client out to a nice restaurant that serves indian food.

I show many restaurants that serve Indian food in that price range. What area would you like to travel to?

Indian food is usually vegetarian friendly, right?

Yes, Indian restaurants typically have vegetarian options. Would you like me to recommend a restaurant for you?

I am looking for an expensive indian restaurant in the area of center.

Might I recommend Saffron Brasserie? That is an expensive Indian restaurant in the center of town. I can book a table for you, if you like.

Sure thing, please book for 6 people at 19:30 on Saturday.

Booking was successful. The table will be reserved for 15 minutes. Your reference number is : RF00JUFQ.

Okay great! Thank you so much. Could you also help me find a 3 star hotel in the area. I don't need wifi either.

The alpha-milton guest house is in the north and is moderately priced. It has 3 stars and no internet or parking. Would you like to book a room?

Does this hotel have good water pressure?

Figure 4.1: A dialogue context in which the user's initial and final target differ. Parts colored violet pertain to the *restaurant* domain and its associated entity, while red-colored parts relate to the *hotel* domain and its corresponding entity.

Addressing granularity to catch the essence of the text data is a primary objective of Document Summarizaton task which is performed through various methods like Extractive Summarization[2] or Abstractive Summarization[3] etc. In various research papers, new methods have been proposed to enhance text summarization by emphasizing sentences that convey more important insights and ignoring the redundant parts [48, 49]. Inspired by the methods proposed in document summarization research and the sequential nature of dialogue contexts, LSTM models can be applied to effectively capture the evolving interaction between the user and the system, thereby enhancing the vector representation of dialogue context for preventing error propagation in HDKR. Specifically, the LSTM network's last output, or the final hidden state, encapsulates the cumulative representation of the entire dialogue context up to the last time step. Key benefits in terms of dialogue understanding include:

- **Long-term Context Retention**: LSTMs are designed to remember information for long periods which provides better understanding of the dialogue context which spans for several exchanges between the user and the system.
- **Sequential Data Processing**: LSTMs are particularly useful for dialogue systems because they process information sequentially and can understand the order of user inputs, thus providing a strong grasp of the conversation flow.
- **User Intent Recognition**: By analyzing the entire dialogue context with sentence-level granularity, the system can accurately determine user intent, which is crucial for generating relevant responses.
- **Complexity Management**: The LSTM's capability to handle and leverage large volumes of sequential data is especially valuable in complex dialogues, characterized by shifts in topics (such as domain changes) or the need to recall specific details from earlier parts of the conversation.
- **Adaptability to User Corrections**: In a dialogue context, user may correct or clarify previous statements. The LSTM model's memory cells can adjust the conversation's context based on new user utterances, which is essential for responding appropriately to user corrections or clarifications.
- **Handling Varied Sentence Structures**: The LSTM's ability to understand long-term dependencies makes it resilient to variations in sentence construction, enabling it to consistently perform well even as users articulate similar concepts using diverse expressions.

[2] Selects and concatenates the most important portions of the text directly from the original document

[3] Generates new sentences to convey the main ideas of the text, often paraphrasing and condensing the original content

In summary, utilizing the final output of an LSTM model is a powerful method for understanding dialogue context, even when there are ambiguities or shift of topics in conversation.

4.1.4 The Framework of the Method

The DSTC11 Track 5 presents a more challenging task compared to its predecessors, offering greater flexibility in the number of entities and knowledge snippets to be retrieved. This increased flexibility introduces additional complexity to the task, necessitating the adoption of different measures. In this work, the HDKR method is integrated into the context of this challenge due to its efficiency and effectiveness. Figure 4.2 illustrates the architecture of the application. Based on specific needs, various techniques are applied at different stages of the pipeline, and each will be discussed in depth individually. To define the relevance between dialogue context and a domain, entity, or knowledge snippet, cosine similarity is employed alongside the appropriate selection strategy depending on the target. The equation for cosine similarity is as follows:

$$\text{cosine similarity}(\mathbf{A}, \mathbf{B}) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} \tag{4.4}$$

Where **A** and **B** are given vectors:

- $\mathbf{A} \cdot \mathbf{B}$ represents the dot product of vectors **A** and **B**, calculated as $\sum_{i=1}^{n} a_i b_i$ for vectors $\mathbf{A} = (a_1, a_2, \ldots, a_n)$ and $\mathbf{B} = (b_1, b_2, \ldots, b_n)$.
- $\|\mathbf{A}\|$ and $\|\mathbf{B}\|$ denote the Euclidean norms (magnitudes) of the vectors **A** and **B**, respectively, calculated as $\|\mathbf{A}\| = \sqrt{\sum_{i=1}^{n} a_i^2}$ and $\|\mathbf{B}\| = \sqrt{\sum_{i=1}^{n} b_i^2}$.
- The Euclidean norm $\|\mathbf{V}\|$ for a vector **V** computes the straight-line distance from the origin of the space to the point **V**, applying Pythagoras' theorem[4].

The similarity score ranges from -1 to 1, where a score close to 1 indicates high similarity between the texts, a score around 0 suggests no similarity, and a score close to -1 implies that the compared texts have completely opposite meanings. Mathematically, we get score 1 when the vectors are in the same direction, -1 when the vectors are exactly in the opposite direction and 0 for orthogonal vectors[5].

[4] In a right-angled triangle, the square of the length of the hypotenuse is equal to the sum of the squares of the lengths of the other two sides

[5] Vectors that meet at a right angle (90 degrees), indicating no linear correlation or dependence between them

Domain Retrieval

In the dataset for the challenge, only two distinct domains are presented: ***hotel*** and ***restaurant***. Passing single utterances through an LSTM layer could deepen context understanding for improved vector representation, or at least offer a comparison with the transformer encoder method. However, in the context of this challenge, this approach loses its significance as the transformer model alone yields highly effective results for this phase of the pipeline.

A transformer model $\mathcal{E}_{\mathcal{D}}$ is fine-tuned to detect the domain of the conversation. Offline the embedding vectors of distinct domains W_D are obtained with the help of $\mathcal{E}_{\mathcal{D}}$. Online the whole dialogue context U is fed to domain encoder $\mathcal{E}_{\mathcal{D}}$ to get vector representation w_U. Relevant domain d_U is determined by comparing dialogue context vector w_U with domain embeddings W_D in terms of cosine similarity score S. Here ***Top-1***[6] selection strategy is applied.

$$d_U = argmax\Big(S(w_U, W_D)\Big). \tag{4.5}$$

Entity Retrieval

DSTC11 Track 5 challenge requires different approach for entity selection. Unlike previous challenges, for a dialogue context more than one entity can be true label. The measure should be taken respectively. Given the 33 distinct entities in the hotel domain and 110 in the restaurant domain, ambiguities in dialogue contexts can lead to the retrieval of incorrect entity (or entities, as required by the situation). Due to this, processing the dialogue context demands special attention. Two entity encoder models are experimented, one with only transformer model, $\mathcal{E}_{\mathrm{Et}}$, another transformer + LSTM, $\mathcal{E}_{\mathrm{E(t+l)}}$. The embedding vector for the dialogue context is obtained for each case as follows:

- **Transformer + LSTM**: Dialogue context U is not handled as a whole, firstly utterances $\{u_1, u2, \ldots, u_t\}$ are fed to transformer model and then passed through LSTM layer one by one to obtain the final dialogue vector representation w_U. The process is depicted in Figure 4.2, Entity retrieval part.

- **Transformer**: Dialogue context U is fed to transformer model as a whole to get the dialogue embedding vector w_U.

Considering all entity embedding vectors W_E obtained by corresponding encoder model are set aside, for retrieval of relevant entity or entities, based on the predefined domain d_U of a dialogue context, only the subset entity embedding vectors

[6] Choosing the option with the highest score or probability from a set of candidates

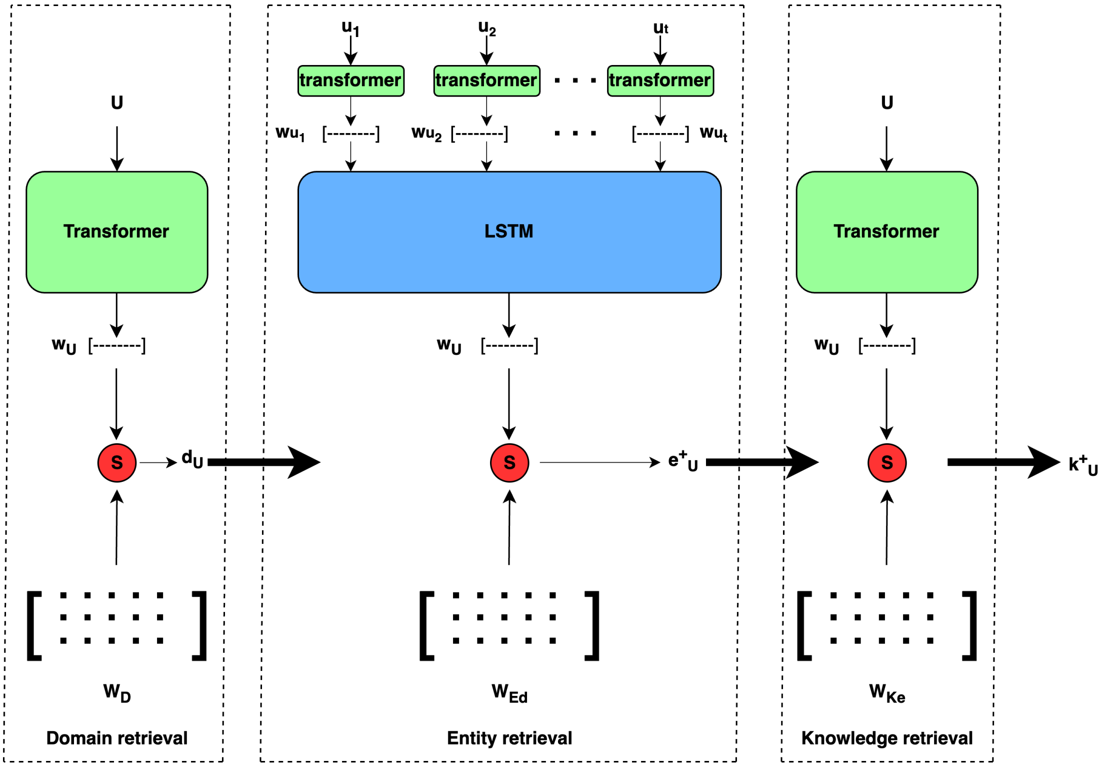


Figure 4.2: Architecture of the hierarchical retrieval process

$W_{E_d} \in W_E$ are used for computation of similarity scores with dialogue embedding w_U. The relevant entities are defined based on pre-defined similarity threshold T (based on evaluation set), retrieving those with scores exceeding this threshold.

$$e_U^+ = \Big(S(w_U, W_{E_d})\Big), \quad where \;\; S \geq T. \tag{4.6}$$

Knowledge Retrieval

This is the final stage of the retrieval process. In this challenge, the number of knowledge snippets to be retrieved is not pre-defined as predecessors (top-1, top-5), thus the threshold is the method applied. Till this part of retrieval, based on the given dialogue context U and the domain d_U and entity or entities e_U^+ concerning the dialogue are defined. Only the knowledge snippets of detected entity or entities are considered for cosine similarity calculation. In this case, since the final knowledge-seeking user utterance u_t is important for knowledge retrieval, the embedding vector of this utterance w_{u_t} is obtained by fine-tuned knowledge encoder model $\mathcal{E}_\mathcal{K}$ and compared to embedding vectors of knowledge snippets $W_{K_e} \in W_K$ belonging to previously retrieved entities. The comparison is done in terms of cosine similarity and the ones with higher similarity scores than the pre-defined similarity threshold T are retrieved.

$$k_{u_t}^{+} = \Big(S(w_{u_t}, W_{K_e})\Big), \quad where \;\; S \geq T. \tag{4.7}$$

Chapter 5

Experiments

In this chapter, the experimental setup and the results are presented. To conduct the experiments presented in this thesis, Google Colab was utilized as the primary development and execution environment. It provides a flexible and accessible platform for running Python code, alongside the benefit of easy access to powerful hardware accelerators [50]. In particular, the experiments leveraged Nvidia's T4 GPUs, selected for their balance of computation power and energy efficiency, which significantly reduced the computational time for complex models and datasets. Additionally, the Colab environment was configured with high RAM capacity to accommodate the extensive data processing and model training requirements of this work.

Also, this work greatly benefited from the use of the Sentence Transformers library. Sentence Transformer models are designed to generate efficient and semantically meaningful sentence embeddings [51]. These models adopt traditional transformer-based architectures like BERT and they are fine-tuned on tasks requiring comprehension of whole sentences, improving upon BERT's methods of creating sentence embeddings. This fine-tuning results in embeddings that capture the semantic essence of sentences more effectively and efficiently, making these models especially useful for NLP tasks such as semantic similarity, clustering, and information retrieval.

5.1 Setup

The training process was conducted in three phases. In each phase, the model was trained to detect a specific target. To facilitate this, the hierarchical structure of the DSTC11 Track 5 knowledge base was exploited. The sub-tasks involved in this process are as follows: Domain Detection, Entity Detection, and Knowledge Detection.

5.1.1 Domain Detection

For the domain detection sub-task, the 'all-MiniLM-L6-v2' model from the Sentence Transformers library was fine-tuned. This model is versatile, designed for a wide range of use cases, and has been trained on a large and diverse dataset comprising over 1 billion training pairs. Its specialty is the balance between speed and average performance. This balance of being fast while maintaining high accuracy makes it the best fit for domain detection among available models. Although it is neither the fastest nor the most accurate model available, the trade-off between speed and accuracy is what sets it apart.

Given that there are only two distinct domains, this sub-task is slightly easier than the others. *TripletLoss* was used for the fine-tuning process. During the data pre-processing phase, triplets are generated from the training data, where the dialogue context serves as the **anchor**, and the true label domain is selected as **positive**. Since there is only one other label for each dialogue context besides the true label, it is automatically assigned as **negative** every time. The dialogue context is reversed for anchors based on the idea that the domain of the conversation is generally mentioned in the final utterances, while transformer models typically focus on the initial parts of the text. At the end, 14768 triplets are generated which is equal to the number of knowledge-seeking turns. The **cosine distance** was utilized as the distance metric between two embeddings, with a *margin* set at 0.25. The model was trained for 4 epochs with a batch size of 16. The *AdamW* optimizer was employed, using a learning rate of $2e - 5$.

5.1.2 Entity Detection

This sub-task is more complex compared to the previous one, as some of the dialogue contexts contain multiple entities as true labels. Consequently, two different methods were experimented with: one using a transformer model and the other combining a transformer with an LSTM. As an encoder, the 'all-mpnet-base-v2' model from the Sentence Transformers library was used. Although this model performs slower than the previous one, it has the highest average performance among all Sentence Transformer models. It is a versatile, all-around model, tuned for a wide range of use-case scenarios and trained on over 1 billion training pairs.

Both models are fine-tuned using triplet loss, with slight variations in anchor generation to accommodate their specific requirements. To keep consistency with the retrieval phase of the work, while generating triplets to train the models for entity detection,the domains of the entities are strictly kept the same for both positive and negative of each triplet pair. For instance, for an anchor and positive pair, the negative is randomly selected among the entities that belong to the same domain of the positive. Since the retrieval process is hierarchical and only the detected domain's entities are considered as potential candidates, this is the best strategy to pursue. Also, if there are more than one true label entity, for instance, in case of two entities are true labels, two anchor, positive pairs are generated by making sure that while selecting negative entity for one pair, another pair's positive is removed from potential negatives list. This approach ensures coherence in training process and aligns with the hierarchical nature of the retrieval mechanism. In the end, 15299 triplets were generated. This is different than the number of knowledge-seeking turns as expected since there are multi-entity dialogue contexts.

Transformer

In this approach, anchors for the model were generated by reversing and concatenating the dialogue context like in the domain detection case. The *cosine distance* was utilized with a *margin* set to 0.30. In total, the model was trained for 6 epochs with batch size of 16. The *AdamW* optimizer was employed with $2e-5$ learning rate.

Transformer + LSTM

In this process, anchors were generated by concatenating each dialogue utterance in their original sequence, using a [SEP] token as a delimiter between them. During the training phase aimed at generating embeddings corresponding to these utterances, the concatenated string is first dissected into individual utterances. Each utterance is then processed by a sentence transformer model to produce an embedding. Subsequently, these embeddings are fed through two LSTM layers to create a final embedding for the dialogue context. This final embedding is then used to perform comparison with positive and negative examples. Two optimizers were employed for adjusting parameters of transformer model and LSTM separately. In both cases, *AdamW* optimizer was the selection, for the transformer with $1e-6$ learning rate,and for LSTM with $1e-5$ learning rate. Early stopping mechanism was applied with patience counter set to 3 to save the best performing model. In total, the model was trained for 25 epochs with batch size of 16. The *cosine distance* was used for triplet loss as usual with a *margin* of 0.30.

5.1.3 Knowledge Detection

This sub-task is particularly interesting. Because unlike previous challenges, here there are **subjective knowledge** snippets additional to FAQs and while retrieval it is possible that the document contains a mix of subjective and FAQ knowledge snippets. Moreover, there is no limit in retrieval process as before, so the number of ground-truth knowledge snippets is flexible.

In the triplet generation process, only the final user utterance from the dialogue context is used to form an anchor. The number of triplet pairs generated for each dialogue context depends on the quantity of available ground-truth knowledge snippets. For each anchor, an equivalent number of pairs are created with positive examples, while negative examples are randomly selected from a pool of negatives. These negatives are chosen to ensure they belong to the same entity as the positive examples, with the additional criterion of excluding any other ground-truth labels from the selection pool for each positive. In the end 56056 triplets were generated.

As in the case of previous sub-task 'all-mpnet-base-v2' model was fine-tuned because of its capability to provide more semantically enriched representation. The *cosine distance* was used for triplet loss with a *margin* of 0.25. In total, the model is trained for 6 epochs with batch size of 16. *AdamW* optimizer was employed with learning rate of $2e-5$.

5.2 Results

This section discusses the results of the experiments. As mentioned in previous chapters, this work concentrates on the Knowledge Selection sub-task. This focus is facilitated by the challenge's internal structure, which allows for concentration on specific parts of the pipeline. Consequently, for the experiments, all dialogue contexts with knowledge-seeking turns are considered by disregarding Turn Detection phase. In Chapter 2, the evaluation metrics are presented mathematically. However, it would be beneficial to describe each metric verbally before delving into the results, to foster a deeper understanding. Respectively, they are:

- **Precision**: measures the proportion of selected items that are relevant out of all the predicted knowledge items. It is an effective way to assess the model's ability to avoid selecting irrelevant knowledge. However, a high precision score does not necessarily indicate that the model is identifying all relevant knowledge.

- **Recall**: measures the proportion of the selected items that are relevant out of all the reference knowledge items. A higher recall indicates that the model is proficient at identifying relevant knowledge. However, since recall does not account for the irrelevant knowledge that is selected alongside the relevant ones, it is possible for precision to be relatively low for the same predictions.

- **F1 Score**: is calculated as the harmonic mean of precision and recall, providing a balanced measure between these two metrics.

- **Exact Match Accuracy**: measures the proportion of instances in which all predicted knowledge items exactly match those of the reference instance. It is considered a strict metric because it disregards the quality of predictions in cases of any slight deviation.

Within the context of the challenge, for each instance, predicted knowledge documents are compared to reference knowledge documents to define true positives[1] (TP), false positives[2] (FP), false negatives[3] (FN), and an exact match metric (scored as either 0 or 1). These metrics are accumulated across all predictions, allowing for the aggregated computation of final scores for Precision, Recall, F1, and Exact Match Accuracy.

Tables 5.1 and 5.2 display the performance of the experiments at different stages of the hierarchical retrieval process for the Transformer-only and the Transformer+LSTM methods, respectively. The metrics have been calculated by considering the results of the preceding retrieval stage. For example, for the Entity phase, the calculation is conducted only for the entities within accurately identified domains, and for the Knowledge Detection phase, it is done by considering only accurately detected entities. This approach ensures that the results reflect how each subsequent stage performs, depending on the accuracy of the previous stage's retrieval. As demonstrated by the tables, the Domain Detection phase results are nearly perfect in both experiments, regardless of whether the test or validation set is used, while the same Transformer model being utilized in each case. The slight differences in scores can be attributed to the subsequent stages. This is because if nothing is retrieved in one of these later stages, the instance is not populated, and the correctly retrieved domain is not accounted for. Given that there were only two domains in this challenge, determining the domain of the dialogue context has been a relatively straightforward task. For the Entity detection phase, in Table 5.1, it can be seen that the results are similarly high as in the Domain Detection case, with performances on the test and validation sets being identical. However, for the Transformer+LSTM method, as indicated in Table 5.2, there is a noticeable drop in performance compared to the Transformer-only model. While precision remains high, recall rates are slightly lower, at 0.78 for the test set and 0.83 for the validation set, indicating a higher incidence of FNs. In the final Knowledge Detection phase, the performance of both methods is comparable across the test and validation sets. This is again because of the same model use. However, there is a notable decrease in precision for the test set compared to recall in both cases. This indicates that the model generated more FPs, despite being effective at minimizing FNs. The likely cause is that the test set contains twice as many unseen instances as the validation set. Overall, the results are quite satisfactory, suggesting that the

[1] Correctly identified items that the model successfully recognizes as relevant or matching the criteria.

[2] Items that the model incorrectly identifies as relevant or matching the criteria when they are not.

[3] Relevant items that the model incorrectly fails to identify or recognize as matching the criteria.

model is fairly good at retrieving the appropriate knowledge documents.

Data Set	Sub-task	Precision (P)	Recall (R)	F1 Score (F1)
Validation	Domain	0.99	0.96	0.98
	Entity	0.97	0.94	0.95
	Knowledge	0.84	0.87	0.85
Test	Domain	0.99	0.97	0.98
	Entity	0.96	0.93	0.94
	Knowledge	0.76	0.86	0.81

Table 5.1: Performance of the Transformer-only models experiment on the Validation and Test sets across different stages of the Hierarchical Retrieval process.

Data Set	Sub-task	Precision (P)	Recall (R)	F1 Score (F1)
Validation	Domain	0.99	0.98	0.99
	Entity	0.94	0.83	0.88
	Knowledge	0.84	0.88	0.86
Test	Domain	0.99	0.98	0.99
	Entity	0.93	0.78	0.85
	Knowledge	0.77	0.86	0.81

Table 5.2: Performance of the Transformer+LSTM model experiment on the Validation and Test sets across different stages of the Hierarchical Retrieval process.

Tables 5.3 and 5.4 present the overall results of the experimented methods for the validation and test sets, respectively, alongside the baseline results provided by the challenge organizers[52]. As expected, the results for all methods are higher on the validation set compared to the test set, a discrepancy attributed to the presence of additional unseen instances in the test set, as previously mentioned. The Transformer-only method outperforms the Transformer+LSTM method on both the validation and test sets. While their precision scores are comparable, the difference in recall is significant, with a gap of 0.08 for the validation set and 0.1 for the test set. This indicates that the Transformer+LSTM method produces more FNs. The performance gap likely demonstrates the Transformer model's superior ability to comprehend the entire dialogue context comprehensively. In the Transformer-only method, the whole dialogue context is processed to generate the final embedding, unlike in the Transformer+LSTM approach, where individual utterances are first encoded separately before obtaining the final embedding through an LSTM. This suggests that feeding the entire dialogue context to the Transformer model not only provides a more thorough understanding of the dialogue but also better captures long-term dependencies. Furthermore, the inclusion of LSTM adds complexity to the task, and it can increase retrieval time. The average retrieval time for the Transformer model experiment is 0.066 seconds for the Test set and 0.064 seconds for the Validation set. Conversely, for the Transformer+LSTM method, the retrieval times are 0.071 seconds for the Test set and 0.062 seconds for the Validation set. Given that the LSTM is applied only during the Entity Detection phase, the increase in retrieval time for the Test set in the Transformer+LSTM method is noteworthy. The performance difference is further evident in both the F1 score and Exact Match accuracy (EM), which are higher for the Transformer-only model.

In the baseline method, the organizers achieve the best result by performing the Knowledge Selection task in two phases. Initially, they apply a fuzzy n-gram matching[4] [53] method to extract relevant entities, followed by employing a Cross-encoder approach [54] with the DeBERTa [55] model for retrieving relevant knowledge snippets. The experimented methods underperform compared to the baseline, likely due to differences in the Entity Detection step. The use of fuzzy n-gram matching, a more straightforward method for detecting entities, operates at the string level, in contrast to the transformer model, which is based on embeddings. It identifies entities through direct comparison, which is particularly effective in dialogue contexts containing multiple entities to be retrieved. Given this, employing a transformer model with a detection threshold performs less effective than directly extracting the appropriate entities through string-level comparison.

Model	Precision (P)	Recall (R)	F1 Score (F1)	Exact Match (EM)
Transformer	0.72	0.77	0.75	0.32
Tr+LSTM	0.70	0.69	0.69	0.30
Baseline	0.80	0.88	0.84	0.40

Table 5.3: Results of the Knowledge Selection task with the experimented methods for the Validation set, including Baseline results from challenge organizers

Overall, the results are quite satisfactory. Following their tradition, the challenge organizers conducted a human evaluation of the best-performing entries, as done in previous challenges (DSTC9 - track 1 and DSTC10 - track 2). For this evaluation, workers were presented with the dialogue context, oracle knowledge snippets, and all responses (including both reference and generated ones). Consistent with previous challenges, the knowledge selection metrics showed the highest correlation with accuracy ratings in the human evaluation (see Fig. 5.1). The F1 score, in

[4] A technique that compares segments of text by breaking them into sequences of n characters (n-grams) and allows for approximate matches, accounting for minor differences or errors.

Model	Precision (P)	Recall (R)	F1 Score (F1)	Exact Match (EM)
Transformer	0.66	0.69	0.67	0.23
Tr+LSTM	0.64	0.59	0.62	0.21
Baseline	0.79	0.79	0.79	0.39

Table 5.4: Results of the Knowledge Selection task with the experimented methods for the Test set, including Baseline results from challenge organizers

particular, demonstrated the highest correlation, indicating that the performance in the experiments conducted is fairly good. The second-highest correlation was with EM accuracy, which is understandable since the final response is grounded in the information provided by retrieved knowledge snippets. However, this metric alone does not fully explain performance due to its sensitivity; any deviation, such as retrieving an additional document or missing one relative to the reference data, results in zero accuracy for the corresponding dialogue context. Therefore, this metric should always be considered alongside other evaluation metrics for a comprehensive assessment.

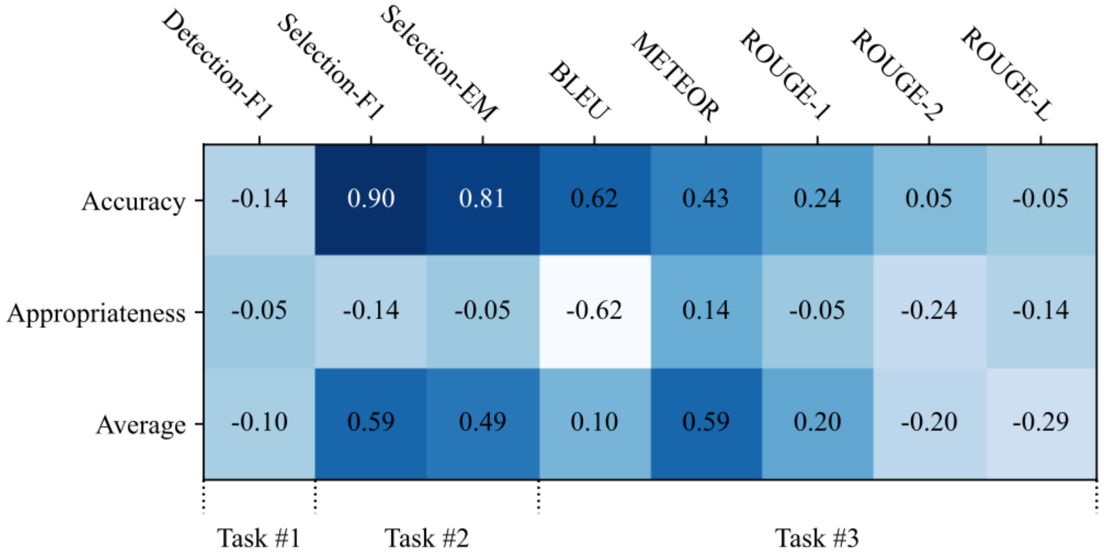


Figure 5.1: Correlations between the objective and human evaluation metrics in Spearman's ρ for DSTC11 Track 5 [52].

Figure 5.2 demonstrates a comparison of the strictest metrics across DSTC9, DSTC10, and DSTC11 based on the entries of all participants. This provides interesting insight because these challenges are continuations of each other, with some added new features and requirements each time. The complexity introduced by the new features results in decreased scores, especially for those metrics that require strict compatibility between the predicted and ground-truth snippets. All of this helps us better understand the complexity of DSTC11. For DSTC9 Track 1 and DSTC10 Track 2, the metric is Recall at 1 (R@1), which measures whether the top-ranked knowledge snippet (the one the system assigns the highest score) is the correct, ground-truth snippet. For DSTC11 Track 5, the metric is Exact Match (EM) accuracy. As previously discussed, it requires that all the predicted knowledge snippets for a particular instance be exactly the same as the ground-truth snippets, making it an even stricter measure than the R@1 metric. The median score and the maximum score observed for a particular metric in the corresponding challenge are given in the figure. The median is provided because it offers a better general overview, especially since the mean score can be misleading for skewed distributions. The maximum score represents the highest entry for the corresponding challenge among all competitors. It can be seen that DSTC11 Track 5 is more challenging, as there is a clear decrease in scores. The gap would be even greater if we compared EM accuracy with R@5, as the latter is less strict than R@1.

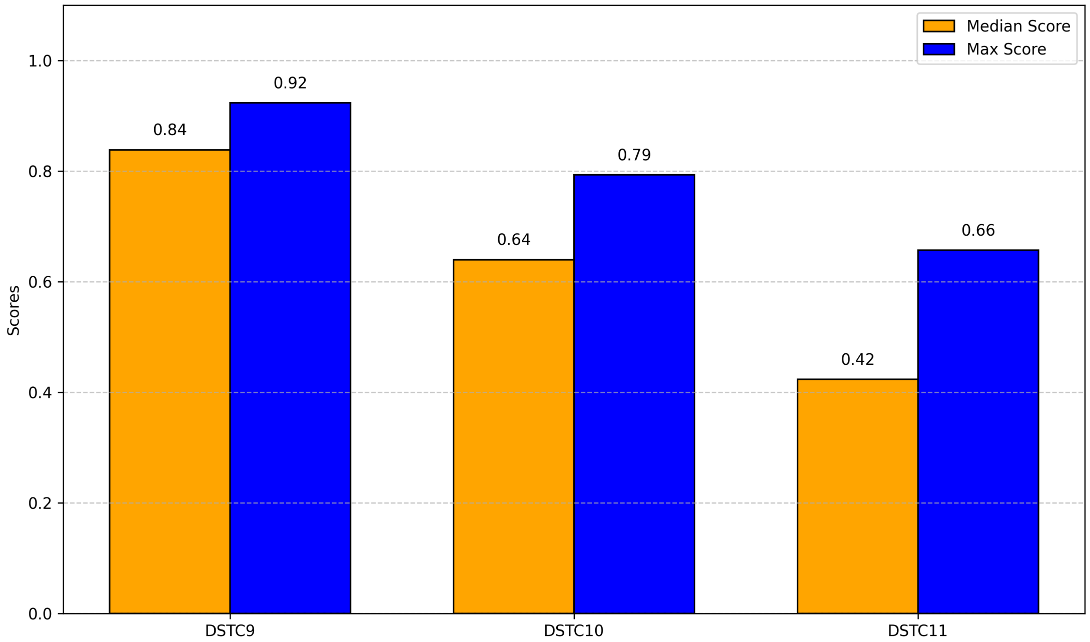


Figure 5.2: Comparison of the strictest metrics for Selection sub-tasks across DSTC9 Track 1, DSTC10 Track 2, and DSTC11 Track 5 challenges: Recall at 1 (R@1) for DSTC9 and DSTC10, and Exact Match Accuracy for DSTC11.

Chapter 6

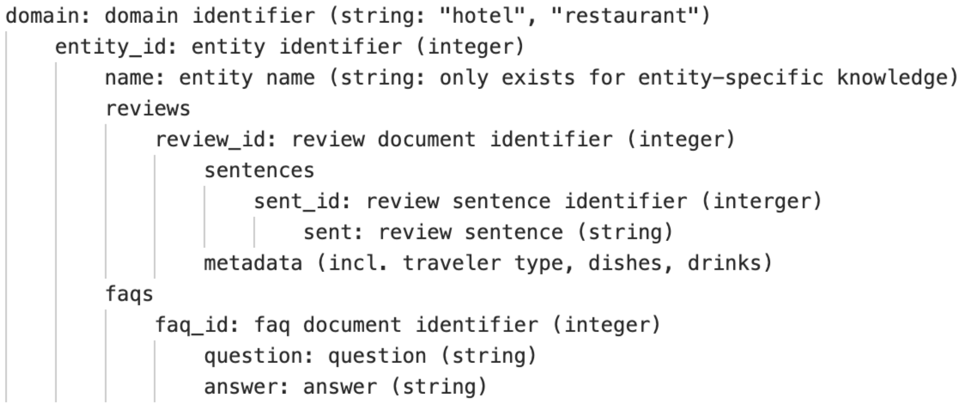
Conclusion

This thesis explored research and experiments aimed at finding efficient and effective solutions for the Knowledge Selection task in DSTC11 Track 5. The objective of this task was to retrieve relevant knowledge snippets from an unstructured knowledge base given the dialogue context. To enhance understanding, the work begins with a comprehensive overview of Conversational Agents, followed by a detailed introduction to the challenge, including a review of previous works. Subsequent sections present related work and offer a detailed explanation of the methodology employed in this study, ensuring a deep understanding of the process. The HDKR method was applied to the task under two different settings: one utilizing only Transformer models for each stage of the retrieval process, and the other incorporating a Transformer+LSTM approach specifically for the Entity Detection phase. The Transformer-only setting demonstrated superior performance compared to the Transformer+LSTM configuration. Overall, the experiments yielded fairly good results, considering the limitations of the methods and the challenge's complexity. A thorough discussion on the strengths and weaknesses of both experimental approaches was provided, comparing their performance to the baseline method in a similar manner.

Appendix A

Appendix

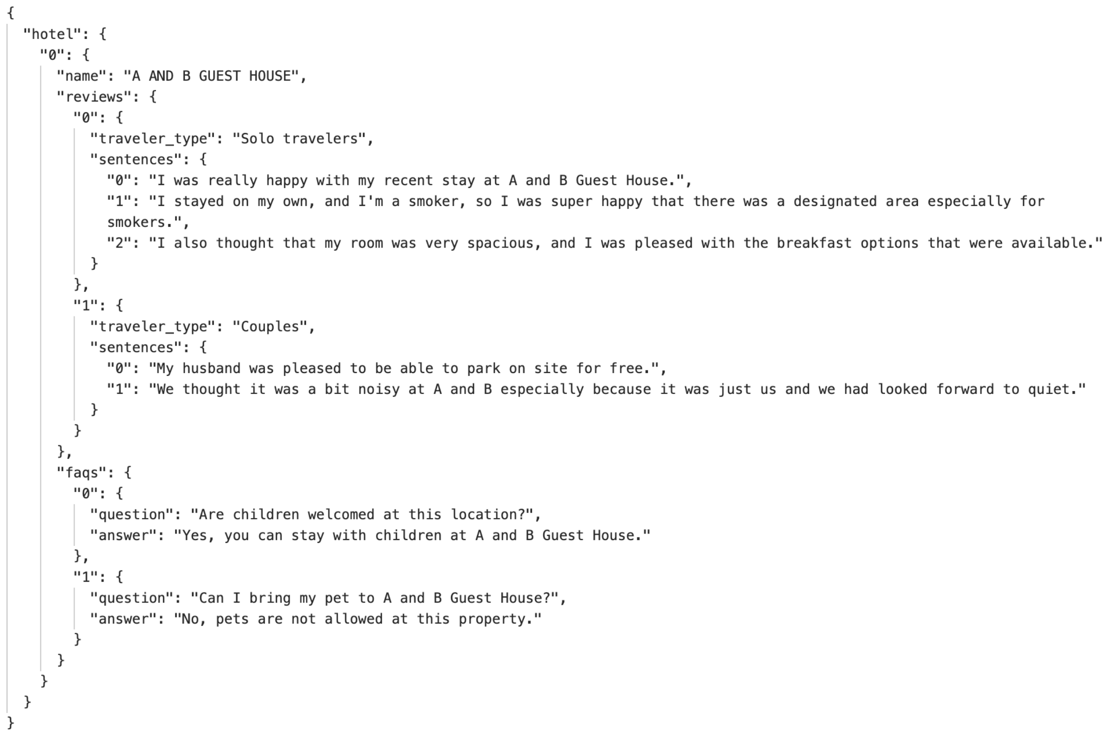
The knowledge base is an unstructured knowledge source, where we refer to select and ground related information in the tasks. Fig. A.1 shows how knowledge base is structured, which consists of domain/entity specific *Review* and *FAQs*.



```
domain: domain identifier (string: "hotel", "restaurant")
    entity_id: entity identifier (integer)
        name: entity name (string: only exists for entity-specific knowledge)
        reviews
            review_id: review document identifier (integer)
                sentences
                    sent_id: review sentence identifier (interger)
                        sent: review sentence (string)
                metadata (incl. traveler type, dishes, drinks)
        faqs
            faq_id: faq document identifier (integer)
                question: question (string)
                answer: answer (string)
```

Figure A.1: The formatting of the knowledge base

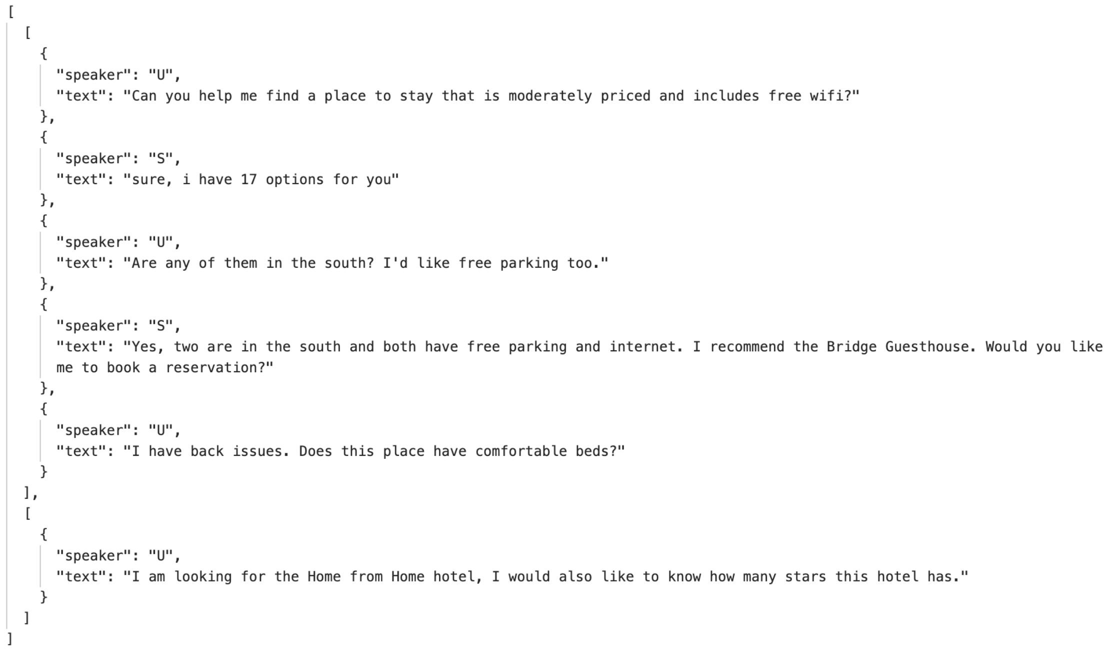
Fig. A.2 shows an abbreviated example section from knowledge base, illustrating selected Reviews and FAQs related to the "A AND B GUEST HOUSE" entity within the hotel domain.



```
{
  "hotel": {
    "0": {
      "name": "A AND B GUEST HOUSE",
      "reviews": {
        "0": {
          "traveler_type": "Solo travelers",
          "sentences": {
            "0": "I was really happy with my recent stay at A and B Guest House.",
            "1": "I stayed on my own, and I'm a smoker, so I was super happy that there was a designated area especially for
            smokers.",
            "2": "I also thought that my room was very spacious, and I was pleased with the breakfast options that were available."
          }
        },
        "1": {
          "traveler_type": "Couples",
          "sentences": {
            "0": "My husband was pleased to be able to park on site for free.",
            "1": "We thought it was a bit noisy at A and B especially because it was just us and we had looked forward to quiet."
          }
        }
      },
      "faqs": {
        "0": {
          "question": "Are children welcomed at this location?",
          "answer": "Yes, you can stay with children at A and B Guest House."
        },
        "1": {
          "question": "Can I bring my pet to A and B Guest House?",
          "answer": "No, pets are not allowed at this property."
        }
      }
    }
  }
}
```

Figure A.2: Abbreviated section from knowledge.json dataset

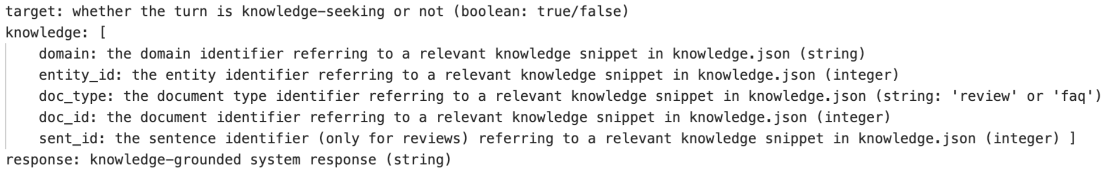
Fig. A.3 shows two example dialogue contexts from original dataset. "U" stands for *user* and "S" stands for *system* turn. Below each of them, *texts* are given which are user requests and system responses in particular. The latest user utterance is either a knowledge-seeking turn or not. First example's last user request is knowledge-seeking turn as it can bee seen from the figure. Second example consists of only one entry from the user and it is not a knowledge-seeking turn.



```
[
  [
    {
      "speaker": "U",
      "text": "Can you help me find a place to stay that is moderately priced and includes free wifi?"
    },
    {
      "speaker": "S",
      "text": "sure, i have 17 options for you"
    },
    {
      "speaker": "U",
      "text": "Are any of them in the south? I'd like free parking too."
    },
    {
      "speaker": "S",
      "text": "Yes, two are in the south and both have free parking and internet. I recommend the Bridge Guesthouse. Would you like
      me to book a reservation?"
    },
    {
      "speaker": "U",
      "text": "I have back issues. Does this place have comfortable beds?"
    }
  ],
  [
    {
      "speaker": "U",
      "text": "I am looking for the Home from Home hotel, I would also like to know how many stars this hotel has."
    }
  ]
]
```

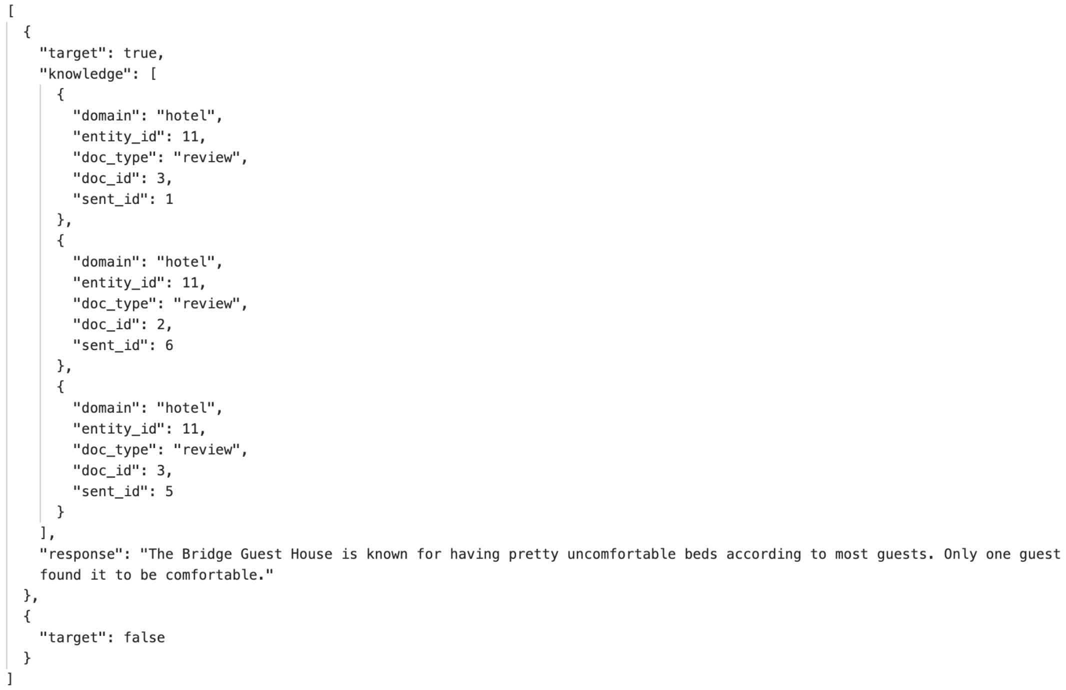
Figure A.3: Abbreviated section from logs.json dataset

Fig. A.4 shows how the labels.json instances are structured and Fig. A.5 shows an example part from it. There are ground-truth label and human response for the final user turn of the first instance in fig. A.3. Because *Knowledge* and *Response* only exist for the instances where the final user request is knowledge-seeking turn.



```
target: whether the turn is knowledge-seeking or not (boolean: true/false)
knowledge: [
    domain: the domain identifier referring to a relevant knowledge snippet in knowledge.json (string)
    entity_id: the entity identifier referring to a relevant knowledge snippet in knowledge.json (integer)
    doc_type: the document type identifier referring to a relevant knowledge snippet in knowledge.json (string: 'review' or 'faq')
    doc_id: the document identifier referring to a relevant knowledge snippet in knowledge.json (integer)
    sent_id: the sentence identifier (only for reviews) referring to a relevant knowledge snippet in knowledge.json (integer) ]
response: knowledge-grounded system response (string)
```

Figure A.4: The formatting of the labels.json



```
[
  {
    "target": true,
    "knowledge": [
      {
        "domain": "hotel",
        "entity_id": 11,
        "doc_type": "review",
        "doc_id": 3,
        "sent_id": 1
      },
      {
        "domain": "hotel",
        "entity_id": 11,
        "doc_type": "review",
        "doc_id": 2,
        "sent_id": 6
      },
      {
        "domain": "hotel",
        "entity_id": 11,
        "doc_type": "review",
        "doc_id": 3,
        "sent_id": 5
      }
    ],
    "response": "The Bridge Guest House is known for having pretty uncomfortable beds according to most guests. Only one guest
    found it to be comfortable."
  },
  {
    "target": false
  }
]
```

Figure A.5: Abbreviated section from labels.json dataset

Bibliography

[1] Johann Hauswald et al. «Designing future warehouse-scale computers for sirius, an end-to-end voice and vision personal assistant». In: *ACM Transactions on Computer Systems (TOCS)* 34.1 (2016), pp. 1–32 (cit. on p. 1).

[2] Petter Bae Brandtzaeg and Asbjørn Følstad. «Chatbots: changing user needs and motivations». In: *interactions* 25.5 (2018), pp. 38–43 (cit. on p. 1).

[3] Jasper Feine, Ulrich Gnewuch, Stefan Morana, and Alexander Maedche. «A taxonomy of social cues for conversational agents». In: *International Journal of Human-Computer Studies* 132 (2019), pp. 138–161 (cit. on p. 1).

[4] Sofia Schöbel, Anuschka Schmitt, Dennis Benner, Mohammed Saqr, Andreas Janson, and Jan Marco Leimeister. «Charting the Evolution and Future of Conversational Agents: A Research Agenda Along Five Waves and New Frontiers». In: *Information Systems Frontiers* (2023), pp. 1–26 (cit. on pp. 1, 3).

[5] Manuel Trinidad, Mercedes Ruiz, and Alejandro Calderon. «A bibliometric analysis of gamification research». In: *IEEE Access* 9 (2021), pp. 46505–46544 (cit. on p. 1).

[6] Joseph Weizenbaum. «ELIZA—a computer program for the study of natural language communication between man and machine». In: *Communications of the ACM* 9.1 (1966), pp. 36–45 (cit. on p. 1).

[7] Douglas R. Hofstadter. *Preface 4 The Ineradicable Eliza Effect and Its Dangers, Epiloguey.* Basic Books, 1996 (cit. on p. 1).

[8] Richard S Wallace. *The anatomy of ALICE In Parsing the Turing Test: Philosophical and Methodological Issues in the Quest for the Thinking Computer (pp. 181–210).* 2009 (cit. on p. 2).

[9] Romal Thoppilan et al. «Lamda: Language models for dialog applications». In: *arXiv preprint arXiv:2201.08239* (2022) (cit. on p. 2).

[10] Mubin Ul Haque, Isuru Dharmadasa, Zarrin Tasnim Sworna, Roshan Namal Rajapakse, and Hussain Ahmad. «"I think this is the most disruptive technology": Exploring Sentiments of ChatGPT Early Adopters using Twitter Data». In: *arXiv preprint arXiv:2212.05856* (2022) (cit. on p. 2).

[11] Chao Zhao et al. «" What do others think?": Task-Oriented Conversational Modeling with Subjective Knowledge». In: *arXiv preprint arXiv:2305.12091* (2023) (cit. on pp. 5, 14–16).

[12] Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. «Wizard of wikipedia: Knowledge-powered conversational agents». In: *arXiv preprint arXiv:1811.01241* (2018) (cit. on p. 5).

[13] Zongsheng Wang, Zhuoran Wang, Yinong Long, Jianan Wang, Zhen Xu, and Baoxun Wang. «Enhancing generative conversational service agents with dialog history and external knowledge». In: *Computer Speech & Language* 54 (2019), pp. 71–85 (cit. on p. 6).

[14] Liliana Laranjo et al. «Conversational agents in healthcare: a systematic review». In: *Journal of the American Medical Informatics Association* 25.9 (2018), pp. 1248–1258 (cit. on p. 6).

[15] Michelle ME Van Pinxteren, Mark Pluymaekers, and Jos GAM Lemmink. «Human-like communication in conversational agents: a literature review and research agenda». In: *Journal of Service Management* 31.2 (2020), pp. 203–225 (cit. on p. 6).

[16] Paul Tarau and Elizabeth Figa. «Knowledge-based conversational agents and virtual storytelling». In: *Proceedings of the 2004 ACM symposium on Applied computing*. 2004, pp. 39–44 (cit. on p. 6).

[17] Karen Myers et al. «An intelligent personal assistant for task and time management». In: *AI Magazine* 28.2 (2007), pp. 47–47 (cit. on p. 7).

[18] Jianyang Deng and Yijia Lin. «The benefits and challenges of ChatGPT: An overview». In: *Frontiers in Computing and Intelligent Systems* 2.2 (2022), pp. 81–83 (cit. on p. 8).

[19] Kleopatra Mageira, Dimitra Pittou, Andreas Papasalouros, Konstantinos Kotis, Paraskevi Zangogianni, and Athanasios Daradoumis. «Educational AI chatbots for content and language integrated learning». In: *Applied Sciences* 12.7 (2022), p. 3239 (cit. on p. 9).

[20] Jason D Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. «Dialog state tracking challenge handbook». In: *Silicon Valley, Microsoft Research* (2012) (cit. on p. 10).

[21] Matthew Henderson, Blaise Thomson, and Jason D Williams. «The third dialog state tracking challenge». In: *2014 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2014, pp. 324–329 (cit. on p. 10).

[22] Seokhwan Kim, Luis Fernando D'Haro, Rafael E Banchs, Jason D Williams, and Matthew Henderson. «The fourth dialog state tracking challenge». In: *Dialogues with Social Robots: Enablements, Analyses, and Evaluation* (2017), pp. 435–449 (cit. on p. 10).

[23] Seokhwan Kim, Luis Fernando D'Haro, Rafael E Banchs, Jason D Williams, Matthew Henderson, and Koichiro Yoshino. «The fifth dialog state tracking challenge». In: *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE. 2016, pp. 511–517 (cit. on p. 10).

[24] Chiori Hori et al. «Overview of the sixth dialog system technology challenge: DSTC6». In: *Computer Speech & Language* 55 (2019), pp. 1–25 (cit. on p. 11).

[25] Luis Fernando D'Haro, Koichiro Yoshino, Chiori Hori, Tim K Marks, Lazaros Polymenakos, Jonathan K Kummerfeld, Michel Galley, and Xiang Gao. «Overview of the seventh dialog system technology challenge: DSTC7». In: *Computer Speech & Language* 62 (2020), p. 101068 (cit. on p. 11).

[26] Seokhwan Kim et al. «Overview of the eighth dialog system technology challenge: DSTC8». In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 29 (2021), pp. 2529–2540 (cit. on p. 11).

[27] Chulaka Gunasekara et al. «Overview of the ninth dialog system technology challenge: Dstc9». In: *arXiv preprint arXiv:2011.06486* (2020) (cit. on p. 11).

[28] Koichiro Yoshino et al. «Overview of the Tenth Dialog System Technology Challenge: DSTC10». In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* (2023) (cit. on p. 11).

[29] Seokhwan Kim, Mihail Eric, Karthik Gopalakrishnan, Behnam Hedayatnia, Yang Liu, and Dilek Hakkani-Tur. «Beyond domain APIs: Task-oriented conversational modeling with unstructured knowledge access». In: *arXiv preprint arXiv:2006.03533* (2020) (cit. on pp. 12, 34).

[30] Seokhwan Kim, Mihail Eric, Behnam Hedayatnia, Karthik Gopalakrishnan, Yang Liu, Chao-Wei Huang, and Dilek Hakkani-Tur. *Beyond Domain APIs: Task-oriented Conversational Modeling with Unstructured Knowledge Access Track in DSTC9*. 2021. arXiv: `2101.09276 [cs.CL]` (cit. on p. 12).

[31] Seokhwan Kim, Yang Liu, Di Jin, Alexandros Papangelis, Karthik Gopalakrishnan, Behnam Hedayatnia, and Dilek Hakkani-Tur. *"How robust r u?": Evaluating Task-Oriented Dialogue Systems on Spoken Conversations*. 2021. arXiv: `2109.13489 [cs.CL]` (cit. on p. 13).

[32] Mihail Eric and Christopher D Manning. «Key-value retrieval networks for task-oriented dialogue». In: *arXiv preprint arXiv:1705.05414* (2017) (cit. on p. 13).

[33] Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. «Multiwoz–a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling». In: *arXiv preprint arXiv:1810.00278* (2018) (cit. on p. 13).

[34] Mihail Eric et al. «MultiWOZ 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines». In: *arXiv preprint arXiv:1907.01669* (2019) (cit. on p. 17).

[35] Robin M Schmidt. «Recurrent neural networks (rnns): A gentle introduction and overview». In: *arXiv preprint arXiv:1912.05911* (2019) (cit. on p. 24).

[36] Sepp Hochreiter and Jürgen Schmidhuber. «Long short-term memory». In: *Neural computation* 9.8 (1997), pp. 1735–1780 (cit. on p. 25).

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. «Attention is all you need». In: *Advances in neural information processing systems* 30 (2017) (cit. on pp. 27, 28).

[38] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. «Bert: Pre-training of deep bidirectional transformers for language understanding». In: *arXiv preprint arXiv:1810.04805* (2018) (cit. on p. 27).

[39] Gokul Yenduri, Gautam Srivastava, Praveen Kumar Reddy Maddikunta, Rutvij H Jhaveri, Weizheng Wang, Athanasios V Vasilakos, Thippa Reddy Gadekallu, et al. «Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions». In: *arXiv preprint arXiv:2305.10435* (2023) (cit. on p. 27).

[40] Rodrigo Nogueira and Kyunghyun Cho. «Passage Re-ranking with BERT». In: *arXiv preprint arXiv:1901.04085* (2019) (cit. on p. 29).

[41] David Thulke, Nico Daheim, Christian Dugast, and Hermann Ney. «Efficient retrieval augmented generation from unstructured knowledge for task-oriented dialog». In: *arXiv preprint arXiv:2102.04643* (2021) (cit. on pp. 30, 33, 35, 36).

[42] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. «Dense passage retrieval for open-domain question answering». In: *arXiv preprint arXiv:2004.04906* (2020) (cit. on p. 30).

[43] Krysta M Svore and Christopher JC Burges. «A machine learning approach for improved BM25 retrieval». In: *Proceedings of the 18th ACM conference on Information and knowledge management*. 2009, pp. 1811–1814 (cit. on p. 30).

[44] Alexander Hermans, Lucas Beyer, and Bastian Leibe. «In defense of the triplet loss for person re-identification». In: *arXiv preprint arXiv:1703.07737* (2017) (cit. on p. 31).

[45] Yinhan Liu et al. «Roberta: A robustly optimized bert pretraining approach». In: *arXiv preprint arXiv:1907.11692* (2019) (cit. on p. 35).

[46] Jeff Johnson, Matthijs Douze, and Hervé Jégou. «Billion-scale similarity search with GPUs». In: *IEEE Transactions on Big Data* 7.3 (2019), pp. 535–547 (cit. on p. 35).

[47] Fabio Caffaro. «Hierarchical Dense Knowledge Retrieval for Knowledge-enhanced Conversational Agents». PhD thesis. Politecnico di Torino, 2022 (cit. on pp. 35, 36).

[48] Yu Zhao, Leilei Wang, Cui Wang, Huaming Du, Shaopeng Wei, Huali Feng, Zongjian Yu, and Qing Li. «Multi-granularity heterogeneous graph attention networks for extractive document summarization». In: *Neural Networks* 155 (2022), pp. 340–347 (cit. on p. 39).

[49] Ruipeng Jia, Yanan Cao, Hengzhu Tang, Fang Fang, Cong Cao, and Shi Wang. «Neural extractive summarization with hierarchical attentive heterogeneous graph network». In: *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)*. 2020, pp. 3622–3631 (cit. on p. 39).

[50] Ekaba Bisong and Ekaba Bisong. «Google colaboratory». In: *Building machine learning and deep learning models on google cloud platform: a comprehensive guide for beginners* (2019), pp. 59–64 (cit. on p. 44).

[51] Nils Reimers and Iryna Gurevych. «Sentence-bert: Sentence embeddings using siamese bert-networks». In: *arXiv preprint arXiv:1908.10084* (2019) (cit. on p. 44).

[52] Seokhwan Kim, Spandana Gella, Chao Zhao, Di Jin, Alexandros Papangelis, Behnam Hedayatnia, Yang Liu, and Dilek Z Hakkani-Tür. «Task-Oriented Conversational Modeling with Subjective Knowledge Track in DSTC11». In: *Proceedings of The Eleventh Dialog System Technology Challenge*. 2023, pp. 274–281 (cit. on pp. 49, 51).

[53] Di Jin, Seokhwan Kim, and Dilek Hakkani-Tur. «Can I be of further assistance? using unstructured knowledge access to improve task-oriented conversational modeling». In: *arXiv preprint arXiv:2106.09174* (2021) (cit. on p. 50).

[54] Thomas Wolf, Victor Sanh, Julien Chaumond, and Clement Delangue. «Transfertransfo: A transfer learning approach for neural network based conversational agents». In: *arXiv preprint arXiv:1901.08149* (2019) (cit. on p. 50).

[55] Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. «Deberta: Decoding-enhanced bert with disentangled attention». In: *arXiv preprint arXiv:2006.03654* (2020) (cit. on p. 50).