Politecnico di Torino

POLYTHECNIC UNIVERSITY OF TURIN

Master's Degree in Data Science and Engineering

July 2022

Hierarchical Dense Knowledge Retrieval for Knowledge-enhanced Conversational Agents

Supervisor:
Prof. Giuseppe RIZZO

Candidate:
Fabio CAFFARO

Summary

In this work will be presented a solution developed to improve the efficiency of the knowledge retrieval systems adopted to extract relevant unstructured documents from a knowledge base given an input query. The experiment relies on the setup delineated in the first track of the ninth Dialog System Technology Challenge (DSTC 9). The goal of this task is to design a frictionless knowledge-enhanced conversational agent able to deal with out-of-scope requests that cannot be addressed simply with the call of an API service but that require access to external knowledge. The knowledge base consists of unstructured textual documents collected from the Frequently Asked Question (FAQ) pages of several entities belonging to five domains. From an analysis of the results of the DSTC9, the knowledge retrieval step resulted as a crucial phase in the knowledge-enhancing process, showing the highest correlation with the human judgment. Indeed, the model that performed the best in this sub-task resulted also as the best model in the final human evaluation phase. However, the best performing models in the retrieval step are based on the Passage Re-Ranking strategy. This strategy requires a point-wise evaluation of all the knowledge documents in the knowledge base, causing the time complexity of the system to scale linearly with the dimension of the knowledge base. For this reason, this approach becomes soon unapplicable to real-case scenarios, where the dimension of the knowledge base can grow really fast. The method developed in this work is based on a Hierarchical Dense Knowledge Retrieval system that exploits the hierarchical structure of the documents present in the knowledge base in order to perform a computationally efficient knowledge retrieval.

Acknowledgements

I would like to express my gratitude to my supervisor, Prof. Giuseppe Rizzo, who guided me throughout this project, and in general to all my colleagues at LINKS Foundation that helped me during the thesis development with valuable advice.

I would also like to thank my friends, my family and in particular my sister, who supported me during my university journey.

Table of Contents

List of Tables vii

List of Figures viii

Acronyms xi

1 Introduction 1

2 Task 5
- 2.1 Knowledge-enhanced Conversational Agents . . . 6
  - 2.1.1 Characteristics of KCAs . . . 7
  - 2.1.2 Architecture . . . 9
- 2.2 The Dialog System Technology Challenge (DSTC) . . . 12
  - 2.2.1 DSTC9 - track 1 . . . 12
- 2.3 The need for efficient knowledge-retrieval systems . . . 19
  - 2.3.1 The importance of the knowledge selection . . . 19
  - 2.3.2 The concerns about the efficiency of the retrieval systems . . . 21

3 Related Work 23
- 3.1 Sentence Embedding . . . 23
  - 3.1.1 Symbol based models . . . 24
  - 3.1.2 Recurrent Neural models . . . 25
  - 3.1.3 Deep Neural models . . . 26
- 3.2 Retrieval methods for unstructured knowledge documents . . . 27
  - 3.2.1 Relationship-based methods . . . 28
  - 3.2.2 Representation-based methods . . . 31
- 3.3 Efficient Knowledge Selection . . . 34
  - 3.3.1 Filtering the documents by keywords . . . 34
  - 3.3.2 Knowledge Distillation . . . 34

4 Method 36
4.1 Finding an efficient retrieval system . . . 37
4.1.1 Advantages of Dense Knowledge Retrieval . . . 37
4.1.2 Problems with DKR for the DSTC9 . . . 39
4.1.3 Exploiting the hierarchical information . . . 41
4.2 Hierarchical Dense Knowledge Retrieval . . . 43
4.2.1 Time complexity . . . 46
4.3 Extension . . . 46
4.3.1 Synthetic data augmentation . . . 47
4.3.2 Enriched domain representation . . . 48
4.3.3 Top-k entity filtering . . . 50

5 Experimental Setup 52
5.1 Dense Domain Retrieval . . . 52
5.2 Dense Entity Retrieval . . . 54
5.3 Dense Knowledge Retrieval . . . 56

6 Results 57
6.1 Dense Domain Retrieval . . . 57
6.2 Dense Entity Retrieval . . . 59
6.3 Dense Knowledge Retrieval . . . 60
6.4 Hierarchical Dense Knowledge Retrieval . . . 61

7 Conclusion 64

A Appendix 65

Bibliography 67

List of Tables

2.1 Dimensionality of the dataset for the first track of the DSTC9. . . . 17
2.2 Objective measures adopted for the three tasks of the DSTC9 track 1. 18
2.3 Final results on the test set for the three sub-tasks of the DSTC9 - track 1. 20

6.1 Results for Dense Domain Retrieval in both val and test sets. μ_t indicates the average inference time and σ_t the standard deviation. . 58
6.2 Results for Dense Entity Retrieval in both val and test sets. The accuracy of the first three entities are considered cumulatively. Acc3 for example measure if the correct name is in the first three entity names retrieved. δ_{domain} indicates the loss in accuracy with respect to the DDR, in order to measure the error propagation. 59
6.3 Results for the Dense Knowledge Retrieval in both val and test sets. δ_{domain} and δ_{entity} indicates respectively the degradation with respect to the DDR and DER. 60
6.4 Final comparison of the HDKR with the Selection baseline of the DSTC9 - track 1. *All* the knowledge-seeking turns have been considered. 62
6.5 Comparison between the DKR method proposed by the team 18 and the HDKR method. Only the knowledge-seeking turns detected by the team 18 have been considered. The number of parameters for the DKR model takes in consideration also the siamese network architecture (two separate encoders). However, the knowledge encoder is not considered in the final *delta* scoring since the knowledge embedding can be computed a-priori. 62
6.6 Comparison between the *Winner* team of the DSTC9 - track 1 and the HDKR method. Only the *team 19* knowledge seeking turns have been considered. 63

List of Figures

2.1 Modular architecture of a KCA. 9

2.2 Example of an unstructured knowledge snippet composed by a simple text document and the corresponding structured knowledge composed of (subject, relation, object) triplets. 11

2.3 Modular architecture for the Interface proposed in the first track of the DSTC9. 15

2.4 Example of two sub-dialogues sampled from the same dialogue context with the corresponding labels. The first request does not need access to the knowledge base, while the second request is a knowledge seeking turn regarding the domain *'restaurant'* and the entity *'Pizza Hut Fen Ditton'*. 16

2.5 Example of five FAQ documents present in the knowledge base. For each document is reported the domain (e.g., *'hotel'*), the entity (e.g., *'A and B Guest House'*) and the actual document composed by question and answer. Further information about the knowledge base can be found in Fig. A.1 on the Appendix. 18

2.6 Correlations between the objective and human evaluation metrics in Spearman's ρ. Task # 1 correspond to the Detection, Task # 2 to the Selection and Task # 3 to the Generation. Image from [10]. . . 21

3.1 Example of a relationship-based retrieval through Passage Re-Ranking for the knowledge Selection in the DSTC9 - track 1. 28

3.2 Three Language Model objectives adopted to pre-train UniLM and Plato-2. Image from Dong et al. [42] 29

3.3 Architecture of Dense Knowledge Retrieval with a siamese network. The embedding of the documents in the knowledge base can be computed *offline* a-priori. At inference time (*online*) only the computation of the embedding vector for the dialogue context U_t is necessary. 32

3.4 Example of the application of the triplet loss. After the training the representation space brings *closer* the positive vector to the anchor. On top is considered the euclidean distance while on the bottom the cosine distance. 33
3.5 Example of the classic knowledge distillation framework in which the student model is trained to mimic the output of the teacher model. 35

4.1 Dense Knowledge Retrieval with a single encoder model. A single encoder is used to compute the embedding of the documents in the knowledge base and of the dialogue context independently. 38
4.2 Example of similar FAQ documents belonging to different domains. The documents are taken from the knowledge base of the DSTC9 - track 1. 39
4.3 Example of three couples of similar FAQ documents sampled from two entities in the same domain. The documents are taken from the knowledge base of the DSTC9 - track 1, ... 40
4.4 Two-dimensional representation of the embedding vectors of the FAQ knowledge documents in the domain hotel obtained through the t-SNE algorithm. From this image it is possible to see clearly how the embedding vector space does not consider the entity. This results in clusters of similar FAQ documents from different entities. 41
4.5 Architecture of the Hierarchical Dense Knowledge Retrieval. The final document is retrieved after three sequential Dense Retrieval tasks in which are respectively retrieved the domain d_U, the entity e_U and the FAQ knowledge snippet k_U given a dialogue context U. . 44
4.6 Example of a synthetic dialogue generated starting from a FAQ knowledge document. 48
50figure.caption.37
4.8 Plot of the average probability distribution of the first 5 Entity names obtained with the DER. 51

A.1 Example of the first three FAQ documents for the entity "A AND B GUEST HOUSE" from the domain hotel. 65
A.2 Example of two sub-dialogues from the same dialogue context. The first user request is not a knowledge-seeking turn, while the second needs access to the knowledge base. 66

Acronyms

AI
artificial intelligence

API
Application Programming Interface

BERT
Bidirectional Encoder Representations from Transformers

BOW
Bag-of-Word

CA
Conversational Agent

DF
Document Frequency

DKR
Dense Knowledge Retrieval

DPR
Dense Passage Retrieval

DST
Dialog State Tracking

DSTC
Dialog System Technology Challenge

FAQs

Frequently Asked Questions

GRU

Gated Recurrent Unit

HDKR

Hierarchical Dense Knowledge Retrieval

IDF

Inverse Document Frequency

IR

Information Retrieval

KB

Knowledge Base

KCA

Knowledge-enhanced Conversational Agent

LM

Language Model

LSTM

Long Short Term Memory

MLM

Masked Language Model

NLG

Natural Language Generation

NLP

Natural Language Processing

NLU

Natural Language Understanding

NSP

Next Sentence Prediction

PLM

Pre-trained Language Model

PRR

Passage Re-Ranking

Q&A

Question-Answering

RNN

Recurrent Neural Network

RoBERTa

Robustly Optimized BERT

TF

Term Frequency

TF-IDF

Term Frequency - Inverse Document Frequency

UniLM

Unified Language Model

Chapter 1

Introduction

For a long time humanity has fantasized about the possibility of creating intelligent artificial agents able to engage and interact with humans in order to help them, assist them in performing some task or simply to entertain them. The origin of this dream can be traced back much further than one can imagine. Mayor [1] reports how stories of creatures that were *"made, not born"* started to appear in Ancient Greece around two thousands years ago. Greeks were also the first to develop the idea of *automata*, self-operating machines designed to automatically follow a sequence of operations. A very famous example coming from the Greek mythology was *Talos*, a giant automaton made of bronze who patrolled the island of Crete circling it's shores three times each day in order to protect princess Europa from pirates and invaders.

In order to be able to properly interact with humans, a fundamental aspect of an artificial agent is the *capability to communicate*. Beyond theorization, the first real endeavors in the creation of artificial systems capable of communicating dates back to the second half of the XVIII century. Describing the first attempts in the field of speech synthesis, Pieraccini [2] reports the work of Wolfgang von Kempelen, a Hungarian inventor working at the court of empress Maria Theresa of Austria, that was able to build the first 'speaking machine'.

Since then, huge strides have been made. With the explosion of the Digital Age, the research moved towards the creation of software-based agents. These systems are generally referred to as Conversational Agents (CAs). In 1966, Dr. Joseph Weizenbaum from Massachusetts Institute of Technology (MIT) presented ELIZA [3], the first real conversational agent able to interact with humans through a textual chat. Despite its simplicity, it had enormous success given its alleged capability to perform intelligent conversations. This led to the birth of the so-called *ELIZA effect* [4]. The incredible impact that this model had was probably due to the fact that it foreshadowed what the future of this technology could have brought. Indeed, it is not a case that in the following years the science fiction production

was filled with iconic artificial companions.

In more recent times, research efforts have moved towards Conversational Agents able to converse with humans with the purpose of *assisting* them in achieving a precise goal (e.g., booking a ticket for a flight). These models are commonly referred to as *assistants*. In 2011, 45 years after the launch of ELIZA, Apple's Siri was presented to the world as the first "intelligent assistant that helps you get things done just by asking".[1] Despite the rapid adoption of this technology, that led to the birth of many alternatives such as Google Assistant, Alexa and Cortana; these "intelligent" assistants are still very far from what we can really feel and think of as intelligent. Generally, they are limited in scope being able to operate only on predefined tracks, and they show just a superficial intelligent behavior that can be unmasked after few interactions with them. Among other things, these models still lack generalization capabilities, commonsense knowledge and context perception from the external environment and in some cases even from the conversation itself. These shortcomings are particularly evident in the case of assistants, where even simple requests out of the pre-established tracks can cause problems to the assistant.

For instance, nowadays assistants are able to help a user booking a table in a restaurant. However, they typically fall short for simple follow-up requests such as *"Do they serve vegetarian food at this restaurant?"*. In this way their primary function of assistants, for which they were originally designed, is ceasing to exist, and they become more as a sort of vocal switch to activate simple services. These defects have repercussions on the user engagements, as shown by recent internal analysis.[2] The rapid lack of interest that plagues modern Conversational Agents can be explained as disillusionment due to a new ELIZA effect. Progresses made in the field of Artificial Intelligence in general and Natural Language Processing (NLP) in particular, have brought models like OpenAI's GPT-3 that are now capable of producing impressive human-like text.[3] The outstanding results obtained had a major impact in society, so much so that in 2020 GPT-3 was elected AI "person" of the year by Forbes.[4] All of this led to an illusion about the real capabilities of these models, which in reality can be defined more as *stochastic parrots* [5] due to their capabilities to only mimic humans without performing any complex reasoning. In fact, from a more detailed analysis it can be seen how these models still tend to lose

[1] https://www.apple.com/nz/newsroom/2011/10/04Apple-Launches-iPhone-4S-iOS-5-iCloud/

[2] https://www.bloomberg.com/news/articles/2021-12-22/amazon-s-voice-controlled-smart-speaker-alexa-can-t-hold-customer-interest-docs

[3] https://www.theguardian.com/commentisfree/2020/sep/08/robot-wrote-this-article-gpt-3

[4] https://www.forbes.com/sites/kenrickcai/2021/01/04/forbes-ai-awards-2020-meet-gpt-3-the-computer-program-that-can-write-an-op-ed/?sh=18aa83d693a7

coherence and contradict themselves [6] particularly when they are interrogated on questions that require some sort of reasoning or on socially important subjects such as morality and law, where they still have near-random accuracy [7].

The main problematic relies on the fact that a conversational interaction is a *semantic activity* much more complex than 'simple' language generation. Indeed, a conversation is a process to create some *meaning* [8], where at turns two agents negotiate the meaning through the sharing of commonsense, personal, and social knowledge. Therefore, additional knowledge is essential to have a deeper semantic understanding of the conversation, that cannot be inferred only by the conversation itself.

For this reason, an important stream of research on Conversational Agents is currently focusing on developing new and efficient methodologies apt to enhance the agents capabilities and the quality of their responses with external additional knowledge. If from one hand knowledge-enhancing techniques have been already studied in the past in the context of open-domain conversations, there are still many problems to address in the domain of task-oriented agents such as the assistants. In particular, one of the most compelling regards the methodologies adopted to select and retrieve relevant knowledge snippets from unstructured knowledge bases. In fact, the task-oriented environment sets up many additional difficulties with respect to its open-domain counterpart. Indeed, distinctly from open-domain, task-oriented agents must be able to deliver completely accurate information and not generally relevant one, in order one to meet the user's needs. For instance, if the user asks for the check-in hours of a specific hotel, a task-oriented system must be able to retrieve the correct document reporting the check-in information from the specific hotel named by the user and not another one.

If from one hand the task-oriented setting imposes strict obligations on the accuracy of the agents, on the other hand the constraints in terms of latency of the response for the assistants impose to the model to be really efficient in the retrieval of the information. However, the state-of-the-art approach in this task scales linearly with the size of the knowledge base, quickly becoming intractable in real case scenarios. For this reason, it is important to investigate solutions for the knowledge retrieval problem in a task-oriented setting that consider also efficiency parameters such as the computational requirements and the latency time for the final retrieval.

Starting from the testbed proposed for the first track of the ninth Dialog System Technology Challenge (DSTC9) [9], this work presents the **Hierarchical Dense Knowledge Retrieval** system. This system is based on a hierarchical reasoning on the input user query aimed at retrieving in dense vector space, unstructured knowledge documents for a knowledge-enhanced conversational agent that operates in a task-oriented setting. This system addresses the problem of efficiency with the intent of proposing a solution applicable in real-case scenarios. Therefore, are

considered as parameter for the evaluation both the time taken to retrieve the documents and the dimension of the models adopted. Aiming at reducing both the two aforementioned efficiency parameters, the goal of the system is to maintain high level of accuracy in the retrieval of the documents with comparison to the current state of the art.

The remaining of the work is structured as follows:

- Chapter 2 provides a description of the task addressed in this work, clarifying the object of study and the related nomenclature (Section 2.1) and detailing the testbed proposed by the first track of the DSTC9 (Section 2.2);
- Chapter 3 presents an exhaustive explanation of the technologies adopted to compute similarity scores between textual information (Section 3.1) and to perform the actual retrieval of the documents (Section 3.2) considering also some of the possible solutions to tackle the efficiency problem (Section 3.3);
- Chapter 4 describes in depth the method adopted to implement the Hierarchical Dense Knowledge Retrieval system starting explaining the path followed to develop the system (Section 4.1) and examining the structure of the system in details (Section 4.2), analyzing the three subsystems of which it is composed that are: Dense Domain Retrieval (DDR), Dense Entity Retrieval (DER) and Dense Knowledge Retrieval (DKR);
- Chapter 5 provides the details of the training for the three subsystems of DDR (Section 5.1), DER (Section 5.2) and DKR (Section 5.3), describing for each one the Transformer encoder adopted, the embedding strategy and the parameters of the fine-tuning;
- Chapter 6 discusses the details of the training for the three subsystems of DDR (Section 6.1), DER (Section 6.2) and DKR (Section 6.3), describing for each one the Transformer encoder adopted, the embedding strategy and the parameters of the fine-tuning and ultimately an overview of the final results for the complete pipeline of the HDKR system comparing them with the original baseline and the current state of the art approaches;
- Chapter 7 finally provides a synopsis of the whole experiment analyzing the points of strength of the HDKR system and considering possible future improvements.

Chapter 2

Task

The experiment presented in this work is based on the second sub-task of the track 1 of the ninth Dialogue System Technology Challenge proposed by Amazon [10]. The second sub-task consisted in retrieving documents containing Frequently Asked Questions (FAQs) knowledge snippets from an unstructured knowledge base. In this chapter is described the framework of the task adopted for this experiment.

Conversational AI suffers from a lack of uniform terminology and clear definitions. Syvänen and Valentini [11] showed how there is a lack of clear and uniform definitions and how many studies use different terms indiscriminately without explicit definitions. Moreover, when definitions are provided these are typically inconsistent among the different studies and they are all similar even when they refer to different terms. For example, terms like *chatbot*, *virtual assistant*, and *conversational agent* are commonly used as synonyms. This heterogeneity makes studies in the field of conversational AI difficult and confusing. For this reason in Section 2.1, it will be explicitly defined the object of investigation, providing a description of its main characteristics and presenting its general architecture.

Afterwards, in Section 2.2 are outlined the guidelines of the framework of the experiment, providing firstly a general introduction on the Dialogue System Technology Challenge (DSTC) and then a specific description of the challenge proposed for the first track of the ninth DSTC edition.

Finally, the focus will be put on the retrieving sub-task of the challenge. Starting from the conclusions drawn by the DSTC9 challenge and the subsequent workshop, in Section 2.3 it is explained the importance of the retrieving system for the final knowledge-enhanced conversational agent and the concerns emerged during the workshop in terms of feasibility of the systems considering efficiency parameters such as the latency time.

2.1 Knowledge-enhanced Conversational Agents

Conversational AI is a branch of research in the field of Natural Language Processing (NLP) that focuses on the development and improvement of techniques adopted to create AI-powered software systems capable of performing a conversation. The object of study of Conversational AI is the **Conversational Agent**.

Definition 1 *A Conversational Agent (CA) a software agent that can engage in natural conversational interactions with humans, with or without a specific purpose.*

We, as humans, are used to dealing with conversational interactions on a daily basis. Nevertheless, a conversational interaction is a complex *semantic activity*. Eggins and Slade [8] describes it as a turn-based negotiation aimed at building some *meaning*. The crafting of this meaning is performed through the sharing of personal or socially shared knowledge. Hence, knowledge is an essential part of each conversation. In fact, from one hand knowledge constitutes the structural basis and the common ground on which we construct our conversational interactions, while on the other hand it represents also the bargaining chip of the sharing process itself.

Knowledge is therefore essential for a CA, both to fully understand the current state of the conversation and also to be able to make some progress in the meaning-construction process, contributing with some additional and coherent information. Considering chit-chat conversations for example, socially shared commonsense knowledge represents the background information that people use during conversation [12, 13] and thus it is essential to establish effective conversational interactions. Indeed, a known problem of CAs that rely only on the context coming from the input utterances, is that they tend to produce trivial responses such as "Yes, I see" or "Me too" [14, 15]. These meaningless responses lack meaningful content and do not contribute to the progress of the conversation. Regarding task-oriented CAs such as the assistants instead, the additional knowledge can help in designing more robust models able to deal with out-of-domain requests in order to offer a *frictionless* conversational interaction [16]. Generally speaking, Sarikaya [17] delineated how in the future CAs must be able to perform both task-oriented in an open-domain setting responding to queries from any domain by making advantage of a variety of *knowledge sources*. For this reason, modern Conversational Agents cannot disregard the use of additional sources of knowledge, that can help them with background and contextual information or specific and sectoral wisdom, in order to provide more accurate and appropriate responses.

Definition 2 *A Knowledge-enhanced Conversational Agent (KCA) is a conversational agent that exploits one or more knowledge sources in order to provide more accurate and appropriate responses.*

2.1.1 Characteristics of KCAs

"Knowledge-enhanced Conversational Agent" is a general term that encloses a wide set of different types of agents. Chatbot [18] and Intelligent Personal Assistant [19] for instance, are often used as synonyms for KCA, even if they refer to substantially different agents with distinct characteristics. Each of these denominations are explicit instances of a KCA that specify a peculiar characteristic of the agent (e.g., assistant).

For this reason, it is important to delineate the various attributes that can characterize a KCA. It is possible to group KCAs considering mainly four different criteria:

1. the ***goal*** of the conversation;
2. the ***domain*** of the conversations;
3. the ***modality*** of the conversational interaction;
4. the ***conversation memory*** of the agent.

Goal. The first and most divisive criterion of distinction is the ***goal*** of the conversation. This is the final scope for which the agent was programmed. In this sense, the agents can be distinguished in:

- *task-oriented*: these are agents designed to perform conversations with users with the scope to assist them to complete certain tasks (e.g, booking a fly ticket or a hotel room) [20].
- *non-task-oriented*: these are agents whose goal is to engage in a *chit-chat* conversation with users, without any specific purpose, except that of maximizing their interest and engagement [21].

Domain. The second criterion of distinction is based on the ***domain*** on which the agent operates. Here we can distinguish three types of agents:

- *single-domain*: agents designed to operate on a single specific domain (e.g., flights);
- *multi-domain*: agents designed to operate on a finite set of domains (e.g., flights, hotel and restaurant);
- *open-domain*: agents untied from any specific domain.

It is common in literature to adopt the term open-domain to refer to non-task-oriented agents as well, since these latter are generally used to engage in conversations unrelated to any specific domain. However, a distinction should be made: non-task-oriented agents can also be limited in scope, focusing for example on a single domain (e.g., BASEBALL [22]) and vice-versa task-oriented agents, should aim in the future to be able to operate in an open-domain setting, tackling requests from any domain.

Modality. The third criterion regards the ***modality*** of the conversation, which is the channel adopted to convey the conversational interaction. Conventionally, this is related to natural language as *text* or *voice*. However, other modalities may also be considered. For example: videos and images, touch input, gestures, etc. Nowadays, modern Conversational Agents usually grant greater flexibility to the users allowing more than a single modality. These agents are referred to as *multi-modal*.

Conversation memory. The last criterion is related to the ***conversation memory*** of the agent. This represents the dialogue history retained by the model during a single session and it can be considered as a sort of short-term memory of the agent that allows it to be more aware of the conversation that is happening. Here there can be distinguished two types of agents:

- *single-turn*: agents that do not keep any information from one turn to another. Each turn behaves like an independent act of the dialogue that involves a single user question and the following system response. This type of dialogues is called *one-shot exchange*.
- *multi-turn*: agents that at each turn are aware of a window composed by the previous n turns. This conversation memory is used for example to solve anaphoric references.[1]

[1]An anaphoric reference is a word or phrase that refers to something mentioned previously.

2.1.2 Architecture

As seen in the previous section, there can be distinguished various types of knowledge-enhanced CA with very diverse characteristics. However, all these types of agents share the same basic root architecture (see Fig. 2.1). Ideally, a knowledge-enhanced CA can be thought as a modular framework composed of three main modules:

1. **Dialogue System**: this is the module that manages the communicating part of the agent from the user conversational input to the agent conversational output;

2. **Memory System**: this is the module that manages all the knowledge resources of the agent;

3. **Interface**: this is the module that has the job of harmonizing the flow of information between the two preceding modules.

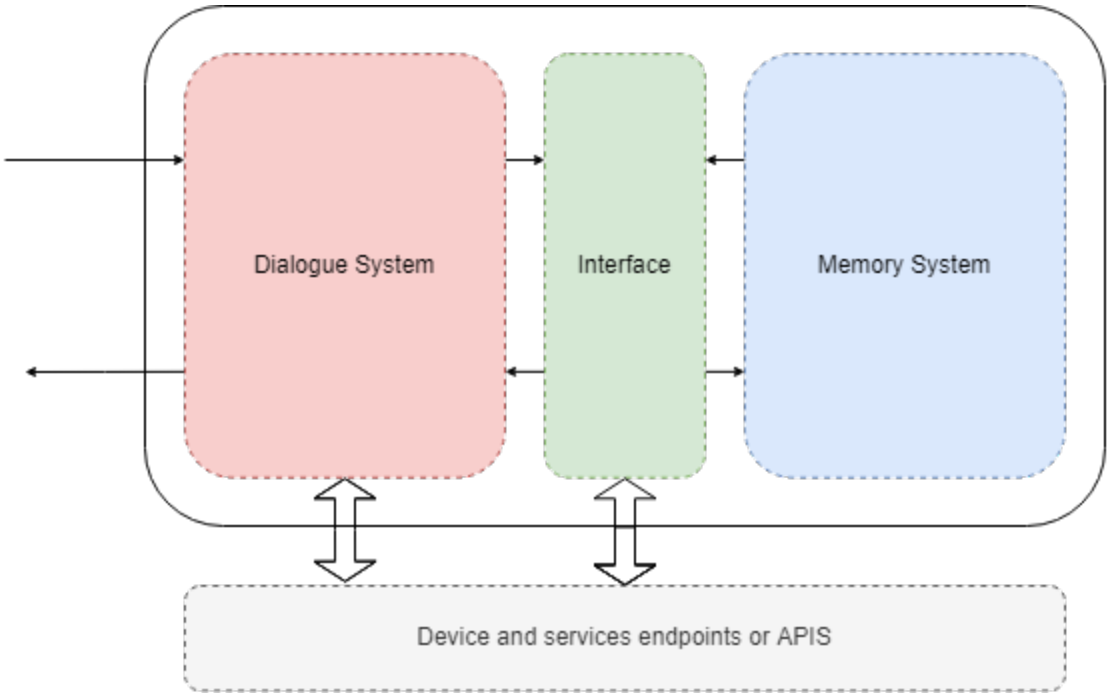

Figure 2.1: Modular architecture of a KCA.

Dialogue System

The Dialogue System is the module that deals with the communicating part of the agent. The task of this system is to manage the conversational input and to prepare an adequate conversational output. In addition, in the case of a multi-modal agent, the dialogue system must deal with issues related to the combination of different modalities adopted to gain the input message from the user and to deliver the response.

The dialogue system is the core part of the agent that mainly characterizes it. Considering natural language conversations, there can be distinguished two types of architectures for Dialogue Systems [20]. The **pipeline architecture** involves a cascade of modules. In this architecture, the input message passes through a series of blocks in order to be decomposed into a semantic representation. This semantic representation is commonly referred to as the *semantic frame* and contains information such as the *intent* of the user or the *domain* of the request. In contrast, **end-to-end architectures** are composed by a single module that takes in input the user request and directly returns in output the system response.

Memory System

The memory system is composed of all the knowledge sources that are directly reachable from the CA. There are different types of knowledge that can be stored. [23] makes a distinction between *internal* and *external* sources of knowledge with respect to the conversation. The former are inferred from the input and can include keywords and linguistic features. The latter are provided from outside sources as knowledge bases. [24] differentiate between *generic* and *domain-dependent* knowledge. While the first offers additional information from any domain (e.g., information about user interaction like in [25]), the second is referred to a specific domain or set of domains [26].

Each knowledge source is generally consisting of a Knowledge Base (KB) composed of a set of *knowledge snippets*. The knowledge snippet represents the atomic knowledge information stored in the KB. Based on the composition of the knowledge snippets, it is possible to distinguish between two types of knowledge (Fig. 2.2):

- **unstructured** knowledge represents the simplest form of knowledge in which the knowledge snippets consist of raw-text documents;
- **structured** knowledge consists of documents that are pre-processed performing a mapping from the raw text to a *semantic representation* of the knowledge that than is typically stored in a specialized architecture.

Unstructured Knowledge	Structured Knowledge
Marie Curie (7 November 1867 - 4 July 1934) was a Polish and naturalized-French physicist and chemist who conducted pioneering research on radioactivity. She is the only person able to win the Nobel Prize in two scientific fields (Chemistry and Physics)	(Marie Curie, date_of_birth, 07/09/1867)
	(Marie Curie, date_of_death, 04/07/1934)
	(Marie Curie, profession, chemist)
	(Marie Curie, profession, physicist)
	(Marie Curie, winner_of, Nobel prize in chemistry)
	(Marie Curie, winner_of, Nobel prize in physics)

Figure 2.2: Example of an unstructured knowledge snippet composed by a simple text document and the corresponding structured knowledge composed of (subject, relation, object) triplets.

Interface

The interface is the module that has the job of harmonizing the information flow between the Dialogue System and the Memory System. Its main job is to prepare an appropriate query to the Memory System based on the information about the user request collected by the Dialogue System and then integrating the knowledge retrieved into the system-generated response. Traditional approaches consisted in designing specialized architectures [23] heavily depending on the type of the dialogue system and in particular on the type of knowledge snippets present in the KB. In the pipeline architecture for example, the agent works in a discrete setting with clear information about the user intent, the domain of the request, etc. This makes it easier to construct a query to the memory system. On the contrary, end-to-end dialogue systems work in a continuous environment. The semantic representation of the user input is made in a latent space and is generally referred to as the *context vector*. In this case there is no explicit state representation of the dialogue that can be exploited and the query must be constructed directly from the context vector. Therefore, the knowledge base interaction is conducted by using intermediate hidden representations [27].

Following the work of Kim et al. [16], in the next section is presented a general framework for the Interface that is based on three sub-tasks that cover three fundamental conceptual aspects involved in the knowledge-enhancing process. The first one is the *Detection* of turns that needs to access to the memory system of the agent, the second is the *Selection* of the correct knowledge from the memory system, and the final one is the *Generation* of an appropriate response conditioned

on the input and the knowledge selected.

2.2 The Dialog System Technology Challenge (DSTC)

The DSTC is a series of competitions held annually where teams of researchers from all around the world can compete on different challenges related to the world of Conversational AI. The series of challenges started in 2013, with the objective of boosting the development of dialog systems. Formerly named *Dialog State Tracking Challenge*, to enforce the initial focus that was reserved in the first editions only towards the Dialog State Tracking (DST) problem;[2] starting from the sixth edition of the challenge evolved and the DSTC rebranded itself as ***"Dialog System Technology Challenge"*** following the necessity to extend the scope of research to various dialog-related challenges such as: tackling multi-modality (e.g., Audio and Video sources), developing valid model evaluation metrics, studying knowledge-enhancing methodologies, and investigating Zero-shot or Few-shot learning.[3]

Moreover, given the remarkable success of the first five editions, and considering the interest of the research community in a wider variety of dialog-related problems, starting from the DSTC6, the challenge started to propose more than a single track for each year.

For the ninth DSTC challenge, four different tracks were proposed: (1) Beyond Domain APIs: Task-oriented Conversational Modeling with Unstructured Knowledge Access, (2) Multi-domain Task-oriented Dialog Challenge II, (3) Interactive Evaluation of Dialog, and (4) SIMMC: Situated Interactive Multi-Modal Conversational AI.

The work presented in this thesis will be centered on the second sub-task of the first of the four tracks of the DSTC9.

2.2.1 DSTC9 - track 1

The first track of the DSTC9 challenge was proposed by Amazon and followed the work of Kim et al. [16]. The framework of the challenge consisted in a ***multi-turn*** and ***multi-domain*** KCA operating in a ***task-oriented*** setting. Commonly, a task-oriented CA is grounded on an API interface. This means that the scope of the CA is limited by the services offered by the API itself. This approach falls down easily even for very simple questions, when the model encounters

[2]The DST problem aims at estimating the user's goal as a dialog progresses

[3]Zero-shot is the ability of a model to generalize its capabilities to domains that it has not seen during the training phase

out-of-scope requests. For instance, consider a case in which a user wants to book a room in a hotel. A task-oriented CA should be able to provide different solutions to the user based on her or his needs. Imagine then, a very reasonable follow-up request like: *"Are pets allowed in this hotel?"*. Requests like these are generally not covered by a specific API service causing difficulties for the model that does not know how to respond to the request and so blocking the flow of the conversation. Generally, these kinds of questions can be easily answered by searching among the Frequently Asked Questions (FAQ) documents of the hotel. In their work, Kim et al. [16] proposed a framework to implement a *frictionless* task-oriented Conversational Agent able to tackle out-of-scope requests enhancing the model capabilities through the incorporation of external unstructured knowledge taken from FAQs documents [9] of different entities.

Framework

A natural conversational interaction can be thought of as an ordered sequence of utterances $U = (u_1, u_2, \ldots, u_n)$. An utterance is commonly defined as the smallest unit of speech, consisting of a continuous piece of speech terminated by a clear pause. In the case of a textual conversation, is considered an utterance a single textual *message* composed by one or more sentences. Each utterance corresponds to an ordered sequence of tokens $u_i = (x_1, \ldots, x_l)$. In Conversational AI it is commonly assumed that the conversation happens in a turn-taking fashion, in which the *user* and the *agent* alternates their utterances. The *dialogue context* $U_{t,w} = (u_{t-w+1}, \ldots u_{t-1}, u_t)$ at time t and with window size w is defined as the ordered set of the last w utterances and that always ends with an utterance u_t coming from the user. Commonly, U refers to the entire dialog context, from the first utterance u_1 to the last utterance u_t of the conversation.

A knowledge base $\mathcal{K}$ is a set of knowledge snippets k_j. In the context of the DSTC9 - track 1, each knowledge snippet consisted of raw-text FAQ document.

Problem (DSTC9 - track 1): *Design a multi-domain multi-turn task-oriented knowledge-enhanced conversational agent, that given in input a dialogue context $U_{t,w}$, is able to generate an appropriate and informative system response $\tilde{u}_{t+1}$ grounded on a set of relevant knowledge snippets $\tilde{\mathcal{K}} \subset \mathcal{K}$.*

In the DSTC9 -track 1 this knowledge-enhancing process is performed by an explicitly designed interface based on three subsystems (Fig. 2.3). Each subsystem covers a fundamental conceptual aspect involved in the knowledge-enhancing task. The three subsystems are:

1. **Detection**: for each turn t, given a dialogue context $U_{t,w}$ the Detection system has to decide if the current input utterance u_t from the *user* requires additional knowledge in order to be addressed.

$$f_D(U_{t,w}) = \begin{cases} 1 & \text{if } U_{t,w} \text{ requires additional knowledge} \\ 0 & \text{else} \end{cases}$$

   A turn that necessitates access to the knowledge request is referred to as *knowledge-seeking turn.*

2. **Selection**: for each knowledge-seeking turn u_t, the Selection system has the job to retrieve a subset $\tilde{\mathcal{K}}_{U_{t,w}} \subset \mathcal{K}$ of knowledge snippets from the knowledge base, that are relevant for the current dialogue context $U_{t,w}$. This can be considered as a binary classification problem on each knowledge snippet k_i:

$$f_S(U_{t,w}, k_i) = \begin{cases} 1 & if\ k_i \in \mathcal{K} \text{ is relevant for } U_{t,w} \\ 0 & else \end{cases}$$

   The final subset is obtained as the set of knowledge snippets relevant for the given dialogue context $U_{t,w}$:

$$\tilde{\mathcal{K}}_{U_{t,w}} = \left\{ k_i \in \mathcal{K} : f_S(U_{t,w}, k_i) = 1 \right\}$$

3. **Generation**: Given a dialogue context $U_{t,w}$ and a set of relevant knowledge snippets $\tilde{\mathcal{K}}_{U_{t,w}}$, the Generation system must be able to generate an appropriate system response $\tilde{u}_{t+1}$ conditioned on the current dialogue context $U_{t,w}$ and the set of relevant knowledge snippets $\tilde{\mathcal{K}}_{U_{t,w}}$.

$$f_G(U_{t,w}, \tilde{\mathcal{K}}_{U_{t,w}}) = \tilde{u}_{t+1}$$

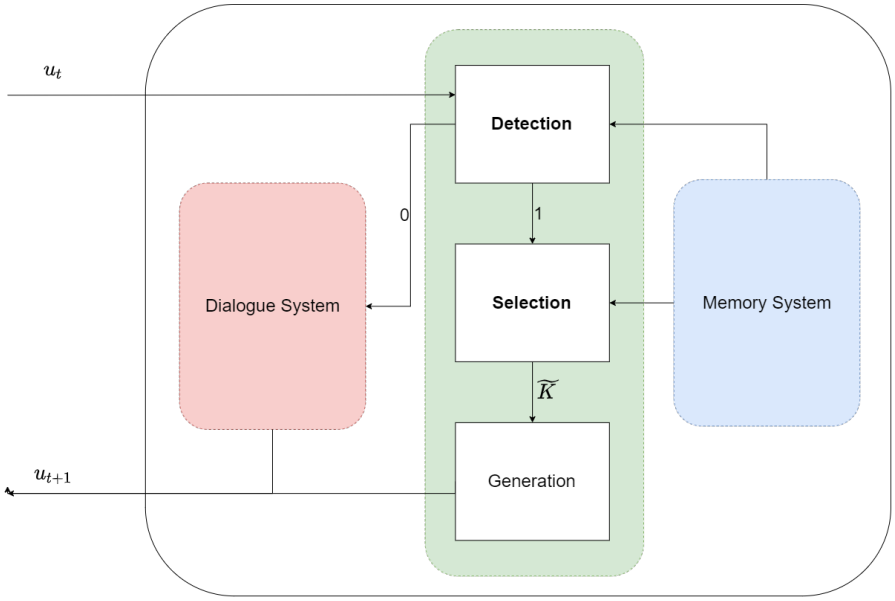

Figure 2.3: Modular architecture for the Interface proposed in the first track of the DSTC9.

Dataset

For this challenge, two different data sets were adopted. The first one is an augmented version of the MultiWOZ 2.1 [28], containing newly introduced knowledge-seeking turns. The data set was collected through a three-step crowdsourcing task [16]:

1. select a position in a given dialog;
2. insert an actual question (knowledge-seeking turn) in the selected position;
3. provide the correct answer to the question.

The final dataset contains a series of dialogue contexts, for which each user utterance is labeled with a boolean value to indicate if that is a knowledge-seeking turn. In the positive case it is also reported the correct FAQ document and the human generated response that is the expected output of the agent (for more information see Fig. A.2).

A total of 22,834 utterance pairs were collected based on 2,900 knowledge snippets from the FAQ webpages about the domains and the entities present in the MultiWOZ data set. The data set contains dialogs spanning from four possible domains: hotel, restaurant, train and taxi. For the first two domains, different

entities were considered (i.e., specific hotels and restaurants) while for the domains of train and taxi only general FAQ documents were gathered.

From each dialogue $U = \{u_1, \ldots, u_n\}$ of dimension n were derived $\lceil \frac{n}{2} \rceil$ samples, considering all the sub-dialogues terminating with a user request. Each sequential user request is commonly referred to as a *turn*. Each sub-dialogue derived from the same dialogue context, started with the same first user request u_1 up to the j-th one, creating a redundancy of information in the data.

$$U = \{u_1, \ldots, u_n\} \mapsto \bigcup_{i=0}^{\lfloor \frac{n}{2} \rfloor} \{u_1, \ldots, u_{2i+1}\}$$

For each sub-dialogue a corresponding label reported if the last user request was a knowledge-seeking turn (i.e., a request that cannot be addressed by the API directly), and in the case it was, signaled also the domain, entity and document information about the correct knowledge snippet in the knowledge base and the human-generated response for the request. An example extracted from the dataset of the DSTC9 - track 1 can be seen in Fig. A.2.

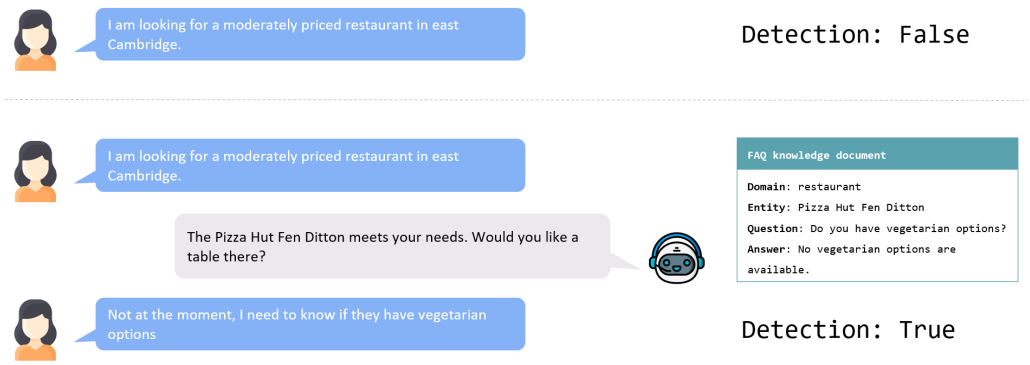

Figure 2.4: Example of two sub-dialogues sampled from the same dialogue context with the corresponding labels. The first request does not need access to the knowledge base, while the second request is a knowledge seeking turn regarding the domain *'restaurant'* and the entity *'Pizza Hut Fen Ditton'*.

The data set was split into three parts: training, validation and test. For the evaluation phase, additional conversations collected from scratch about touristic information for San Francisco were also considered. To evaluate the capabilities of generalization of the models, the new conversations covered knowledge requests and entities coming from the four aforementioned domains plus a new unseen domain. Moreover, the new conversations introduced an additional order of difficulty considering multi-modal dialogues coming both from written and spoken conversations.

DSTC9 - track 1 dataset

Source	**Split**	**Total dialogs**	**Num. instances**	**Knowledge seeking turns**
MultiWOZ	Train	7190	71348	19184
	Valid	1000	9663	2673
	Test	977	2084	977
SF	Written	900	1834	900
	Spoken	100	263	104

Table 2.1: Dimensionality of the dataset for the first track of the DSTC9.

Finally, the second important element of the dataset was $\mathcal{K}$ the unstructured Knowledge Base (KB). The KB was constructed starting from Frequently Asked Questions (FAQs) from several different entities among the five aforementioned domains. Each FAQ knowledge document $k_i \in \mathcal{K}$ in the KB consisted of a tuple containing three information (see Fig. 2.5)

$$k_i = (domain, entity\_name, FAQ\ document)$$

where FAQ document was the actual unstructured textual information containing a question and the corresponding answer.

Domain	Entity Name	FAQ Document
hotel	A and B Guest House	**Q:** Can I bring my pet to A and B Guest House? **A:** No, pets are not allowed at this property.
restaurant	Pizza Hut Fen Ditton	**Q:** Do you have vegetarian options? **A:** No vegetarian options are available.
train	*	**Q:** Can I bring my pets to trains? **A:** We happily welcome dogs and cats up to 20 pounds for trips up to seven hours.
taxi	*	**Q:** How will I receive my booking confirmation? **A:** Booking confirmations will be sent via text messages shortly.
attraction	7D Experience	**Q:** Are there age restrictions for 7D Experience? **A:** No, there are no age restrictions for 7D Experience.

Figure 2.5: Example of five FAQ documents present in the knowledge base. For each document is reported the domain (e.g., *'hotel'*), the entity (e.g., *'A and B Guest House'*) and the actual document composed by question and answer. Further information about the knowledge base can be found in Fig. A.1 on the Appendix.

Evaluation method

For the evaluation of the models submitted to the challenge, two approaches were taken. First, each submission was evaluated through task-specific objective metrics (Tab. 2.2). This first evaluation step was adopted to perform a filter of the best performing models.

Task	**Metrics**
Detection	Precision/Recall/F-Measure
Selection	Mean Reciprocal Rank/Recall@1/Recall@5
Generation	BLEU-1/BLEU-2/BLEU-3/BLEU-4/METEOR/ROUGE-1/ROUGE-2/ROUGE-L

Table 2.2: Objective measures adopted for the three tasks of the DSTC9 track 1.

Due to the sequential structure of the three subsystems designed for the DSTC9 - track 1, the results of the Selection and Generation were depending on the Detection step. To tackle the dependency between the three tasks, the final scores for the Selection task and Generation task have been normalized considering the Detection recall and precision performances as follow:

$$\tilde{f}_1(x) = \begin{cases} 1 & \text{if } x \text{ is predicted as a knowledge-seeking turn} \\ 0 & \text{otherwise} \end{cases}$$

$$S_p(X) = \frac{\sum_{x_i \in X} \left(s(x_i) \cdot f_1(x_i) \cdot \tilde{f}_1(x_i) \right)}{\sum_{x_i \in X} \tilde{f}_1(x_i)};$$

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

$$S_f(X) = \frac{2 \cdot S_p(X) \cdot S_r(X)}{S_p(X) + S_r(X)};$$

where $s(x)$ is the selection or generation score in a target metric for a single instance $x \in X$. Finally, the best models in the objective measures were selected and passed through a second evaluation phase. This consisted of a human judgement in which each model response was evaluated through a crowd sourcing task considering two criteria:

- **Appropriateness**: how coherent is the system output with respect to the conversation on a scale of 1-5.
- **Accuracy**: how accurate is the system output based on the provided reference knowledge on a scale of 1-5.

2.3 The need for efficient knowledge-retrieval systems

2.3.1 The importance of the knowledge selection

The DSTC9 - track 1 has been a success in terms of involvement, with a total of 24 participant teams and an overall number of 105 entries submitted. This helped in gaining a lot of information about what are the key factors to develop accurate and appropriate knowledge-enhanced task-oriented conversational agents able to deal with out-of-scope requests. One of the most interesting results that emerged during the evaluation phase was the importance of the selection system

in the whole knowledge-enhancing pipeline. In particular, developing an accurate knowledge-retrieval system is fundamental in order to create an accurate and appropriate conversational agent.

Indeed, the final winner of the challenge with the highest scores for both Accuracy and Appropriateness in the human evaluation phase, was not the team 3[29] that resulted as the overall best team in the objective metrics evaluation. However, it was the team 18[30] that resulted as the best team in both the Mean Reciprocal Rank (MRR) score and Recall at 1 (R@1) (i.e., first selected knowledge snippet) for the knowledge Selection task. This suggests how much important is the selection of the correct knowledge snippets. In particular, it emerged the importance of the selection of the first document from the knowledge base. This is not surprising at all, considering the task-oriented setting of the challenge. In this setting in fact, the user queries the agent with the aim to retrieve information about a specific entity (e.g., a specific hotel) and not a general similar one. Therefore, it is not enough to retrieve a general related document from the knowledge base as if it was the case of an open-domain conversation, but the system has to accurately retrieve the correct knowledge snippet in the correct domain and from the exact entity.

	Detection			**Selection**			**Generation**							
Team	Precision	Recall	F-1	R@1	R@5	MRR	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR	ROUGE-1	ROUGE-2	ROUGE-L
3	**0.9964**	**0.9859**	**0.9911**	0.9013	**0.9840**	0.9395	**0.3864**	**0.2539**	0.1692	**0.1190**	0.3914	**0.4332**	**0.2115**	**0.3885**
15	0.9933	0.9677	0.9803	0.8975	0.9460	0.9195	0.3779	0.2532	**0.1731**	0.1175	**0.3931**	0.4204	0.2113	0.3765
18	0.9954	0.9818	0.9886	**0.9235**	0.9814	**0.950**	0.3803	0.2449	0.1590	0.1081	0.3869	0.4192	0.1976	0.3738

Table 2.3: Final results on the test set for the three sub-tasks of the DSTC9 - track 1.

To further investigate this aspect of the competition, Kim et al. [10] performed an analysis of the correlation between the objective metrics and the human-evaluation metrics showing the significance of the knowledge selection performances with respect to the final human judgement. Figure 2.6 shows Spearman's rank correlation coefficient [31] between the various metrics. As it can be seen, the knowledge selection measures showed stronger correlations to the final ranking than all the other metrics. R@1 in particular, has the strongest connection to the mean human assessment rating with a value of 0.8601, which is substantially higher than 0.7692 for the F-measure for the detection and 0.6503 for the BLEU-1 in the response generation. As a result, is it possible to affirm that knowledge retrieval is a key task for improving the overall performance of the agent.

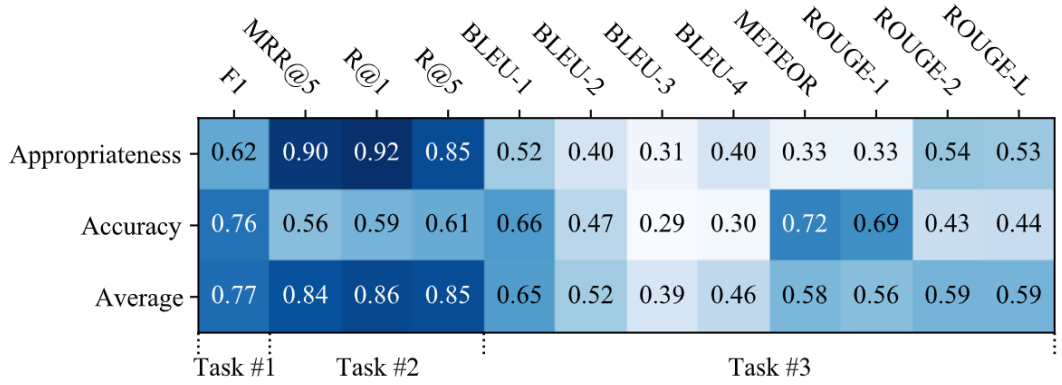

Figure 2.6: Correlations between the objective and human evaluation metrics in Spearman's ρ. Task # 1 correspond to the Detection, Task # 2 to the Selection and Task # 3 to the Generation. Image from [10].

2.3.2 The concerns about the efficiency of the retrieval systems

Another important point of reflection was drawn from the workshop of the DSTC9 held at the 35th Association for the Advancement of Artificial Intelligence (AAAI) conference, where some concern emerged about the real usability of the current best models. The majority of the teams adopted strategies based on the so-called Passage Re-Ranking (PRR) [32] framework. Although, PRR models settled as the state of the art, achieving impressive results in many question answering tasks, these models are also typically really expensive, both in term of latency time and computational resources (energy, hardware cost), making many of these models impractical especially under resource constraints. Indeed, the PRR strategy requires at inference time a point-wise evaluation of all the knowledge snippets present in the knowledge base. This means that the time consumption of the methods scales linearly with the dimension of the KB becoming quickly intractable in real-case scenarios. This is not doable in a real application, where the knowledge base of documents can grow really fast, but where the user's requirements in terms of latency of the system response remain independent on the size of KB.

Regarding the DSTC9 - track 1 huge models were used such as PLATO-2 [33] adopted by the final winner team. In addition, many of the solutions proposed made advantage of ensembles of multiple models [16] to compute the final score as a weighted average or through majority voting strategies considering different parameters. For instance, He et al. [30] employed 5 different extra pre-trained models, including PLATO-2 both in the two versions with 32 layers and 24 layers, BERT-base [34] and ALBERT-xlarge [35]. If from one hand this approach can

offer a boost in accuracy of the retrieval system; on the other hand, it is also an unpractical solution that is typically not usable in a normal setting where there can be strict limitations especially in terms of computational resources (e.g., memory usage).

Considering the fundamental role of the knowledge retrieval system in a task-oriented knowledge-enhanced conversational agent, this work presents an *efficient* solution for the knowledge selection task, taking into consideration the latency time for the retrieval of the documents as a fundamental parameter and the dimension in terms of memory consumption of the models adopted. At the same time, the model aims at maintaining as much as possible the accuracy of the retrieval with respect to the best performing models.

Chapter 3

Related Work

In this chapter will be provided an exhaustive description of the technologies and the methodologies adopted to face up the knowledge retrieval problem, considering the state-of-the-art approaches in question-answering settings and alternative solutions more keen on an efficient retrieval of the information.

Knowledge retrieval is a part of the more general **Information Retrieval (IR)**, a field of study that focuses on solutions to select relevant information from a huge collection of data given an input query. A fundamental aspect of information retrieval is the representation of the objects of the collection. For instance, regarding the DSTC9 - track 1 the collection of objects to retrieve was composed of unstructured textual FAQ knowledge documents.

Sec. 3.1 provides an introduction on various *Sentence Embedding* methodologies that can be adopted to convert the textual knowledge documents into a numerical *embedding vector*. Then are discussed the main approaches in the context of information retrieval for unstructured textual documents, considering the two principal approaches: relationship-based methods and representation-based methods. These two approaches are based on two corresponding methodologies respectively called *Passage Re-Ranking* [32] and *Dense Passage Retrieval* (DPR) [36] that are described in Sec. 3.2. To conclude, in Sec. 3.3 are considered different possible approaches that could be taken to improve the performance of the information retrieval systems in the context of the DSTC9 - track 1.

3.1 Sentence Embedding

Computers are unable to directly manipulate textual information. In order to be able to adopt software-based systems to operate with text and strings, it is necessary to develop techniques to convert textual information into numerical one. This problem is commonly referred to as **Sentence Emebedding** and is one of

the oldest and most interesting areas of research in the field of Natural Language Processing (NLP).

Sentence embedding is the process of vectorization of a textual string into a real-valued representation vector also referred to as the **embedding vector**. The basic idea behind sentence embedding techniques is to craft a vector space V that is *semantically enriched*. This means that the positional information within this vector space is used to encode semantic information. An Encoder E is a model that performs a sentence embedding mapping a textual document d into an embedding vector $\mathbf{w}_d$ able to convey the original semantic message of the document into a numerical form:

$$\mathbf{w}_d = E(d)$$

Numerical text representation is a crucial task for many Natural Language Processing applications such as sentence categorization, sentiment analysis, machine translation and also Question-Answering (Q&A), such as the one proposed in the context of the DSTC9 - track 1.

3.1.1 Symbol based models

The first solutions adopted in the dominion of sentence embedding approached this problem with Bag-of-Word (BOW) strategies. These models were based on *symbols*, generally considering words or groups of words as the minimum unit for the semantic representation. The set composed by the union of all the symbols x_i represented the *vocabulary* $\mathcal{V} = \bigcup_{i=1}^{n} x_i$ of the model. In these models, each sentence was encoded as a vector of the dimension of the vocabulary size n, and each element in the vector represented the *importance* of the corresponding symbol for the given sentence.

The most straightforward BOW method is the Term Frequency (TF) representation. Considering a vocabulary composed by n distinct symbols x_i (e.g., words), each symbol can be represented as a vector w_i of the type:

$$\mathbf{w}_i = [0, \ldots, 0, \underbrace{1}_{i\text{-th position}}, 0, \ldots, 0] \in \mathbb{R}^n$$

This vectorization process is commonly referred to as called One-Hot Encoding. Given a sentence $s = \{x_1, x_2, \ldots, x_l\}$ of length l is then possible to define a sentence embedding based on the one-hot word encoding:

$$\mathbf{tf_s} = \frac{\sum_{i=1}^{l} \mathbf{w}_i}{l}$$

This representation is called Term Frequency since each element in the resulting vector corresponds exactly to the fraction of occurrences of the corresponding word

in the given sentence. TF is a flat representation that does not take in consideration the importance of the specific words. Some words contribute deeply to the semantic information of the sentence, other words such as articles appear equally in all the sentences and therefore their contribution is almost irrelevant. For this reason, more advanced methods have been developed.

Given a corpus of sentences $\mathcal{S}$, the Inverse Document Frequency (IDF) of a symbol $x_i \in \mathcal{V}$ measure the importance of that symbol in the vocabulary, based on $\mathcal{S}$:

$$idf_{x_i} = \log \frac{|\mathcal{S}|}{df_{x_i}}$$

where $|\mathcal{S}|$ is the number of the sentences in $\mathcal{S}$, while df_{x_i} represents the Document Frequency (DF) of the word x_i in the whole corpus $\mathcal{S}$. The Term Frequency - Inverse Document Frequency (TF-IDF) sentence representation encodes each sentence s in a corpus of sentences $\mathcal{S}$, considering both the term frequency of the single symbols in the sentences and also the specific importance of the symbols in the whole corpus $\mathcal{S}$:

$$\textbf{tf-idf}_s = \textbf{tf}_s \odot \textbf{idf}$$

where $\textbf{idf} = [idf_{x_1}, \ldots, idf_{x_n}] \in \mathbb{R}^{|\mathcal{V}|}$ is a vector containing the idf scores for all the symbols $x_i \in \mathcal{V}$ in the vocabulary and $\odot$ represent the element-wise multiplication.

3.1.2 Recurrent Neural models

BOW approaches presented two fundamental problems. The first one regards the final dimension of the representation vectors. In these methods in fact, the dimension of the representation vectors scales linearly with the number of symbols present in the vocabulary. However, the vocabulary size can reach really fast orders of tens or hundreds of thousands of symbols even for corpora with a moderate number of documents. Moreover, typically each sentence contains only a small subset of symbols. Hence, the resulting representation vectors will be composed by a vast majority of zeros and only some real-valued non-zero entries, adding a sparsity issue to the problem. Secondly, sentences present complicated structures, including both sequential and hierarchical patterns, that are necessary for the total comprehension of the meaning of the sentence itself. Nevertheless, Bag-of-Word based methods completely ignore the structural information of the text, disregarding even the sequential order of the symbols.

Neural language methods have been proposed as a possible solution to try to tackle these issues. Recurrent Neural Network (RNN) [37] constitute a milestone

in the field of NLP. Their contribution to the field can be summarized in two main points: first, they introduced a procedure to deal with different sized inputs, generating output vectors of a pre-defined dimension independently from the dimension of the input and from the number of symbols; second, they established a new sequential procedure to take into account the temporal structure of the inputs.

The main characteristic of RNN models is that they operate jointly on the input space and on an internal state space. At each time step t, given an input x_t the model computes an internal representation of the current state h_t from the previous one h_{t-1} and the input itself. This internal representation is usually called *hidden state* vector. The final output at time t is the vector y_t computed starting from the hidden state h_t:

$$h_t = \tanh\left(W_h h_{t-1} + W_x x_t\right)$$
$$y_t = W_y h_t$$

where W_h, W_x and W_y are weight matrices that the model learns during the training phase.

However, also RNN based methods have their own set of problems. One major issue is that RNNs can not be parallelized because they take one input at a time processing each token by token. Hence, training such a model on a big dataset will take a lot of time.

3.1.3 Deep Neural models

In 2017, Google drastically changed the course of NLP history with a revolutionary paper [38]. In their work they presented the ***Transformer***, an encoder-decoder architecture based on a series of *multi-head attention layers* that radically changed the way of approaching many natural language-related problems.

In the context of sentence embedding, the encoder part is the most interesting one. In 2018 was presented the Bidirectional Encoder Representations from Transformers (BERT), the forerunner of all the transformer-based encoders. BERT inaugurated also the era of Pre-trained Language Model (PLM) in NLP, huge models trained on large corpora of text data whose knowledge can be redirected towards a wide range of NLP tasks with a simple fine-tuning process or small changes in the architecture such as the addition of a specific output layer.

BERT is a transformer encoder model designed to generate bidirectional encodings of single strings or pairs of strings of text by jointly conditioning on both left and right context. Two objective functions were designed for the pre-training stage: Masked Language Model (MLM), based on a *Cloze* task [39] and Next Sentence Prediction (NSP). The result of this process is a model able to capture various language knowledge for downstream supervised tasks. In fact, BERT can be fine-tuned for any NLP task, whose input consists of a single text or text

pairs, that can represent sentence pairs in paraphrase, hypothesis-premise pairs in entailment, question-passage pairs in Q&A, and simply a single text for text classification task or sequence tagging. As for the output, BERT can produce the token-level representation for each token, that can be used to sequence tagging or question answering tasks. Besides, the special token [CLS] in BERT is fed into the classification layer for sequence classification.

With the advent of BERT, Pre-trained Language Models became the standard in NLP and many more advanced encoder models were proposed. For instance, RoBERTa (Robustly Optimized BERT Pretraining Approach)[40] and XLNet[41] proposed some variants of BERT with improved pretraining tasks for better general language representation and overcoming some discrepancy among the pretrain and finetune of the original BERT. Other models extended the incredible encoding capacity of bidirectional models such as BERT presenting a Unified Language Model (UniLM)[42] able to encode and generate text at the same time, making advantage of both bidirectional and unidirectional attention masks.

3.2 Retrieval methods for unstructured knowledge documents

Information Retrieval (IR) is the process of finding useful resources or information from a huge collection of data. A classical application of IR are search engines, that must be able to return relevant information such as web sites or images based on a given user query.

Given a query q, neural based Information Retrieval systems aim to retrieve a subset of documents that are relevant for a given query, estimating a *relevance score* s_{q,d_i} for each document d_i. The relevance score is computed taking into account the ***Semantic Similarity*** between the two.

The traditional setup for IR consists of a list of search queries and a collection of documents. In the context of the Selection sub-task of the DSTC9 - track 1 the search queries are direct requests from the users about commodities or services offered by different hotels, restaurants, train, taxi and attractions. These requests are posed in the form of a dialog context $U = \{u_i, \ldots, u_n\}$, composed of a series of alternated utterances of which only the last utterance u_n is the real user request. Regarding the documents to retrieve instead, each FAQ knowledg document k_i in the knowledge base $\mathcal{K}$ consists of a tuple composed by three information: the domain of the document, the entity at which the document belongs and a pair of Q&A.

Typically, the retrieval is performed through a *ranking system*. Given a query q, all the documents k_i in the Knowledge Base are scored against the query, in order to generate a ranking of the documents based on their semantic relevance score

s_{q,d_i} with the query itself. The documents are then filtered considering a given threshold th considering only the set of knowledge documents $\{d_i : s_{q,k_i} > th\}$ or alternatively selecting only the top-k ones. These ranking models can be grouped into two main classes: relationship-based and representation-based.

3.2.1 Relationship-based methods

Relationship-based ranking methods aim at estimating the relevance between a query q and a document k_i through *word-level interaction patterns* between the two. Considering a transformer encoding model, this can be done by passing jointly the query q and a specific knowledge document k_i as input to the model. The two are commonly concatenated with a special token (e.g., [SEP] for BERT) in-between them and the model is trained to estimate the score s_{q,k_i} of how relevant a candidate passage k_i is to the query q. This approach is commonly referred to as **Passage Re-Ranking** [32]. In Fig. 3.1 is displayed an example in the context of the Selection task for the DSTC 9 - track 1. In this case the dialogue context U_t represents the query on which the retrieval is performed.

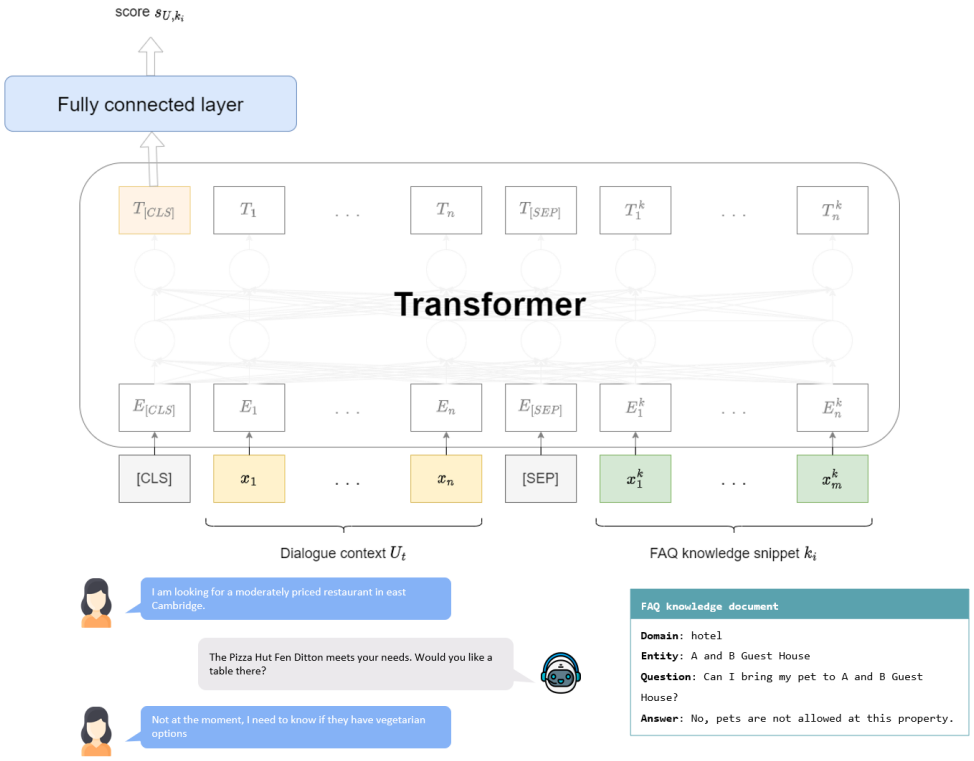

Figure 3.1: Example of a relationship-based retrieval through Passage Re-Ranking for the knowledge Selection in the DSTC9 - track 1.

Relationship-based retrieval methods through PRR was the most adopted strategy for the Selection task of the DSTC9 - track 1. The majority of the models were based on the same variation of the task using a PLM transformer model as the encoder to compute the similarity score between a concatenation of information containing the dialog context U_t, the question and answer from a FAQ knowledge document k_i and in some cases additional information such as the domain or the entity name [43].

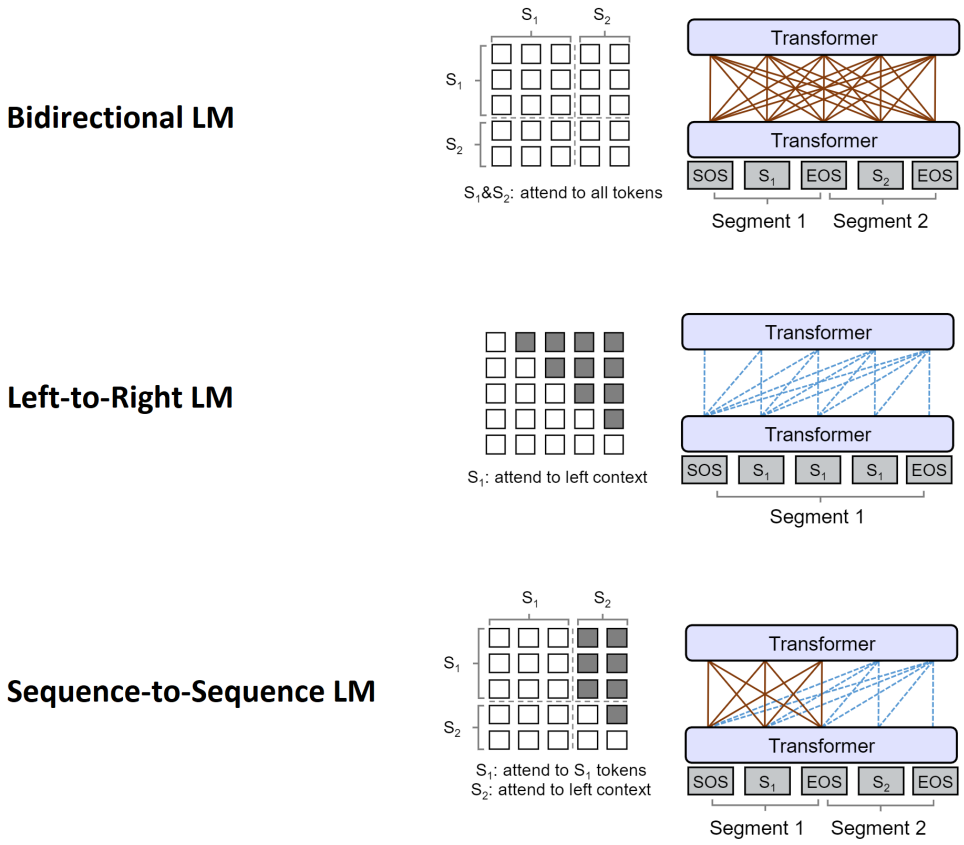

Figure 3.2: Three Language Model objectives adopted to pre-train UniLM and Plato-2. Image from Dong et al. [42]

He et al. [30] proposed a model that performed the best in the Selection sub-task, resulting also as the winner team in the final human evaluation score. In their work, they adopted a passage re-ranking strategy using an ensemble of different transformer models, among which PLATO-2 [33], a huge transformer model composed by 32 head attention layers and 1.6 billion parameters and based on the UniLM architecture that was jointly pre-trained on three types of unsupervised Language Model (LM) objectives (Fig. 3.2):

- **Left-to-Right LM**: predict a masked word based on the words on its left;
- **Bidirectional LM**: predict a masked word based on both the left and right context;
- **Sequence-to-sequence LM**: the word is predicted based on all the words of the first target sequence and the left context of the current sequence.

Training procedure and Negative Sampling

Depending on the task, relationship-based models can be trained in a regression fashion, where the output is the score s_{q,k_i} that measures the similarity between the knowledge document k_i and the query q. Then, this score can be used to generate a ranking of the documents based on their relevance. Alternatively, the model can also be trained in a binary classification setting, with output 0 to signal that the document k_i is not relevant to the query q or 1 if it's relevant for the given query.In this case it is not possible to perform a sorting of the selected documents based on their relevance score and the final output of the model will be a set of relevant documents.

Re-ranking models are commonly fine-tuned using a cross-entropy loss:

$$L = \sum_{j \in J_{pos}} \log(s_{q,j}) - \sum_{j \in J_{neg}} \log(1 - s_{q,j})$$

where J_{pos} is the set containing the indexes of the positive documents (i.e., relevant documents for the query q) and J_{neg} is the set containing the indexes of non-relevant documents. He et al. [30] showed how the selection of the set of negative examples J_{neg} has a major impact on the ability of the final model to filter relevant knowledge snippets. In particular, they designed a procedure to select J_{neg} based on four steps with increasing difficulty. The negative snippets were selected in order:

- Randomly: sampling non-relevant documents randomly among all the documents in the knowledge base;
- In-Domain: sampling non-relevant documents considering only the documents from the same domain of the positive document;

- In-Entity: sampling non-relevant documents considering only the documents from the same entity of the positive document;
- Cross-Entity: sampling non-relevant documents considering only the documents from entities previously mentioned in the dialogue context.

3.2.2 Representation-based methods

Representation-based methods focus on the construction of an informative vector space in which is performed the retrieval process. Given an input query q and a set of documents to retrieve, these are transformed into embedding vectors with a Sentence Encoder E. Then, the semantic similarity score s_{q,d_i} between the query q and a generic knowledge document k_i is computed simply considering some similarity measure S (e.g., cosine similarity) between the two representation in vector space:

$$s_{q,k_i} = S\Big(E(q), E(k_i)\Big)$$

Traditional approaches adopted BOW solutions using symbol based encoders and a similarity measure like the *cosine similarity* as scoring function S. An example still used is the Okapi BM25 [44] algorithm that uses a variation of the TF-IDF encoding. Mi et al. [29] adopted this method as one of an ensemble of five different ranking models to perform the Selection task in the DSTC9 - track 1.

However, recent studies in the context of Question-Answering showed how Pre-trained Language Models can be effectively used to compute continuous sentence embedding in order to perform information retrieval tasks. Sentence-BERT (SBERT) [45] proposed a Siamese Architecture (Fig. 3.3) composed by two separate embedding pipelines for the query and the documents, in which the two encoders in the architecture (e.g., BERT in the image) have tied weights.

This approach is commonly referred to as **Dense Knowledge Retrieval** [46] or **Dense Passage Retrieval** [36] due to the fact that the model maps all the textual information into dense embedding vectors, and then uses these to compute a similarity score.

In the context of the DSTC9 - track 1, Thulke et al. [46] performed a study aimed at proposing an efficient solution for the Selection task. In their work they proposed a Dense Knowledge Retrieval based method adopting a siamese architecture with two RoBERTa-large models as encoders. The main advantage of this approach is that all the knowledge snippets can be encoded offline before running the system, leaving to encode online only the dialogue input U_t. This means that one single passage through the Transformer encoder is needed in order to perform the inference, in contrast to the PRR methods that require as much as passages as the number of documents in the knowledge base. This approach

reduced the computing time up to more than 2000 times with respect to the original baseline.

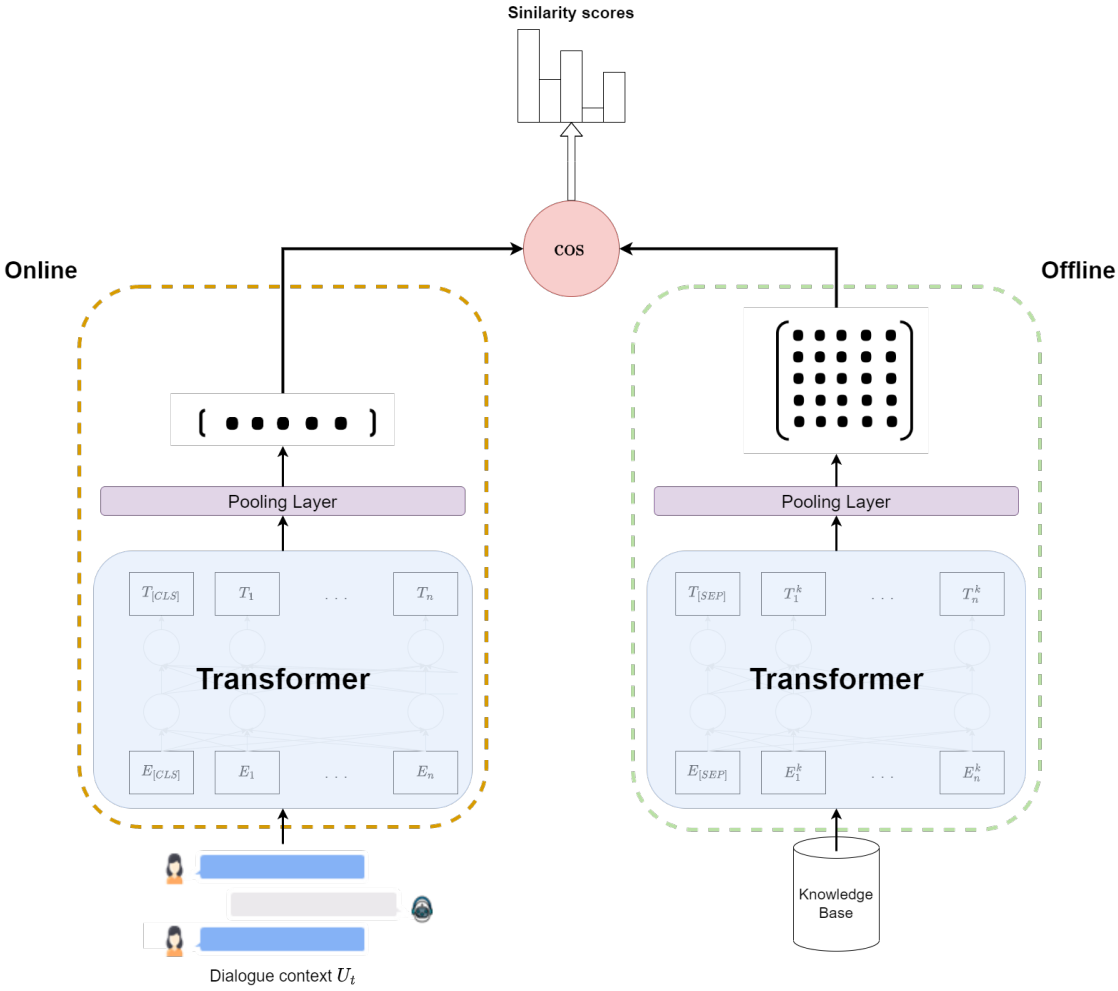

Figure 3.3: Architecture of Dense Knowledge Retrieval with a siamese network. The embedding of the documents in the knowledge base can be computed *offline* a-priori. At inference time (*online*) only the computation of the embedding vector for the dialogue context U_t is necessary.

Training procedure and Triplet Loss

During the fine-tuning phase of a siamese network, the goal is to update the weights of the encoder model such that the embedding vectors of similar texts are *closer* than the ones of dissimilar texts. This concept of closeness is related to the measure function adopted to score the similarity between the embedding vectors (e.g., the cosine similarity in Fig. 3.3). To perform this type of training is commonly adopted the **Triplet Loss** [47, 48]. Given a distance function D, the Triplet loss is a loss function that aims at modifying an embedding such that a reference input called *anchor* is closer to a matching input called *positive* and a far from a non-matching input called *negative* based on D. The general formula of the Triplet Loss can be written as:

$$\mathcal{L}(a, p, n) = \max\Big(D(a, p) - D(a, n) + \alpha, 0\Big)$$

The triplet loss ensures that the couple ap is closer with respect to the couple an with a margin α based on the distance function D (Fig. 3.4).

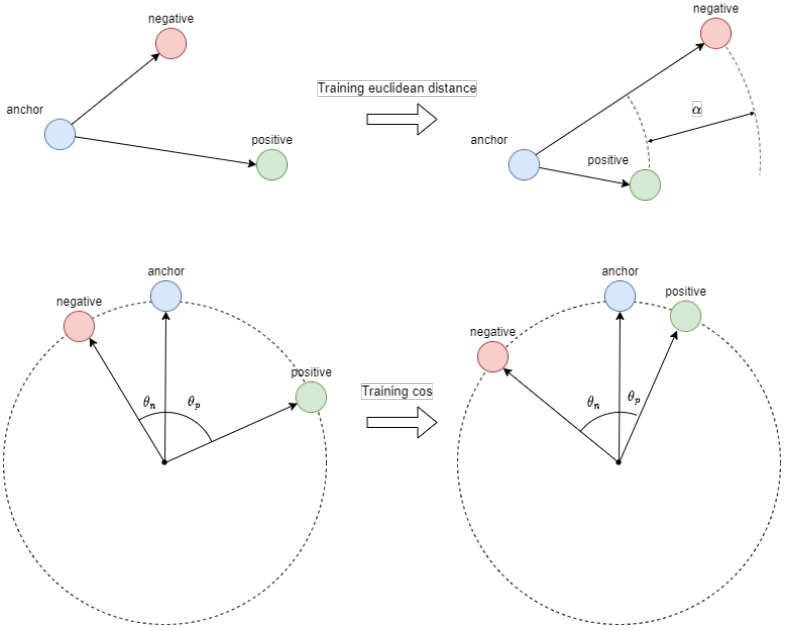

Figure 3.4: Example of the application of the triplet loss. After the training the representation space brings *closer* the positive vector to the anchor. On top is considered the euclidean distance while on the bottom the cosine distance.

3.3 Efficient Knowledge Selection

Although, PRR based methods can achieve impressive results in question answering tasks, they are also typically really expensive, both in terms of latency time and computational resources required (e.g., hardware required), making many of these models impractical especially under resource constraints.

A major problem with this approach is in fact that at inference time the model needs to check one by one all the knowledge snippets in the knowledge base $\mathcal{K}$ in order to retrieve which are the most relevant ones. This check is performed by computing the similarity score on the concatenation of the query q (e.g., the dialog context U_t in the DSTC9 challenge) and each possible knowledge snippet (e.g., each FAQ document in the DSTC9 challenge). The number of computations is therefore linearly dependent on the size of the knowledge base $|\mathcal{K}|$. Hence, also the time consumption for the retrieval scales linearly with the dimension of the KB becoming quickly intractable, in real-case scenarios. In addition, each single computation is not a unitary operation since it requires a full passage through an entire Transformer model involving many matrix multiplications. In the following are discussed some possible solutions to alleviate these problems.

3.3.1 Filtering the documents by keywords

In order to limit the number of comparisons that the PRR model needs to perform, different filtering strategies can be applied to the documents in the knowledge base. Typically these strategies aim at filtering a subset of possible candidates based on keywords extracted from the user query.

Different filtering strategies can be adopted. Considering the DSTC9 - track 1, Tan et al. [49] proposed a method named *Retrieve&Rank* that consists in a two-step approach: first entity tracking is performed to retrieve all the entities named in the dialog context that are then used to filter and select only related knowledge snippets, second the knowledge ranking task described above is performed. More fine-grained filters can be also defined.

3.3.2 Knowledge Distillation

In order to limit the memory requirements of the encoder models, one of the most promising approaches is the **Knowledge Distillation** (KD) process. KD is the process of learning a small model called *student* from a large model called *teacher*. Generally, the teacher model is a pre-trained model that is used to supervise the training phase of the student model. This process was formally popularized as knowledge distillation by Hinton et al. [50].

The key problem with KD is how to transfer knowledge from a large teacher model to a small student model. Basically, a knowledge distillation system is composed of three key components:

1. the **knowledge** that must be transferred from the teacher model to the student model;
2. the **distillation algorithm** adopted to perform the transfer;
3. the **teacher-student architecture** that defines the interaction methodology between the two models.

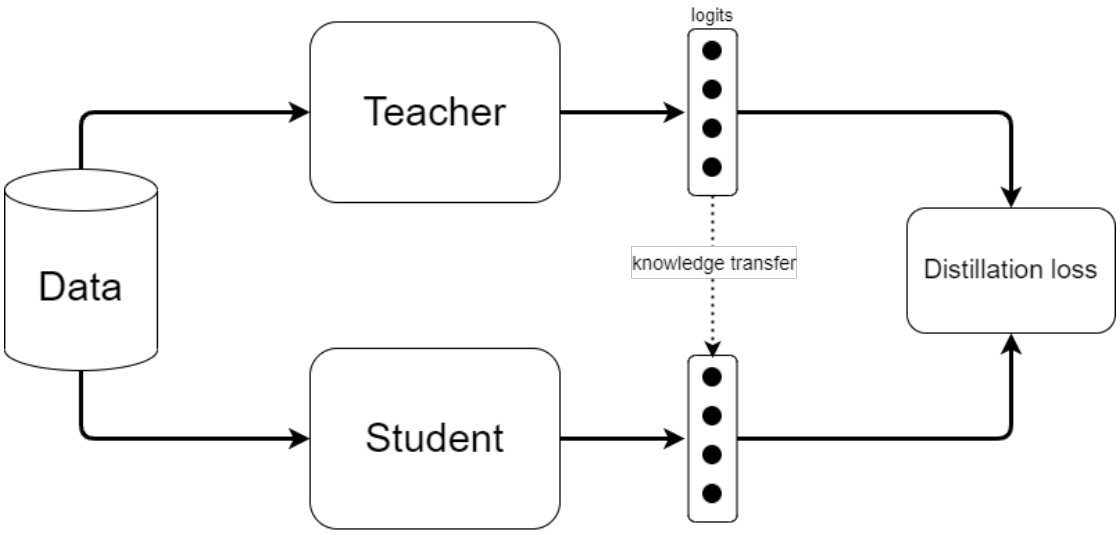

Figure 3.5: Example of the classic knowledge distillation framework in which the student model is trained to mimic the output of the teacher model.

Chapter 4

Method

Ranking methods based on Passage Re-Ranking achieved with transformer encoders achieved impressive results in many Q&A tasks [32]. However, their main limitation is that at inference time the final system must perform many singular comparisons. In particular, given a user query q the model must match the query with each single document present in the knowledge base in order to be able to create a ranking of all the documents based on their semantic relevance with respect to the given query. Therefore, the time complexity of the system grows linearly with the numbers of documents present in the knowledge base.

Moreover, the single comparison is not a simple arithmetical operation, but involves the concatenation of the query with the specific document and the passage through a Transformer model that requires many matrix multiplications. Given a knowledge base $\mathcal{K}$ the complexity of the Passage Re-Ranking strategy is $\mathcal{O}(T_E \cdot |\mathcal{K}|)$, where $|\mathcal{K}|$ is the dimension of the knowledge base and T_E indicates the time to perform a full passage through the Transformer encoder model E. This is a variable that takes into account the fact that the encoding of a sentence (or a pair of sentences as in the case of PRR) is not a simple unitary operation but it is union of many layered matrix multiplications whose number depends on the transformer adopted for the task. Obviously, the bigger is the transformer model (i.e., the higher is the number of attention layers) the more operations the model must perform in order to compute the final relevance score. With reference to the DSTC9 - track 1, the majority of the teams adopted strategies derived from the Passage Re-Ranking idea, using as base encoder huge transformer models such as PLATO-2 [33], a model whose architecture is composed by 32 transformer blocks and 32 attention heads with an hidden embedding space with a dimension of 2048. As the authors report, in order to use this model in inference it is required a 32GB NVIDIA Tesla

V100 GPU.[1]

In addition, many of the solutions proposed adopted ensembles composed by multiple transformer models in order to compute the final relevance score as a weighted average of the single scores or through majority voting strategies based on different parameters. For instance, He et al. [30] employed 5 different extra pre-trained models, including PLATO-2 both in the two versions with 32 layers and 24 layers, BERT-base [34] and ALBERT-xlarge [35]. Mi et al. [29], that were awarded as the winning team on the objective evaluation metrics, proposed an ensemble of 7 models composed by RoBERTa and more traditional models such as the BM25. If from one hand this approach can certainly offer a boost in accuracy of the retrieval system, on the other hand it is also generally not doable in a real application, where differently from a challenge framework, the limitations especially in terms of computational resources are one of the main concerns.

In this chapter is described the method developed, starting from the work of Thulke et al. [46], for the retrieval of the FAQ documents for the Selection task of the track 1 of the DSTC9. Firstly, in Sec. 4.1 are discussed some of the ideas that brought to the design of the Hierarchical Dense Knowledge Retrieval (HDKR) method. Then, in Sec. 4.2 is presented the HDKR method itself. Finally, in Sec. 4.3 are explained three extension to the base model that helped in achieving better performances. Namely, these are: the data augmentation through synthetic data (Sec. 4.3.1), the enriched domain embedding (Sec. 4.3.2) and the top-k entity filtering (Sec. 4.3.3).

4.1 Finding an efficient retrieval system

4.1.1 Advantages of Dense Knowledge Retrieval

Thulke et al. [46] focused their research effort aiming at developing an efficient solution for the retrieval of the knowledge documents for the selection task of the DSTC9 - track 1. As explained in their paper, the limited time that a conversational agent has to prepare an appropriate response sets tight limits on the time available for the retrieval system to find the correct document. To account for these constraints, they proposed a solution called Dense Knowledge Retrieval (DKR) with a Siamese network architecture. The main advantage of this solution with respect to the original Passage Re-Ranking strategy consists in the fact that it disjoins the embedding procedures for the knowledge documents and for the dialogue contexts into two separate and independent streams. This means that the

[1] https://github.com/PaddlePaddle/Knover/blob/develop/projects/PLATO-2/README.md

embedding procedure for the documents collected in the knowledge base can be performed *offline*, before the conversational agent goes *online* and start to receive requests from the users.

The method developed in this work follows the DKR strategy. However, differently from the original DKR method that adopted a siamese network architecture, in order to diminish the requirements in terms of memory consumption, a single encoder model was considered for both the embedding of the documents in the knowledge base and the dialogue contexts (Fig. 4.1).

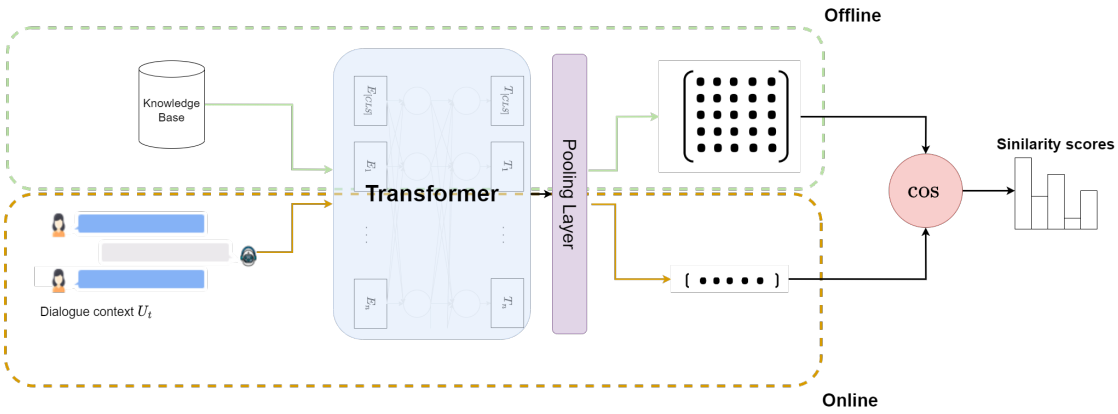

Figure 4.1: Dense Knowledge Retrieval with a single encoder model. A single encoder is used to compute the embedding of the documents in the knowledge base and of the dialogue context independently.

In this case, given a Transformer encoder model E, when the conversational agent is *online* and receives a dialog context U_t where the last user request u_t is a knowledge-seeking turn, in order to perform the retrieval task the agent needs only to vectorize the dialogue context itself and perform a similarity score between the embedding vector of the dialogue context and the embedding matrix of all the knowledge documents. This means that the time complexity for the inference is not bind to the dimension of the knowledge base since a single full passage through the encoder E is needed. Hence, the time complexity is:

$$\mathcal{O}\big(T_E + T_S\big)$$

where T_E is the time for the encoding of the dialogue context that depends on the encoder E and T_S is the time to compute the similarity score S (e.g., the cosine similarity in Fig. 4.1). Considering that the encoding procedure involves many matrix multiplication while the scoring function is typically a single matrix operation (e.g., the dot product), is it possible to bound the time complexity of the inference for DKR as:

$$\mathcal{O}\Big(T_E + T_S\Big) < \mathcal{O}\Big(2T_E\Big)$$

4.1.2 Problems with DKR for the DSTC9

Obviously the DKR approach offers a huge computational advantage with respect to the Passage Re-Ranking strategy. However, DKR is a representation retrieval method based on the similarity between the embedded vectors of the dialogue context and of the knowledge documents. This means that the final scores on which is performed the ranking of the documents depend simply on the semantic similarity encoded in the embedding representation. This method suits very well general open-domain question-answering tasks, where the request is to retrieve generic similar documents given an input query. However, this framework does not entirely fit the DSTC9 - track 1 Selection task. In this challenge in fact, it is asked to face an information retrieval task where the collection of documents is composed of FAQ knowledge snippets from several entities spanning from different domains. Despite the fact that there exist a hierarchy among the documents, there is not a clearly defined taxonomy among them. This means that there is an overlapping between FAQs documents belonging to different domains or even to different entities within the same domain. For instance, different domains can have similar FAQ documents. Consider the request *"Can I bring my dog?"*, this could be asked with reference to the domain of hotel or restaurant, but even regarding the domain of train and attraction (Fig. 4.2).

Domain	Entity Name	FAQ Document
hotel	A and B Guest House	**Q:** Can I bring my pet to A and B Guest House? **A:** No, pets are not allowed at this property.
restaurant	Burma Love	**Q:** Can you bring in your pet? **A:** Animals are not allowed at Burma Love.
train	*	**Q:** Can I bring my pets on board? **A:** You may bring 2 domestic pets, they must be kept on a short lead, or carried in a basket size 85x60x60cm.
attraction	Cable Car Museum	**Q:** Can I bring my dog to Cable Car Museum? **A:** Sorry, you're not allowed to bring a pet to the museum.

Figure 4.2: Example of similar FAQ documents belonging to different domains. The documents are taken from the knowledge base of the DSTC9 - track 1.

Going forward, the FAQ documents consider requests that are *frequent* and so are general requests commonly not tied to a specific entity. For this reason, it is very likely that FAQs knowledge snippet taken from different entities in the same domain contains similar requests (Fig. 4.3).

Domain	Entity Name	FAQ Document
hotel	A and B Guest House	**Q:** Can I bring my pet to A and B Guest House? **A:** No, pets are not allowed at this property.
hotel	ACORN GUEST HOUSE	**Q:** Are pets welcomed at this property? **A:** People cannot bring pets to Acorn Guest House.
hotel	A and B Guest House	**Q:** What time is check-in there? **A:** Check-in time is from 3:30pm - 9:00pm.
hotel	ACORN GUEST HOUSE	**Q:** What time is check in for Acorn Guest House? **A:** Check in is between 2:00pm and 10:00pm.
hotel	A and B Guest House	**Q:** Is there wifi available? **A:** There is free wifi available.
hotel	ACORN GUEST HOUSE	**Q:** Is there free wifi at Acorn Guest House? **A:** Acorn guest house has free WiFi.

Figure 4.3: Example of three couples of similar FAQ documents sampled from two entities in the same domain. The documents are taken from the knowledge base of the DSTC9 - track 1,

This can be a problem, especially under the constraints of a task-oriented setting such as in the DSTC9. In this case in fact, the user is not satisfied simply with a general similar document but given a question about a domain and a specific entity, her or him requires exactly the precise document that refers to the named domain and entity and that answers the request stated. However, considering all the documents together, different documents belonging to different domains or entities will be grouped based only on their semantic content making more ambiguous the retrieval. In Fig. 4.6 it is plotted a two dimensional representation of the embedding vectors of all the FAQ knowledge documents within the domain 'hotel'. The documents have been encoded using the transofrmer encoder all-MiniLM-L6-v2, a transformer model pretrained on open-domain Q&A tasks.[2] The representation is obtained reducing the dimensionality of the data with the t-SNE algorithm [51]. Unsurprisingly, it is possible to observe from the picture how the documents are grouped based only on the semantic content of the FAQ knowledge snippet and independently on the entity they belong to. This means that, given an input query it embedding vector will be end up in a very fit clustr containing a lot of similar knowledge documents form different entities, making more difficult the retrieval of the correct knowledge document from the correct entity.

[2]https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

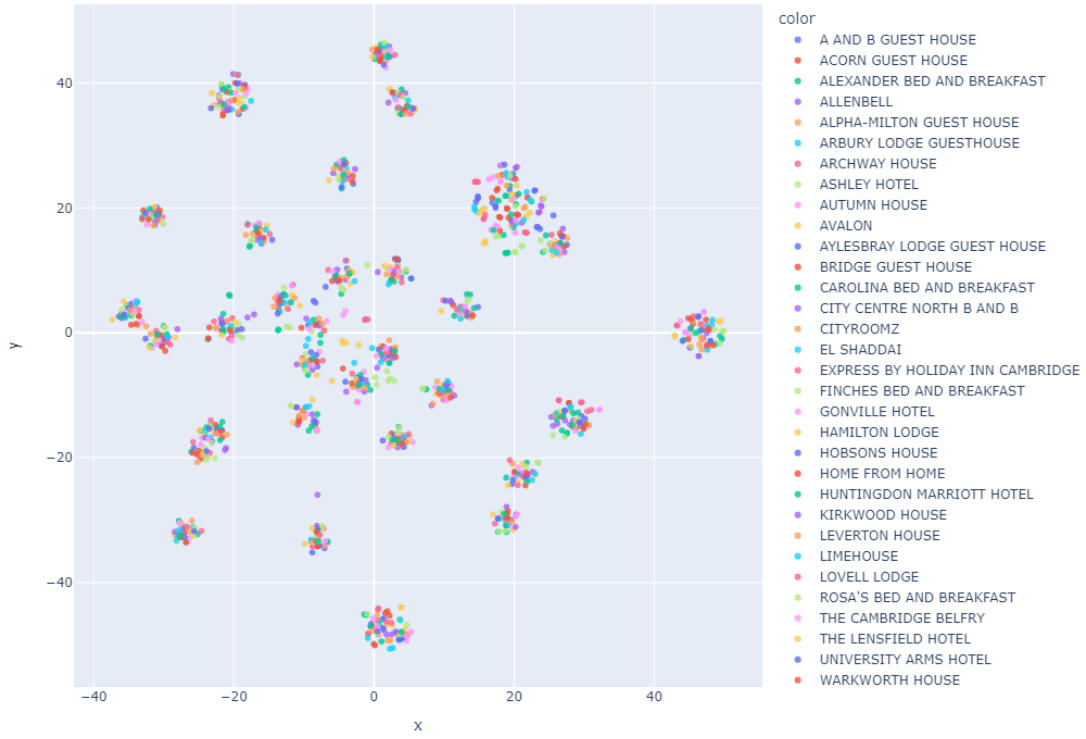

Figure 4.4: Two-dimensional representation of the embedding vectors of the FAQ knowledge documents in the domain hotel obtained through the t-SNE algorithm. From this image it is possible to see clearly how the embedding vector space does not consider the entity. This results in clusters of similar FAQ documents from different entities.

4.1.3 Exploiting the hierarchical information

To help the dense retrieval process is it possible to exploit the hierarchical information attached to each FAQ knowledge snippet. Indeed, some proposals in this sense have submitted also during the DSTC9 [46, 30]. However, the adopted strategy was always based on a PRR strategy considering three pipelined re-ranking or classification tasks: the first one for the domain, the second one for the entity and the final one for the knowledge snippet.

Consider the set of all the domains $\mathcal{D}$:

$$\mathcal{D} = \{\text{hotel}, \text{restaurant}, \text{train}, \text{taxi}, \text{attraction}\};$$

than consider $\mathcal{E}$ the set of all the entities in the knowledge base:

$$\mathcal{E} = \bigcup_{d \in \mathcal{D}} \mathcal{E}_d$$

where $\mathcal{E}_d$ is the set of all the entities belonging to the domain $d \in \mathcal{D}$. Finally consider the knowledge base $\mathcal{K}$ containing the set of all the FAQ knowledge documents k_i as the union of the FAQ knowledge documents all the entities in all the domains:

$$\mathcal{K} = \bigcup_{d \in \mathcal{D}} \bigcup_{e \in \mathcal{E}_d} \mathcal{K}_e$$

where $\mathcal{K}_e$ is the set of all the FAQ knowledge documents belonging to entity e:

$$\mathcal{K}_e = \Big\{ (\mathit{domain}, \mathit{entity}, \mathit{FAQ\ document}) \in \mathcal{K} : \mathit{entity} = e \Big\}$$

Given three separate PRR encoders, E_d for the domain, E_e for the entity and E_k for the knowledge snippet, the final time complexity of the hierarchical PRR strategy can be estimated as:

$$\mathcal{O}\Big(T_{E_d} |\mathcal{D}| + T_{E_e} \frac{|\mathcal{E}|}{|\mathcal{D}|} + T_{E_k} \frac{|\mathcal{K}|}{|\mathcal{E}|} \Big)$$

where T_{E_i} indicates the inference time for the encoder E_i, $\frac{|\mathcal{E}|}{|\mathcal{D}|}$ indicates the average number of entities for each domain and $\frac{|\mathcal{K}|}{|\mathcal{E}|}$ indicates the average number of FAQ knowledge documents for each entity.

Even if this approach can alleviate the scalability problem associated with the original PRR strategy, it does not solve the problem entirely remaining dependent on the ratio of knowledge documents per each entity and especially on the average number of entities per each domain (e.g., there can be many hotels in a single knowledge base). Thulke et al. [46] obtained their best result with a hierarchical PRR model showing how this strategy can reduce the inference time for the knowledge retrieval from around 276.53 seconds for the baseline model to 13.79 seconds, remaining still not doable for a real case scenario.

4.2 Hierarchical Dense Knowledge Retrieval

The ***Hierarchical Dense Knowledge Retrieval (HDKR)*** method developed for this work aims at providing an efficient knowledge retrieval system while maintaining high levels of accuracy. To do this, the HDKR method maintains the computational advantages provided by the DKR strategy one and improves the accuracy of the retrieval system through the exploitation of the hierarchical information of the FAQ documents present in the knowledge base. However, differently from the hierarchical PRR strategy, this method is based on three sequential Dense Retrieval tasks:

1. *Dense Domain Retrieval*: given a dialogue context U the domain of the dialogue context d_U is retrieved among the set of all the domains $\mathcal{D}$ present in the knowledge base;

2. *Dense Entity Retrieval*: given a dialogue context U and its domain d_U, the entity $e_U \in \mathcal{E}_{d_U}$ at which the dialogue context U is referring to is retrieved among the set $\mathcal{E}_{d_U}$ of all the entities in the domain d_U;

3. *Dense Knowledge Retrieval*: given a dialogue context U, its domain d_U and its entity e_U, the final document k_U at which the dialogue context is referring to is retrieved among the set $\mathcal{K}_{e_U}$ of all the FAQ documents belonging to the entity $e_U \in \mathcal{E}_{d_U}$.

Moreover, in order to reduce the memory usage, instead of siamese networks as in the architecture proposed by Thulke et al. [46], a single small transformer model is adopted for each dense retrieval task. In total three Transformer encoder models are needed for the HDKR: E_{domain}, E_{entity} and $E_{knowledge}$.

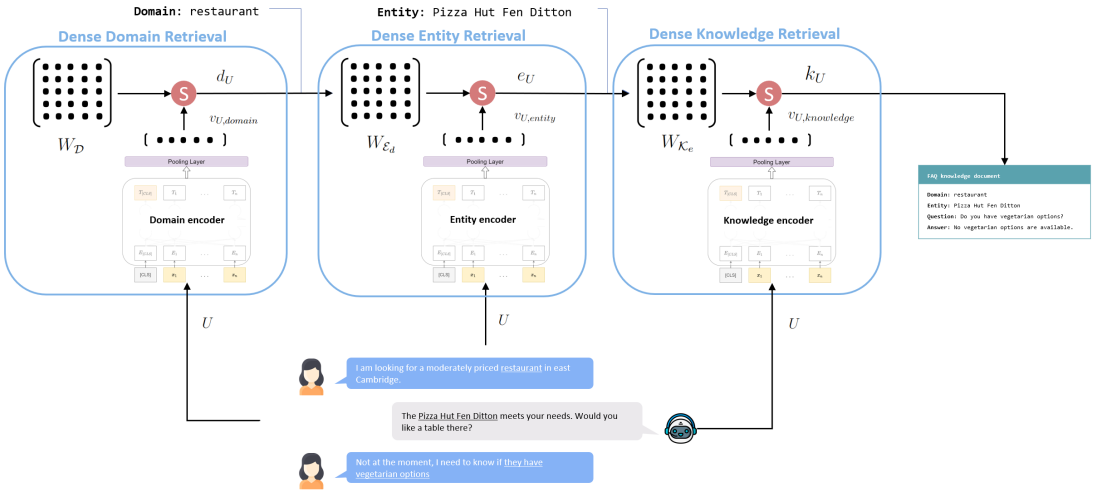

Figure 4.5: Architecture of the Hierarchical Dense Knowledge Retrieval. The final document is retrieved after three sequential Dense Retrieval tasks in which are respectively retrieved the domain d_U, the entity e_U and the FAQ knowledge snippet k_U given a dialogue context U.

Dense Domain Retrieval (DDR): In the first step is performed the dense retrieval of the domain. A domain encoder transformer model E_{domain} is trained to bring closer the representation of a dialogue context U and the representation of the corresponding domain $d_U \in \mathcal{D}$. Offline, before inference, the domain encoder E_{domain} is used to obtained the embedding vectors of all the domains. For instance, given the domain *hotel*, its embedding is computed performing the encoding of the word 'hotel' with the domain encoder E_{domain}:

$$w_{hotel} = E_{domain}(\text{"}hotel\text{"})$$

All the domain embedding are grouped into a domain embedding matrix $W_{\mathcal{D}}$:

$$W_{\mathcal{D}} = E_{domain}(\mathcal{D})$$

At inference time the model E_{domain} is used to convert a given dialogue context U into an embedding vector $v_{U,domain}$:

$$v_{U,domain} = E_{domain}(U)$$

Then, the domain of the dialogue context d_U is retrieved computing a scoring function S between the encoded dialogue context $v_{U,domain}$ and the domain embedding matrix $W_{\mathcal{D}}$ and taking the domain corresponding to the highest score:

$$d_U = \arg\max_{d \in \mathcal{D}} \Big(S\big(v_{U,domain}, W_{\mathcal{D}}\big) \Big)$$

Dense Entity Retrieval (DER): Similarly, in the second step, given the same dialogue context U and its domain d_U, the entity to which it belongs is retrieved among the set of entities $\mathcal{E}_{d_U}$. First, as in the previous case, the encoder E_{entity} is used offline to retrieve the entity embedding matrices $W_{\mathcal{E}_d}$ of the set of entities $\mathcal{E}_d$ for all the domains $d \in \mathcal{D}$ in the KB:

$$W_{\mathcal{E}_d} = E_{entity}(\mathcal{E}_d) \quad \forall d \in \mathcal{D}$$

Each entity embedding vector is calculated as the encoding of the entity name with the entity Transformer encoder E_{entity}.Then, at inference the dialogue context is encoded with the encoder model E_{entity} in order to obtain another embedding vector $v_{U,entity}$:

$$v_{U,entity} = E_{entity}(U)$$

Finally the entity e_U of the given dialogue context U is retrieved among the set $\mathcal{E}_{d_U}$ of all the entities in the domain d_U, as the one that maximize the scoring function S:

$$e_U = \arg\max_{e \in \mathcal{E}_{d_U}} \Big(S\big(v_{U,entity}, W_{\mathcal{E}_{d_U}}\big) \Big)$$

Dense Knowledge Retrieval (DKR): Finally, in the last step, given the dialogue context U, its domain d_U and the entity which it is referring to e_U, the FAQ document answering the user request is retrieved considering only the set of documents $\mathcal{K}_{e_U}$ from the entity $e_U \in \mathcal{E}_{d_U}$. First, the encoder model $E_{knowledge}$ is used offline to compute a knowledge embedding matrix for the set of FAQ knowledge documents for each entity in the knowledge base:

$$W_{\mathcal{K}_e} = E_{knowledge}(\mathcal{K}_e) \quad \forall e \in \mathcal{E}$$

Then, at inference, a third embedding vector $v_{U,knowledge}$ for the dialogue context U is obtained with the knowledge encoder model $E_{knowledge}$:

$$v_{U,knowledge} = E_{knowledge}(U)$$

Finally the FAQ knowledge snippet k_U corresponding to the request made by the user in the last utterance of the dialogue context is retrieved among the set $\mathcal{K}_{e_U}$ of all the FAQ knowledge snippets from the entity e_U. The final snippet is the one that maximize the scoring function S:

$$k_U = \underset{k \in \mathcal{K}_{e_U}}{\arg\max} \left(S\Big(v_{U,knowledge}, W_{\mathcal{K}_{e_U}}\Big)\right)$$

4.2.1 Time complexity

Similarly to the original DKR method, the Hierarchical Dense Knowledge Retrieval has the advantage that the computation of the matrix embedding for the domain, the entities and the knowledge documents can be computed one single time apriori for all the dialogue contexts. For this reason, the inference time of the HDKR method depends only on the time needed to perform the three vectorization $v_{U,domain}$, $v_{U,entity}$, and $v_{U,knowledge}$. Considering the three encoders E_{domain}, E_{entity}, and $E_{knowledge}$, the time complexity for the inference is:

$$\mathcal{O}(T_{E_{domain}} + T_{E_{entity}} + T_{E_{knowledge}} + 3T_S)$$

It is worth to notice how the computational time T_E required for to retrieve the sentence embedding with an encoder model E can vary a lot based on the Transformer encoder model. The adoption of smaller models or knowledge distilled models can help in reducing the computational time with respect to the base DKR method, as it will be showed in Sec. 6. Moreover, differently from the original DKR method, where the time T_S required for the computation of the similarity score considered a single embedding matrix with $|\mathcal{K}|$ embedding vectors (i.e., 12039 FAQ knowledge documents); in the HDKR three similarity scores are computed considering three smaller matrices W_D with $|\mathcal{D}|$ embedding vectors (e.g., 5 domains), $W_{\mathcal{E}_d}$ with an average of $\frac{|\mathcal{E}|}{|\mathcal{D}|}$ embedding vectors (i.e., 133.6 average entities for each domain) and finally $W_{\mathcal{K}_e}$ with an average of $\frac{|\mathcal{K}|}{|\mathcal{E}|}$ embedding vectors (i.e., 18.02 average documents for each entity). Hence, the decomposition of the computation of the similarity score helped to reduce the total number of comparisons with respect to the original DKR method.

4.3 Extension

In this section are explained three extension applied to the base HDKR model presented in the previous section. In Sec. 4.3.1 is discussed the procedure of data augmentation performed with synthetic data generated starting from the knowledge base, in Sec. 4.3.2 is discussed an alternative semantically enriched representation for the Dense Domain Retrieval and finally in Sec. 4.3.3 is explained a strategy adopted to take in consideration different entities with similar names from the same domain.

4.3.1 Synthetic data augmentation

Regarding task-oriented KCAs it is critical to develop robust models able to generalize to unseen domains. At the moment, task-oriented KCAs are limited in scope to only a small subset of domains. However, the hope in the future is to be able to generate models that can generalize their assisting capabilities to all the domains offering an open-domain frame of assistance [17]. This issue was taken into consideration also for the DSTC9 - track 1 where the domain of *attraction*, unseen during the training phase, was added in the evaluation phase of the challenge. To do this, new labeled dialogues contexts were considered and the knowledge base was extended with new knowledge snippets. However, there is an important distinction to do between the labeled dialogue contexts and the FAQ knowledge documents in the knowledge base. The final KB in fact was provided before the closure of the challenge. Indeed, the knowledge base is not really a part of the dataset. As a matter of fact, it does not contain any labeled data since it is from the FAQ pages containing factual knowledge provided by the owners of the various entities.

Imagine a real-case scenario, with a frictionless task-oriented KCA able to operate on a specific set of domains. Consider the case in which the owners of the KCA want to extend its capabilities to a new domain like for example *attraction*. The first thing that they have to do is to consider a new set of entities $\mathcal{E}_{attraction}$ belonging to this domain and to extend the knowledge base with an additional set of FAQs documents, collected by scraping the pages of the entities selected or alternatively allowing to the entities to be added to the service providing their set of FAQ knowledge snippets. Then to broaden the capabilities of the model to the new unseen domain it is necessary to extend the three set of domains, entities and knowledge as:

$$\tilde{\mathcal{D}} = \mathcal{D} \cup \{attraction\}$$
$$\tilde{\mathcal{E}} = \mathcal{E} \cup \mathcal{E}_{attraction}$$
$$\tilde{\mathcal{K}} = \mathcal{K} \cup \mathcal{K}_e \ \ \forall\, e \in \mathcal{E}_{attraction}$$

and compute the corresponding embedding considering the previously mentioned three encoder models: E_{domain}, E_{entity} and $E_{knowledge}$.

However, in this manner we are not completely exploiting all the information present in this new collected data. This data in fact is *factual knowledge* created directly from the various owners of the entities that does not require any labeling. This data contains various typical requests and already carry the information about the belonging domain and entity. In this sense these knowledge snippets can be exploited to create ***synthetic dialogues*** composed by a single requests with attached the label of the corresponding domain. This synthetic data then can be used to train the domain encoder model E_{domain} to bring closer the typical requests

asked for the new domain that we want to add to the service and the encoding of the label itself.

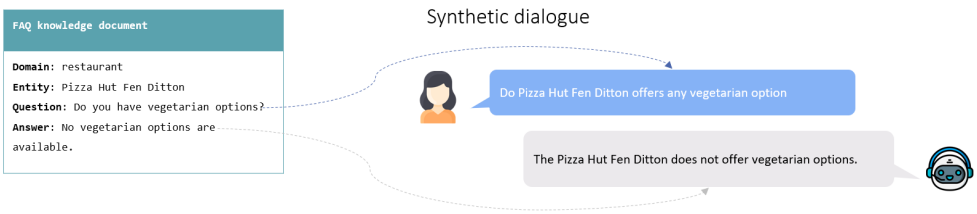

Figure 4.6: Example of a synthetic dialogue generated starting from a FAQ knowledge document.

4.3.2 Enriched domain representation

The task-oriented environment of the DSTC9 - track 1 adds a layer of difficulty to the retrieval task. Differently from open-domain settings, where a general similar document may be enough, in this setting it is crucial to retrieve the exact knowledge document belonging to the correct domain at which the dialogue context is referring to, regarding the specific named entity and concerning the exact document answering the user request.

The hierarchical structure of the architecture proposed in Sec. 4.2, imposes rigid standards of accuracy on the domain and entity retrieval. Due to the pipelined structure of the HDKR method in fact, the error propagation of the model could drastically reduce the retrieval capabilities of the system. With respect to the simple Dense Knowledge Retrieval approach, where all the documents $\mathcal{K}$ were firstly converted into a single embedding matrix $W_{\mathcal{K}}$ that were than scored against the vector embedding of the dialogue context; in the Hierarchical Dense Knowledge Retrieval, the final document selection is performed only on the subset $\mathcal{K}_{e_U}$ of documents resulting from the retrieval of the domain d_U of the dialogue context and the specific entity $e_U \in E_{d_U}$. For this reason, for a wrong retrieval of domain or entity, the final ranking of documents would it be completely wrong, with a consequential contribution to the final selection score in R@1, R@5 and MRR always equal to 0.

Generally speaking, given a query and a corpus of documents, Dense Retrieval systems can be distinguished into two categories:

- Symmetrical: where the query and the documents in the knowledge base are of about the same length;

- Asymmetrical: where the query is shorter than the documents in the knowledge base (e.g, finding related articles from a keyword);

However, in the Dense Domain Retrieval the situation is reversed compared to the typical asymmetrical setting. In fact, the query consists of a dialogue context that can be composed by many utterances forming a long snippet of text, while the documents to retrieve in this case consist simply of the domain labels: 'hotel', 'restaurant', 'taxi', 'train' and 'attraction'.

However, the single domain label (e.g., 'train') carries very low semantic information making more difficult the retrieval in the embedded space.

To solve this problem a ***semantically enriched representation*** of the domains has been designed. The idea is simple; given a domain $d \in \mathcal{D}$, the domain embedding vector w_d is computed starting from the whole set of FAQ knowledge documents $\mathcal{K}_d = \bigcup_{e \in \mathcal{E}_d} \mathcal{K}_e$ belonging to the domain d. This set is used to create the cluster $\mathcal{C}_d$ composed by the embedding vectors of all the FAQ knowledge documents belonging to the domain d:

$$\mathcal{C}_d = \left\{ c_i = E_{domain}(k) \; \forall \; k \in \mathcal{K}_d \right\}$$

Starting from this cluster then, the embedding vector w_d can be computed as a *representative object* of the cluster $\mathcal{C}_d$, such as the *centroid* or the *medoid*:

$$w_{d,centroid} = \frac{\sum_{c \in \mathcal{C}_d} c}{|\mathcal{C}_d|}$$

$$w_{d,medoid} = \arg\min_{c \in \mathcal{C}_d} \sum_{i=i}^{|\mathcal{C}_d|} \|c - c_i\|^2$$

where $|\mathcal{C}_d|$ corresponds to the number of points in the cluster.

The main advantage of this approach is that instead of considering the domain representation simply as the encoding of a single word it is considered a summary of all the documents belonging to it. In this sense, for instance the embedded representation of the domain *'hotel'* would be a semantic average of all the general requests (i.e., FAQs) provided by the entities in the domain 'hotel'. This method helps the domain retrieval procedure, providing semantically enriched embedded representation of the domain w_d.

4.3.3 Top-k entity filtering

For the reasons explained in the previous section, also the Dense Entity Retrieval system has to withstand strict accuracy requirements. The task-oriented setting in fact, requires to retrieve the exact named entity. However, investigating further the names of the entities present in the knowledge base, it is possible to observe how a lot of entities have close names (Fig. 4.7). This makes the Dense Entity Retrieval task ambiguous in many cases.

Entity 1	Entity 2	Sentence Similarity
Orchard Garden Hotel	Orchard Hotel	0.932
Hotel Zoe Fisherman's Wharf	Holiday Inn Fisherman's Wharf	0.803
Chancellor Hotel on Union Square	Fitzgerald Hotel Union Square	0.803
YIPPEE NOODLE BAR	JINLING NOODLE BAR	0.763

Figure 4.7: Example of similar entity names. The Sentence Similarity scores have been computed with the Transformer model all-mpnet-base-v2 from Sentence Transformers.[3]

Considering only the top-1 entity that maximizes the scoring $S(E_{entity}(U), W_{\mathcal{K}_{e_U}})$ can be a risk, because if the model wrongly retrieves an entity with a very close name to the one mentioned in the dialogue context, but that is not the correct one, the final knowledge snippets retrieved would be completely wrong anyway.

To try to alleviate this problem, a softer approach based on a top-k strategy has been taken. Instead of considering only the entity that maximizes the scoring function, it is considered the probability distribution for each entity:

$$\mathbf{p}_e = \text{soft}\max\left(S(v_e, W_{\mathcal{E}_{d_U}})\right)$$

where $v_e = E_{entity}(U)$ is the vector embedding of the dialogue context U obtained with the entity encoding model E_{entity}.

In order to decide the number k of entities to retrieve, it has been performed an analysis of the probability distributions obtained with various dialogue context. In Fig. 4.8 it is plotted the average probability distribution for the first top-5 entity names. As it can be seen there is clear disproportion of probability between the top-1 ranked entity name and the other positions. Nevertheless, also the entities in second and third position have a not negligible probability. A cut-off of 95% of probability corresponding to the **top-3** entities have been considered.

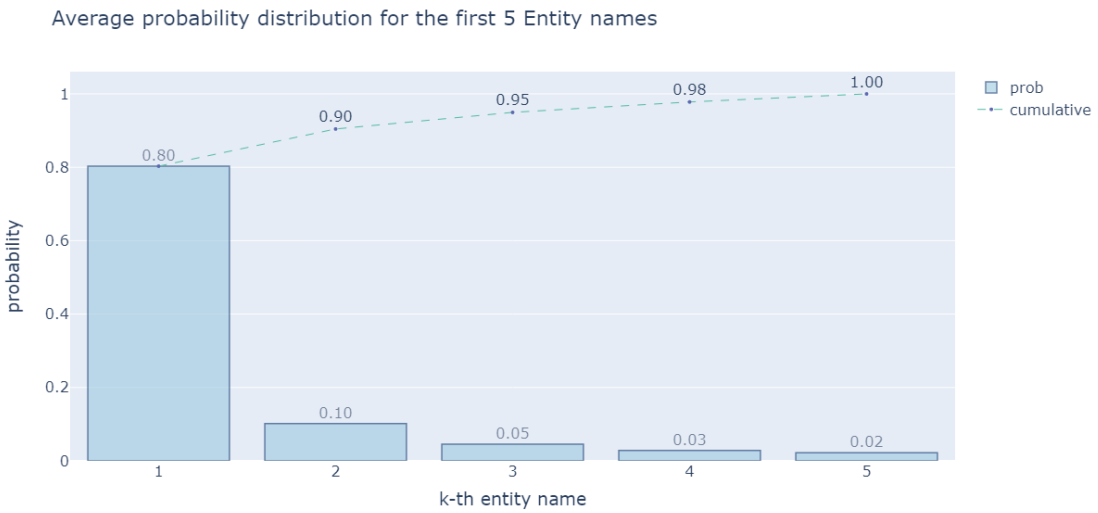

Figure 4.8: Plot of the average probability distribution of the first 5 Entity names obtained with the DER.

Chapter 5

Experimental Setup

In this chapter is delineated the experiment framework adopted to test the *Hierarchical Dense Knowledge Retrieval* system explained in the previous chapter, in the context of the Selection subtask of the DSTC9 - track 1.

The HDKR system consists of three sub-systems corresponding to the three Dense Retrieval tasks of: dense domain retrieval, dense entity retrieval and dense knowledge retrieval. These three sub-systems have been developed and trained separately on all the dialogues and evaluated sequentially. In the following section are discussed the details of the systems.

5.1 Dense Domain Retrieval

The Dense Domain Retrieval (DDR) constitutes the first model in the pipeline architecture. Given in input a dialogue context $U = \{u_1, \ldots, u_n\}$ where the last utterance u_n is a user request, the DDR tackles the task of retrieving in a dense embedding space the domain at which the utterance u_n is referring to.

Encoder. For the task of dense domain retrieval it has been used the transformer encoder model all-MiniLM-L6-v2 from Sentence Transformer.[1] This transformer model is obtained by finetuning the model MiniLM-L6-H384-uncased that is a 6 layers version realized by keeping only every second layer from the transformer encoder model MiniLM-L12-H384-uncased developed by Microsoft [52]. This model is obtained through knowledge distillation process from the BERT-base Transformer model [34]. The student model is trained by distilling the self-attention module of the last Transformer layer of the teacher model.

[1] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

All-MiniLM-L6-v2 has been finetuned on 1B sentence pairs with a contrastive learning objective in order to pair related sentences. As the authors report, this model "maps sentences and paragraphs to a 384 dimensional dense vector space and can be used for tasks like clustering or semantic search".

Embedding and scoring. The embedding vector $v_{U,d}$ of the dialogue context $U = \{u_1, \ldots, u_n\}$ is obtained by concatenating all the utterances and truncating the string to the last 256 tokens (i.e., the maximum amount of tokens that the model can handle).

$$v_{U,domain} = E_{entity}(< u_n >< u_{n-1} > ... < u_1 > |_{256})$$

Reversing the order of the utterances showed an improvement in performances. A possible explanation to this behavior can be found in the positional bias present in many famous Q&A datasets adopted to train these sentence embedding Transformers. For instance, Hofstätter et al. [53] showed how two widely adopted datasets MS MARCO and SQuAD 2.0 tend to strongly favor earlier positions in the paragraphs causing the final model to pay more attention at the first positions of the text. Due to the structure of the task, the domain to retrieve is more probable that has been mentioned in the last user utterances than in the first ones, for this reason moving these to the first positions can assist the model in finding the correct answer.

For this system different extensions have been tested. First of all three different domain embedding strategies have been considered. In the *base* model the domain embedding are computed using simply the labels of the domain. For example, the embedding vector for the domain 'hotel' is obtained simply as:

$$w_{hotel} = E_{domain}('hotel')$$

Then, two alternative embedding strategies have been considered computing the embedding vector of a domain d as the *centroid* or the *medoid* of the corresponding cluster $\mathcal{C}_d$ that contains all the FAQ knowledge snippets from all the entities in the domain d. For each knowledge snippet the corresponding point in the cluster is computed as the encoding of the concatenation of all the information in the knowledge snippet k:

$$\mathcal{C}_d = \left\{ E_{domain}\Big(< d >< e >< q >< a > |_{256}) \; \forall \; e \in \mathcal{E}_d \; \forall \; k \in \mathcal{K}_e \right\}$$

where d is the domain, e is the entity name, q is the question and a is the answer. The **cosine similarity** has been used as scoring function S to compute the relevance score $s_{U,d}$ between a dialog context U and a domain d:

$$s_{U,d} = \cos(v_{U,domain}, w_d) = \frac{v_{U,domain} \cdot w_d}{\|v_{U,domain}\| \|w_d\|}$$

Training. Each model is trained for four epochs. For each epoch 2000 triplets are randomly sampled from the dialogs present in the training set. Each triplet is of the type (U, d^+, d^-) where d^+ represent the correct domain of the dialogue context, and d^- a wrong domain. A cosine distance function has been considered for the triplet loss:

$$D_{\cos}(v_{U,domain}, w_d) = 1 - \cos(v_{U,domain}, w_d) \in [0, 2]$$

$D_{\cos}$ ranges between 0 in case the two vectors are parallel and facing the same direction, and 2 in case the two vectors are parallel and facing opposite directions. The encoder model is finetuned with the triplet loss with the cosine distance function:

$$\mathcal{L}_{tri,domain}(U, d^+, d^-) = \max\Big(D_{\cos}(v_{U,domain}, w_{d^+}) - D_{\cos}(v_{U,domain}, w_{d^-}) + \alpha, 0\Big)$$

the parameters α indicates the margin and is set to 0.25. AdamW is used as optimizer with a learning rate of $2e-5$.

An alternative training procedure has been tested augmenting for each epoch the 2000 original samples with 1000 new synthetic triplets randomly sampled from the knowledge base. For instance, given a knowledge snippet k the corresponding triplet was formed as $(< q >< a >, d^+, d^-)$, where the concatenation of the question and the answer $< q >< a >$ was used as anchor, the positive example d^+ was the correct domain of k and the negative example d^- was a randomly sampled domain from $\mathcal{D} \setminus \{d^+\}$.

5.2 Dense Entity Retrieval

The Dense Entity Retrieval (DER) represents the second step of the retrieval process. Given in input a dialogue context $U = \{u_1, \ldots, u_n\}$ where the last utterance u_n is a user request and the corresponding retrieved domain d, the DER tackles the task of retrieving in a dense embedding space the entity name at which the last user utterance u_n is referring to.

Encoder. To perform the DER it has been used as entity encoder E_{entity} the Transformer encoder model MPNet from sentence transformers.[2] MPNet stands for Masked Permuted Network and is a model developed by Microsoft by inheriting the advantages of both BERT [34] and XLNet [41]. This model maps a sentence into

[2]https://huggingface.co/sentence-transformers/all-mpnet-base-v2

a 768-dimensional dense vector space and it is designed to be used for clustering tasks or semantic search tasks.

Embedding and scoring. The embedding of the dialog context is performed as in the domain case. Given a dialogue context U, the utterances are concatenated in reverse order and passed to the entity encoder in order to obtain the dialogue context embedded vector into the entity vector space:

$$v_{U,entity} = E_{entity}(< u_n >< u_{n-1} > ... < u_1 > |_{384})$$

By default, input text longer than 384 word pieces is truncated. As explained in the previous chapter, differently from the domain in this case the clustering strategy was not applicable. For this reason the entity embedding w_e is obtained simply as the encoding of the domain name. For example given an entity e whose name is *"Hilton hotel"* its embedding in the entity vector space is obtained as:

$$w_{Hilton\ hotel} = E_{entity}(\text{“}Hilton\ hotel\text{”})$$

Training. The DER model has been trained in two steps for a total of six epochs following a curriculum learning approach [54]. In the first three epochs is performed a coarse grained training, where the model is trained to distinguish between entities belonging to different domains. During each epoch 2000 triplets of the type (U, e^+, e^-) are randomly sampled.

In the second step, is performed an in-domain fine grained training. In this step, the model is trained for other three epochs to distinguish between entity names belonging to the same domain. For this step 3000 triplets were randomly sampled at each epoch from the training set. In addition, 1000 triplets were synthetically created from the knowledge snippets. The batch size was set to 16.

Since the domain of 'train' and 'taxi' contained only domain-level documents(a single entity is present), these are excluded for the training of the DER. In fact, the DER model must perform a retrieval of an entity name among a list of names within the same domain. Hence, the model needs to learn to distinguish entities among the same domain and this distinction is not needed for the cases of 'train' and 'taxi'.

For both the two training phases the optimizer AdamW has been used with a learning rate of $1e-5$. As in the previous case the triplet loss with the cosine distance was used as loss function in order to bring closer the dialogue context embedding vector and the corresponding entity name embedding vector in the entity vector space. The margin *alpha* was set again to 0.25

5.3 Dense Knowledge Retrieval

The Dense Knowledge Retrieval (DKR) represent the last task of the three-step pipeline retrieval process. Given in input a dialogue context $U = \{u_1, \ldots, u_n\}$ where the last utterance u_n is a user request and the corresponding domain d_U retrieved by the DDR and entity e_U retrieved by the DER, the DKR tackles the task of retrieving in a dense embedding space a set of relevant knowledge snippet with respect to the last u_n.

Encoder Also in this case the model all-mpnet-base-v2 from sentence transformers is used as encoder $E_{knowledge}$. Hence, resulting in a 768-dimensional knowledge dense vector space.

Embedding In contrast to the previous two retrieval tasks, the DKR works at a different granularity. Indeed, considering the utterances in the dialogue context U, if for the domain and entity retrieval all the utterances are needed since the domain or the entity name can be mentioned in any utterance; for the knowledge retrieval only the last user utterance u_n is needed. This in fact represents the user request whose response we want to retrieve from the FAQ documents. For this reason, given a dialogue context U, differently from the previous two cases the dialogue context embedding vector $v_{U,knowledge}$ in the knowledge dense vector space is obtained considering only the last user utterance:

$$v_{U,k} = E_{knowledge}(u_n|_{384})$$

Instead, the embedding vector of each knowledge snippet was computed considering the concatenation of all the information and computing the sentence embedding with the Transformer encoder model $E_{knwoledge}$:

$$w_k = E_{knowledge}(< d >< e >< q >< a >)$$

Training The DKR model has been trained for ten epochs. During each epoch 2000 triplets of the type (U, k^+, k^-) are sampled within the same domain and the same entity. Also in this case AdamW has been used as optimizer with a learning rate of $2e-5$. As in the previous case the triplet loss with the cosine distance was used as loss function with a margin of 0.25 and batch size of 4.

Chapter 6

Results

In this chapter are discussed the result obtained in the experiment. The three sub-systems addressing the domain, entity and document retrieval that compose the architecture of the Hierarchical Dense Knowledge Retrieval model have been evaluated in a pipeline fashion.

Given the pipeline framework of the DSTC9 track 1 composed by the three tasks of Detection, Selection and Generation, the retrieval step happened if and only if the given dialogue context was classified as a knowledge seeking-turn. For this reason, different teams worked on different data for the Selection and Generation phases, depending on their Detection results. In order to perform a fair comparison with other models, additional care has been taken in the computation of the final results, performing the measurements of the HDKR model on the same Detection labels.

6.1 Dense Domain Retrieval

For the training of the Dense Domain Retrieval (DDR) all the knowledge seeking turns have been considered. In addition, two extensions have been tested considering three different domain embedding strategies and performing a synthetic data augmentation.

In Tab. 6.1 is it possible to see the results of the different models both on the validation set and on the test set.

In the first half of the table are reported the results for the DDR models without synthetic data augmentation. The subscript *base*, *centroid*, and *medoid* indicates the different domain embedding strategies. As it is possible to observe, the results for the validation are equal for all three the embedding strategies in terms of accuracy, with a slight variation of computational time. However, it is important to notice how things change when we extend the dialogues to an unseen domain.

This is the case of the test set, where the new domain of attraction is added to the four domains of hotel, train, taxi and restaurant. in this zeroshot setting, it can be seen from the table how the base model has a huge drop of more than 11 percentage points in terms of accuracy, scoring 0.868 versus 0.987 for the validation set. Nonetheless, both the centroid and medoid methods can mitigate the zero shot problem for the new domain reaching respectively 0.959 and 0.947, with an increase with respect to the base model of 8-9%.

In the second half of the table are reported the results for the synthetic data augmentation. As in the previous case, all the models perform the same in the validation set. Moreover, it is possible to observe how the data augmentation help all these models in slightly boosting the performances, reaching a final accuracy score on the validation set of 0.99. Interestingly, in this case the models perform similarly also in the test set case with the additional unseen domain of attraction. Hence, the synthetic data can help in mitigate the zeroshot problem also in the *base* domain embedding case. The centroid and medoid solutions, combined with the synthetic data augmentation provide a slight improvement in terms of accuracy.

Since all the models adopt the same Transformer encoder and the same similarity function, the inference time is around 3.6 milliseconds for all the models.

	Validation			Test		
	R@1	μ_t **(s)**	σ_t **(s)**	**R@1**	μ_t **(s)**	σ_t **(s)**
DDR_{base}	0.987	0.00359	0.00053	0.868	0.00361	0.00059
$DDR_{centroid}$	0.987	0.00357	0.00050	0.959	0.00361	0.00059
DDR_{medoid}	0.987	0.00359	0.00054	0.947	0.00360	0.00060
$DDR_{synth+base}$	0.989	0.00360	0.00059	0.970	0.00360	0.00059
$DDR_{synth+centroid}$	**0.990**	0.00357	0.00051	**0.971**	0.00360	0.00060
$DDR_{synth+medoid}$	**0.990**	0.00359	0.00059	**0.971**	0.00362	0.00059

Table 6.1: Results for Dense Domain Retrieval in both val and test sets. μ_t indicates the average inference time and σ_t the standard deviation.

6.2 Dense Entity Retrieval

The Dense Entity Retrieval system has been trained considering the output of the DDR model. As explained in the previous chapters the clustering strategy was not adaptable to the entity name retrieval. For this reason only the synthetic data augmentation has been tested in this case. In addition, a curriculum learning strategy has been tried in order to help the convergence of the training phase starting from a coarse-grained finetuning to a fine-grained one.

In Tab. 6.2 are reported the results of the experiments. As in the previous case, it is possible to see how all the models behave similarly on the validation set, where they encounter the same domains on which they have been trained. However, the main differences between the different models can be appreciated in the test set where it is considered also the unseen domain of attraction. In this case the synthetic data augmentation helps the model improving the accuracy in the top-1 entity retrieval of 1%. In addition, the curriculum learning training seems to help in training a more robust model. As it can be seen from the table, this model in fact shows the best overall results in all the metrics.

	Validation				
	R@1	**R@2**	**R@3**	μ_t **(s)**	σ_t **(s)**
DER_{base}	0.969	0.985	0.986	0.00544	0.00061
DER_{synth}	0.971	0.986	**0.987**	0.00545	0.00061
$DER_{synth+curriculum}$	**0.972**	**0.987**	**0.987**	0.00545	0.00062
δ_{domain}	- 0.018	- 0.003	- 0.003		
	Test				
	R@1	**R@2**	**R@3**	μ_t **(s)**	σ_t **(s)**
DER_{base}	0.917	0.954	0.961	0.00694	0.00059
DER_{synth}	0.927	0.960	0.966	0.00687	0.00057
$DER_{synth+curriculum}$	**0.930**	**0.962**	**0.967**	0.00684	0.00056
δ_{domain}	- 0.041	- 0.009	- 0.004		

Table 6.2: Results for Dense Entity Retrieval in both val and test sets. The accuracy of the first three entities are considered cumulatively. Acc3 for example measure if the correct name is in the first three entity names retrieved. δ_{domain} indicates the loss in accuracy with respect to the DDR, in order to measure the error propagation.

The δ_{domain} in the table shows the loss in accuracy with respect to the domain retrieval, and is a measure of the error propagation. In the HDKR pipeline in fact the errors are propagate between the three models. It is possible to see how limiting the choice only to the top-1 entity name, the models suffers a loss of 1.8% in the validation set and of 4.1% in the test set. Considering the accuracy score for the top-2 and top-3 entity names it can be observed how the correct entity name is among the first three entities retrieved in the 98.7% of the cases on the validation set and in the 96.7% on the test set, this means that the correct entity name is not among the first three choices only on the 0.3% and 0.4% of the cases respectively. Regarding the test set, it is possible to appreciate how the top-k name retrieval strategy can help in retaining as much accuracy as possible. In this case in fact, considering for example only the top-1 name means that the final document retrieval accuracy cannot be higher than 93%.

6.3 Dense Knowledge Retrieval

The Dense Knowledge Retrieval (DKR) was evaluated considering the output of the DER. For the knowledge retrieval the top-3 entity names have been considered.

	Validation				
	R@1	**R@5**	**MRR**	μ_t **(s)**	σ_t **(s)**
DKR_{base}	0.942	0.984	0.960	0.00604	0.00076
δ_{domain}	-0.048	- 0.06			
δ_{entity}	-0.045	- 0.03			
	Test				
	R@1	**R@5**	**MRR**	μ_t **(s)**	σ_t **(s)**
DKR_{base}	**0.904**	**0.963**	**0.927**	0.00614	0.00039
δ_{domain}	-0.067	- 0.08			
δ_{entity}	-0.063	- 0.04			

Table 6.3: Results for the Dense Knowledge Retrieval in both val and test sets. δ_{domain} and δ_{entity} indicates respectively the degradation with respect to the DDR and DER.

In Tab. 6.3 are reported the results for the final DKR model. The final recall for the top-1 knowledge snippet is 0.942 for the validation set and 0.904 for the test set. Regarding the ranked list of the first 5 knowledge snippets, the recall grows to 0.984 for the validation set and 0.963 for the test set. Considering the domain retrieval accuracy of 0.99 for the validation set and 0.971 for the test set and given the strict filter imposed by the hierarchical architecture proposed, is it possible to observe that if the domain is correctly retrieved in the first step the final top-5 ranked list of knowledge snippet contains the correct snippet in over the 99% of the cases.

Finally, the inference time for the knowledge retrieval takes in average around 6 milliseconds both for the validation and test set.

6.4 Hierarchical Dense Knowledge Retrieval

In this section are reported the final results for the Hierarchical Dense Knowledge Retrieval considering the full pipeline. In order to assess the goodness of the HDKR method, three comparisons have been performed:

1. a comparison with the *baseline* system;
2. a comparison with the original DKR system proposed by Thulke et al. [46] (team 18);
3. a comparison with the model awarded as the final winner of the competition proposed by He et al. [30] (team 19).

In order to take a fair comparison between different solutions, it has been taken in consideration the fact that the results for the knowledge selection sub-task for the DSTC9 - track 1 was depending on the results on the knowledge detection sub-task. For this reason different measure for the HDKR model has been taken, considering *all* the knowledge-seeking turns or only the ones **detected** by the *team18* and the *team19*.

Comparison with the baseline: In the first comparison it is possible to observe how the HDKR model outperforms by a large margin the baseline system in all the metrics. In particular, the R@1 that measures the performances on the first ranked knowledge snippet, increases by almost 30%. The gap decreases considering the top-5 knowledge snippets. Nonetheless, also in this case there was an increase of almost 10%. However, the most impressive result is obtained considering the runtime of the models. The baseline system based on a Passage Re-Ranking strategy requires almost two minutes to compute the inference on the user request and output the top-5 knowledge documents. In contrast, the HDKR requires less than three milliseconds with an improvement of over 4000 times.

	Test				
	R@1	**R@5**	**MRR**	μ_t **(s)**	**num. parameters**
baseline	0.620	0.877	0.726	112.83	**117M**
HDKR (all)	**0.904**	**0.963**	**0.927**	**0.028**	241M
δ	+ 0.284	+0.086	+0.201	-112.80	+124M

Table 6.4: Final comparison of the HDKR with the Selection baseline of the DSTC9 - track 1. *All* the knowledge-seeking turns have been considered.

Comparison with DKR: In Tab. 6.5 are reported the results for the original DKR method proposed by Thulke et al. [46] (team 18) and the HDKR. In order to compare fairly the two models, only the knowledge-seeking turns detected by the team 18 during the Detection phase have been considered. In this case it is possible to observe how the HDKR model is able to outperform the original DKR method in almost every metric. In particular, the model is more accurate on the first prediction with an increase of 3.6%. It is worth to remember that this resulted as the most correlated measure with the final human judgement. Since the three encoders adopted for the HDKR are smaller than the encoder adopted for the DKR, it is possible to observe a reduction of the 30% of the average inference time with respect to the original DKR method. Finally, as it is possible to observe from the table, the model loose in R@5. This is due to the strict filter that the hierarchical structure impose to the retrieval system.

	Test				
	R@1	**R@5**	**MRR**	μ_t **(s)**	**num. parameters**
DKR (team18)	0.838	**0.945**	0.888	0.040	355M (+355M)
HDKR (team18)	**0.874**	0.926	**0.894**	**0.028**	**241M**
δ	+0.036	- 0.019	+ 0.006	- 0.012	- 469M

Table 6.5: Comparison between the DKR method proposed by the team 18 and the HDKR method. Only the knowledge-seeking turns detected by the team 18 have been considered. The number of parameters for the DKR model takes in consideration also the siamese network architecture (two separate encoders). However, the knowledge encoder is not considered in the final *delta* scoring since the knowledge embedding can be computed a-priori.

Comparison with the winning model (team 19): In Tab. 6.6 is reported the comparison between the model awarded as the winner of the DSTC9 - track 1 and the HDKR evaluated on the same knowledge-seeking turns. Unfortunately, it was not possible to measure the inference time for the winner model. However, considering the fact that this model adopted a similar PRR strategy with respect to the baseline model, the number of parameters can give a sense of the dimension of this model. This model, in fact used an ensemble of 5 transformer models among which also Plato-2, a transformer model with 1.6B parameters that requires a V100 32GB for inference. The final model resulted almost ten times bigger with respect to the HDKR model and 20 times bigger with respect to the original baseline model. In this comparison it is possible to appreciate how the HDKR is fairly close to the best model on the Selection task of the DSTC9, suffering a degradation between 2.5% and 3%, while maintaining an inference time and a number of parameters applicable also in a real-case scenario.

	Test				
	R@1	**R@5**	**MRR**	μ_t **(s)**	**num. parameters**
Winner (team19)	**0.924**	**0.981**	**0.950**	/	2082M
HDKR (team19)	0.899	0.952	0.920	0.028	**241M**
δ	-0.025	- 0.029	-0.030	/	-1841M

Table 6.6: Comparison between the *Winner* team of the DSTC9 - track 1 and the HDKR method. Only the *team 19* knowledge seeking turns have been considered.

Chapter 7

Conclusion

In this work it has been presented an efficient solution for the knowledge selection in the context of the DSTC9 - track 1. The goal of the task was to retrieve unstructured FAQ knowledge snippets in a task-oriented setting. The Hierarchical Dense Knowledge Retrieval showed interesting results. On one hand, it improved the efficiency of the retrieval, reducing the computational time during inference from more than 112s for the Passage Re-Ranking baseline to only 0.028s and improving also the original DKR with a decrease of 30% in time. At the same time the model caused also a minor impact in terms of memory consumption reducing of almost ten times the number of parameters of the model with respect to the solution proposed by the winner team. On the other hand, the HDKR model was able to improve the accuracy of the original DKR model being more precise on the selection of the correct knowledge snippet of 3.6%.

This model showed to be applicable also in real case applications at a cost of a deterioration in the selection performance of only 2.5% in the selection of the correct knowledge snippet with respect to the best model.

Finally, as shown by the final comparisons, the HDKR method suffered more on the R5 where it performed worse in comparison to both the Winning team (team 19) and the original DKR (team 18). Further improvements in the model could come from the development of the domain retrieval, with a consequent reduction of the error propagation.

Appendix A

Appendix

Fig. A.1 shows an example of three FAQ documents in the knowledge base. Each knowledge document present in the KB consisted of a raw-textual document divided in two parts: the question, called *title* and the corresponding answer, called *body*. Despite each knowledge snippet was unstructured, the knowledge base was organized in a hierarchical manner considering the *domain* (e.g., 'hotel' in the figure), then the list of entities, and finally for each entity the list of documents. Each entity is identified by an entity_id and has a *name* (e.g., "A AND B GUEST HOUSE" in the figure). Similarly each document is identified by a doc_id (e.g., "0", "1" and "2" in the image).


```
{
  "hotel": {
    "0": {
      "name": "A AND B GUEST HOUSE",
      "docs": {
        "0": {
          "title": "Are children welcomed at this location?",
          "body": "Yes, you can stay with children at A and B Guest House."
        },
        "1": {
          "title": "Can I bring my pet to A and B Guest House?",
          "body": "No, pets are not allowed at this property."
        },
        "2": {
          "title": "Do you have onsite parking for your guests?",
          "body": "There is onsite parking at A and B Guest House but it costs extra."
        }
```


Figure A.1: Example of the first three FAQ documents for the entity "A AND B GUEST HOUSE" from the domain hotel.

Fig. A.2 shows an example of two sub-dialogues sampled from the same original dialogue context. Each dialogue is composed by a series of utterances. For each

utterance it is reported the *speaker* as *U* for user or *S* for system and the actual content of the utterance, called *text*. Each sub-dialogue has a label containing a boolean value called *target* that indicates if the last user utterance in the sub-dialogue is a knowledge-seeking turn. In the case it is, there are also reported the information about the correct *knowledge* snippet and the human-generated *response*.

```
[
  [
    {
      "speaker": "U",
      "text": "I need a 4-star hotel in the financial district of San Francisco"
    }
  ],
  [
    {
      "speaker": "U",
      "text": "I need a 4-star hotel in the financial district of San Francisco"
    },
    {
      "speaker": "S",
      "text": "I locate Hilton San Francisco Financial District,"+
      "would like me to check availability?"
    },
    {
      "speaker": "U",
      "text": "Not at the moment, I need to know if they speak French."
    }
  ]
]
```

```
[
  {
    "target": false
  },
  {
    "target": true,
    "knowledge": [
      {
        "domain": "hotel",
        "entity_id": 11,
        "doc_id": 12
      }
    ],
    "response": "The hotel does not speak French."+
    "Would you like to make a reservation?"
  }
]
```


Figure A.2: Example of two sub-dialogues from the same dialogue context. The first user request is not a knowledge-seeking turn, while the second needs access to the knowledge base.

Bibliography

[1] Adrienne Mayor. «Gods and Robots, Myths, Machines, and Ancient Dreams of Technology». In: (2018). DOI: 10.1354/books/unregistered/9780691185446 (cit. on p. 1).

[2] Roberto Pieraccini. «The Voice in the Machine: Building Computers That Understand Speech». In: (2012). ISSN: 978026230. DOI: 10.7551/mitpress/9072.001.0001. URL: https://doi.org/10.7551/mitpress/9072.001.0001 (cit. on p. 1).

[3] Joseph Weizenbaum. «ELIZA—a Computer Program for the Study of Natural Language Communication between Man and Machine». In: *Commun. ACM* 9.1 (Jan. 1966), pp. 36–45. ISSN: 0001-0782. DOI: 10.1145/365153.365168. URL: https://doi.org/10.1145/365153.365168 (cit. on p. 1).

[4] Douglas R. Hofstadter. *Preface 4 The Ineradicable Eliza Effect and Its Dangers, Epilogue*. USA: Basic Books, Inc., 1996. ISBN: 0465024750 (cit. on p. 1).

[5] Emily Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. «On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? » In: *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*. FAccT '21. Virtual Event, Canada: Association for Computing Machinery, 2021, pp. 610–623. ISBN: 9781450383097. DOI: 10.1145/3442188.3445922. URL: https://doi.org/10.1145/3442188.3445922 (cit. on p. 2).

[6] Tom Brown et al. «Language Models are Few-Shot Learners». In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin. Vol. 33. Curran Associates, Inc., 2020, pp. 1877–1901. URL: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf (cit. on p. 3).

[7] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Xiaodong Song, and Jacob Steinhardt. «Measuring Massive Multitask Language Understanding». In: *ArXiv* abs/2009.03300 (2021) (cit. on p. 3).

[8] S. Eggins and D. Slade. *Analysing Casual Conversation*. Equinox textbooks and surveys in linguistics. Equinox, 2004. ISBN: 9781845530464. URL: https://books.google.it/books?id=h-85jgEzyOcC (cit. on pp. 3, 6).

[9] 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». In: *arXiv* (2021). eprint: 2101.09276 (cit. on pp. 3, 13).

[10] 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». In: *arXiv* (2021) (cit. on pp. 5, 20, 21).

[11] Salla Syvänen and Chiara Valentini. «Conversational agents in online organization–stakeholder interactions: a state-of-the-art analysis and implications for further research». In: *Journal of Communication Management* 24.4 (2020), pp. 339–362. ISSN: 1363-254X. DOI: 10.1108/jcom-11-2019-0145 (cit. on p. 5).

[12] Marvin Minsky. «Society of Mind: A Response to Four Reviews». In: *Artif. Intell.* 48 (1991), pp. 371–396 (cit. on p. 6).

[13] Ivana S Marková, Per Linell, Michèle Grossen, and Anne Salazar Orvig. «Dialogue in Focus Groups: Exploring Socially Shared Knowledge». In: 2007 (cit. on p. 6).

[14] Lili Mou, Yiping Song, Rui Yan, Ge Li, Lu Zhang, and Zhi Jin. «Sequence to Backward and Forward Sequences: A Content-Introducing Approach to Generative Short-Text Conversation». In: *COLING*. 2016 (cit. on p. 6).

[15] Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, M. Zhou, and Wei-Ying Ma. «Topic Aware Neural Response Generation». In: *AAAI*. 2017 (cit. on p. 6).

[16] 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* (2020). eprint: 2006.03533 (cit. on pp. 6, 11–13, 15, 21).

[17] Ruhi Sarikaya. «The Technology Behind Personal Digital Assistants: An overview of the system architecture and key components». In: *IEEE Signal Processing Magazine* 34.1 (2017), pp. 67–81. ISSN: 1053-5888. DOI: 10.1109/msp.2016.2617341 (cit. on pp. 6, 47).

[18] Shafquat Hussain, Omid Ameri Sianaki, and Nedal Ababneh. «A Survey on Conversational Agents/Chatbots Classification and Design Techniques». In: *Web, Artificial Intelligence and Network Applications*. Ed. by Leonard Barolli, Makoto Takizawa, Fatos Xhafa, and Tomoya Enokido. Cham: Springer International Publishing, 2019, pp. 946–956. ISBN: 978-3-030-15035-8 (cit. on p. 7).

[19] Allan de Barcelos Silva, Marcio Miguel Gomes, Cristiano André da Costa, Rodrigo da Rosa Righi, Jorge Luis Victoria Barbosa, Gustavo Pessin, Geert De Doncker, and Gustavo Federizzi. «Intelligent Personal Assistants: A Systematic Literature Review». In: *Expert Systems with Applications* 147 (2020), p. 113193. ISSN: 0957-4174. DOI: 10.1016/j.eswa.2020.113193 (cit. on p. 7).

[20] Hongshen Chen, Xiaorui Liu, Dawei Yin, and Jiliang Tang. «A Survey on Dialogue Systems: Recent Advances and New Frontiers». In: 19 (2017), pp. 25–35. ISSN: 1931-0145. DOI: 10.1145/3166054.3166058 (cit. on pp. 7, 10).

[21] Minlie Huang, Xiaoyan Zhu, and Jianfeng Gao. «Challenges in Building Intelligent Open-Domain Dialog Systems». In: *ACM Trans. Inf. Syst.* 38.3 (Apr. 2020). ISSN: 1046-8188. DOI: 10.1145/3383123. URL: https://doi.org/10.1145/3383123 (cit. on p. 7).

[22] Bert F. Green, Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. «Baseball: An Automatic Question-Answerer». In: *Papers Presented at the May 9-11, 1961, Western Joint IRE-AIEE-ACM Computer Conference*. IRE-AIEE-ACM '61 (Western). Los Angeles, California: Association for Computing Machinery, 1961, pp. 219–224. ISBN: 9781450378727. DOI: 10.1145/1460690.1460714. URL: https://doi.org/10.1145/1460690.1460714 (cit. on p. 8).

[23] Wenhao Yu, Chenguang Zhu, Zaitang Li, Zhiting Hu, Qingyun Wang, Heng Ji, and Meng Jiang. «A Survey of Knowledge-Enhanced Text Generation». In: *arXiv* (2020) (cit. on pp. 10, 11).

[24] Eleni Adamopoulou and Lefteris Moussiades. «Chatbots: History, technology, and applications». In: *Machine Learning with Applications* 2 (2020), p. 100006. ISSN: 2666-8270. DOI: https://doi.org/10.1016/j.mlwa.2020.100006. URL: https://www.sciencedirect.com/science/article/pii/S2666827020300062 (cit. on p. 10).

[25] Ting-Hao (Kenneth) Huang, Joseph Chee Chang, and Jeffrey P. Bigham. «Evorus: A Crowd-Powered Conversational Assistant Built to Automate Itself Over Time». In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2018, pp. 1–13. ISBN: 9781450356206. URL: https://doi.org/10.1145/3173574.3173869 (cit. on p. 10).

[26] Dayiheng Liu et al. «GLGE: A New General Language Generation Evaluation Benchmark». In: *FINDINGS*. 2021 (cit. on p. 10).

[27] Zheng Zhang, Ryuichi Takanobu, Qi Zhu, MinLie Huang, and XiaoYan Zhu. «Recent advances and challenges in task-oriented dialog systems». In: *Science China Technological Sciences* 63.10 (2020), pp. 2011–2027. ISSN: 1674-7321. DOI: 10.1007/s11431-020-1692-3 (cit. on p. 11).

[28] Mihail Eric et al. «MultiWOZ 2.1: A Consolidated Multi-Domain Dialogue Dataset with State Corrections and State Tracking Baselines». In: *arXiv* (2019). eprint: 1907.01669 (cit. on p. 15).

[29] Haitao Mi, Qiyu Ren, Yinpei Dai, Yifan He, Yongbin Li, Jian Sun, Jing Zheng, and Peng Xu. «Towards Generalized Models for Beyond Domain API Task-oriented Dialogue». In: (2021) (cit. on pp. 20, 31, 37).

[30] Huang He, Hua Lu, Siqi Bao, Fan Wang, Hua Wu, Zhengyu Niu, and Haifeng Wang. «Learning to Select External Knowledge with Multi-Scale Negative Sampling». In: *arXiv* (2021). eprint: 2102.02096 (cit. on pp. 20, 21, 30, 37, 41, 61).

[31] C. Spearman. *The Proof and Measurement of Association Between Two Things.* Studies in individual differences: The search for intelligence. Pages: 58. East Norwalk, CT, US: Appleton-Century-Crofts, 1961. DOI: 10.1037/11491-005 (cit. on p. 20).

[32] Rodrigo Nogueira and Kyunghyun Cho. «Passage Re-ranking with BERT». In: *ArXiv* abs/1901.04085 (2019) (cit. on pp. 21, 23, 28, 36).

[33] Siqi Bao, Huang He, Fan Wang, Hua Wu, Haifeng Wang, Wenquan Wu, Zhen Guo, Zhibin Liu, and Xinchao Xu. «PLATO-2: Towards Building an Open-Domain Chatbot via Curriculum Learning». In: *arXiv* (2020). eprint: 2006.16779 (cit. on pp. 21, 30, 36).

[34] 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 pp. 21, 37, 52, 54).

[35] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. «Albert: A lite bert for self-supervised learning of language representations». In: *arXiv preprint arXiv:1909.11942* (2019) (cit. on pp. 21, 37).

[36] Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Yu Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. «Dense Passage Retrieval for Open-Domain Question Answering». In: *ArXiv* abs/2004.04906 (2020) (cit. on pp. 23, 31).

[37] Tomas Mikolov, Martin Karafiát, Lukás Burget, Jan Honza Cernocký, and Sanjeev Khudanpur. «Recurrent neural network based language model». In: *INTERSPEECH*. 2010 (cit. on p. 25).

[38] 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*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf (cit. on p. 26).

[39] Wilson L. Taylor. «"Cloze Procedure": A New Tool for Measuring Readability». In: *Journalism Quarterly* 30.4 (1953), pp. 415–433. DOI: 10.1177/107769905303000401. eprint: https://doi.org/10.1177/107769905303000401. URL: https://doi.org/10.1177/107769905303000401 (cit. on p. 26).

[40] Yinhan Liu et al. «RoBERTa: A Robustly Optimized BERT Pretraining Approach». In: *ArXiv* abs/1907.11692 (2019) (cit. on p. 27).

[41] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. «XLNet: Generalized Autoregressive Pretraining for Language Understanding». In: *NeurIPS*. 2019 (cit. on pp. 27, 54).

[42] Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, M. Zhou, and Hsiao-Wuen Hon. «Unified Language Model Pre-training for Natural Language Understanding and Generation». In: *ArXiv* abs/1905.03197 (2019) (cit. on pp. 27, 29).

[43] Liang Tang, Qinghua Shang, Kaokao Lv, Zixi Fu, Shijiang Zhang, Chuanming Huang, and Zhuo Zhang. «RADGE: Relevance Learning and Generation Evaluating Method for Task-Oriented Conversational Systems». In: (2021) (cit. on p. 29).

[44] Stephen Robertson and Hugo Zaragoza. «The Probabilistic Relevance Framework: BM25 and Beyond». In: *Foundations and Trends® in Information Retrieval* 3.4 (2009), pp. 333–389. ISSN: 1554-0669. DOI: 10.1561/1500000019. URL: http://dx.doi.org/10.1561/1500000019 (cit. on p. 31).

[45] Nils Reimers and Iryna Gurevych. «Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks». In: *ArXiv* abs/1908.10084 (2019) (cit. on p. 31).

[46] David Thulke, Nico Daheim, Christian Dugast, and Hermann Ney. «Efficient Retrieval Augmented Generation from Unstructured Knowledge for Task-Oriented Dialog». In: *arXiv* (2021). eprint: 2102.04643 (cit. on pp. 31, 37, 41–43, 61, 62).

[47] Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. «Large Scale Online Learning of Image Similarity Through Ranking». In: *Journal of Machine Learning Research* 11.36 (2010), pp. 1109–1135. URL: http://jmlr.org/papers/v11/chechik10a.html (cit. on p. 33).

[48] Florian Schroff, Dmitry Kalenichenko, and James Philbin. «FaceNet: A unified embedding for face recognition and clustering». In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 815–823. DOI: 10.1109/CVPR.2015.7298682 (cit. on p. 33).

[49] Chao-Hong Tan et al. «Learning to Retrieve Entity-Aware Knowledge and Generate Responses with Copy Mechanism for Task-Oriented Dialogue Systems». In: *arXiv* (2020). eprint: 2012.11937 (cit. on p. 34).

[50] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. «Distilling the Knowledge in a Neural Network». In: *ArXiv* abs/1503.02531 (2015) (cit. on p. 34).

[51] Laurens van der Maaten and Geoffrey Hinton. «Visualizing Data using t-SNE». In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html (cit. on p. 40).

[52] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. «MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers». In: *ArXiv* abs/2002.10957 (2020) (cit. on p. 52).

[53] Sebastian Hofstätter, Aldo Lipani, Sophia Althammer, Markus Zlabinger, and Allan Hanbury. «Mitigating the Position Bias of Transformer Models in Passage Re-Ranking». In: *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28 – April 1, 2021, Proceedings, Part I*. Berlin, Heidelberg: Springer-Verlag, 2021, pp. 238–253. ISBN: 978-3-030-72112-1. DOI: 10.1007/978-3-030-72113-8_16. URL: https://doi.org/10.1007/978-3-030-72113-8_16 (cit. on p. 53).

[54] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. «Curriculum learning». In: *ICML '09*. 2009 (cit. on p. 55).