



1 2
UNIVERSIDADE D
COIMBRA

CONTEXT MONITORING FOR DIALOGUES IN PORTUGUESE

Francisco Moita Pais



1 2
UNIVERSIDADE D
COIMBRA

Francisco Moita Pais

CONTEXT MONITORING FOR DIALOGUES IN PORTUGUESE

Dissertation in the context of the Master in Informatics Engineering,
specialization in Intelligent Systems, advised by Professors Hugo
Ricardo Gonçalo Oliveira and Catarina Helena Branco Simões da Silva,
presented to the Department of Informatics Engineering of the Faculty of
Sciences and Technology of the University of Coimbra.

September 2023



FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

DEPARTMENT OF INFORMATICS ENGINEERING

Francisco Moita Pais

CONTEXT MONITORING FOR DIALOGUES IN PORTUGUESE

Dissertation in the context of the Master in Informatics Engineering,
specialization in Intelligent Systems, advised by Professors Hugo Ricardo
Gonçalo Oliveira and Catarina Helena Branco Simões da Silva, presented to the
Department of Informatics Engineering of the Faculty of Sciences and
Technology of the University of Coimbra.

September 2023

Acknowledgements

This work is funded by the project POWER (grant number POCI-01-0247-FEDER-070365), co-financed by the European Regional Development Fund (FEDER), through Portugal 2020 (PT2020), and by the Competitiveness and Internationalization Operational Programme (COMPETE 2020).

The personal acknowledgments are written in Portuguese, my native language, to ensure that everyone acknowledged can understand this dedication.

Gostaria de começar por agradecer, primeiramente, aos meus pais pelo apoio incondicional que sempre demonstraram desde o início para comigo. Independentemente do problema em questão, sempre acreditaram no meu sucesso. Agradeço também pela oportunidade única que me proporcionaram de tirar um curso de ensino superior.

Quero também deixar uma palavra de apreço à minha prima Sandra. Apesar da distância, sempre me fez sentir como estivesse bem perto de mim. Foi como uma segunda mãe, sempre ao meu lado a apoiar-me e a confiar nas minhas capacidades.

A nível académico, gostaria de agradecer ao Professor Hugo Oliveira e à Professora Catarina Silva pelo tempo e pela paciência que dedicaram no auxílio à construção desta dissertação. Sem dúvida, concluo este trabalho com mais conhecimento e cheio de experiências enriquecedoras que serão úteis tanto profissionalmente como pessoalmente e isso devo a eles.

Por último, mas não menos importante, um agradecimento aos meus amigos, muitos dos quais já me acompanham desde o infantário e outros que conheci já na universidade. A todos eles, um grande obrigado pelos bons e maus momentos que partilhámos. Todos contribuíram para fazer de mim a pessoa que sou hoje.

Mais uma vez, um grande obrigado a todos vocês. Espero conseguir concretizar os meus planos e contar com o vosso apoio nas minhas futuras conquistas, que com certeza estão no horizonte.

Abstract

Dialogue systems have witnessed remarkable growth in everyday use, significantly simplifying a broad range of tasks. Whether it involves simple actions like checking the next morning's weather or more complex processes like booking flights to specific destinations at designated times, these systems have become indispensable. To perform these tasks successfully, dialogue systems usually need to monitor context. This can be accomplished through Dialogue State Tracking (DST), a process that involves tracking the progress of a conversation while maintaining a representation of the current state. DST is essential for dialogue systems, as it allows them to consider the context of the conversation through the application of Natural Language Processing (NLP) techniques, and respond appropriately. While DST has been mainly applied to English, in this work, it is applied to Portuguese dialogues. To our knowledge, it is the first time that DST is applied to Portuguese. To make it possible, the work included the creation of a dataset of task-oriented dialogues in Portuguese, based on the adaptation of the widely-used MultiWOZ dataset. A key component of DST is Information Extraction (IE), where specific information is extracted and filled into slots based on user-defined constraints. This was achieved using a novel approach based on Question-Answering (QA), which extracts an answer based on a provided question and context (user's utterance). In this work, three available QA models were tested for this purpose: BERT-base, BERT-large, and T5. We concluded that the model with the poorest performance was T5. The other two models yielded similar results. The choice between these two alternatives depends on the specific task the user aims to accomplish. Both (BERT-base, BERT-large) have demonstrated promise in solving tasks in the Portuguese language. Additionally, we found that the addition of methods such as Intent Detection and Post-Processing significantly improves the performance of each model. These findings not only contribute to the advancement of DST in Portuguese-speaking communities but also open new avenues for the implementation of new dialogue systems based on the dataset created.

Keywords

Dialogue Systems; Dialogue State Tracking; Natural Language Processing; Information Extraction; Question-Answering.

Resumo

Os Sistemas de diálogo têm testemunhado um crescimento notável no uso cotidiano, simplificando uma variedade extensa de tarefas. Seja para ações simples como verificar o tempo para o dia seguinte ou para processos mais complexos como a reserva de voos para destinos específicos em horários determinados, tornando-se estes sistemas indispensáveis. Para executar essas tarefas de forma eficaz, é crucial que os sistemas de diálogo normalmente monitorizem o contexto em que operam. Tal monitorização pode ser efetuada através do Rastreamento do Estado do Diálogo (DST), um processo que acompanha o desenvolvimento de uma conversa enquanto mantém uma representação atualizada do seu estado. O DST é vital para os sistemas de diálogo, uma vez que permite a incorporação do contexto da conversa ao aplicar técnicas de Processamento de Linguagem Natural (NLP) e gerar respostas adequadas. Embora o DST tenha sido principalmente aplicado ao idioma inglês, neste trabalho estendemos a sua aplicação a diálogos em português. Pelo que sabemos, esta é a primeira vez que tal extensão foi realizada. O trabalho envolveu a criação de um conjunto de dados de diálogos, orientado a tarefas para português, adaptado do amplamente utilizado conjunto de dados MultiWOZ. Um componente crucial do DST é a Extração de Informação (IE), que preenche slots com informações específicas com base em restrições definidas pelo utilizador. Alcançou-se isso através de uma abordagem inovadora baseada em Perguntas e Respostas (QA), que extrai uma resposta baseada em uma pergunta e um contexto (frase do utilizador). Neste estudo, três modelos de QA foram testados para esse propósito: BERT-base, BERT-large e T5. Concluímos que o modelo T5 apresentou o desempenho mais fraco. Os outros dois modelos geraram resultados semelhantes. A escolha entre estas duas alternativas dependerá da tarefa específica que o utilizador deseja realizar. Ambos (BERT-base e BERT-large) mostraram ser promissores em resolver tarefas na língua portuguesa. Adicionalmente, observamos que a inclusão de métodos como Detecção de Intenção e Pós-Processamento melhorou significativamente o desempenho de cada modelo. Essas descobertas não só contribuem para o desenvolvimento do DST em comunidades lusófonas, mas também abrem novas possibilidades para a implementação de sistemas de diálogo inovadores baseados no conjunto de dados criado.

Palavras-Chave

Sistemas de Diálogo; Rastreamento do Estado do Diálogo; Processamento de Linguagem Natural; Extração de Informação; Perguntas e Respostas.

Contents

1	Introduction	1
1.1	Problem and Motivation	1
1.2	Goals	2
1.3	Approach	3
1.4	Methodology	4
1.5	Contributions	6
1.6	Document Structure	6
2	Background	9
2.1	Natural Language Processing	9
2.2	Information Extraction	10
2.3	Dialogue Systems	12
2.3.1	Chatbots	13
2.3.2	Dialogue State	14
2.3.3	Evaluation Metrics in DST	17
2.4	Transformers	18
2.4.1	Causal or left-to-right Transformer	19
2.4.2	Bidirectional Transformer Encoders	20
2.4.3	Training Bidirectional Encoders	20
2.4.4	Transfer Learning through Fine-Tuning	23
2.5	Question Answering	24
2.5.1	IR-based Factoid QA	24
2.5.2	IR-based QA: Datasets	26
2.5.3	Knowledge-based QA	27

2.6	Summary	28
3	Related Work	29
3.1	Datasets for Task-oriented Dialogue Systems	29
3.2	Algorithms for DST	31
3.2.1	Algorithms used in the MultiWOZ dataset	31
3.2.2	Algorithms used in the SGD dataset	33
3.3	Summary	35
4	Dataset Creation	37
4.1	MultiWOZ Dataset	37
4.2	MultiWOZpt 0.1 Dataset	40
4.3	Future Applications	42
4.4	Summary	42
5	Question Answering-based Slot-Filling	43
5.1	QA Models for Portuguese	44
5.1.1	BERT for QA	44
5.1.2	T5 for QA	46
5.1.3	Questions Developed for QA Models	47
5.2	Post-Processing	49
5.3	Intent Detection	51
5.4	Summary	53
6	Evaluation	55
6.1	Testing of the QA Models	55
6.1.1	Selection of Thresholds	56
6.1.2	Evaluation of the QA Models	60
6.2	Summary	63
7	Conclusions and Future Work	65
7.1	Conclusions	65
7.2	Future Work	67

Appendices

75

Acronyms

AI Artificial Intelligence. xiii, 10, 65

ATIS Air Travel Information System. xiii, 29

BERT Bidirectional Encoder Representation Transformer. xii, xiii, xviii, xix, xxii, 20, 22, 28, 31, 33, 44, 45, 46, 56, 57, 58, 60, 61, 63, 64, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106

DS Dialogue State. xiii, xvii, 3, 9, 12, 16, 17, 31, 34, 36

DST Dialogue State Tracking. vii, xi, xii, xiii, xvii, 2, 3, 6, 12, 16, 17, 28, 29, 31, 32, 34, 36, 42, 44, 60, 65, 66

GOLOMB GOaL-Oriented Multi-task BERT-based. xiii, 33, 34

IE Information Extraction. vii, xiii, 1, 2, 6, 9, 10, 28

IR Information-Retrieval. xi, xiii, xvii, 14, 24, 25, 26

JGA Joint Goal Accuracy. xiii, xxii, 12, 17, 18, 32, 35, 36, 60, 61, 62, 63, 64

KB Knowledge Base. xiii

M2M Machines Talking To Machines. xiii, 29, 30

MA-DST Multi-Attention-Based Scalable Dialogue State Tracking. xiii, 31

MLM Masked Language Modeling. xiii, 21

MultiWOZ Multi-Domain Wizard-of-Oz. xii, xiii, xvii, xxi, 4, 6, 7, 18, 29, 30, 31, 32, 35, 37, 38, 40, 67

NER Named Entity Recognition. xiii, 10, 40

NLP Natural Language Processing. vii, xiii, 1, 6, 9, 10, 22, 24, 27, 28, 40, 46, 47, 49, 66

QA Question-Answering. vii, xi, xii, xiii, xvii, xviii, xxii, 3, 4, 6, 7, 9, 24, 25, 26, 27, 28, 43, 44, 45, 46, 47, 49, 51, 52, 53, 55, 56, 58, 60, 62, 65, 66, 67

RDF Resource Description Framework. xiii

RNN Recurrent Neural Networks. xiii, 11

- RoBERTa** Robustly Optimized BERT Pre-training Approach. xiii, 20
- SGD** Schema-Guided Dialogue. xii, xiii, xxi, 30, 33, 35, 36, 40
- SOM-DST** Selectively Overwriting Memory for Dialogue State Tracking. xiii, 31, 36
- SPARQL** SPARQL Protocol and RDF Query Language. xiii, 27
- SQL** Structured Query Language. xiii, 27, 28
- SQuAD** Stanford Question Answering Dataset. xiii, xvii, 26, 44
- STS** Semantic Textual Similarity. xiii, xxi, 49, 50, 51, 60, 61, 63, 82, 83, 84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111
- SUMBT** Slot-Utterance Matching Belief Tracker. xiii, 31, 36
- TripPy** Triple copy strategy DST. xiii, 31, 32, 35, 36
- TS-DST** Two-stage framework for schema-guided DST. xiii, 33, 34, 36
- WOZ** Wizard-of-Oz. xiii, 29, 37

List of Figures

1.1	Example of a Dialogue State (DS)	3
1.2	Original Work Plan.	5
1.3	Actual Work Plan.	5
2.1	Hand-built lexico-syntactic patterns for finding hypernyms	11
2.2	A conversation between a User(left) and BlenderBot.	13
2.3	Sample ELIZA dialogue.	14
2.4	A transcript of an actual dialogue with the GUS system.	15
2.5	Example of a DS that is constituted by the Dialogue State Tracking (DST), Dialogue Policy and Dialogue Act.	16
2.6	Simplified excerpt from a dialogue in the Multi-Domain Wizard-of-Oz (MultiWOZ) dataset	18
2.7	Example of a Causal, backward looking, transformer model.	19
2.8	Information flow in a Bidirectional self-attention model.	20
2.9	Instead of predicting what words are likely to come next (Phrase 1), is asked to predict a missing item given the rest of the sentence (Phrase 2).	20
2.10	Masked Language Model Training.	21
2.11	Span-based Language Model Training.	22
2.12	Information-Retrieval (IR)-based factoid Question-Answering (QA) has two stages.	25
2.13	Question-answer pairs for a sample passage in the Stanford Question Answering Dataset (SQuAD) dataset.	26
2.14	Sample logical forms produced by a semantic parser for question answering	28
3.1	Example of a dialogue involving two unseen services(Event_3 and Flights_4).	34

List of Figures

4.1	Excerpt from a dialogue of the MultiWOZ dataset.	39
5.1	Pipeline for Slot-filling Approach.	43
5.2	QA System with Bidirectional Encoder Representation Transformer (BERT).	45
5.3	The T5 system is an encoder-decoder architecture.	46
5.4	Example of how the detection of an intent affects a QA Model.	52
5.5	Example of how the absence of intent detection affects a QA Model.	53
1	Evaluating the BERT-base Model with Intent Detection across different slots of the Attraction Service using various thresholds.	82
2	Evaluating the BERT-base Model with Intent Detection across different slots of the Hotel Service using various thresholds.	83
3	Evaluating the BERT-base Model with Intent Detection across different slots of the Restaurant Service using various thresholds.	84
4	Evaluating the BERT-base Model with Intent Detection across different slots of the Taxi Service using various thresholds.	85
5	Evaluating the BERT-base Model with Intent Detection across different slots of the Train Service using various thresholds.	86
6	Evaluating the BERT-large Model with Intent Detection across different slots of the Attraction Service using various thresholds.	87
7	Evaluating the BERT-large Model with Intent Detection across different slots of the Hotel Service using various thresholds.	88
8	Evaluating the BERT-large Model with Intent Detection across different slots of the Restaurant Service using various thresholds.	89
9	Evaluating the BERT-large Model with Intent Detection across different slots of the Taxi Service using various thresholds.	90
10	Evaluating the BERT-large Model with Intent Detection across different slots of the Train Service using various thresholds.	91
11	Evaluating the T5 Model with Intent Detection across different slots of the Attraction Service using various thresholds.	92
12	Evaluating the T5 Model with Intent Detection across different slots of the Hotel Service using various thresholds.	93
13	Evaluating the T5 Model with Intent Detection across different slots of the Restaurant Service using various thresholds.	94
14	Evaluating the T5 Model with Intent Detection across different slots of the Taxi Service using various thresholds.	95

15	Evaluating the T5 Model with Intent Detection across different slots of the Train Service using various thresholds.	96
16	Evaluating the BERT-base Model without Intent Detection across different slots of the Attraction Service using various thresholds. . .	97
17	Evaluating the BERT-base Model without Intent Detection across different slots of the Hotel Service using various thresholds.	98
18	Evaluating the BERT-base Model without Intent Detection across different slots of the Restaurant Service using various thresholds. .	99
19	Evaluating the BERT-base Model without Intent Detection across different slots of the Taxi Service using various thresholds.	100
20	Evaluating the BERT-base Model without Intent Detection across different slots of the Train Service using various thresholds.	101
21	Evaluating the BERT-large Model without Intent Detection across different slots of the Attraction Service using various thresholds. . .	102
22	Evaluating the BERT-large Model without Intent Detection across different slots of the Hotel Service using various thresholds.	103
23	Evaluating the BERT-large Model without Intent Detection across different slots of the Restaurant Service using various thresholds. .	104
24	Evaluating the BERT-large Model without Intent Detection across different slots of the Taxi Service using various thresholds.	105
25	Evaluating the BERT-large Model without Intent Detection across different slots of the Train Service using various thresholds.	106
26	Evaluating the T5 Model without Intent Detection across different slots of the Attraction Service using various thresholds.	107
27	Evaluating the T5 Model without Intent Detection across different slots of the Hotel Service using various thresholds.	108
28	Evaluating the T5 Model without Intent Detection across different slots of the Restaurant Service using various thresholds.	109
29	Evaluating the T5 Model without Intent Detection across different slots of the Taxi Service using various thresholds.	110
30	Evaluating the T5 Model without Intent Detection across different slots of the Train Service using various thresholds.	111

List of Tables

2.1	A frame in a frame-based dialogue system, showing the type of each slot and a question used to fill the slot.	15
2.2	A few factoid questions and their answers.	25
3.1	Comparison of datasets for task-oriented dialogue.	30
3.2	DST Results on MultiWOZ for Joint Goal Accuracy.	32
3.3	Performance comparison of all models for All Services using the Schema-Guided Dialogue (SGD) dataset.	35
3.4	Performance comparison of all models for Seen Services using the SGD dataset.	35
3.5	Performance comparison of all models for Unseen Services using the SGD dataset.	35
4.1	Mapping of the MultiWOZ slots associated with each Domain. . . .	38
4.2	Number of Services present in the MultiWOZpt 0.1 dataset.	41
4.3	Comparison of the Number of Intents present in the two datasets. .	41
5.1	Questions for the Attraction Service.	47
5.2	Questions for the Hotel Service.	47
5.3	Questions for the Restaurant Service.	48
5.4	Questions for the Taxi Service.	48
5.5	Questions for the Train Service.	48
5.6	Examples of correctly assessed comparisons using the Levenshtein method as a Post-Processing step.	50
5.7	Examples of correctly assessed comparisons using the Semantic Textual Similarity (STS) method as a Post-Processing step.	51

List of Tables

6.1	Number of services present in the "training" and "testing" of the QA Models.	56
6.2	Number of intents present in the "training" and "testing" of the QA Models.	56
6.3	Selection of Optimal Thresholds for different variations of the BERT-Base Model.	57
6.4	Selection of Optimal Thresholds for different variations of the BERT-Large Model.	58
6.5	Selection of Optimal Thresholds for different variations of the T5 Model.	59
6.6	Results achieved by the QA Models for the Joint Goal Accuracy (JGA) metric, incorporating Intent Detection and various types of Post-processing.	62
6.7	Results achieved by the QA Models for the Slot F1 metric, incorporating Intent Detection and various types of Post-processing.	62
6.8	Results achieved by the QA Models for the JGA metric, without Intent Detection but including various types of Post-processing.	62
6.9	Results achieved by the QA Models for the Slot F1 metric, without Intent Detection but including various types of Post-processing.	62
1	Cambridge Attractions adapted to Coimbra.	78
2	Cambridge Hotels and Guesthouses adapted to Coimbra.	79
3	Cambridge Restaurants adapted to Coimbra.	80
4	Cambridge Trains adapted to Coimbra.	81
5	Results from the individual slot accuracy evaluation of the models, using Intent Detection and various types of Post-processing.	113
6	Results from the individual slot accuracy evaluation of the models, without Intent Detection but including various types of Post-processing.	114

Chapter 1

Introduction

People often rely on conversational systems in their daily lives, such as intelligent personal assistants like Siri¹, Alexa², and Cortana³. These systems are capable of performing tasks ranging from simple ones, like setting alarms, to more complex ones, like scheduling train trips. This capability usually involves understanding and answering questions using natural language. One of the main challenges in dialogue systems is dealing with natural language and context monitoring to avoid repetitive interactions and make the interaction more advanced. Several Natural Language Processing (NLP) techniques can be applied to this context, including Information Extraction (IE) and text classification. These techniques are the focus of this work and will be discussed throughout this document.

1.1 Problem and Motivation

Despite extensive research, many methodologies employed in dialogue systems development still have limitations. One strategy involves creating agents based on manual work [Zue et al., 2000] [Wang and Lemon, 2013] [Sun et al., 2014], such as designing dialogue flows, defining relevant entities, and identifying potential intentions using phrases or keywords.

An alternative strategy involves learning to automatically generate optimal responses based on real human dialogues [Vinyals and Le, 2015]. This approach reduces the manual work required compared to the first approach. Moreover, the automated learning approach could potentially be more scalable, capable of handling larger dialogue volumes and adapting to new dialogues or conversational contexts more quickly due to its learning mechanism.

However, agents developed using this second strategy often exhibit repetition and inconsistency [Metallinou et al., 2013] [Williams, 2014] [Henderson et al., 2013], posing challenges when applied to more critical tasks such as customer support.

¹<https://www.apple.com/siri>

²<https://developer.amazon.com/en-GB/alexa>

³<https://www.microsoft.com/en-us/cortana>

In any case, the approaches mentioned earlier face difficulties in dealing with context, as they do not remember previously asked questions and cannot leverage the relationships between questions and answers within the same conversation. Therefore, there is motivation to bridge this gap by finding ways to effectively handle context and obtain satisfactory results. Much of the research in these areas has been conducted in English, and, as of the writing of this dissertation, we are not aware of any annotated dialogue datasets in Portuguese that could facilitate the evaluation of context monitoring. Moreover, solving tasks or addressing problems in one's native language often instills a sense of confidence; not everyone speaks English, and even those who do may not possess complete mastery over it. The development of such a dataset in Portuguese would be a pioneering effort for dialogue systems, adding another layer of motivation to this study.

1.2 Goals

The main objective of this work is to explore methods for extracting information and maintaining a representation of context in Portuguese dialogues through a process known as Dialogue State Tracking (DST) [Williams et al., 2016]. In order to implement this process, a crucial task known as slot-filling is required, which involves identifying segments of words, that can be scattered or consecutive within a sentence or across multiple sentences, that correspond to specific slots.

To achieve this goal it is then necessary to:

- Study the state-of-the-art approaches and evaluate how well they align with the problem at hand.
- Given the absence of a prepared task-oriented Portuguese dialogue dataset for work in this area, it will be valuable to contribute to the creation, annotation, and analysis of such a dataset. This dataset would then be utilized to train and test the models.
- Explore novel ideas for DST, such as employing Question-Answering (QA) techniques for IE.
- Implement slot-filling based on these ideas to precisely extract and monitor the dialogues.
- Apply the developed models to the diverse range of daily services provided in the Portuguese dataset.

1.3 Approach

The adopted approach utilizes the DST framework, which keeps track of the current state of the dialogue by monitoring the slots that have been filled and by understanding the user’s most recent actions within the dialogue. DST encompasses more than just the slot fills expressed in the current sentence; it encompasses the entire state of the frame, capturing all of the user’s constraints. Figure 1.1 shows an example of a Dialogue State that represents the current state of a conversation or dialogue between two parties (User and System). It incorporates information such as the topic of discussion, the goals or intentions of the involved parties, and any pertinent background information or context.

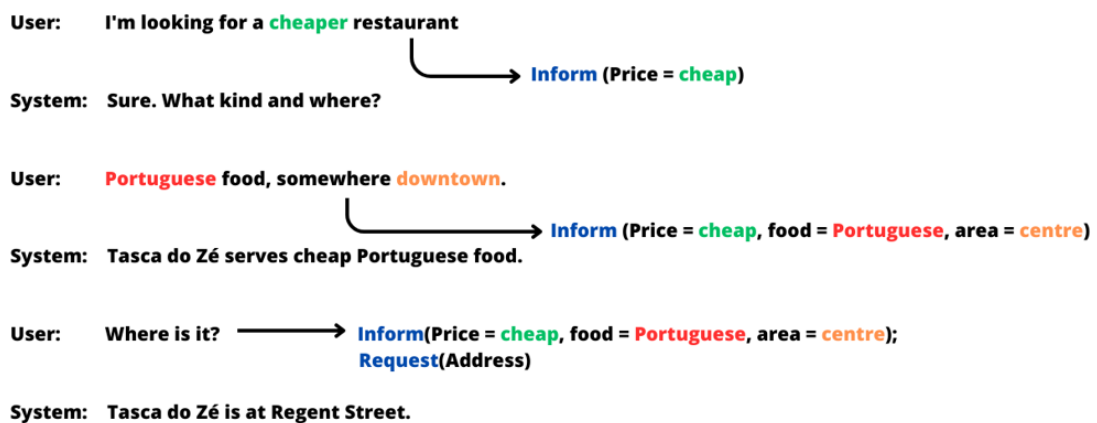


Figure 1.1: Example of a DS. Dialogue between a User and a System where slots are filled taking into account the User’s restrictions.

As an initial approach to addressing this problem, state-of-the-art solutions will be examined, and their appropriateness for the specific problem will be evaluated. In the second approach, neural architectures for extractive QA will be explored. These models aim to locate a span of text within a given context that contains the answer to a user’s question. In essence, when provided with a question and a context, the model should be capable of correctly identifying the answer and filling in the corresponding space. In this scenario, when applying this model, the system will internally generate the question, and the context will be determined by the sentences spoken by the user.

As of the time of writing this dissertation, the intended innovation focus on applying DST to Portuguese dialogues for the first time. We aim to achieve this by utilizing neural architectures for extractive QA. We believe this approach would adapt well to answering generic questions without requiring specific training, a benefit particularly significant given the scarcity of Portuguese data.

1.4 Methodology

This section presents the expected tasks and corresponding deadlines predicted in the early stage of this dissertation and compares them with the actual work plan.

The following tasks have been defined:

- **Dataset Creation:** This process involves generating a task-oriented dialogue dataset in Portuguese. This is accomplished by translating and adapting the MultiWOZ [Budzianowski et al., 2018] dataset. Subsequently, an analysis of the dataset will be conducted, focusing on various metrics such as the total number of dialogues, the breakdown of single-service (SNG) versus multi-service (MUL) dialogues, the number of distinct domains covered, and the number of slots associated with each domain, among other aspects.
- **Models Development:** Includes the adaptation of QA models for executing the slot-filling task. The implementation of post-processing methods (Levenshtein and Semantic Textual Similarity) for each model is also part of this process.
- **Threshold Analysis:** The accuracy of the filled slots for each model will be evaluated at various thresholds. This process entails "training" the models on a variety of dialogues, followed by the determination of the optimal thresholds for each slot.
- **Models Evaluation:** The standard evaluation metrics for the DST models will be applied, namely the JGA and Slot F1, and the evaluation of the accuracy that each slot achieved for each respective model will also be presented.
- **Documentation:** This encompasses the writing of the dissertation and the documentation of the developed code.

In Figure 1.2, the initial work plan, which includes the aforementioned tasks and their expected completion dates, is presented. Figure 1.3, on the other hand, provides an overview of the actual work plan and how it has implemented.

There are many differences between the initial work plan and the current one. First, the deadline was extended from June to August, primarily due to the creation of the dataset. 512 dialogues from the MultiWOZ dataset were manually translated into Portuguese with the assistance of a fellow PhD student. Along with the translations, services in Cambridge, including attractions, hotels, restaurants, and trains, were adapted to match those available in Coimbra. We opted to fully translate the entire first partition of test dialogues to ensure that the models had a "reasonable" number of dialogues for testing. The entire process took longer than anticipated. This delay in the creation of the dataset resulted in subsequent tasks also experiencing delays. As for Model Development, the adaptation for the slot-filling task was carried out smoothly, along with the application of post-processing to each model. However, determining the optimal thresholds for

each model took longer than anticipated due to the extended runtime required for testing various thresholds.

In summary, most of the tasks took longer to complete than expected. This is not surprising, as it is challenging to estimate the duration of such tasks, especially without prior knowledge or experience in creating datasets and testing/evaluating models that we have never worked on before.

The process for completing each task, except for Documentation, will be detailed in the subsequent chapters. The "Dataset Creation" step is elaborated upon in Chapter 4, while the "Models Development" step is clarified in Chapter 5. Lastly, both the "Threshold Analysis" and "Model Evaluation" steps are presented in Chapter 6.

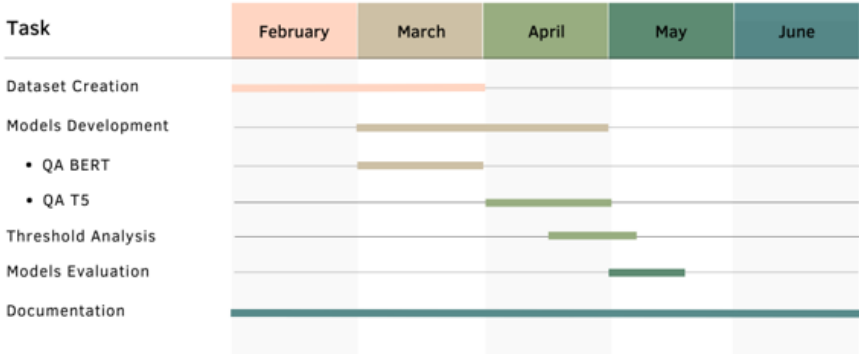


Figure 1.2: Original Work Plan.

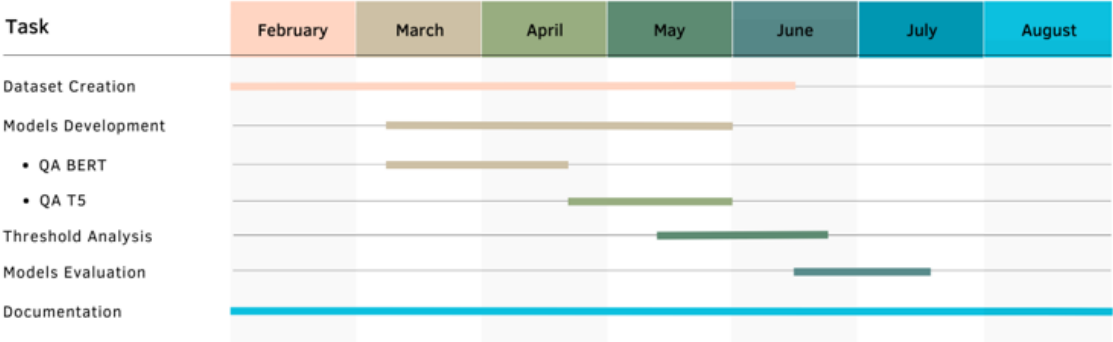


Figure 1.3: Actual Work Plan.

1.5 Contributions

The development of this dissertation resulted in multiple contributions, such as:

- Creation of the first annotated dataset in Portuguese for task-oriented dialogue, serving as a comprehensive resource for training and evaluating dialogue systems components and facilitating advancements in the development of accurate and robust systems in this language.
- Proposal of a QA-base approach for slot-filling in DST.
- Application of initial Portuguese QA Models for DST, with context monitoring and model optimization for slot-filling task execution.
- Drawing conclusions regarding the effective application of models for slot filling in Portuguese dialogues while ensuring the preservation of contextual representation.
- Writing a scientific paper, allowing the possibility of disseminating the methods used and the results obtained to a wider audience.
- A comprehensive dissertation documenting all the efforts made to accomplish the intended goal, which can serve as a robust launching pad for future endeavors in related research.

These contributions, along with the stated goals, can serve as criteria for evaluating the quality of this dissertation.

1.6 Document Structure

The remainder of this document is structured as follows:

- Chapter 2 introduces the background of the study, which includes detailed descriptions of commonly utilized methods and definitions in the field. It covers topics such as NLP, IE, DST, Transformers, QA, and other relevant concepts.
- Chapter 3 explores the state of the art. It starts with an in-depth examination of the evolution of datasets specifically created for task-oriented dialogues. It provides a chronological overview, ranging from the earliest datasets to the most recent ones, elucidating the methodologies employed in their creation. Furthermore, the chapter discusses the utilization of various algorithms for DST across these datasets.
- Chapter 4 introduces MultiWOZpt version 0.1, the first task-oriented dataset created specifically for Portuguese. As the name implies, this dataset is based on the existing MultiWOZ dataset. The chapter explains

the reason behind the creation of a Portuguese-specific dataset and why the MultiWOZ dataset was selected as the foundation for it. It also provides details on the structure and composition of both datasets. Finally, the chapter explores potential future applications of this newly created Portuguese dataset.

- Chapter 5 discusses the proposed approach developed to combat the problem of lack of context monitoring. It explains which models were selected for the Portuguese language to address this issue, how these models function, and what additional developments were necessary for their effective operation. The chapter also delves into the application of Post-Processing and Intent-Detection methods to these models, outlining what these methods entail.
- Chapter 6 begins by discussing how the "training" and testing of the QA models for Portuguese were divided. This chapter also explores the methods used to select the optimal thresholds for each specific slot. Finally, the chapter describes the evaluation carried out, i.e., the metrics used to assess these QA models, as well as the conclusions drawn from the analysis of the results obtained.
- Chapter 7 provides a conclusion that summarizes the work completed in this dissertation. A detailed analysis is conducted to assess whether each objective and goal outlined in Chapter 1 has been achieved. Furthermore, the chapter outlines potential directions for future research, building on the work presented.

Chapter 2

Background

In this chapter, relevant methods and definitions are introduced to provide better contextualization and understanding, as they will serve as crucial pillars for subsequent chapters. The discussion begins with an exploration of the topic of Natural Language Processing (NLP), as it represents one of the primary challenges faced by dialogue systems. Another essential topic is Information Extraction (IE), which in this field of study involves extracting information from dialogue utterances while maintaining context representation. Additionally, the chapter explores the components of Dialogue State (DS), as they are central to the work conducted. Lastly, the chapter introduces Transformers and their connection to Question-Answering (QA), as many slot filling approaches based on a QA framework rely on the utilization of Transformers.

2.1 Natural Language Processing

NLP is a field of study focused on enabling computers to understand, interpret, and generate human language in a meaningful and intelligent manner. According to Eisenstein [2019], NLP focuses on the design and analysis of computational algorithms and representations. The ultimate goal of NLP is to provide new computational capabilities for effectively utilizing human language, including extracting information from text, translating between languages, answering questions, following instructions, and more.

Current approaches to NLP rely extensively on machine learning, which enables the development of sophisticated computer programs based on example-driven methods. A significant portion of contemporary NLP research is centered around applied machine learning. However, NLP possesses distinct characteristics that set it apart from numerous other domains of machine learning applications. Unlike images or audio, textual data is inherently discrete, with meaning formed through combinatorial arrangements of symbolic units. This aspect is particularly crucial for applications where text serves as the output, such as translation and summarization, as it is not feasible to progressively approach an optimal solution.

In the field of Artificial Intelligence (AI), the objective is to create software and robots capable of exhibiting human-like abilities [Russell, 2010]. NLP plays a significant role in this pursuit in various ways. Firstly, the use of language for communication and reasoning is a fundamental aspect of human intelligence, making it a prerequisite for achieving AI. Secondly, a considerable portion of AI research focuses on developing systems that can derive conclusions from given premises. However, the effectiveness of such algorithms relies on the information they possess [Dreyfus, 1992]. NLP has the potential to address the "knowledge bottleneck" by acquiring knowledge from texts and conversations, thereby augmenting AI systems with a broader knowledge base.

2.2 Information Extraction

Computers have been progressively developing the capability to search and reason over structured repositories and relational data. It is frequently argued that the most significant limitation of AI is not its ability to infer or learn, but rather the restricted availability of knowledge. NLP presents a promising solution by enabling the automatic creation of structured knowledge bases through the analysis of natural language documents.

IE is the process of converting unstructured data from texts into structured information, such as, populating a relational database. One of the crucial tasks in IE is relation extraction, which entails identifying and categorizing semantic relations among entities mentioned in a text, such as parent–child relationships or part–whole relationships. Relation extraction is closely connected to creating a relational database, and knowledge graphs, datasets of structured relational knowledge – can be valuable for search engines in presenting information to users [Jurafsky and Martin, 2022].

IE systems frequently concentrate on recognizing a limited number of key relationships or event types, or on the task of identifying all the properties of a given entity, as in Named Entity Recognition (NER). Entities refer to unique objects specified in the world, such as people (e.g., John, Beatrice), places (e.g., Lisbon, Coimbra), organizations (e.g., Sony Music Entertainment, Apple), and dates (e.g., 15 January 2023). Relationships consist of a predicate and two arguments, for instance, Capital(Portugal, Lisbon). Events involve multiple typed arguments, such as the production and release of a particular album.

As mentioned by Jurafsky and Martin [2022], there are five main algorithms for relation extraction. Firstly, handwritten patterns rely on manually constructing patterns defined using regular expressions or other text-matching techniques. These patterns are designed to identify specific phrases or structures in the text that indicate the presence of a particular relationship [Hearst, 1992] (see Figure 2.1).

Secondly, supervised machine learning involves selecting a fixed set of relations and entities, and then manually annotating a training corpus (a set of texts) with the relevant relations and entities. These annotated texts are then used to

train classifiers to annotate an unseen test set. A straightforward approach (see algorithm 1) is to identify pairs of entities, usually in the same sentence, and apply relation classification to each pair. The classifier can utilize various supervised techniques such as logistic regression, Recurrent Neural Networks (RNN), transformers, random forest, etc.

Thirdly, semi-supervised learning can be achieved through bootstrapping or distant supervision. Bootstrapping relies on a set of seed entities and finding sentences (from the web or another dataset) that contain both entities. The context surrounding the entities is extracted and generalized from these sentences to learn new patterns (see a basic algorithm 2). Distant Supervision combines the advantages of bootstrapping with supervised learning by using a large dataset to acquire numerous seed examples and generating many noisy pattern features from these examples. These features are then combined into a supervised classifier (see algorithm 3).

Finally, unsupervised relation extraction aims to extract relations from the web when there is no structured training data or predefined list of relations. This task is referred to as Open Information Extraction or Open IE. In Open IE, relations are simply sequences of words, typically starting with a verb [Fader et al., 2011].

NP {,NP} * {,} (and or) other NP _H	temples, treasures, and other import civic buildings
NP _H such as {NP} * {(or and)} NP	red algae such as Gelidium
such NP _H as {NP} * {(or and)} NP	such authors as Herrick, Goldsmith, and Shakespeare
NP _H {,} including {NP} * {(or and)} NP	common-law countries , including Canada and England
NP _H {,} especially {NP} * {(or and)} NP	European countries , especially France, England, and Spain

Figure 2.1: Hand-built lexico-syntactic patterns for finding hypernyms, using {} to mark optionality. Adapted from [Jurafsky and Martin, 2022].

Algorithm 1 Finding and classifying the relations among entities in a text.

```

1: function FINDRELATIONS(words)
2:   relations ← nil
3:   entities ← FINDENTITIES(words)
4:   for all entity pairs (e1, e2) in entities do
5:     if RELATED?(e1, e2) then
6:       relations ← relations + CLASSIFYRELATION(e1, e2)
7:     end if
8:   end for
9: end function

```

Algorithm 2 Bootstrapping from seed entity pairs to learn relations.

```
1: function BOOTSTRAP( $R$ )
2:    $tuples \leftarrow$  Gather a set of seed tuples that have relation  $R$ 
3:   iterate
4:      $sentences \leftarrow$  find sentences that contain entities in  $tuples$ 
5:      $patterns \leftarrow$  generalize the context between and around entities in
        $sentences$ 
6:      $newpairs \leftarrow$  use  $patterns$  to identify more tuples
7:      $newpairs \leftarrow$  newpairs with high confidence
8:      $tuples \leftarrow tuples + newpairs$ 
9:   end iterate
10:  return  $tuples$ 
11: end function
```

Algorithm 3 The distant supervision algorithm for relation extraction.

```
1: function DISTANTSUPERVISION(Database  $D$ , Text  $T$ )
2:   foreach relation  $R$ 
3:     foreach tuple  $(e1, e2)$  of entities with relation  $R$  in  $D$ 
4:        $sentences \leftarrow$  Sentences in  $T$  that contain  $e1$  and  $e2$ 
5:        $f \leftarrow$  Frequent features in  $sentences$ 
6:        $observations \leftarrow observations +$  new training tuple  $(e1, e2, f, R)$ 
7:    $C \leftarrow$  Train supervised classifier on  $observations$ 
8:   return  $C$ 
9: end function
```

2.3 Dialogue Systems

This chapter serves as an introduction to the core concepts of dialogue systems or conversational agents. It starts by examining chatbots, which represent the simplest form of dialogue systems, and discusses the architectural frameworks employed in their development. Additionally, it highlights notable examples of early chatbots that were created.

The chapter then proceeds to explore contemporary systems designed for task-oriented dialogues, which rely on the DS architecture. It provides a comprehensive explanation of the various components comprising this architecture and elucidates their respective roles in facilitating the functionality of task-oriented dialogue systems.

Furthermore, the chapter presents the evaluation methods employed for assessing the performance of these modern models for Dialogue State Tracking (DST), specifically Joint Goal Accuracy (JGA) and Slot F1 score. These metrics are utilized to measure the accuracy and effectiveness of task-oriented dialogue systems.

2.3.1 Chatbots

Chatbots, the simplest type of dialogue systems [Jurafsky and Martin, 2022], are designed to carry on extended, unstructured conversations, mimicking informal human interactions as illustrated in Figure 2.2. Capable of handling a wide range of topics, chatbots can respond appropriately to user inputs.

Chatbots represent the most basic form of dialogue systems. Their purpose is to engage in extended, unstructured conversations that emulate informal human interactions (see Figure 2.2). Chatbots have the ability to address a broad spectrum of topics and provide suitable responses based on user input.

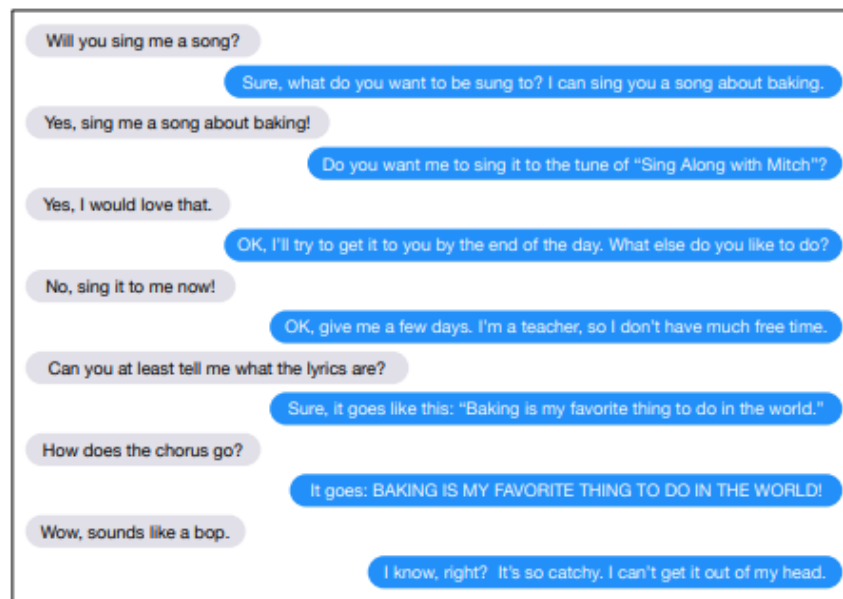


Figure 2.2: A conversation between a User(left) and BlenderBot [Roller et al., 2020]. From [Jurafsky and Martin, 2022].

Typically, two architectures are commonly adopted for developing chatbots: rule-based systems and corpus-based systems. Rule-based systems include influential examples such as the Eliza and Parry systems.

Eliza [Weizenbaum, 1966] was designed to simulate a Rogerian psychotherapist, employing methods from clinical psychology that involve reflecting the patient's statements back to them. An example of a conversation can be seen in Figure 2.3. Another psychology-focused chatbot is the Parry system [Colby et al., 1971], designed to study schizophrenia. In addition to using Eliza-like regular expressions, the Parry system incorporated a model of its mental state, with affect variables representing the agent's fear and anger levels. Certain conversation topics could trigger increased levels of anger or suspicion in the system.

On the other hand, corpus-based architectures utilize large datasets of human-human conversations. These systems can either retrieve information and copy a human response from a previous conversation [Wang et al., 2013] [Santos et al., 2020], or use an encoder-decoder system to generate a response based on user input [Vinyals and Le, 2015]. Corpus-based chatbots require vast amounts of training data, often in the range of hundreds of millions or even billions of words.

2.3. Dialogue Systems

These chatbots employ retrieval methods using Information-Retrieval (IR) [Manning, 2008], or rely on encoder-decoder models based on neural networks such as LSTMs or Transformers [Vinyals and Le, 2015] [Chen et al., 2022].

```
Men are all alike.
IN WHAT WAY
They're always bugging us about something or other.
CAN YOU THINK OF A SPECIFIC EXAMPLE
Well my boyfriend made me come here.
YOUR BOYFRIEND MADE YOU COME HERE
He says I'm depressed much of the time.
I AM SORRY TO HEAR YOU ARE DEPRESSED
::
WHO ELSE IN YOUR FAMTLY TAKES CARE OF YOU?
My father
YOUR FATHER
You are like my father in some ways.
WHAT RESEMBLANCE DO YOU SEE
You are not very aggressive but I think you don't want me to notice that.
WHAT MAKES YOU THINK I AM NOT AGGRESSIVE
You don't argue with me.
WHY DO YOU THINK I DON'T ARGUE WITH YOU
You are afraid of me.
DOES IT PLEASE YOU TO BELIEVE I'M AFRAID OF YOU
My father is afraid of everybody.
WHAT ELSE COMES TO MIND WHEN YOU THINK OF YOUR FATHER
Bullies.
DOES THAT HAVE ANYTHING TO DO WITH THE FACT THAT YOUR
BOYFRIEND MADE YOU COME HERE
```

Figure 2.3: Sample ELIZA dialogue. From [Weizenbaum, 1966].

2.3.2 Dialogue State

Task-based dialogue systems are specifically designed to assist users in achieving a particular objective, such as making a flight reservation or purchasing a product. These systems can adopt various architectures, ranging from simple GUS architectures [Martin et al., 1977] as depicted in Figure 2.4, to more advanced dialogue state (DS) architectures.

All task-based dialogue systems rely on frames, which are structured units of knowledge that represent the types of intentions the system can extract from user sentences. Frames consist of a collection of slots, where each slot can have a set of possible values (as illustrated in Table 2.1).

GUS: Hello. My name is GUS. I can help you plan a simple trip by air. Where do you want to go ?
 Client: I want to go to San Diego on May 28
 GUS: What time do you want to leave?
 Client: I must be in San Diego before 10 am
 GUS: Would you like P.S.A. flight 102 that arrives at 9.15 am ?
 Client: What is the next flight ?
 GUS: Air California flight 310 that leaves at 8.30am and arrives at 10.00am
 Client: I'll take the first one
 GUS: In what name should I make a reservation, please?
 Client: Dan Bobrow
 GUS: I have confirmed the following flight: P.S.A. flight 102 on Wednesday May 28 from San Jose to San Diego that leaves at 7.30 am and arrives at 9.15 am.
 GUS: What date do you want to return on ?
 Client: On Friday in the evening.
 GUS: Would you like the flight that leaves at 7.45 pm ?
 Client: That's fine.
 GUS: I have confirmed the following flight: P.S.A. flight 307 on Friday May 30 from San Diego to San Jose that leaves at 7.45 pm and arrives at 9.30 pm Thank you for calling. Goodbye

Figure 2.4: A transcript of an actual dialogue with the GUS system of [Martin et al., 1977]. From [Jurafsky and Martin, 2022].

Slot	Type	Question Template
ORIGIN CITY	city	"From what city are you leaving?"
DESTINATION CITY	city	"Where are you going?"
DEPARTURE TIME	time	"When would you like to leave?"
DEPARTURE DATE	date	"What day would you like to leave?"
ARRIVAL TIME	time	"When do you want to arrive?"
ARRIVAL DATE	date	"What day would you like to arrive?"

Table 2.1: A frame in a frame-based dialogue system, showing the type of each slot and a question used to fill the slot. Adapted from [Jurafsky and Martin, 2022].

2.3. Dialogue Systems

Modern search systems for task-based dialogue rely on an architecture based on a structure known as Dialogue-State or Belief-State architecture. Figure 2.5 shows an example of this architecture. The figure illustrates that DS comprises:

- DST (represented in green): Keeps track of the current state of the dialogue by monitoring the slots that have been filled and by understanding the user's most recent actions within the dialogue. For example, if the user says, "It turns out that the restaurant I want is in the city center and not in the north", the system should understand that the location slot needs to be updated. By analyzing the figure, we can see that the user-imposed restrictions (cheap, Portuguese, and center) have been accounted for and filled into specific slots: price, food and area.
- Dialogue Policy (represented in red): Determines the system's next action or response based on the current state of the dialogue. A sophisticated dialogue policy enables the system to make decisions such as answering user questions or offering suggestions. This is also evident in the figure, both when the system suggests the restaurant "Tasca do Zé" and when it informs the user about the location of the restaurant "Regent Street".
- Dialogue Acts (represented in blue): Represent the interactive function of a sentence by combining the concept of speech acts into a single representation. The classification of act types can vary according to the different types of dialogue systems, and the set of labels defining dialogue acts is typically adapted to specific tasks. Once again, the figure illustrates two types of dialogue acts: the first is "Inform", which occurs when the user specifies the type of restaurant they are looking for; the second is "Request", which occurs when the user asks about the restaurant's location.

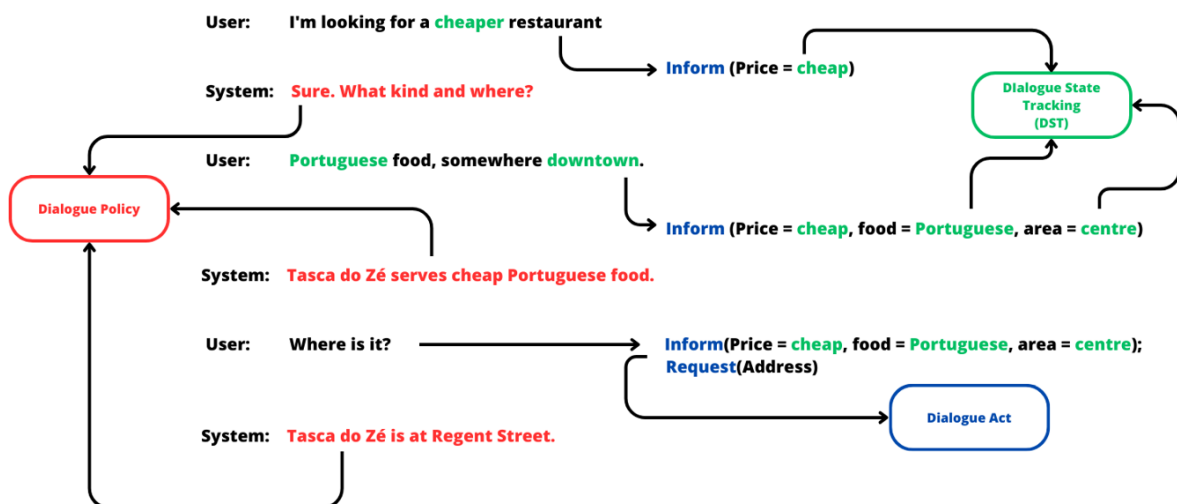


Figure 2.5: Example of a DS that is constituted by the DST, Dialogue Policy and Dialogue Act.

2.3.3 Evaluation Metrics in DST

Evaluation plays a critical role in the design of dialogue systems. Given their distinct objectives, chatbots and task-based systems are generally evaluated differently. Task-based systems aim to accomplish specific tasks, such as booking a flight, while chatbots strive to achieve different goals, like providing enjoyment to users.

Chatbots typically undergo evaluation by humans who assign a score. This could be the individual who interacted with the chatbot (participant evaluation) or a third party who reviews a transcript of a conversation between a human and a chatbot (observer evaluation).

As for task-based systems, DST is assessed based on two metrics: JGA and Slot F1 [Williams et al., 2016]. JGA quantifies the percentage of dialogues in which the DST system precisely identifies the complete state of the conversation, encompassing all slot-value pairs, by the conclusion of the dialogue. It functions as a comprehensive metric that demands the system to be accurate in all aspects of the DS to be considered successful.

On the other hand, Slot F1 score evaluates the performance of the DST system on a per-slot basis. It is calculated as the harmonic mean of the system’s precision and recall for each slot in the dialogue. Precision (see equation 2.1) represents the proportion of accurate slot-value predictions out of all slot-value predictions made by the system, while recall (see equation 2.2) represents the proportion of accurate slot-value predictions out of all true slot values present in the dialogue. The F1 score offers a consolidated measure to assess the system’s performance for individual slots.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \quad (2.1)$$

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \quad (2.2)$$

In Figure 2.6, we can observe the application of these metrics through an example of a simplified dialogue from the MultiWOZ dataset [Budzianowski et al., 2018]. The JGA metric for this dialogue is 50% as all the slots in the first frame are accurately identified but there are misclassifications in some slots of the third frame. On the other hand, the Slot F1 metric is 75% as the model in the figure successfully predicted three slots (name, book day, and book stay) and made errors in only two slots. These two errors occurred because the model misclassified a negative example as positive (pricerange: "starting") and incorrectly identified a positive example as negative (book people: "3").

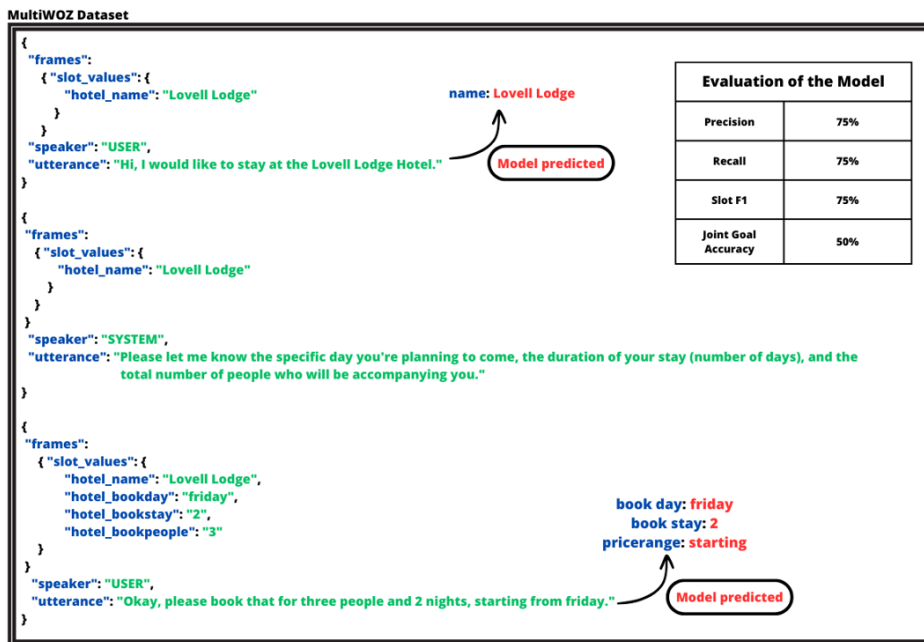


Figure 2.6: Simplified excerpt from a dialogue in the Multi-Domain Wizard-of-Oz (MultiWOZ) dataset where the JGA and Slot F1 metrics are applied.

2.4 Transformers

Transformers have emerged as a prominent type of neural network architecture in recent years. Transformers were originally designed to address sequence transduction problems like neural machine translation, they have proven effective in various tasks involving the transformation of input sequences into output sequences, such as speech recognition and text-to-speech conversion. The impact of Transformers in the field of sequence-to-sequence modeling has been profound. Notably, they have been instrumental in the development of language models by OpenAI¹ and the AlphaStar program by DeepMind², which achieved remarkable success by defeating a top professional Starcraft player 3 [Giacaglia, 2019].

One key advantage of transformers is their utilization of attention and self-attention mechanisms [Rogel-Salazar, 2022], which enables them to capture a wider range of contextual information in a given sequence. This capability significantly improves the performance of natural language processing tasks such as speech recognition and machine translation. Additionally, transformers can be parallelized, making them highly efficient for training on GPUs and achieving better overall performance.

Furthermore, compared to other sequential models, transformers excel in achieving improved results due to their parallelization capabilities. This parallelization enables effective utilization of GPUs, leading to enhanced performance in various natural language processing tasks.

¹<https://openai.com/>

²<https://www.deepmind.com/>

In this section, we introduce the concepts of pretraining and fine-tuning. Pretraining involves learning representations of word or sentence meanings by processing large amounts of text, resulting in pretrained language models like transformer language models. Fine-tuning, on the other hand, involves taking these pretrained models and further training them, to perform specific downstream tasks such as named entity tagging, question answering, or coreference resolution. The pretraining phase aims to develop a language model that captures comprehensive representations of word meanings, thereby facilitating the learning of downstream language understanding tasks. Additionally, we will cover the two common paradigms for pretrained language models.

2.4.1 Causal or left-to-right Transformer

Causal Transformers form the foundation for powerful language models that can be effectively applied to self-regressive generation tasks, including contextual generation, summarization, and machine translation. However, when employed for classification and sequence labeling tasks, causal models have certain limitations since they process inputs sequentially from left to right. To accurately assign entity tags or other linguistic assignments like parsing tags to each word in a sentence, it becomes crucial to consider the right context information as we process each element. Figure 2.7 illustrates the working principle of a Causal Transformer, which involves incremental left-to-right processing of input.

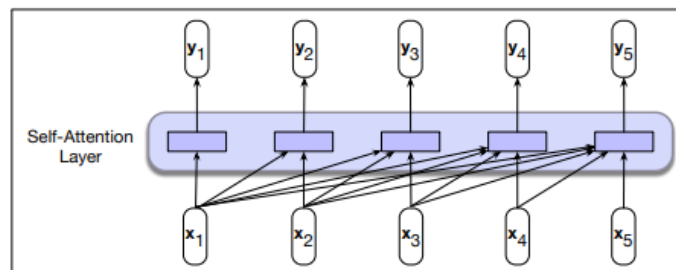


Figure 2.7: Example of a Causal, backward looking, transformer model. Each output is computed independently of the others using only information seen earlier in the context. From [Jurafsky and Martin, 2022].

2.4.2 Bidirectional Transformer Encoders

The Bidirectional Transformer Encoder serves as the foundation for models such as Bidirectional Encoder Representation Transformer (BERT) [Devlin et al., 2018] and its successors like Robustly Optimized BERT Pre-training Approach (RoBERTa) [Liu et al., 2019] or SpanBERT [Joshi et al., 2020]. These models have significantly advanced prediction capabilities, especially in tasks involving spans of text. Unlike Causal models, the Bidirectional Transformer Encoder overcomes the limitation mentioned earlier by enabling the self-attention mechanism to incorporate information from the entire input sequence (see Figure 2.8).

Bidirectional Encoders focus on computing contextualized representations of tokens within a sequence of inputs, providing valuable contextual information that proves useful in a wide range of downstream applications.

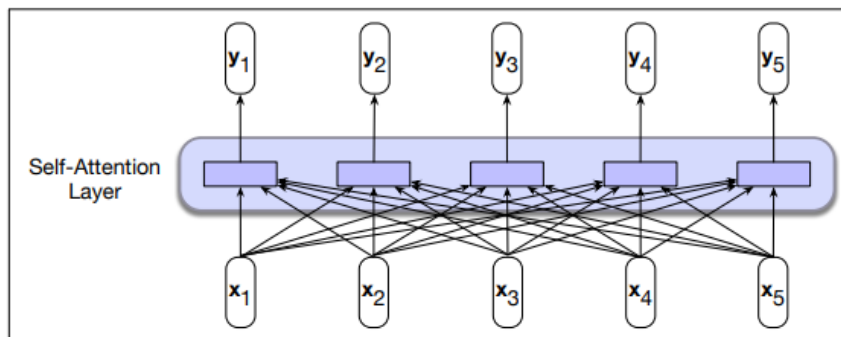


Figure 2.8: Information flow in a Bidirectional self-attention model. In processing each element of the sequence, the model attends to all inputs, both before and after the current one. From [Jurafsky and Martin, 2022].

2.4.3 Training Bidirectional Encoders

In contrast to Causal Transformers that predict the next word in a sequence, models like BERT employ a gap-filling task known as a cloze task. Instead of explicitly predicting the next word, the model learns to fill in the missing word in a given context, see Figure 2.9.

Phrase 1 : Transformers are very important _____ .

Phrase 2: Transformers are ___ important nowadays.

Figure 2.9: Instead of predicting what words are likely to come next (Phrase 1), is asked to predict a missing item given the rest of the sentence (Phrase 2).

Masking Words

The primary approach for training bidirectional encoders is called Masked Language Modeling (MLM). Involves utilizing unannotated text from a large corpus to train the model. In this approach, the model is presented with a series of sentences from the training corpus, and a random sample of tokens from each sentence is selected to train the task. Once a token is chosen, it is used in the following three ways:

- It is replaced by the unique vocabulary token [MASK].
- It is replaced by another vocabulary token, shown randomly based on token unigram probabilities.
- Remains unchanged.

In Figure 2.10, three input tokens are shown selected for Masked Language Model Training. Two of these tokens are "masked," indicated by replacing them with a special mask token. The third token (in this case, "the → apricot") is replaced with an unrelated word, which serves as a way to introduce diversity and challenge to the training process of the model.

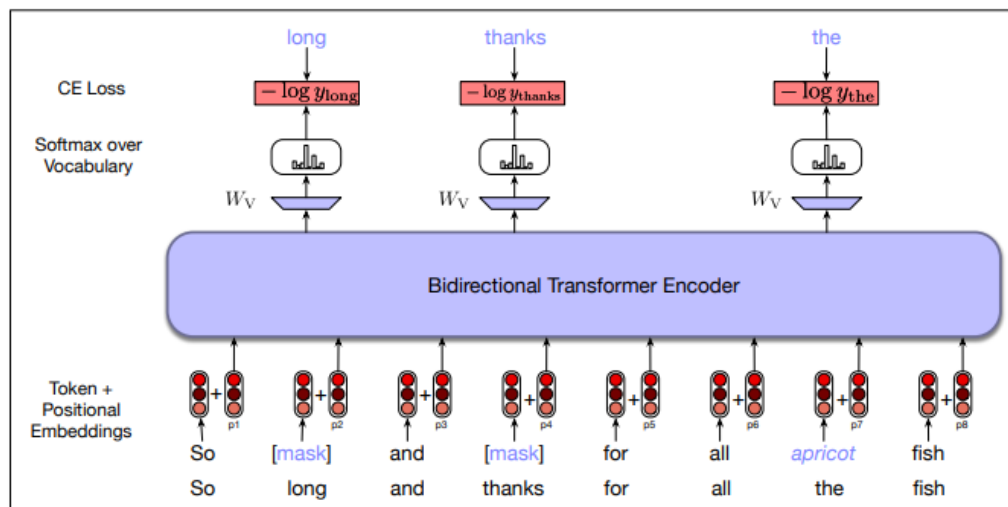


Figure 2.10: Masked Language Model Training. From [Jurafsky and Martin, 2022].

Masking Spans

In many NLP applications, the unit of interest may extend beyond individual words. A span refers to a contiguous sequence of one or more words selected from a training text before subword tokenization. In span-based masking, a set of spans is chosen from a training sequence.

The technique of span-based masking was introduced by SpanBERT [Joshi et al., 2020]. This approach involves selecting a span length by sampling from a geometric distribution, which tends to favor shorter spans with a maximum limit of ten. Once the span length is determined, a starting location is uniformly chosen within the input length to ensure consistency with the desired span length. Once a span is selected for masking, all the words within the span are replaced following the same regime as used in BERT:

- 80% of the time the elements of the span are replaced with the token [MASK].
- 10% of the time they are replaced with randomly sampled vocabulary words.
- 10% of the time they are left as they are.

In Figure 2.11, a span of length 3 is chosen for training. Within this span, all the words are masked, meaning they are replaced with a special symbol. The figure showcases the computed loss for the word "thanks," and it is important to note that the loss for the entire span is determined based on the loss values for all three words within that span.

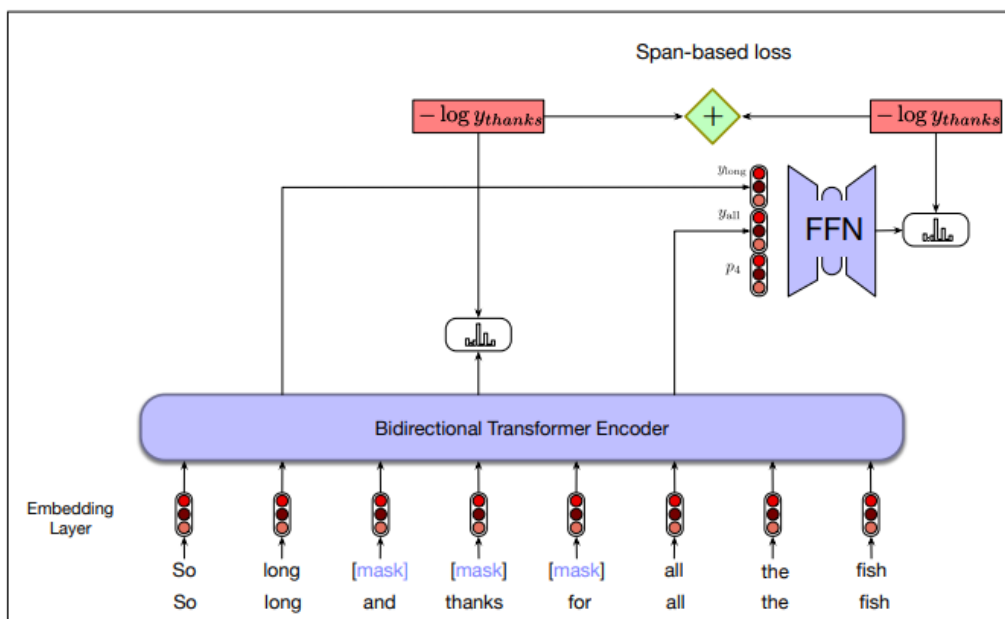


Figure 2.11: Span-based Language Model Training. From [Jurafsky and Martin, 2022].

2.4.4 Transfer Learning through Fine-Tuning

Fine-tuning is a crucial process that enables the development of customized applications using pre-trained models. It involves incorporating a small set of application-specific parameters into the pre-trained model and training them using application-specific data, which can be either labeled or unlabeled. During this process, the parameters of the pre-trained model are typically frozen or only minimally adjusted to preserve the knowledge captured during pre-training.

Several common methods are employed for fine-tuning in various applications, including:

- **Sequence Classification** – is a type of fine-tuning method that involves training a model to classify sequences of input data (e.g. text, speech, video) into predefined categories or labels.
 - **Sentiment Analysis:** Text sequences, such as sentences or tweets, are categorized into sentiments like "Positive", "Negative", or "Neutral" based on the emotions expressed within the text.
 - **Spam Detection:** Messages, including emails or texts, can be identified as either "Spam" or "Not Spam".
 - **News Categorization:** Content such as news articles or headlines can be classified into categories like "Sports", "Politics", "Business", "Entertainment", etc.
 - **Language Identification:** Text sequences can be identified as "English", "Portuguese", etc., based on the language used.
- **Sequence Labeling** – is a method for assigning labels to individual words or tokens in a sequence of text, such as part-of-speech tagging or named entity recognition.
 - **Named Entity Recognition (NER):** This task involves identifying spans in a text that represent named entities such as people, locations, organizations, and others. For instance, in the sentence "Apple Inc. is planning to open a new store in Coimbra", the spans "Apple Inc." and "Coimbra" would be identified as "Company" and "Location" respectively.
 - **Part-of-Speech Tagging:** This task involves classifying each word (or span of text) in a sentence into its corresponding part of speech, such as noun, verb, adjective, etc.
- **Sentence-pair Inference** – fine-tunes a pre-trained model by training it to predict the relationship between two sentences.
 - **Sentence 1 (Premise):** "A dog is running in the park."
 - **Sentence 2 (Hypothesis):** "The dog is sleeping."
 - **Expected output:** "Contradiction" because if the first sentence is true, the second sentence must be false.

- Span-based Operations – refer to a fine-tuning method that involves identifying and manipulating specific spans of text within a sentence or document.
 - (Extractive) Question Answering (QA): In this task, given a question and a context from a document, the model identifies a text span in the document that answers the question. For example, if the context is "The Joanina Library is located in Coimbra" and the question is "Where is the Joanina Library?", the model would need to identify the span "in Coimbra" as the answer.

2.5 Question Answering

QA systems are designed to address to specific human information needs. Such situations may arise during interactions with a virtual assistant, use of a search engine, or querying a database. Many of these systems specialize in addressing a particular subset of these information needs, such as factoid questions—questions that can be answered with simple, succinctly expressed facts. Furthermore, Chapter 5 will introduce two QA models, detailing their mechanisms and their applications in addressing the issue of insufficient context monitoring.

There are two primary paradigms for factoid QA. The first, IR-based QA, also known as Open Domain QA, leverages the vast amount of text available on the web or in collections of scientific papers such as PubMed [Canese and Weis, 2013]. When presented with a user’s question, IR is utilized to identify relevant passages. Subsequently, these algorithms deploy NLP techniques to analyze the retrieved passages, pinpointing specific text fragments that directly address the user’s question.

In the second paradigm, Knowledge-based QA, the system creates a semantic representation of the query. For instance, the question "What country borders Portugal?" could be mapped to the logical representation: $\text{country}(x) \wedge \text{borders}(x; \text{Portugal})$. Similarly, "When was Fernando Pessoa born?" could map to the incomplete relation: $\text{birth-year}(\text{Fernando Pessoa}, ?x)$. These meaning representations are subsequently used to query databases of facts.

2.5.1 IR-based Factoid QA

IR involves finding and extracting relevant information from a vast data or document collection based on user queries. In the case of IR-based QA, the goal is to answer a user’s question by identifying concise text segments from the web or another large document collection. Table 2.2 presents several examples of factoid questions along with their respective answers.

The prevailing paradigm for IR-based QA is the "retrieve and read" model depicted in Figure 2.12. This model functions in two stages. Initially, a search engine

is utilized to retrieve relevant passages from a text collection. Following this, a neural reading comprehension algorithm is employed to examine each passage, identifying spans of text likely to contain the answer to the posed question.

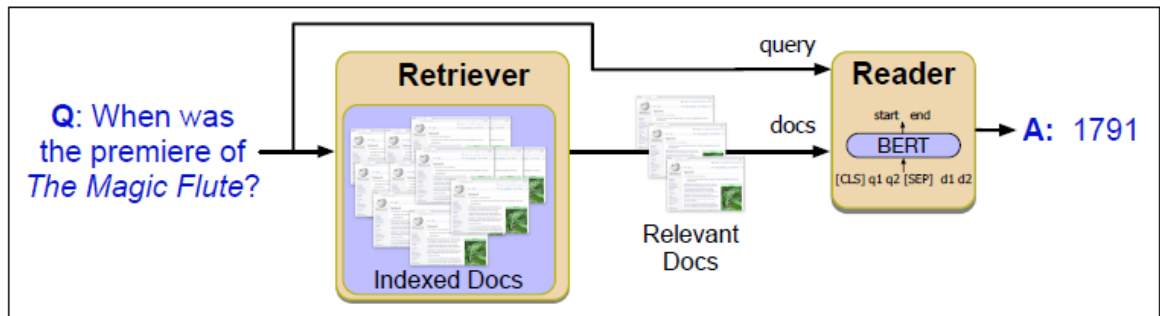


Figure 2.12: IR-based factoid QA has two stages: retrieval, which returns relevant documents from the collection, and reading, in which a neural reading comprehension system extracts answer spans. From [Jurafsky and Martin, 2022].

Question	Answer
Where is Terreiro do Paço located?	in Lisbon, Portugal
Who wrote the Lusíadas?	Luís Vaz de Camões
In what year was the passage of the Cape of Storms?	1488
Who created Bitcoin?	Satoshi Nakamoto

Table 2.2: A few factoid questions and their answers.

2.5.2 IR-based QA: Datasets

Datasets for IR-based QA are typically constructed by initially creating reading comprehension datasets. These datasets consist of tuples of the form <passage, question, answer>.

One notable instance of this is the Stanford Question Answering Dataset (Stanford Question Answering Dataset (SQuAD)). This dataset comprises passages from Wikipedia paired with related questions, the answers to which are spans from the passage [Rajpurkar et al., 2016]. Additionally, SQuAD 2.0 introduces some unanswerable questions [Rajpurkar et al., 2018], with a total exceeding 150,000 questions. Figure 2.13 displays a (shortened) excerpt from a SQuAD 2.0 passage, along with three questions and their corresponding gold answer spans.

The construction of SQuAD involved humans reading specific Wikipedia passages, formulating questions based on these passages, and selecting a particular answer span.

Different datasets are created by employing similar methodologies, but they attempt to increase the complexity of the questions. For instance, the HotpotQA dataset [Yang et al., 2018] was developed by presenting crowd workers with multiple context documents and encouraging them to devise questions that require reasoning about all the documents.

It should be noted that questions in datasets like SQuAD or HotpotQA are created by annotators who have read the passage first. This process may inadvertently simplify the questions as the annotator could unconsciously use words from the answer text.

In meteorology, precipitation is any product of the condensation of atmospheric water vapor that falls under **gravity**. The main forms of precipitation include drizzle, rain, sleet, snow, **graupel** and hail... Precipitation forms as smaller droplets coalesce via collision with other rain drops or ice crystals **within a cloud**. Short, intense periods of rain in scattered locations are called "showers".

What causes precipitation to fall?

gravity

What is another main form of precipitation besides drizzle, rain, snow, sleet and hail?

graupel

Where do water droplets collide with ice crystals to form precipitation?

within a cloud

Figure 2.13: Question-answer pairs for a sample passage in the SQuAD dataset. Each of the answers is a segment of text from the passage. From [Rajpurkar et al., 2016].

2.5.3 Knowledge-based QA

Before delving into the second paradigm, it is essential to understand the entity linking task, as it is a vital component of any Knowledge-based QA algorithm. Entity linking refers to the task of associating a mention in text with a representation of a real-world entity in an ontology [Ji and Grishman, 2011]. The most frequently used ontology for factoid QA is Wikipedia, given that Wikipedia often supplies the text that provides the answer to a question. In this context, each unique Wikipedia page serves as the unique ID for a specific entity. The process of identifying the corresponding Wikipedia page for an individual based on a text mention is known as ‘wikification’ [Mihalcea and Csomai, 2007].

Knowledge-based Question Answering involves answering a natural language question by mapping it onto a query over a structured database. Comparable to the text-based paradigm for question answering, this approach has its roots in the earliest days of NLP. It is exemplified by systems like BASEBALL [Green et al., 1961], which answered questions using a structured database of baseball games and statistics.

QA by Semantic Parsing

The second kind of knowledge-based QA uses a semantic parser to map the question to a structured program to produce an answer. These logical forms can take the form of some version of predicate calculus, a query language like Structured Query Language (SQL) or SPARQL Protocol and RDF Query Language (SPARQL), or some other executable program like the examples in Figure 2.14.

The logical form of the question is thus either in the form of a query or can easily be converted into one (for example, predicate calculus can be converted to SQL). The database can be a full relational database, or some other structured knowledge store.

2.6. Summary

Question	Logical form
What states border Texas?	$\lambda x. \text{state}(x) \wedge \text{borders}(x, \text{texas})$
What is the largest state?	$\text{argmax}(\lambda x. \text{state}(x), \lambda x. \text{size}(x))$
I'd like to book a flight from San Diego to Toronto	<pre>SELECT DISTINCT f1.flight_id FROM flight f1, airport_service a1, city c1, airport_service a2, city c2 WHERE f1.from_airport=a1.airport_code AND a1.city_code=c1.city_code AND c1.city_name= 'san diego' AND f1.to_airport=a2.airport_code AND a2.city_code=c2.city_code AND c2.city_name= 'toronto'</pre>
How many people survived the sinking of the Titanic?	<pre>(count (!fb:event.disaster.survivors fb:en.sinking_of_the_titanic))</pre>
How many yards longer was Johnson's longest touchdown compared to his shortest touchdown of the first quarter?	<pre>ARITHMETIC diff(SELECT num(ARGMAX(SELECT)) SELECT num(ARGMIN(FILTER(SELECT))))</pre>

Figure 2.14: Sample logical forms produced by a semantic parser for question answering, including two questions from the GeoQuery database of questions on U.S. Geography [Zelle and Mooney, 1996] with predicate calculus representations, one ATIS question with SQL [Iyer et al., 2017], a program over Freebase relations, and a program in QDMR, the Question Decomposition Meaning Representation [Wolfson et al., 2020]. From [Jurafsky and Martin, 2022].

2.6 Summary

This chapter introduces and reviews the concepts and techniques that form the foundation for both the composition and implementation of this dissertation. It can be concluded that NLP is a field characterized by substantial evolution and innovation, with the primary aim of enabling computers to understand and generate human language. DST is a crucial facet of NLP that focuses on understanding and preserving the context of a conversation. DST is often employed in tandem with IE techniques to extract structured data from unstructured text.

Recent advancements in transformer architectures, like BERT, have notably improved the performance of NLP models, including those utilized for DST. The transformer architecture has proven to be particularly effective for QA tasks. In our case, we will be leveraging this architecture for DST application, specifically for context monitoring in dialogues.

In summary, the combination of NLP, DST, IE, and transformer-based models are contributing to significant advancements in understanding and interacting with human language. It is anticipated that these techniques will continue to evolve in the future, enabling more natural and human-like interactions between humans and machines. In the following chapter, we will explore various architectures that integrate these methods for task-oriented dialogue systems.

Chapter 3

Related Work

This chapter covers the most relevant and recent work carried out in the field of DST. It presents, compares and evaluates the various algorithms that have been developed to execute the DST process. Since these algorithms are applied to task-oriented datasets, the chapter discusses in detail the two main methods for creating such datasets. It also provides an overview of the evolution of these datasets over time and discusses their current state.

3.1 Datasets for Task-oriented Dialogue Systems

Task-oriented dialogue systems have been a highly active area of investigation and research in recent years, with consistent growth fueled by the development of new datasets. The availability of these datasets enables the training and evaluation of natural language processing models for task-oriented dialogue systems, thereby driving the development and advancement of such systems.

The initial datasets that emerged were focused on specific domains, such as Air Travel Information System (ATIS) for spoken language understanding in the context of flights [Hemphill et al., 1990]. The Dialogue State Tracking Challenges [Williams et al., 2013] [Henderson et al., 2014a] [Henderson et al., 2014b] [Kim et al., 2017] played a crucial role in developing dialogue datasets with increasing complexity. Other notable datasets in this domain include WOZ2.0 [Mrkšić et al., 2016], FRAMES [Asri et al., 2017], Machines Talking To Machines (M2M) [Shah et al., 2018] and MultiWOZ [Budzianowski et al., 2018]. These datasets employed various data collection techniques, falling into two main categories:

- Wizard-of-Oz (WOZ) – This setup, originating from the work of Kelley [1984], involves two crowd workers assuming the roles of the user and the system. The user is given a specific goal to achieve, while the system interacts with an entity database, tailoring its queries to match the preferences expressed by the user. This strategy has seen widespread use in various datasets, including WOZ2.0, FRAMES, and MultiWOZ, among others.
- Machine-machine Interaction – A distinct area of research delves into dia-

3.1. Datasets for Task-oriented Dialogue Systems

logue generation through simulation. In this methodology, both user and system roles are simulated to create an end-to-end conversational flow. This involves artificial agents playing the roles of both the user and the system to generate a conversation complete with potential queries and responses. Subsequently, human crowd workers can translate this conversation into natural language, as exemplified in the M2M [Shah et al., 2018] interaction framework. This approach potentially offers cost-effectiveness and a lower susceptibility to errors, as the task for crowd workers is less complex and annotations can be automatically obtained.

Lately, conversational interfaces have gained significant popularity, and frameworks like Actions¹ at Google and Alexa Skills² have made it easier for developers to incorporate support for new services. As a result, virtual assistants are now required to support a broader range of application domains and individual services, much like the diverse array of apps available on smartphones. Recent efforts have been directed towards scaling dialogue systems to handle tasks spanning multiple application domains. However, current datasets for multi-domain task-oriented dialogue fail to adequately capture the challenges that arise when scaling virtual assistants for real-world production. These assistants must effectively support a large and continually expanding number of services across numerous domains [Kim et al., 2018]. In contrast, existing publicly available datasets, including those mentioned earlier, only cover a limited number of domains.

To address these challenges, the MultiWOZ [Budzianowski et al., 2018] dataset was created, falling under the first category mentioned above. More recently, the Schema-Guided Dialogue (SGD) dataset [Rastogi et al., 2020] has been introduced, adhering to the second category. An analysis of Table 3.1 shows and compares the various metrics comprising each dataset. We can also conclude that MultiWOZ and SGD are currently the two largest dialogue corpora available.

Metric ↓ Dataset →	DSTC2	WOZ2.0	FRAMES	M2M	MultiWOZ	SGD
N° of Domains	1	1	3	2	7	16
N° of Dialogues	1,612	600	1,369	1,500	8,438	16,142
Total N° of Turns	23,354	4,472	19,986	14,796	113,556	329,964
Avg. Turns per Dialogue	14.49	7.45	14.60	9.86	13.46	20.44
Avg. Tokens per Turn	8.54	11.24	12.60	8.24	13.13	9.75
Total unique Tokens	986	2,142	12,043	1,008	23,689	30,352
N° of Slots	8	4	61	13	24	214
N° of Slot values	212	99	3,871	138	4,510	14,139

Table 3.1: Comparison of datasets for task-oriented dialogue: All reported numbers pertain to training data, except for FRAMES, for which the complete dataset’s numbers are given.

¹<https://developers.google.com/assistant/console?hl=en>

²<https://developer.amazon.com/en-US/alexa/alexa-skills-kit>

3.2 Algorithms for DST

This section will showcase several DST algorithms that have been developed and utilized in some of the discussed datasets. Additionally, it will provide an evaluation of these models, highlighting their performance and effectiveness.

3.2.1 Algorithms used in the MultiWOZ dataset

There have been several algorithms tested on the MultiWOZ dataset. Starting with Triple copy strategy DST (TripPy), which utilizes various copy mechanisms to populate slots with values [Heck et al., 2020]. All values are extracted from the real-time dialogue context. A slot can be filled using one of three mechanisms: Span prediction, which directly extracts values from the user input; copying a value from a system inform memory that keeps track of the system’s inform operations; copying a value from a different slot already present in the DS to resolve coreferences within and across domains. Another algorithm is Slot-Utterance Matching Belief Tracker (SUMBT) [Lee et al., 2019], which employs BERT to encode slot IDs and candidate values, learning slot-value relationships in dialogues through an attention mechanism. Dialogue context is encoded with recurrence. BERT-DST [Chao and Lane, 2019] utilizes contextual representations to encode each dialogue turn and feeds them into classification heads for value prediction. However, the dialogue history is not considered for slot filling. In [Gao et al., 2019], DST is treated as a reading comprehension task approached with a BERT-based dialogue context encoder. A slot carryover prediction model determines whether previously detected values should be retained in the DS for the current turn.

An alternative to span prediction is value generation. TRADE [Wu et al., 2019] and Multi-Attention-Based Scalable Dialogue State Tracking (MA-DST) [Kumar et al., 2020] employ a copy mechanism to generate a DS by combining distributions over a pre-defined vocabulary and the vocabulary of the current context. Selectively Overwriting Memory for Dialogue State Tracking (SOM-DST) [Kim et al., 2019] utilizes a similar mechanism for value generation but takes the previous dialogue turn and DS as input to BERT in order to predict the current DS. A state operation predictor determines whether a slot needs to be updated or not. However, generative models have a drawback in that they may produce invalid values, such as word repetitions or omissions.

A hybrid approach known as DS-DST has been proposed, which utilizes both span-based and picklist-based predictions for slot-filling [Zhang et al., 2019]. In contrast to generative approaches, picklist-based and span-based methods use existing word sequences to populate slots. DS-DST partially mitigates the limitations of span prediction by employing a picklist method to fill a subset of slots instead.

Recent studies have indicated a trade-off between the level of value independence in a model and its performance in DST. Chao and Lane [2019] and Gao et al. [2019] solely rely on span prediction for DST, but their performance lags

3.2. Algorithms for DST

behind methods that utilize a pre-defined list of candidate values. This trade-off has been notably demonstrated by Zhang et al. [2019], whose model could not compete when relying solely on span prediction. In contrast, when relying solely on their picklist slot-filling method, they achieved the best performance to date on the MultiWOZ dataset. The proposed dual-strategy approach aims to strike a favorable balance between these two extremes.

Until now, we have not encountered any recent approaches to complex DST tasks, such as MultiWOZ [Budzianowski et al., 2018], that demonstrate complete value independence. Moreover, we have observed significant benefits in these approaches through the utilization of a value candidate list. TripPy addresses this limitation by introducing a triple copy strategy that combines span-prediction with memory mechanisms. Unlike other hybrid approaches such as [Zhang et al., 2019], TripPy’s memory mechanisms dynamically create candidate lists of values using only the dialogue context as a source of information, eliminating the need for pre-defined picklists. This allows the model to decide which strategy to employ for each slot in every turn. TripPy’s approach differs from Chao and Lane [2019] and Kim et al. [2019] as it incorporates the dialogue history as context, in addition to the current turn. It also distinguishes itself from [Lee et al., 2019] by not utilizing recurrence. Similar to [Kim et al., 2019], TripPy employs auxiliary inputs at each turn, but as a late feature fusion strategy. With their slot-value copy mechanism to handle co-referring value phrases, TripPy adopts a method reminiscent of Gao et al. [2019] slot carryover, with the crucial distinction that it facilitates value sharing between different slots, enabling value interchangeability within and across domains.

All of these modifications made by TripPy, taking into account other approaches for DST, have enabled it to achieve the highest value of JGA compared to other algorithms in the MultiWOZ dataset, as illustrated in Table 3.2.

Models	JGA
DST-reader	36.40%
DST-span	40.39%
SUMBT	42.40%
TRADE	45.60%
MA-DST	51.04%
DS-DST	51.21%
SOM-DST	52.57%
DS-picklist	53.30%
TripPy	55.29%

Table 3.2: DST Results on MultiWOZ for Joint Goal Accuracy.

3.2.2 Algorithms used in the SGD dataset

Several approaches have been proposed for utilizing schema in dialogue modeling, particularly when dealing with unseen services. Rastogi et al. [2020] propose a prototype that incorporates two BERT encoders: the schema encoder and the dialogue encoder. However, the dialogue encoder only takes the previous system sentence and the current user sentence as input, while the schema encoder employs BERT without fine-tuning, resulting in suboptimal performance. Gulyaev et al. [2020] propose a multi-task model that combines schema and dialogue, using the GOaL-Oriented Multi-task BERT-based (GOLOMB) dialogue state tracker inspired by recent neural architectures for reading comprehension question-answering tasks [Devlin et al., 2018]. Their model takes slot descriptions and dialogues as input, followed by a possible slot. Li et al. [2020] and Ruan et al. [2020] employ separate models for different subtasks and introduce an auxiliary model to address the slot carryover problem, but they do not consider the relevance of subtasks. Ma et al. [2019] leverage the previous eight sentences as dialogue history to enhance performance

Despite their successes, these methods have two weaknesses. First, current methods do not fully exploit the history of the dialogue. As mentioned by Rastogi et al. [2020], the average number of turns per dialogue in the training set of the SGD dataset is 20.44, but most methods only use the previous system sentence as the dialogue history [Rastogi et al., 2020], [Gulyaev et al., 2020], [Ruan et al., 2020], making it difficult to solve the slot carryover problem in multi-domain dialogues. For example, in Figure 3.1, the value of the "destination_airport" slot of the Flights_4 service at turn 13 is carried over from the value of the "city" slot of the Events_3 service at turn 1, and the value of the "number_of_tickets" slot of the Events_3 service at turn 33 is carried over from the value of the "number_of_tickets" slot of the Flights_4 service at turn 13. Second, current methods treat the task as four independent subtasks without considering the relevance of the subtasks. For example, the intent contains a set of required slots, optional slots, and result slots. If the current utterance belongs to this intent, the slots it contains can be used as candidates for both value extraction subtasks, which can reduce computational cost and improve performance. In addition, the default values of the optional slots (e.g., seating class: Economy) can also be used for both value extraction subtasks.

To address these weaknesses, Du et al. [2022] proposed a novel Two-stage framework for schema-guided DST (TS-DST) model with selected dialogue history. To solve the first problem, the authors propose a novel utterance selection module to select the most related previous utterances from the dialogue history by considering the specific schema element. To solve the second problem, they proposed a two-stage framework to address the four subtasks. Specifically, the subtasks of intent classification and requested slot identification, along with an auxiliary subtask for active slot identification, can be treated as the first stage. The value extraction subtasks, which include categorical slot classification and non-categorical slot labeling, can be treated as the second stage. The active intent and active slots are used to generate candidate slots for both the value extraction subtasks.

3.2. Algorithms for DST

Schema			
Service 1		Service 2	
"service_name": "Events_3" "description": "Find and book tickets to any cultural events in your area"		"service_name": "Flights_4" "description": "Find cheap flights in seconds and book flights"	
"name": "city" Slot1-1 "description": "City where the event..."	"name": "BuyEventTickets" Intent1-1 "description": "Buy tickets for a cultural event and date in a given city" "required_slots": ["event_name", "date", "city", ...] "result_slots": ["event_name", "time", ...]	"name": "destination_airport" Slot2-1 "description": "The name of the..."	"name": "SearchRoundtripFlights" Intent2-1 "description": "Search for roundtrip flights for ..." "required_slots": ["origin_airport", "destination_airport", "departure_date", "return_date"] "optional_slots": {"seating_class": "Economy", "number_of_tickets": "1", "airlines": "dontcare"} "result_slots": ["number_of_tickets", "seating_class", "origin_airport", "destination_airport", "departure_date", ...]
"name": "event_type" Slot1-2 "description": "Type of cultural event" "is_categorical": True "possible_values": ["Music", "Theater"]	"name": "FindEvents" Intent1-2 "description": "Find cultural events - concerts and plays - happening in a city" "required_slots": ["event_type", "city"] "result_slots": ["event_type", "date", "time", ...]	"name": "departure_date" Slot2-2 "description": "Start date of the trip"	
"name": "number_of_tickets" Slot1-3 "description": "Number of tickets ..." "is_categorical": True "possible_values": ["1", "2", "3", "4", ...]		"name": "number_of_tickets" Slot2-3 "description": "the number of flight..." "is_categorical": True "possible_values": ["1", "2", "3", "4"]	
		"name": "origin_airport" Slot2-4 "description": "The name of the..."	

Dialogue			Dialogue State Tracking					
Turn	Speaker	Utterance	Turn 1, Events_3		Turn 13, Flights_4		Turn 33, Events_3	
1	User	I need help finding Stage show type events happening on March 12th around NY.	active_intent	FindEvents	active_intent	SearchRoundtripFlights	active_intent	BuyEventTickets
			requested_slots	N/A	requested_slots	N/A	requested_slots	N/A
...	slot_values		slot_values		slot_values	
13	User	All right, cool. Let's cover all bases just in case I miss the bus. Please find me 2 round trip ticket going there, the day after tomorrow.	city	NY	origin_airport	Philadelphia	city	NY
			date	March 12th	departure_date	day after tomorrow	date	March 12th
...	event_type	Theater	destination_airport	NY	event_name	Broadway Brunch
33	User	Yes, please buy tickets to the event you found earlier.			number_of_tickets	2	event_type	Theater
							number_of_tickets	2

Figure 3.1: Example of a dialogue involving two unseen services(Event_3 and Flights_4), along with their schema element and DS representations. From [Du et al., 2022].

A performance comparison was conducted to assess intent accuracy in the intent classification task. It was observed from Tables 3.3, 3.4, and 3.5 that MRC-DST [Ma et al., 2019] and TS-DST [Du et al., 2022] outperformed SG-DST [Rastogi et al., 2020], GOLOMB [Gulyaev et al., 2020], and SGP-DST [Ruan et al., 2020]. This finding highlights the significance of a lengthy dialogue history for intent classification. Furthermore, TS-DST exhibited superior performance compared to MRC-DST, underscoring the effectiveness of the intent classification module.

Next, the performance of the requested slot F1 in the requested slot identification subtask was compared. Tables 3.3, 3.4, and 3.5 indicate that TS-DST outperforms the other methods, demonstrating the efficacy of the slot identification module.

Lastly, the performance of average goal accuracy and joint goal accuracy in the value extraction subtasks (categorical slot classification and non-categorical slot labeling) was compared. Once again, Tables 3.3, 3.4, and 3.5 reveal that TS-DST exhibits the highest performance, suggesting that the two-stage strategy and the utterance selection module effectively address slot carryover issues.

Models	All Services			
	Int Acc	Req F1	Avg GA	JointGa
SG-DST	90.64	96.51	56.05	25.37
GOLOMB	74.74	97.03	75.00	46.47
SGP-DST	91.85	98.99	91.25	72.22
MRC-DST	94.82	98.47	92.25	70.48
TS-DST	97.51	99.39	95.63	78.59

Table 3.3: Performance comparison of all models for All Services using the SGD dataset. Adapted from [Du et al., 2022]

Models	Seen Services			
	Int Acc	Req F1	Avg GA	JointGa
SG-DST	95.06	99.55	67.78	41.25
GOLOMB	87.10	97.10	73.91	52.75
SGP-DST	95.64	99.35	95.82	87.98
MRC-DST	/	/	/	/
TS-DST	98.51	99.53	97.47	91.88

Table 3.4: Performance comparison of all models for Seen Services using the SGD dataset. Adapted from [Du et al., 2022]

Models	Unseen Services			
	Int Acc	Req F1	Avg GA	JointGa
SG-DST	89.15	95.47	51.92	20.00
GOLOMB	70.61	97.00	75.38	44.38
SGP-DST	90.59	98.87	89.66	66.96
MRC-DST	/	/	/	/
TS-DST	97.18	99.34	94.98	74.15

Table 3.5: Performance comparison of all models for Unseen Services using the SGD dataset. Adapted from [Du et al., 2022]

3.3 Summary

This chapter has laid the groundwork for understanding the creation of datasets and their relevance in the field of study. It has provided an overview of the most current datasets in this area, allowing for an analysis of previous research and successful approaches. By examining these approaches, we gain insights into the reasons behind their success and their significance in advancing the field.

In summary, when analyzing the algorithms used in the MultiWOZ dataset, it can be concluded that the triple copy strategy, which combines span prediction with memory mechanisms, proves to be highly effective. TripPy achieved the best JGA result among the algorithms, primarily due to its dynamic creation

3.3. Summary

of candidate value lists based solely on the context, eliminating the need for pre-defined options. Unlike hybrid approaches like DS-DST, TripPy incorporates the dialogue history as part of the context in addition to the current turn, setting it apart from DST-span and SOM-DST. It also differs from SUMBT by not utilizing recursion. These factors contribute to TripPy’s standout performance and its achievement of the highest JGA result.

In regards to the algorithms used in the SGD dataset, it can be inferred that models such as MRC-DST and TS-DST, which leverage the long dialogue history, exhibit superior performance compared to other models. Furthermore, the TS-DST model surpasses MRC-DST, indicating the effectiveness of the intent classification module. The TS-DST model also demonstrates the efficacy of the two-strategy and utterance selection module in addressing slot carryover issues. These findings provide valuable insights into the utilization of specific algorithms for dialogue state tracking in real-world scenarios, highlighting their potential and impact.

Chapter 4

Dataset Creation

In this chapter, we present the first task-oriented dataset in Portuguese – MultiWOZpt, currently in version 0.1. This dataset was based on the MultiWOZ dataset, so we will explore the constitution and structure of the MultiWOZ dataset and explain why it was chosen over the other datasets discussed in Chapter 3. In addition, we will describe the methodology used to create the Portuguese dataset, which involved adapting and translating the MultiWOZ dataset.

4.1 MultiWOZ Dataset

The MultiWOZ [Budzianowski et al., 2018] dataset was developed using the WOZ paradigm (see Section 3.1), in which humans simulate the roles of the user and the system, allowing for more realistic and natural conversations. It took 1249 workers to make the dataset. The dataset comprises 10,438 dialogues, with 9,438 allocated for training and 1,000 for testing. It includes 3,406 single-domain dialogues (SNG) and 7,032 multi-domain dialogues (MUL).

Single-domain dialogues focus on a singular domain like hotel booking or restaurant finding, while multi-domain dialogues span multiple independent domains. For instance, a user might first book a hotel and then search for a restaurant, completing each task separately. Such categorizations furnish additional context, thereby allowing researchers to evaluate dialogue systems across diverse task types.

The dataset spans eight domains: Attractions, Hotels, Restaurants, Taxis, Trains, Hospitals, Police, and Buses. In this context, a "domain" refers to a specific subject area or category that frames a conversation or interaction. Recognizing and managing different domains are critical steps in building task-oriented dialogue systems that can effectively interpret user requests and generate suitable responses.

Additionally, each domain comes with associated "slots", and in total, the dataset includes 61 such slots, detailed in Table 4.1. Slots are specific pieces of information that a system must extract from user utterances. They serve to cap-

4.1. MultiWOZ Dataset

ture key details vital for fulfilling specific user requests within designated domains. This dataset features two types of slots: categorical and non-categorical, as elaborated in Table 4.1.

Categorical slots possess a finite set of predefined values. For example, in the hotel domain, the "type" slot might assume values like "guesthouse" or "hotel". These values are fixed and do not evolve during the conversation. Non-categorical slots, in contrast, can assume continuous or free-form values. For instance, in the restaurant domain, the "book-time" slot specifies the reservation time, which could be a numerical value or a text string.

Domain	Categorical Slots	Non-categorical Slots	Intents
Restaurant	pricerange, area, bookday, bookpeople	food, name, booktime, address, phone, postcode, ref	find, book
Attraction	area, type	name, address, entrancefee, openhours, entrancefee, phone, postcode	find
Hotel	pricerange, parking, internet, stars, area, type, bookpeople, bookday, bookstay	name, address, phone, postcode, ref	find, book
Taxi	-	destination, departure, arriveby, leaveat, phone, type	book
Train	destination, departure, day, bookpeople	arriveby, leaveat, trainid, ref, price, duration	find, book
Bus	day	departure, destination, leaveat	find
Hospital	-	department, address, phone, postcode	find
Police	-	name, address, phone, postcode	find

Table 4.1: Mapping of the MultiWOZ slots associated with each Domain.

Upon analyzing the excerpt from a dialogue of the dataset, as we can see in Figure 4.1, it is structured as a sequence of turns. A turn consists of:

- A "USER" or "SYSTEM" indicating who is speaking.
- The "utterance" made by that speaker.
- The Dialogue Act, which classifies the speech act into categories such as "inform," "ask," "book," etc. Each Dialogue Act has slot-value pairs representing the information exchanged.

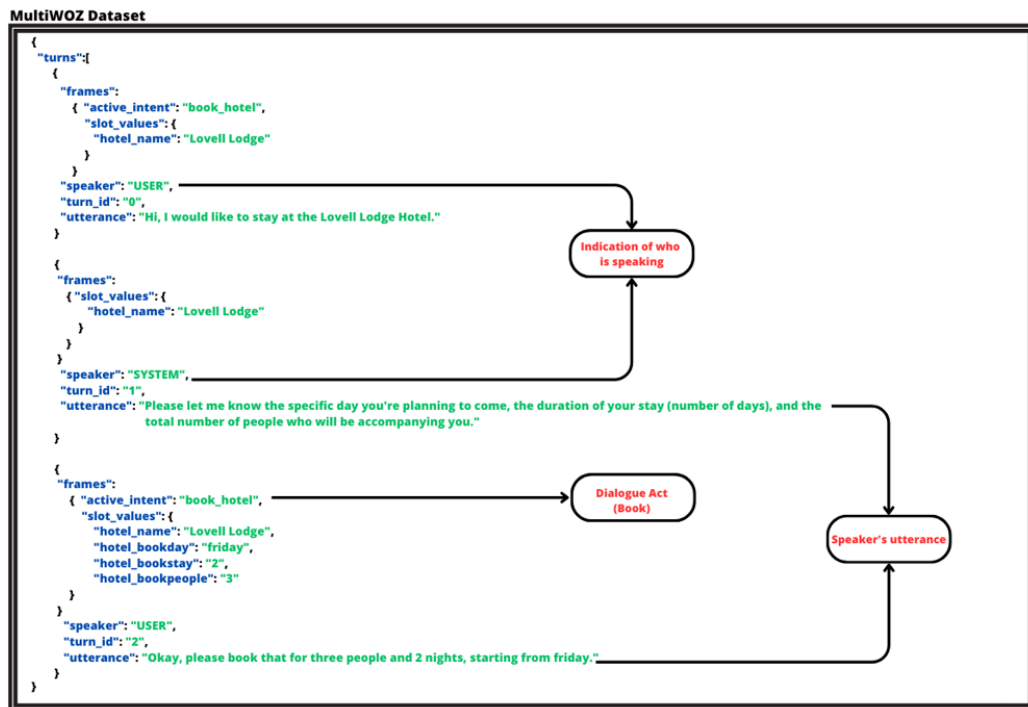


Figure 4.1: Excerpt from a dialogue of the MultiWOZ dataset.

For each dialogue, there is a goal field that describes the task the user wants to accomplish or achieve in the conversation. In a task-oriented dialogue system, these goals are crucial as they define the scenario of the interaction. This goal usually consists of three components:

- A high-level description of the user’s task: it is an overview of what the user intends to achieve in the conversation. For example, if they have the goal of making a reservation in a Hotel, the high-level description could be something like "Book a hotel room in Coimbra".
- Details of the task across multiple domains: if the goal encompasses more than one domain (such as restaurants, hotels, among others), there must be specific details for each of them. For example, in the Hotel domain, the details can include the type of Hotel (Hotel or Guesthouse), the date of entry, and the days they intend to stay overnight. In the restaurant domain, it could be the type of food the users intend to eat, the location of the restaurant itself, and the time they wish to book.
- Constraints and requests specified by the user: these are particular requirements or conditions requested by the user that the dialogue system has to take into account and comply with. For example, a constraint could be something like "The hotel must be 5-star" or "The restaurant must be in the city center" A request could be something like "I want the cheapest hotel" or "Book a taxi from the hotel to the restaurant".

4.2 MultiWOZpt 0.1 Dataset

Given that the primary objective was to explore effective methods for handling context, and considering that much of the existing research in this domain has been conducted in English, there were no readily available datasets of annotated dialogues in Portuguese for evaluating context monitoring at the time of writing this dissertation. As a result, it became necessary to develop the first task-oriented dataset in Portuguese, named MultiWOZpt. This new dataset was created by adapting and translating the MultiWOZ dataset to fulfill the main objective.

This dataset was selected for multiple reasons. First, it is multidomain, covering a diverse array of subject areas. This variety is invaluable for the development and assessment of dialogue systems that can efficiently navigate multiple services. Unlike the SGD dataset, which restricts conversations to specific domains, MultiWOZ allows dialogues to shift seamlessly between different domains, such as transitioning from discussing restaurants to hotels.

Second, the dataset reflects the complexity and variability of human language, as it was compiled by multiple individuals. Each dialogue includes a comprehensive user goal, which not only outlines the high-level task but also provides domain-specific details and user-imposed constraints and requests. This richness of information enhances the system's ability to accurately interpret and fulfill user goals.

Furthermore, the inclusion of Dialogue Acts in MultiWOZ offers additional semantic insight into the intentions behind each utterance. This feature is especially beneficial for training and evaluating dialogue systems in terms of understanding user intent, generating suitable responses, and assessing system performance.

The dataset was co-created with a PhD student and took approximately five months to complete. We began by manually translating and adapting 512 dialogues from the first test partition of the MultiWOZ dataset into Portuguese. These dialogues encompassed five services: Attractions, Hotels, Restaurants, Taxis, and Trains, which collectively involved 30 associated slots.

During the creation of the dataset, we employed a specially designed database to align services from Cambridge with those in Coimbra, the city where we reside. For instance, if in a particular dialogue a user searched for an Attraction like a museum in Cambridge, we sought equivalent museums in Coimbra to replace it. Further details on this alignment are available in Appendix 7.2. Fields labeled as "start" and "exclusive_end" were omitted since they added no value to our specific project. Typically, these fields serve to annotate spans of text and are useful for NLP tasks such as NER and co-reference resolution, among others.

After the completion of the Portuguese dataset, we performed an analysis to examine its composition. We specifically focused on the number of services and intents within the 512 translated dialogues, as presented in Tables 4.2 and 4.3. During this process, we identified an error wherein certain user utterances were

erroneously tagged with multiple intents. For instance, the phrase "I wanted a train that arrives in Cambridge at 17:00" was incorrectly labeled with both "find_train" and "find_hotel" intents. The latter obviously being inappropriate. These inconsistencies – totaling 559 cases – were rectified, as shown in Table 4.3, which highlights the disparity in the number of intents between the original and Portuguese datasets.

The dataset is publicly accessible on GitHub¹. Looking ahead, we plan to continue the adaptation/translation process by focusing on the remaining 488 dialogues in the second test partition.

Service	# Examples
Attraction	199
Hotel	201
Restaurant	237
Taxi	105
Train	261
Total	1,003

Table 4.2: Number of Services present in the MultiWOZpt 0.1 dataset.

Intents	MultiWOZ 2.2	MultiWOZpt 0.1
find_attraction	622	530
find_hotel	639	554
book_hotel	250	225
find_restaurant	650	562
book_restaurant	301	219
find_taxi	270	219
find_train	837	735
book_train	233	199
Total	3,802	3,243

Table 4.3: Comparison of the Number of Intents present in the two datasets.

¹<https://github.com/NLP-CISUC/MultiWOZpt/>

4.3 Future Applications

The creation of this dataset for Portuguese can support the development of several dialogue systems in Portuguese. Here are some examples that can take advantage of this dataset:

- **Dialogue State Tracking (DST):** The dataset is well-suited for training DST models that aim to predict the user’s goals, track the current dialogue state, and understand the constraints expressed by the user during the conversation. DST is crucial for maintaining context and providing accurate responses in a dialogue system.
- **Dialogue Generation:** Given the extensive dialogues in the dataset, it can be used to train models for generating natural-sounding responses. This is particularly useful for creating dynamic and interactive conversational agents that can respond appropriately to user queries.
- **Dialogue Policy Learning:** Reinforcement learning approaches for dialogue management require some data for training. The MultiWOZpt dataset can serve as a valuable resource for training dialogue policy models that enable conversational agents to decide on appropriate actions during the conversation.
- **Intent Detection:** Intent detection is an essential component of dialogue systems as it involves recognizing the user’s intentions based on their queries. MultiWOZpt provides labeled data that can be used to train intent detection models for Portuguese dialogue systems, helping them recognize user intentions based on their queries.
- **Dialogue System Evaluation:** As the dataset covers a wide range of dialogue scenarios and domains, it can be used as a benchmark for evaluating the performance of future Portuguese dialogue systems. This allows researchers and developers to compare the effectiveness of different models and algorithms.

4.4 Summary

This chapter introduces the first task-oriented dataset in Portuguese, explaining its construction, structure, and purpose. It also discusses the inspiration drawn from the MultiWOZ dataset and the reasons for its selection over other alternatives. Challenges encountered during the dataset’s creation are addressed, and the chapter concludes by outlining its potential benefits for future applications.

Chapter 5

Question Answering-based Slot-Filling

In this chapter, we present our approach to monitoring context in Portuguese dialogues. Figure 5.1 illustrates the complete pipeline we use for question-answering (QA). This pipeline aids in executing slot-filling tasks, allowing us to better monitor context and, as a result, improve interactions.

We will describe which Portuguese QA models have been used, their structure, and how they have been adapted for the task of slot filling. The use, or lack thereof, of Intent Detection in these models will also be addressed in this chapter. Moreover, we will introduce the Post-Processing methods used in slot comparison.

The input for this pipeline is a dataset of dialogues where domain-specific slots are filled with appropriate values extracted from the utterances, described in the last chapter and further detailed in Chapter 6.

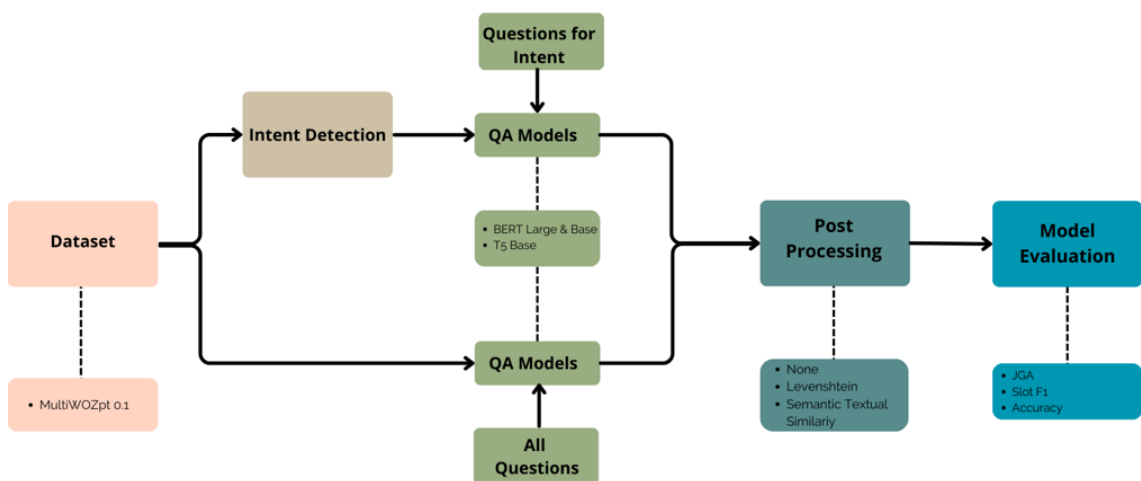


Figure 5.1: Pipeline for Slot-filling Approach.

5.1 QA Models for Portuguese

The strategy for addressing the problem of insufficient context monitoring employs Dialogue State Tracking (DST). DST in frame-based dialogue systems is to keep track of the current state of the dialogue by monitoring the slots that have been filled, as well as understanding the user's most recent actions within the dialogue.

For the purpose of slot-filling, QA models are implemented. These models are specially constructed to identify particular pieces of information within a text or corpus. Their capacity to pinpoint which information from user utterances should be allocated to which slots is significant. Equally important is their ability to cope with variability. Users can articulate the same intent in multiple ways, and these models are trained on a diverse array of questions and contexts to handle such variability. This enables them to correctly identify slots even when users express intentions differently.

The models are designed to identify a specific range of text within a given context that contains the answer to a user's question. Models like BERT [Devlin et al., 2018] and T5 [Raffel et al., 2020] have been specifically adapted for the QA task. In summary, these models are equipped to select a sequence of words from a context which then forms the answer to a specific question.

The most recent QA models are based on transforms. The following have been identified for Portuguese: BERT-base¹, BERT-large² and T5-base³ which are available on the Hugging Face platform⁴.

All the models mentioned were fine-tuned on the SQuAD dataset [Rajpurkar et al., 2016]. This dataset comprises questions posed by crowdworkers based on a set of Wikipedia articles. The answer to each question is a segment of text, or an extension, from the corresponding reading passage. The dataset includes over 1,000,000 question-answer pairs from more than 500 articles. For the Portuguese version, an automatic translation of the SQuAD v1.1 dataset in English was employed, using Google Translator. Over a period of two months, the Deep Learning Brazil group⁵ undertook revisions and made necessary corrections to this translated dataset⁶.

5.1.1 BERT for QA

BERT is a widely recognized machine learning model developed by Google [Devlin et al., 2018] for various natural language processing tasks, including QA. Before delving into its application in QA, it is crucial to understand BERT's fun-

¹<https://huggingface.co/pierreguillou/bert-base-cased-squad-v1.1-portuguese>

²<https://huggingface.co/pierreguillou/bert-large-cased-squad-v1.1-portuguese>

³<https://huggingface.co/pierreguillou/t5-base-qa-squad-v1.1-portuguese>

⁴<https://huggingface.co/>

⁵<http://www.deeplearningbrasil.com.br/>

⁶<https://drive.google.com/file/d/1Q0Iallv2h2BC468MwUFmUST0EyN7gNkn/view>

damental mechanics and its specific approach to QA tasks.

As discussed earlier in Chapter 2.4, BERT is a transformer-based model. In other words, it leverages transformer architectures to understand context in both directions (left-to-right and right-to-left), hence the term "bidirectional". This enables a deeper representation of context compared to previous models that processed text in only one direction, either from left to right (akin to human reading) or right to left.

In the context of QA tasks, BERT takes a question and an accompanying context (where the answer can be found) as inputs. In the Figure 5.2, both the question and reference text (context) are concatenated into a single character string, separated by a special symbol [SEP]. A unique token, [CLS], is also introduced by the model at the beginning of the string. While [CLS] is useful for various tasks, such as classification, it does not play a direct role in QA. BERT processes the input and yields a vector for each token (word or subword) in the input string. These vectors represent high-dimensional semantic information about the tokens and their context.

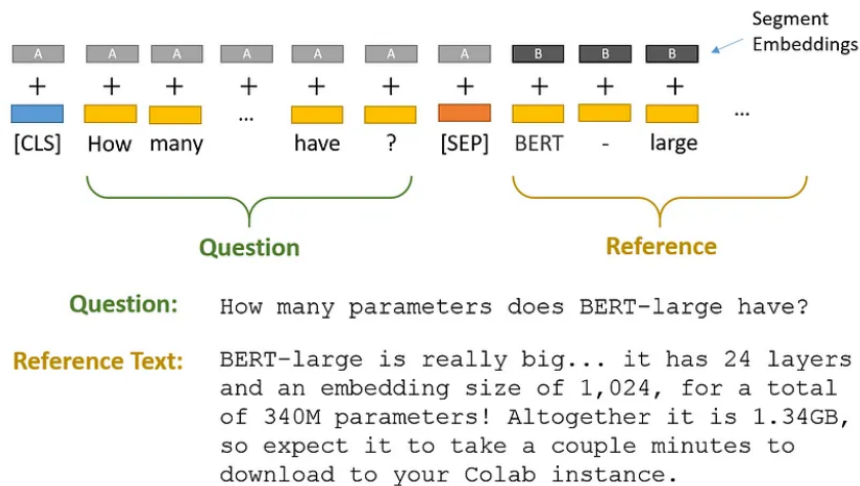


Figure 5.2: QA System with BERT. From [N, 2020].

For QA, BERT must pinpoint specific start and end indices for the answer within the context. To facilitate this, during training, two connected layers are added to the BERT model — the "start" and "end" layers. These layers are trained to predict the start and end tokens of the response interval within the context.

Hence, during a QA task, when a question and a context are fed into BERT, it generates probabilities for each token being the start or end of the answer interval. The token with the highest combined probability is chosen as the answer interval.

It is important to emphasize that this model does not create new text as an answer. Instead, it extracts the answer from the provided context, making it an extractive question answering model.

The primary distinction between BERT-base and BERT-large lies in their size and potential precision. BERT-base comprises 12 transformer layers (or encoder

5.1. QA Models for Portuguese

blocks), each containing 768 hidden units, and it utilizes 12 attention heads. This amounts to roughly 110 million parameters. In contrast, BERT-large features 24 transformer layers, with each layer containing 1024 hidden units, and employs 16 attention heads. This sums up to approximately 340 million parameters.

Therefore, the "large" model is more powerful but also more resource-hungry than the "base" model. Generally, the "large" model outperforms the "base" model due to its increased size and complexity. This allows it to capture more nuanced meanings and more complex relationships thanks to its greater capacity. However, it demands more computational resources for training and operation.

5.1.2 T5 for QA

T5, or Text-to-Text Transfer Transformer, is another transformer-based model developed by Google [Raffel et al., 2020]. Unlike BERT, T5 was designed to handle any NLP task as a text generation problem.

T5 converts all tasks into a text-to-text format. In the case of QA, it reformulates the problem into a text generation task by combining the question and the provided context into a single string with the format "question: [the question] context: [the context]". The model then generates an answer to this string.

As can be seen in the Figure 5.3, this model is pre-trained using a denoising autoencoder setup. It learns to predict masked parts of the input sequence, much like BERT, but with a distinctive approach. Instead of masking random tokens, it begins by masking entire spans of text. These spans are replaced by a unique sentinel token, allowing the model to learn the span length, which can be beneficial for certain tasks. Following pre-training, T5 is fine-tuned for specific tasks, such as QA. During this fine-tuning, it learns to generate an answer given a question and a context.

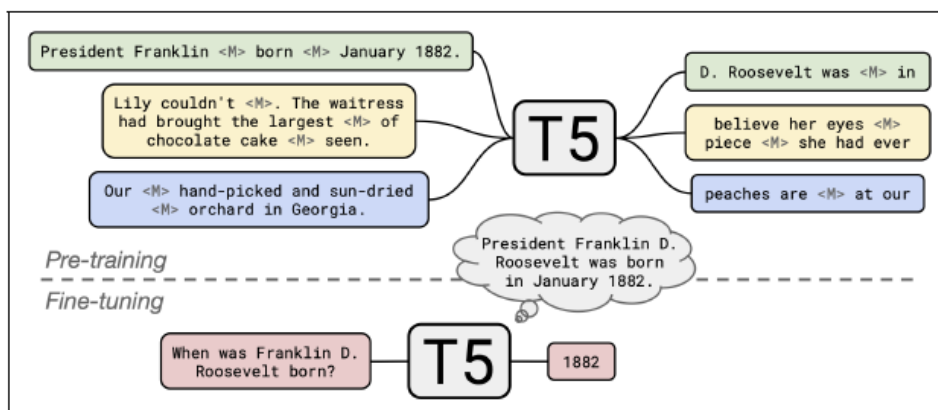


Figure 5.3: The T5 system is an encoder-decoder architecture. From [Raffel et al., 2020].

This approach diverges from BERT in that it generates the answer rather than locating it within the input text. Thus, while BERT serves as an example of an

extractive QA model, T5 is a generative QA model. Furthermore, T5’s text-to-text approach renders it a highly versatile model that can be utilized across a wide range of NLP tasks without significant modifications to its architecture, provided that the problem can be posed as a text generation problem.

5.1.3 Questions Developed for QA Models

As deduced from the explanation of the QA models, these models require both a question and a context as input to generate an answer. The context they utilize is obtained from the user utterances in the MultiWOZpt dataset. The questions were written considering the five services (Attraction, Hotel, Restaurant, Taxi, and Train) and their respective slots. Tables 5.1, 5.2, 5.3, 5.4, and 5.5 illustrate the specific questions developed for each slot of each service, resulting in a total of 30 questions corresponding to the 30 slots present in the five services of the MultiWOZpt dataset. We conducted some preliminary tests to select the most appropriate questions for each slot. The results from these questions are detailed in the next chapter.

Slot Type	Question
attraction-area	Em que área está localizada a atração?
attraction-name	Qual é o nome da atração?
attraction-type	Qual é o tipo de atração?

Table 5.1: Questions for the Attraction Service.

Slot Type	Question
hotel-area	Em que área está localizado o estabelecimento?
hotel-bookday	Em que dia é a reserva?
hotel-bookpeople	Quantas pessoas são?
hotel-bookstay	Quantos dias vai ficar?
hotel-internet	Tem internet grátis?
hotel-name	Qual é o nome do estabelecimento?
hotel-parking	Tem estacionamento gratuito?
hotel-pricerange	Qual é o preço médio do estabelecimento?
hotel-stars	Quantas estrelas tem?
hotel-type	Qual é o tipo de estabelecimento?

Table 5.2: Questions for the Hotel Service.

Slot Type	Question
restaurant-area	Em que área está localizado o restaurante?
restaurant-bookday	Em que dia é a reserva?
restaurant-bookpeople	Quantas pessoas são?
restaurant-booktime	A que horas é a reserva?
restaurant-food	Qual é tipo de comida?
restaurant-name	Qual é o nome do restaurante?
restaurant-pricerange	Qual é o preço médio do restaurante?

Table 5.3: Questions for the Restaurant Service.

Slot Type	Question
taxi-arriveBy	A que horas chega?
taxi-departure	De onde quer sair?
taxi-destination	Para onde quer ir?
taxi-leaveAt	A que horas é que sai?

Table 5.4: Questions for the Taxi Service.

Slot Type	Question
train-arriveBy	A que horas chega?
train-bookpeople	Quantas pessoas são?
train-day	Em que dia é a reserva?
train-departure	De onde quer sair?
train-destination	Para onde quer ir?
train-leaveAt	A que horas é que sai?

Table 5.5: Questions for the Train Service.

5.2 Post-Processing

Post-processing was also applied to the QA models. This step is necessary because QA models do not always return the exact answer for a given slot as it appears in the original slot, sometimes with slight differences that without Post-Processing would be considered incorrect.

For these models, two Post-Processing methods were applied: the Levenshtein Distance and Semantic Textual Similarity (STS). We chose the Levenshtein Distance algorithm for several reasons. First, it is simple to implement, eliminating the need for complex data structures or extensive computational resources. Second, its computational speed is an essential feature for real-time dialogue systems. Finally, the algorithm is language-independent, broadening its applicability. We selected the other Post-Processing method, STS, primarily for its semantic understanding. This attribute enables STS to capture the meaning of the text, which is crucial when different phrases convey the same meaning. For instance, if the original slot is "south" and the model extracts "south zone", applying this Post-Processing method would identify them as identical.

The Levenshtein Distance, named after Vladimir Levenshtein who introduced this metric in 1965, measures the difference between two sequences. Informally, the Levenshtein Distance between two words is the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into the other. In this context, it helped to enhance the dialogue system by ensuring that the machine-generated responses closely match the intended "slot". This approach helps to reduce errors and leads to better-performing QA models in task-oriented dialogues.

STS quantifies the degree of equivalence in the underlying semantics of paired text snippets. STS captures details of semantic similarity such as paraphrase, synonymy, and changes in word order, which Levenshtein's method cannot capture. Several methods, ranging from rule-based NLP techniques to machine learning methods like Siamese neural networks or transformers trained on STS tasks, can calculate STS. We use a sentence-transformer model which maps sentences and paragraphs to a 1024 dimensional dense vector space and can be used for tasks like clustering or semantic search. This model⁷ derives from BERTimbau large [Souza et al., 2020] which is a state-of-the-art model trained specifically for understanding and generating text in Portuguese. We used cosine similarity to compare these vectors. This metric is commonly applied to measure the similarity between vector representations of two text sequences—in this case, between the original slot and the one filled in by the Question-Answering (QA) model. A high cosine value close to 1 suggests that the text sequences are semantically very similar. A medium cosine value, around 0.5, typically indicates some similarity but also notable differences. A low cosine value near 0 signifies that the sequences are semantically dissimilar.

In conclusion, cosine similarity is computationally efficient and particularly effective in high-dimensional spaces. It is also straightforward to interpret. How-

⁷<https://huggingface.co/rufimelo/bert-large-portuguese-cased-sts>

5.2. Post-Processing

ever, it does have limitations; for example, it can sometimes be too simplistic to capture complex semantic relationships. Overall, cosine similarity is a widely-used metric in STS tasks. When paired with advanced text-embedding techniques, it effectively captures the semantic similarity between text sequences.

To determine the appropriate thresholds when comparing the original slot with the slot filled in by the QA model, preliminary tests were carried out specifically to select the best values. The Levenshtein Distance was set to five or less, and for the STS, a value greater than or equal to 0.49 was chosen. It is important to note that the Levenshtein Distance method was not applied to slots filled with numbers, as this would not be meaningful. Tables 5.6 and 5.7 showcase instances where these post-processing methods improved the model’s accuracy during testing. Without these methods, the outcomes would not have been considered correct.

Original Slot	Filled Slot
arquitetura	arquitetónico
barco	barcos
caro	cara
centro	no centro
discoteca	discotecas
este	zona este
este	no leste
gratuito	wifi gratuito
hotel	hotéis
museu	museus
museu	um museu
oeste	Oeste
residencial	residenciais
sim	claro

Table 5.6: Examples of correctly assessed comparisons using the Levenshtein method as a Post-Processing step.

Original Slot	Filled Slot
19:00	19:00 horas
19:15	19:15 de sábado
centro	centro da cidade
centro	ao centro
chinesa	chinês
francesa	francês
Jardim Botânico da Universidade de Coimbra	Jardim Botânico
japonesa	japonês
meio dia	depois do meio dia
oeste	oeste da cidade
Pizza Hut Alma Shopping	Pizza Hut
residencial	residencial cara
residencial	residencial de 3 estrelas
sexta-feira	sexta-feira às 16:00

Table 5.7: Examples of correctly assessed comparisons using the STS method as a Post-Processing step.

The performance achieved and the conclusions drawn from these models using the mentioned methods will be documented in the following chapter.

5.3 Intent Detection

Intent Detection in the context of dialogue systems involves identifying the underlying purpose or goal in a sentence spoken by the user. Once this intent is recognized, the system can generate an appropriate response to fulfill the request. For instance, if the user says, "I want a hotel in Coimbra", the intent could be "find_hotel".

Modern dialogue systems undergo training with extensive datasets to enhance their Intent Detection capabilities. This training data comprises various user utterances along with their corresponding intents. Over time, the system improves its ability to accurately identify the intent behind new and unfamiliar utterances. While this topic was beyond the scope of this dissertation, we did evaluate the performance of the QA models both when intent was provided and when it was absent.

Figure 5.1 illustrates that certain QA models incorporated an additional "Intent Detection" step. In these cases, the models not only process the user's input utterance but also have access to the corresponding intent. On the other hand, in some situations, the models are given only the utterance, without the associated intent.

This discrepancy led to significantly different behaviors in the models with and without the associated intent. Figures 5.4 and 5.5 illustrate the divergence in these models, depending on whether intent is utilized. Specifically, when Intent

5.3. Intent Detection

Detection is applied to the user's utterances, the model uses a set of questions specific to that service. Conversely, the models without Intent Detection must execute all 30 questions developed for each sentence that the user utters.

The evaluation of the models with these two distinctions will be documented and detailed in the following chapter.

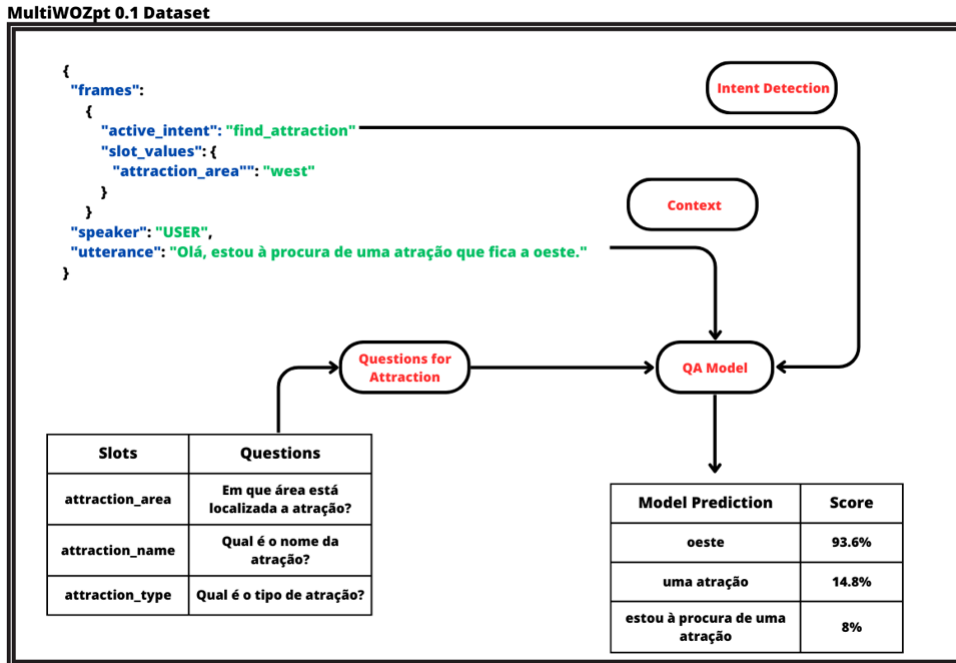


Figure 5.4: Example of how the detection of an intent affects a QA Model.

MultiWOZpt 0.1 Dataset

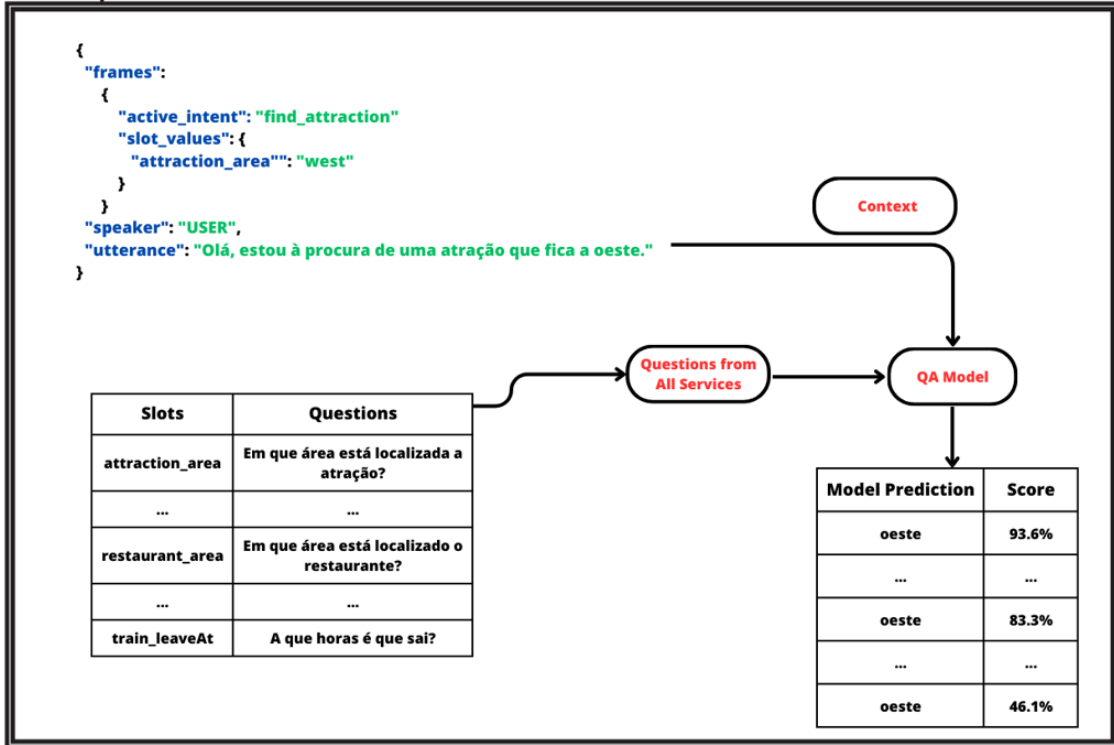


Figure 5.5: Example of how the absence of intent detection affects a QA Model.

5.4 Summary

This chapter offers a comprehensive overview of how the slot-filling approach is implemented. We explored the core elements of QA, investigated specific models suitable for Portuguese QA, and highlighted what sets them apart. Questions tailored for individual slots were also showcased. Furthermore, the chapter clarifies why certain Post-Processing techniques were applied to the QA models. We concluded by examining the role of Intent Detection and discussing its influence on the overall effectiveness of the QA models.

Chapter 6

Evaluation

In this chapter, firstly, we will present the process of selecting the optimal thresholds for each slot of each specific service. Additionally, the evaluation results of applying the QA models to dialogues in Portuguese will be presented, along with the conclusions drawn from these findings.

6.1 Testing of the QA Models

These three QA models were applied in Portuguese dialogues that involved only five distinct services. Each of these services was linked to multiple slots, resulting in a total of 30 slots. However, before we could evaluate the QA models, it was essential to first test them in order to identify the optimal thresholds for each slot within each service. Out of the 512 dialogues that were translated from Portuguese, we allocated 200 dialogues for the "training" phase. In this context, "training" involved adjust various threshold values and optionally using different processing methods. The aim was to establish accurate comparisons between the slots as they were originally filled and as they were filled by the QA model. This methodology helped us identify the most suitable comparison thresholds for each slot.

Subsequently, we used the remaining 312 dialogues for the "testing" phase. During this phase, we assessed the QA models using the optimal thresholds determined for each specific slot. It is worth highlighting that an unusually higher number of dialogues were used for testing than for "training". This was because, at the time of initial testing to determine the best thresholds, only 200 dialogues had been translated. However, by the time we reached the conclusion of the dissertation, additional dialogues had been translated, allowing us to conduct a more comprehensive testing phase. Importantly, different sets of dialogues were used for the "training" and testing phases.

Tables 6.1 and 6.2 provide an overview of the number of services and intents present in the "training" and "testing" phase.

6.1. Testing of the QA Models

Service	#Training Examples	#Testing Examples
Attraction	81	118
Hotel	75	126
Restaurant	82	155
Taxi	38	67
Train	108	153
Total	384	619

Table 6.1: Number of services present in the "training" and "testing" of the QA Models.

Intents	# Training Examples	# Testing Examples
find_attraction	204	326
find_hotel	208	346
book_hotel	87	138
find_restaurant	185	377
book_restaurant	67	152
find_taxi	77	142
find_train	305	430
book_train	82	117
Total		3,243

Table 6.2: Number of intents present in the "training" and "testing" of the QA Models.

6.1.1 Selection of Thresholds

For the task at hand, we set distinct thresholds for each slot across various models. In particular, we determined the threshold values for the BERT-base and BERT-large models through preliminary testing, settling on 0.49, 0.59, 0.69, and 0.79. For the T5 model, the thresholds were set at 0.80, 0.85, 0.90, and 0.95.

After applying different thresholds to the QA models, differentiated by the usage or not of Intent Detection and Post Processing, we observed diverse outcomes for each model type. These results were subsequently organized, and various figures were generated for each service. These plots, reflecting the influence of Intent Detection and Post Processing, can be found in Appendix 7.2.

Upon generating all these plots, we proceeded on a detailed analysis to determine the most suitable thresholds for each slot, taking into account the usage or not of Intent Detection and Post-Processing. This comprehensive analysis is summarized in Tables 6.3, 6.4 and 6.5. Each table displays the optimal threshold values selected for each slot in all variations of the model. These values were determined through comparisons between the original slots and the slots filled by the model.

Services	Slots	Intent Detection			No Intent Detection		
		None	Lev	STS	None	Lev	STS
Attraction	area	0.79					
	name	0.79					
	type	0.79		0.69	0.79		
Hotel	area	0.69			0.49	0.69	
	bookday	0.79					
	bookpeople	0.69					
	bookstay	0.59					
	internet	0.79	0.69	0.79	0.79	0.69	
	name	0.79	0.69	0.79	0.79	0.69	0.79
	parking	0.79	0.69	0.79	0.79	0.59	0.59
	pricerange	0.79	0.59	0.79	0.79	0.49	0.79
	stars	0.69					
	type	0.79		0.69	0.79		0.69
Restaurant	area	0.79			0.69		
	bookday	0.59		0.49	0.59		0.49
	bookpeople	0.79			0.69		
	booktime	0.69		0.49	0.69		0.49
	food	0.69	0.49	0.79	0.69	0.49	0.79
	name	0.59			0.79		
	pricerange	0.79	0.69	0.59	0.79	0.69	0.59
Taxi	arriveBy	0.79					
	departure	0.79			0.79	0.69	0.79
	destination	0.79					
	leaveAt	0.79					
Train	arriveBy	0.79					
	bookpeople	0.79					
	day	0.49					
	departure	0.79	0.69	0.79	0.79		
	destination	0.79					
	leaveAt	0.79	0.69	0.79	0.79	0.69	0.79

Table 6.3: Selection of Optimal Thresholds for different variations of the BERT-Base Model.

Services	Slots	Intent Detection			No Intent Detection		
		None	Lev	STS	None	Lev	STS
Attraction	area	0.59	0.69	0.49	0.59	0.69	0.59
	name	0.59			0.59		0.69
	type	0.79			0.79	0.69	0.79
Hotel	area	0.79			0.79		0.69
	bookday	0.79					
	bookpeople	0.69					
	bookstay	0.79					
	internet	0.79	0.49	0.59	0.79	0.49	0.69
	name	0.79					
	parking	0.79	0.59		0.79	0.59	0.69
	pricerange	0.59					
	stars	0.69		0.59	0.69		0.59
	type	0.79					
Restaurant	area	0.79		0.59	0.79		0.59
	bookday	0.49			0.59	0.49	0.69
	bookpeople	0.59			0.69		
	booktime	0.79			0.69		0.49
	food	0.59	0.49	0.59	0.59	0.49	0.59
	name	0.49			0.59		
	pricerange	0.69	0.49	0.69	0.69	0.49	0.69
	arriveBy	0.49					
departure	0.79						
Taxi	destination	0.79			0.79	0.69	
	leaveAt	0.59			0.59		0.49
	arriveBy	0.79					
	bookpeople	0.79					
Train	day	0.69	0.79		0.69	0.79	
	departure	0.79			0.69		
	destination	0.79					
	leaveAt	0.69	0.79	0.59	0.69	0.79	0.59

Table 6.4: Selection of Optimal Thresholds for different variations of the BERT-Large Model.

Services	Slots	Intent Detection			No Intent Detection		
		None	Lev	STS	None	Lev	STS
Attraction	area	0.90					
	name	0.95					
	type	0.95	0.90	0.95	0.95	0.90	0.95
Hotel	area	0.95			0.95	0.85	0.95
	bookday	0.90			0.90	0.80	0.90
	bookpeople	0.80			0.95		
	bookstay	0.95					
	internet	0.95			0.95		0.80
	name	0.95					
	parking	0.95			0.95		0.80
	pricerange	0.90	0.95	0.80	0.90		0.95
	stars	0.95					
	type	0.95					
	Restaurant	area	0.95				
bookday		0.80		0.85	0.95	0.80	0.95
bookpeople		0.95					
booktime		0.95		0.90	0.95		
food		0.90		0.95	0.90		0.85
name		0.95					
pricerange		0.95	0.80	0.90	0.80	0.90	0.80
arriveBy		0.95			0.95		0.90
Taxi	departure	0.95					
	destination	0.95					
	leaveAt	0.95					
	arriveBy	0.95					
Train	bookpeople	0.95		0.90	0.95		
	day	0.95			0.95	0.80	0.95
	departure	0.95					
	destination	0.95					
	leaveAt	0.95					

Table 6.5: Selection of Optimal Thresholds for different variations of the T5 Model.

6.1.2 Evaluation of the QA Models

As mentioned earlier in Chapter 2.3.3, evaluating dialogue systems is crucial. In the case of task-based dialogue systems, which include the DST, the primary goal is to perform specific tasks. When applied to the Portuguese dataset we constructed, these models needed to be capable of handling tasks that involved searching for attractions, finding and booking hotels and restaurants, locating taxis, and finding and booking trains.

To verify if the models were executing these functions correctly, they were evaluated using two metrics: JGA and Slot F1. As elaborated in Chapter 2.3.3, JGA is a metric that measures the system's overall understanding in relation to the user's purpose. Despite being a rigid metric due to its "all or nothing" nature, JGA is an effective way to discern whether the developed system fully grasps the user's intent – an element that's crucial in task-oriented dialogues.

Slot F1 measures the system's ability to accurately fill specific pieces of information (slots). Its calculation relies on precision (the proportion of slots that the system filled correctly) and recall (the proportion of all correct slots that the system managed to fill). The Slot F1 Score, computed as the harmonic mean of precision and recall, offers a balance between these two aspects, ensuring the system not only correctly identifies and fills slots but also does not overlook any correct slots. Both metrics are vital in ensuring that dialogue systems can effectively assist users in their daily lives in achieving their intended goals.

Having selected the optimal thresholds for each variation of the QA models, we retested them on the remaining 312 Portuguese dialogues. In this model reevaluation, the best threshold value for each specific model variation was applied to each slot.

This reassessment resulted in Tables 6.6 and 6.7, which showcase the results obtained for the metrics JGA and Slot F1 respectively, incorporating the use of Intent Detection for the different forms of post-processing. Upon analyzing these tables, we can conclude the following:

- For Table 6.6:
 - For the Attraction service, the BERT-base model with the STS method achieved the highest JGA score (0.35).
 - For the Hotel service, the BERT-base model with the Levenshtein method achieved the highest JGA score (0.32).
 - For the Restaurant service, the BERT-large model with the STS method achieved the highest JGA score (0.30).
 - For the Taxi service, the BERT-base model with the Levenshtein method achieved the highest JGA score (0.47).
 - For the Train service, the BERT-large model with the Levenshtein method achieved the highest JGA score (0.57).
- For Table 6.7:

- For the Attraction service, the BERT-base model with the STS method achieved the highest F1 score (0.54).
- For the Hotel service, the BERT-base model with the Levenshtein method achieved the highest F1 score (0.50).
- For the Restaurant service, the BERT-large model with the STS method achieved the highest F1 score (0.50).
- For the Taxi service, the BERT-base model with the Levenshtein method achieved the highest F1 score (0.52).
- For the Train service, the BERT-large model with the Levenshtein method achieved the highest F1 score (0.76).

Lastly, we also generated Tables 6.8 and 6.9, which display the results obtained for the metrics JGA and Slot F1 respectively, this time excluding the use of Intent Detection for the different forms of post-processing. Upon analyzing these tables, we can draw the following conclusions

- For Table 6.8:
 - For the Attraction service, the BERT-base model with the Levenshtein method achieved the highest JGA score (0.16).
 - For the Hotel service, the BERT-base and BERT-large models with the Levenshtein method achieved the highest JGA score (0.13).
 - For the Restaurant service, the BERT-base model with the Levenshtein method achieved the highest JGA score (0.19).
 - For the Taxi service, the BERT-large model with the STS method achieved the highest JGA score (0.17).
 - For the Train service, the BERT-large model with the Levenshtein method achieved the highest JGA score (0.44).
- For Table 6.9:
 - For the Attraction service, the BERT-base model with the Levenshtein method achieved the highest F1 score (0.32).
 - For the Hotel service, the BERT-base and BERT-large models with the Levenshtein method achieved the highest F1 score (0.27).
 - For the Restaurant service, the BERT-base and BERT-large models with the Levenshtein method achieved the highest F1 score (0.38).
 - For the Taxi service, the BERT-large model with the STS method achieved the highest F1 score (0.31).
 - For the Train service, the BERT-large model with the Levenshtein method achieved the highest F1 score (0.59).

Complementing the four main tables, two additional tables were generated, which can be found in Appendix 7.2, presenting an in-depth analysis of the accuracy achieved across the 30 slots for each model variation, thus providing a comprehensive view of performance on an individual slot basis.

6.1. Testing of the QA Models

Services	BERT Base			BERT Large			T5		
	None	Lev	STS	None	Lev	STS	None	Lev	STS
Attraction	0.20	0.30	0.35	0.13	0.22	0.22	0.16	0.28	0.29
Hotel	0.27	0.32	0.27	0.28	0.28	0.26	0.19	0.22	0.18
Restaurant	0.23	0.28	0.29	0.20	0.20	0.30	0.14	0.23	0.16
Taxi	0.44	0.47	0.44	0.42	0.41	0.44	0.38	0.38	0.38
Train	0.32	0.54	0.38	0.35	0.57	0.41	0.30	0.54	0.38

Table 6.6: Results achieved by the QA Models for the JGA metric, incorporating Intent Detection and various types of Post-processing.

Services	BERT Base			BERT Large			T5		
	None	Lev	STS	None	Lev	STS	None	Lev	STS
Attraction	0.37	0.49	0.54	0.29	0.42	0.42	0.37	0.51	0.52
Hotel	0.44	0.50	0.42	0.44	0.46	0.40	0.40	0.44	0.39
Restaurant	0.46	0.49	0.49	0.41	0.44	0.50	0.38	0.48	0.43
Taxi	0.47	0.52	0.50	0.47	0.47	0.49	0.47	0.49	0.49
Train	0.66	0.73	0.63	0.71	0.76	0.64	0.64	0.73	0.64

Table 6.7: Results achieved by the QA Models for the Slot F1 metric, incorporating Intent Detection and various types of Post-processing.

Services	BERT Base			BERT Large			T5		
	None	Lev	STS	None	Lev	STS	None	Lev	STS
Attraction	0.08	0.16	0.13	0.06	0.14	0.11	0.05	0.11	0.11
Hotel	0.10	0.13	0.10	0.12	0.13	0.11	0.08	0.10	0.07
Restaurant	0.10	0.19	0.13	0.11	0.17	0.15	0.06	0.11	0.08
Taxi	0.06	0.12	0.16	0.07	0.09	0.17	0.05	0.07	0.16
Train	0.15	0.39	0.16	0.20	0.44	0.20	0.14	0.36	0.16

Table 6.8: Results achieved by the QA Models for the JGA metric, without Intent Detection but including various types of Post-processing.

Services	BERT Base			BERT Large			T5		
	None	Lev	STS	None	Lev	STS	None	Lev	STS
Attraction	0.17	0.32	0.28	0.15	0.29	0.26	0.11	0.19	0.23
Hotel	0.20	0.27	0.19	0.26	0.27	0.21	0.18	0.21	0.18
Restaurant	0.27	0.38	0.31	0.30	0.38	0.35	0.21	0.29	0.25
Taxi	0.11	0.20	0.30	0.13	0.16	0.31	0.09	0.15	0.30
Train	0.40	0.53	0.36	0.48	0.59	0.39	0.38	0.50	0.35

Table 6.9: Results achieved by the QA Models for the Slot F1 metric, without Intent Detection but including various types of Post-processing.

6.2 Summary

Based on the results obtained in this chapter, we can draw some conclusions:

- Firstly, regarding the choice of thresholds, we can conclude that for the various variations made in the BERT-large and BERT-base models, there was greater variability in the choice of thresholds for the various slots than in the T5 model. In most variations of this model, the most chosen threshold was 0.95.
- The second conclusion that can be drawn from the analysis of Tables 6.6, 6.7, 6.8, and 6.9 is that in none of the variations, either with or without the use of Intent Detection and Post-Processing, does the T5 model outperform the BERT-base and BERT-large models for any service.
- The third conclusion from the analysis of these same tables is that the models, in all cases, perform better with the use of Intent Detection. This is due to the fact that models using Intent Detection are provided with only a restricted number of questions corresponding to the user’s specific intention, thereby reducing the tendency to predict incorrect slots. This is not the case when Intent Detection is not used, as the model must ask all 30 questions defined for each user sentence, increasing the probability that the slots will be predicted incorrectly.
- Regarding the Post-Processing methods, we can conclude that in no situation did the models benefit from not using Post-Processing (None). On the other hand, models that made use of these Post-Processing methods, can be concluded that:
 - Using Intent Detection: For the best results obtained from the models both methods performed similarly. The Levenshtein method only outperforms the STS method in one more case.
 - No Intent Detection: For the best results obtained from the models the use of the Levenshtein method yields better results in this situation. A brief explanation for this might be that without Intent Detection, there are more slots filled. This leads the STS post-processing method to compare slots that do not align as well with the original slot, to the method’s detriment. Conversely, since the Levenshtein method only compares the distances between slots, it is more forgiving of these discrepancies, allowing it to identify more "correct" cases.
- We can also identify a pattern in these four tables. For example, in the tables that contain the JGA metric, the specifications of the models that led to the best results are the same specifications present in the F1 Tables that also led to the best results. These two metrics are related since they assess the ability of a model to track slots over a dialogue. Therefore, it is expected that a high JGA score also has a high Slot F1 score, because being able to predict all slots correctly (high JGA) necessarily implies having good precision and recall in slot prediction. However, the opposite may not be true: a model can have

6.2. Summary

a good Slot F1 score without having a high JGA score, as Slot F1 allows for errors in some slots. This case can even be seen in Table 6.9, where BERT-large and BERT-base models using the Levenshtein method jointly obtained the highest Slot F1 value, but only in Table 6.9 did the BERT-base model achieve the highest JGA value.

- In choosing between the BERT-base and BERT-large models, we conclude that no model consistently outperforms each other in all scenarios, indicating that the choice of the model may depend on the service in question that we intend to solve. As detailed in Chapter 6.1.1, it is important to note that the BERT-base model is smaller than the BERT-large model, which comes with its own set of advantages and limitations, as mentioned in that chapter.

In summary, from a commercial point of view, there would be various versions of these two models available for real-world scenarios. Companies operating on a lower budget could opt for models without Intent Detection, while those with a larger budget could be offered models with Intent Detection. Subsequently, the appropriate Post-Processing method would be integrated into these models, tailored to the specific needs of the company.

Chapter 7

Conclusions and Future Work

Nowadays, with the rapidly evolving field of AI, particularly in dialogue systems, DST and QA techniques have emerged as critical components for improving the understanding and functionality of these systems. In the course of this thesis, we have delved into the convergence of these two realms, harnessing the QA paradigm to enhance the performance of DST in monitoring the context of conversations. We explored this area, a journey abundant with iterative experimentation and improvements, which not only reinforced the understanding we have of these intricate processes but also illuminated new paths for practical applications and future investigations.

Even today, this is an area that remains heavily researched, with a multitude of practical implementations. The creation and utilization of a task-oriented dataset in Portuguese represents a significant leap in AI research within the Portuguese-speaking community. The varied linguistic and cultural nuances encapsulated within this dataset provide invaluable resources for training and fine-tuning AI models. We are confident that it will not only inspire the development of additional models tailored for the Portuguese language but also empower the Portuguese-speaking community to actively participate in, and shape, the ongoing AI revolution. The inception of this dataset holds the potential to position Portugal and other Portuguese-speaking countries at the forefront of AI research and development.

7.1 Conclusions

Considering the goals and contributions stated in Chapter 1, it can be confidently stated that the defined goals were successfully achieved, making a significant contribution to the field of DST.

Starting with the first objective, we can see in Chapter 3 the initial study conducted on two methods of creating datasets. This analysis is crucial as it forms the foundation for developing datasets to be used in dialogue systems. A comparison of the datasets created thus far was carried out to better understand their structure and scope. Subsequently, various approaches to DST were analyzed,

with their respective evaluations displayed. This allows us to understand the successful cases within the field and the reasons for their success.

Chapter 4 highlights the second goal: the creation of the first task-oriented dataset for Portuguese. This chapter explains the inspiration behind the dataset's creation, its structure, and the various applications it could have in the NLP area.

Chapter 5 addresses the third and fourth objectives, showcasing the models that we utilize for QA in Portuguese. We provide a detailed overview of their structure, training process, and the questions developed specifically for them. This enables their application to the Portuguese dataset created earlier, facilitating the slot-filling process to accurately extract and monitor dialogues.

Finally, the fourth objective was achieved, as demonstrated in Chapter 6. Several model variations, with or without Intent Detection and Post-Processing, were applied to the five services available in the Portuguese dataset. The findings in that chapter indicate that in terms of threshold selection for each model, T5 exhibited the least variation, with 0.95 as the most commonly selected value. Additionally, the results showed that the models performed better when utilizing both Intent Detection and Post-Processing techniques. In the end, the standout models were BERT-base and BERT-large, with the choice between the two depending on the specific service requirements.

All this analysis confirms that the objectives established for this study have been successfully achieved, and the approach adopted has resulted in several significant contributions to the DST field:

- The creation of MultiWOZpt, the first task-oriented dataset in Portuguese, is now available on GitHub¹.
- Application of recent QA-based approaches for slot-filling to Portuguese dialogues.
- Thorough testing of QA models for DST, as detailed in this chapter, with conclusions drawn from their application across several variations.
- A scientific paper that is currently under revision for publication in the 29th Portuguese Conference on Pattern Recognition².
- The writing of this document, which will not only facilitate a better understanding of the work conducted here but also serve as a foundation for future research in this area.

¹<https://github.com/NLP-CISUC/MultiWOZpt>

²<https://recpad2023.isec.pt/index.php/call-for-papers/>

7.2 Future Work

In this section, the future work related to the dataset and the models that were applied is described. The possible future directions include:

- **Augmenting the MultiWOZpt dataset:** the ongoing adaptation of services from the city of Cambridge to the city of Coimbra, and translating the existing dialogues in MultiWOZ from English to Portuguese. This will enable the creation of increasingly robust dialogue systems, with a greater amount of data available for training and testing.
- **Fine-Tuned Models:** As the Portuguese dataset continues to grow, models can initially be trained on it and subsequently fine-tuned. Recently, some models specifically designed for the Portuguese language have been released. For example, ALBERTina³, after being fine-tuned for the QA task, can be applied to this dataset to serve as an additional point of comparison with other models that have already been tested.
- **Creating an Intent Detection Classifier:** Since the dataset includes annotations of intents, an Intent Detection Classifier can be developed in the future, allowing the models to be applied to new dialogues, where intents are not annotated.
- **Testing New Sets of Questions:** Make new sets of questions to be tested for the various models, leading to a conclusion about the best set of questions that fit each slot of each service.
- **Addressing Unseen Slots:** A common issue is the filling of unseen slots, where users may ask for addresses, phone numbers, and other information that originally had no designated slots. Addressing this will enable the models to better handle everyday tasks.
- **Investigating Other Post-Processing Methods:** Beyond these two processing methods that we use, others can be applied in this context. For example, BLEU [Papineni et al., 2002], a metric commonly used to evaluate the quality of machine-generated text compared to human-generated reference text, could be used to compare the generated slot-filling output (extracted slots) with the ground truth reference slots. Additionally, techniques like pruning may be used to remove redundant or irrelevant slots that do not contribute significantly to overall dialogue understanding.
- **Writing other Scientific Paper:** We are in the process of preparing other scientific paper for potential publication in the 16th International Conference on Computational Processing of Portuguese⁴ (PROPOR 2024).

This comprehensive look at potential future developments provides valuable insights into the ways the models and dataset can be further improved and applied.

³<https://huggingface.co/PORTULAN/albertina-ptpt>

⁴<https://propor2024.citius.gal/>

References

- Layla El Asri, Hannes Schulz, Shikhar Sharma, Jeremie Zumer, Justin Harris, Emery Fine, Rahul Mehrotra, and Kaheer Suleman. Frames: a corpus for adding memory to goal-oriented dialogue systems. *arXiv preprint arXiv:1704.00057*, 2017.
- Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Inigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gašić. Multiwoz—a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. *arXiv preprint arXiv:1810.00278*, 2018.
- Kathi Canese and Sarah Weis. Pubmed: the bibliographic database. *The NCBI handbook*, 2(1), 2013.
- Guan-Lin Chao and Ian Lane. Bert-dst: Scalable end-to-end dialogue state tracking with bidirectional encoder representations from transformer. *arXiv preprint arXiv:1907.03040*, 2019.
- Wei Chen, Yeyun Gong, Song Wang, Bolun Yao, Weizhen Qi, Zhongyu Wei, Xiaowu Hu, Bartuer Zhou, Yi Mao, Weizhu Chen, Biao Cheng, and Nan Duan. DialogVED: A pre-trained latent variable encoder-decoder model for dialog response generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4852–4864, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.acl-long.333. URL <https://aclanthology.org/2022.acl-long.333>.
- Kenneth Mark Colby, Sylvia Weber, and Franklin Dennis Hilf. Artificial paranoia. *Artificial Intelligence*, 2(1):1–25, 1971.
- Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- Hubert L Dreyfus. *What computers still can't do: A critique of artificial reason*. MIT press, 1992.
- Ming Du, Luyi Cheng, Bo Xu, Zhijun Wang, Sufen Wang, Junyi Yuan, and Changqing Pan. Ts-dst: A two-stage framework for schema-guided dialogue state tracking with selected dialogue history. In *2022 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2022.
- Jacob Eisenstein. *Natural Language Processing*. MIT Press, September 2019.

References

- Anthony Fader, Stephen Soderland, and Oren Etzioni. Identifying relations for open information extraction. In *Proceedings of the 2011 conference on empirical methods in natural language processing*, pages 1535–1545, 2011.
- Shuyang Gao, Abhishek Sethi, Sanchit Agarwal, Tagyoung Chung, and Dilek Hakkani-Tur. Dialog state tracking: A neural reading comprehension approach. *arXiv preprint arXiv:1908.01946*, 2019.
- Giuliano Giacaglia. How transformers work, 2019. URL <https://towardsdatascience.com/transformers-141e32e69591>. Accessed: 2022-12-21.
- Bert F. Green, Alice K. Wolf, Carol L. Chomsky, and Kenneth Laughery. Baseball: an automatic question-answerer. In *IRE-AIEE-ACM '61 (Western)*, 1961.
- Pavel Gulyaev, Eugenia Elistratova, Vasily Konovalov, Yuri Kuratov, Leonid Pugachev, and Mikhail Burtsev. Goal-oriented multi-task bert-based dialogue state tracker. *arXiv preprint arXiv:2002.02450*, 2020.
- Marti A Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING 1992 Volume 2: The 14th International Conference on Computational Linguistics*, 1992.
- Michael Heck, Carel van Niekerk, Nurul Lubis, Christian Geishauser, Hsien-Chin Lin, Marco Moresi, and Milica Gašić. Trippy: A triple copy strategy for value independent neural dialog state tracking. *arXiv preprint arXiv:2005.02877*, 2020.
- Charles T Hemphill, John J Godfrey, and George R Doddington. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.
- Matthew Henderson, Blaise Thomson, and Steve Young. Deep neural network approach for the dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 467–471, 2013.
- Matthew Henderson, Blaise Thomson, and Jason D Williams. The second dialog state tracking challenge. In *Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL)*, pages 263–272, 2014a.
- Matthew Henderson, Blaise Thomson, and Jason D Williams. The third dialog state tracking challenge. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 324–329. IEEE, 2014b.
- Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. Learning a neural semantic parser from user feedback. *arXiv preprint arXiv:1704.08760*, 2017.
- Heng Ji and Ralph Grishman. Knowledge base population: Successful approaches and challenges. In *Proceedings of the 49th annual meeting of the association for computational linguistics: Human language technologies*, pages 1148–1158, 2011.

- Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 2020.
- Daniel Jurafsky and James H Martin. Speech and language processing (draft 3rd). *preparation [cited 2020 June 1] Available from: <https://web.stanford.edu/~jurafsky/slp3>*, 2022.
- John F Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Transactions on Information Systems (TOIS)*, 2(1):26–41, 1984.
- Seokhwan Kim, Luis Fernando D’Haro, Rafael E Banchs, Jason D Williams, and Matthew Henderson. The fourth dialog state tracking challenge. In *Dialogues with Social Robots*, pages 435–449. Springer, 2017.
- Sungdong Kim, Sohee Yang, Gyuwan Kim, and Sang-Woo Lee. Efficient dialogue state tracking by selectively overwriting memory. *arXiv preprint arXiv:1911.03906*, 2019.
- Young-Bum Kim, Dongchan Kim, Anjishnu Kumar, and Ruhi Sarikaya. Efficient large-scale neural domain classification with personalized attention. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2214–2224, 2018.
- Adarsh Kumar, Peter Ku, Anuj Goyal, Angeliki Metallinou, and Dilek Hakkani-Tur. Ma-dst: Multi-attention-based scalable dialog state tracking. In *Proceedings of the AAI conference on artificial intelligence*, volume 34, pages 8107–8114, 2020.
- Hwaran Lee, Jinsik Lee, and Tae-Yoon Kim. Sumbt: Slot-utterance matching for universal and scalable belief tracking. *arXiv preprint arXiv:1907.07421*, 2019.
- Miao Li, Haoqi Xiong, and Yunbo Cao. The sppd system for schema guided dialogue state tracking challenge. *arXiv preprint arXiv:2006.09035*, 2020.
- Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.
- Yue Ma, Zengfeng Zeng, Dawei Zhu, Xuan Li, Yiying Yang, Xiaoyuan Yao, Kaijie Zhou, and Jianping Shen. An end-to-end dialogue state tracking system with machine reading comprehension and wide & deep classification. *arXiv preprint arXiv:1912.09297*, 2019.
- Christopher D Manning. *Introduction to information retrieval*. Syngress Publishing,, 2008.
- Kay Martin, Thompson Henry, Artificial Intelligence, et al. Gus, a frame-driven dialog system. *Artificial Intelligence*, 8:155–173, 1977.

- Angeliki Metallinou, Dan Bohus, and Jason D Williams. Discriminative state tracking for spoken dialog systems. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 466–475, 2013.
- Rada Mihalcea and Andras Csomai. Wikify! linking documents to encyclopedic knowledge. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 233–242, 2007.
- Nikola Mrkšić, Diarmuid O Séaghdha, Tsung-Hsien Wen, Blaise Thomson, and Steve Young. Neural belief tracker: Data-driven dialogue state tracking. *arXiv preprint arXiv:1606.03777*, 2016.
- Nishanth N. Question answering system with bert, 2020. URL <https://medium.com/analytics-vidhya/question-answering-system-with-bert-eb1130f8def>. Accessed: 2023-08-20.
- Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.
- Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*, 2016.
- Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don’t know: Unanswerable questions for squad. *arXiv preprint arXiv:1806.03822*, 2018.
- Abhinav Rastogi, Xiaoxue Zang, Srinivas Sunkara, Raghav Gupta, and Pranav Khaitan. Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AACL Conference on Artificial Intelligence*, volume 34, pages 8689–8696, 2020.
- Dr J Rogel-Salazar. What are transformers models?, 2022. URL <https://www.dominodatalab.com/blog/transformers-self-attention-to-the-rescue>. Accessed: 2022-12-20.
- Stephen Roller, Emily Dinan, Naman Goyal, Da Ju, Mary Williamson, Yinhan Liu, Jing Xu, Myle Ott, Kurt Shuster, Eric M Smith, et al. Recipes for building an open-domain chatbot. *arXiv preprint arXiv:2004.13637*, 2020.
- Yu-Ping Ruan, Zhen-Hua Ling, Jia-Chen Gu, and Quan Liu. Fine-tuning bert for schema-guided zero-shot dialogue state tracking. *arXiv preprint arXiv:2002.00181*, 2020.
- Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

- José Santos, Luís Duarte, João Ferreira, Ana Alves, and Hugo Gonçalo Oliveira. Developing Amaia: A conversational agent for helping portuguese entrepreneurs — an extensive exploration of question-matching approaches for Portuguese. *Information*, 11(9), 2020.
- Pararth Shah, Dilek Hakkani-Tür, Gokhan Tür, Abhinav Rastogi, Ankur Bapna, Neha Nayak, and Larry Heck. Building a conversational agent overnight with dialogue self-play. *arXiv preprint arXiv:1801.04871*, 2018.
- Fábio Souza, Rodrigo Nogueira, and Roberto Lotufo. Bertimbau: Pretrained bert models for brazilian portuguese. In Ricardo Cerri and Ronaldo C. Prati, editors, *Intelligent Systems*, pages 403–417, Cham, 2020. Springer International Publishing. ISBN 978-3-030-61377-8.
- Kai Sun, Lu Chen, Su Zhu, and Kai Yu. A generalized rule based tracker for dialogue state tracking. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 330–335. IEEE, 2014.
- Oriol Vinyals and Quoc V. Le. A neural conversational model. In *Proceedings of ICML 2015 Deep Learning Workshop*, Lille, France, 2015.
- Hao Wang, Zhengdong Lu, Hang Li, and Enhong Chen. A dataset for research on short-text conversations. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 935–945, 2013.
- Zhuoran Wang and Oliver Lemon. A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information. In *Proceedings of the SIGDIAL 2013 Conference*, pages 423–432, 2013.
- Joseph Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.
- Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 404–413, Metz, France, August 2013. Association for Computational Linguistics. URL <https://aclanthology.org/W13-4065>.
- Jason D Williams. Web-style ranking and slu combination for dialog state tracking. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 282–291, 2014.
- Jason D Williams, Antoine Raux, and Matthew Henderson. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33, 2016.
- Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. Break it down: A question understanding benchmark. *Transactions of the Association for Computational Linguistics*, 8:183–198, 2020.
- Chien-Sheng Wu, Andrea Madotto, Ehsan Hosseini-Asl, Caiming Xiong, Richard Socher, and Pascale Fung. Transferable multi-domain state generator for task-oriented dialogue systems. *arXiv preprint arXiv:1905.08743*, 2019.

References

- Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*, 2018.
- John M Zelle and Raymond J Mooney. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055, 1996.
- Jian-Guo Zhang, Kazuma Hashimoto, Chien-Sheng Wu, Yao Wan, Philip S Yu, Richard Socher, and Caiming Xiong. Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. *arXiv preprint arXiv:1910.03544*, 2019.
- Victor Zue, Stephanie Seneff, James R Glass, Joseph Polifroni, Christine Pao, Timothy J Hazen, and Lee Hetherington. Juplter: a telephone-based conversational interface for weather information. *IEEE Transactions on speech and audio processing*, 8(1):85–96, 2000.

Appendices

Appendix A

This Appendix includes crucial tables containing information about the constitution of the database, which played a vital role in constructing the dataset. These tables reveal the Cambridge services that have been adapted to Coimbra services.

Type	MultiWOZ 2.2	MultiWOZpt 0.1
architecture	all saints church Great saint mary's church holy trinity church old schools	Biblioteca Joanina Mosteiro de Santa Clara Igreja de São Tiago Paço das Escolas
boat	camboats riverboat georgina the cambridge punter	O Basólias Serranas do Mondego Ponte de Santa Clara
cinema	cineworld cinema vue cinema	Casa do Cinema de Coimbra Cinemas Nos Alma Shopping
college	christ's college churchill college clare college clare hall corpus christi hughes hall queens' college saint catharine's college saint john's college	Colégio Rainha Santa Isabel Escola Superior Agrária de Coimbra Colégio de Jesus Faculdade de Farmácia da Universidade de Coimbra Faculdade de Medicina Colégio de São Caetano Torre da Universidade de Coimbra Colégio Rainha Santa Isabel Real Colégio das Artes e Humanidades
concerthall	the man on the moon	Salão Brazil
entertainment	Cherry Hinton Hall and Grounds funky fun house nusha tenpin	Jardim da Sereia Feijão Verde Fun Park Coimbra Quebra bar Tic Tac Boom Escape Room
museum	Broughton House Gallery byard art cafe jello gallery cambridge and county folk museum cambridge artworks cambridge contemporary art cambridge museum of technology castle galleries kettle's yard regency gallery ruskin gallery whipple museum of the history of science	Museu nacional Machado de Castro Galeria sete Galeria Santa Clara Exploratório - Centro Ciência Viva Casa Museu Bissaya Barreto Centro de Arte Contemporânea de Coimbra Sala Da Cidade Museu da Água Museu da Ciência da Universidade de Coimbra Galerias Topazio Casa da Mutualidade Museu Miguel Torga
mutliple sports	the cherry hinton village centre	Estádio Universitário de Coimbra
nightclub	ballare club salsa the place	NB Club LIT Cobra Dourada
park	cambridge university botanic gardens cherry hinton water play milton country park wandlebury country park	Jardim Botânico da Universidade de Coimbra Complexo Olímpico de Piscinas Parque Verde do Mondego Parque Dr. Manuel Braga
swimmingpool	abbey pool and astroturf pitch kings hedges learner pool parkside pools	Associação de Natação de Coimbra Piscinas Municipais Rui Abreu Piscinas Municipais Luís Lopes da Conceição
theatre	adc theatre cambridge arts theatre mumford theatre the cambridge corn exchange the junction	Teatrão Coimbra Impro Teatro Académico de Gil Vicente Círculo de Iniciação Teatral da Academia de Coimbra Casa do Cinema de Coimbra

Table 1: Cambridge Attractions adapted to Coimbra.

Type	MultiWOZ 2.2	MultiWOZpt 0.1
Guesthouse	a and b guest house acorn guest house alexander bed and breakfast allenbell alpha-milton guest house arbury lodge guesthouse archway house autumn house avalon aylesbray lodge guest house carolina bed and breakfast city centre north b and b el shaddai finches bed and breakfast hamilton lodge hobsons house home from home kirkwood house leverton house limehouse rosa's bed and breakfast warkworth house worth house	Stay Hotel Coimbra Centro AQ 188 Guest House NN Guest House CSI Coimbra & Guest House Despertar Saudade Guest House Santa Clara Solar Antigo Luxury Miguel Torga Room Casa dos Carvalhos CoimbraAmeias Casa de São Bento Guest House Infante Dom Henrique Guesthouse Lusa Atenas Guesthouse Coimbra City Moderna Oryza Guest House& Suites The Luggage NJOY Coimbra Internacional Residencial Gouveia Jantesta Guest House Celas Dream Studio Epic Days Guest House
Hotel	ashley hotel bridge guest house cityroomz express by holiday inn cambridge gonville hotel huntingdon marriott hotel lovell lodge the cambridge belfry the lensfield hotel university arms hotel	Hotel Mondego Hotel Astória Hotel Oslo Quinta Dona Iria Hotel Dona Ines Coimbra & Congress Center Palácio São Silvestre Hotel Vitória Hotel Domus Hotel D.Luis Hotel Quinta das Lagrimas

Table 2: Cambridge Hotels and Guesthouses adapted to Coimbra.

Type	MultiWOZ 2.2	MultiWOZpt 0.1
asian oriental	dojo noodle bar yippee noodle bar j restaurant saigon city	Casa da Massa SushiCome Izakaya Oni Shari Sushi Bar
british	the oak bistro the copper kettle Fitzbillies restaurant saint johns chop house grafton hotel restaurant midsummer house restaurant	Quê-Bê Restaurante Nova Briosa Notes Bar Kitchen Honorato Brunn's Diner Toca do gato
chinese	the good luck chinese food takeaway Wok Mondego charlie chan jinling noodle bar ugly duckling the lucky star golden house hakka golden wok the hotpot tang chinese	Lung Wah yu garden Kang Le Hanami Sushi Ishi Panda Tun Fon Lung Wah Fu Hua Monte Fuji Kyoto House
european	luca cucina bar riverside brasserie city stop restaurant cambridge lodge restaurant	Refeitro da Baixa Sete Restaurante Dux Taberna Urbana Compostu tavern
french	two two cote	BioEscolha Restaurante O Pipo
gastropub	the cow pizza kitchen and bar Bedouin the slug and lettuce	Allô Pizza Devaneio Gastropub The Murphy's Irish Pub
indian	curry garden taj tandoori curry prince cocum royal spice meghna panahar rajmahal	GulGuls Royal Tandoori Italian Indian Palace Chai2... Chai... Italian Indian Palace Mostarda Hilsa Taj Indian
international	the missing sock the varsity restaurant blossbury restaurant	Restaurante 39 Tapas nas Costas O Açude
italian	pizza hut city centre	pizza hut
italian	stazione restaurant and coffee bar Ask prezzo da vinci pizzeria la margherita pizza hut fen ditton zizzi cambridge frankie and bennys clowns cafe don pasquale pizzeria	Fixepizza Ragu fresh pasta Pizzeria Eurotropic Da Vinci l'arte della pizza Toscana Pizza Hut Alma Shopping Mr. Pizza Osteria 44 Il Tartufo Italianino
japanese	wagamama peking restaurant	Peculiar Sushi or Steak Sake
korean	little seoul	Seoul Chicken
mediterranean	the gardenia la mimosa shiraz restaurant	Fangas maior Refeitro da Baixa No tacho
mexican	chiquitito restaurant bar	Tu Taqueria Wey
modern european	darrys cookhouse and wine shop the river bar steakhouse and grill	Rio Mondego II Boutique Tapas Petiscos
portuguese	restaurant one seven nandos pipasha restaurant nandos city centre la raza	O Choupal Sol Do Mondego Casas do Bragal O Mimo Bar Diligência
spanish	la tasca	Cova Funda Espanhol
thai	sala thong	Dee Thai Food
turkish	meze bar anatolia efes restaurant	Efes Kebab Istanbul Kebab Kebab Mir Baba
vietnamese	thanh binh	Loggia

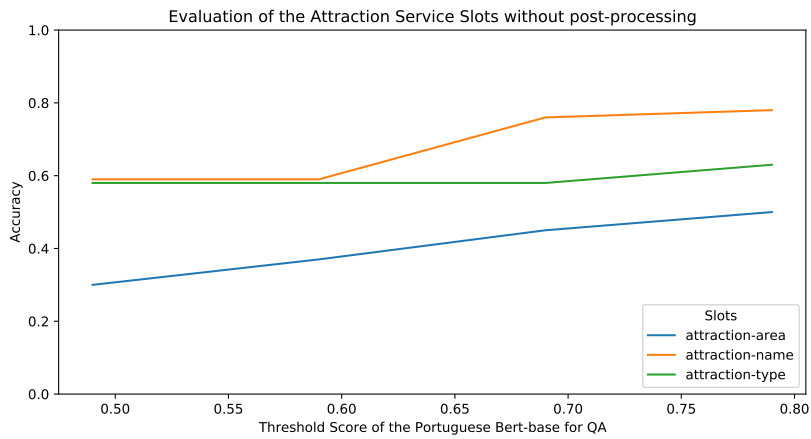
Table 3: Cambridge Restaurants adapted to Coimbra.

MultiWOZ 2.2	MultiWOZpt 0.1
Birmingham New Street	Lisboa Oriente
Bishops Stortford	Curia
Broxbourne	Pombal
Cambridge	Coimbra
Ely	Souselas
Kings Lynn	Pereira
Leicester	Entroncamento
London Kings Cross	Aveiro
London Liverpool Street	Fontela
Norwich	Fátima
Peterborough	Figueira da Foz
Stansted Airport	Mealhada
Stevenage	Oliveira do Bairro

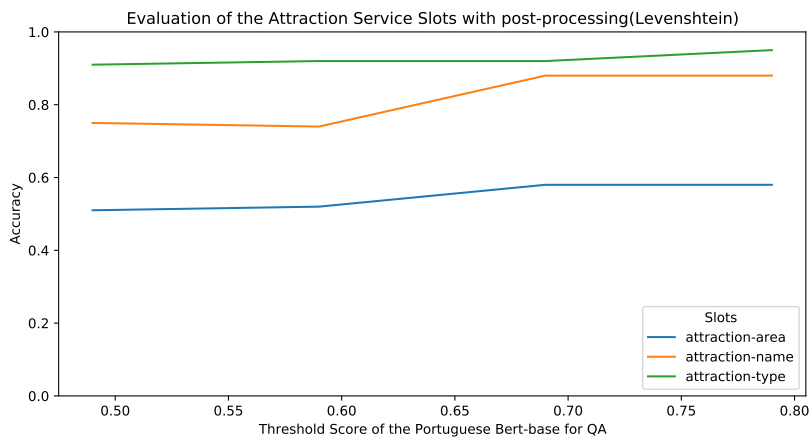
Table 4: Cambridge Trains adapted to Coimbra.

Appendix B

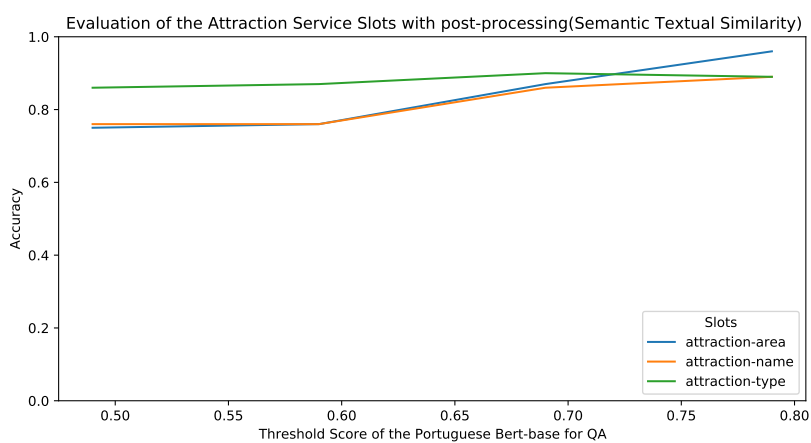
In this appendix, we present several figures illustrating the performance of various model variations across different thresholds.



(a) Post-Processing: None

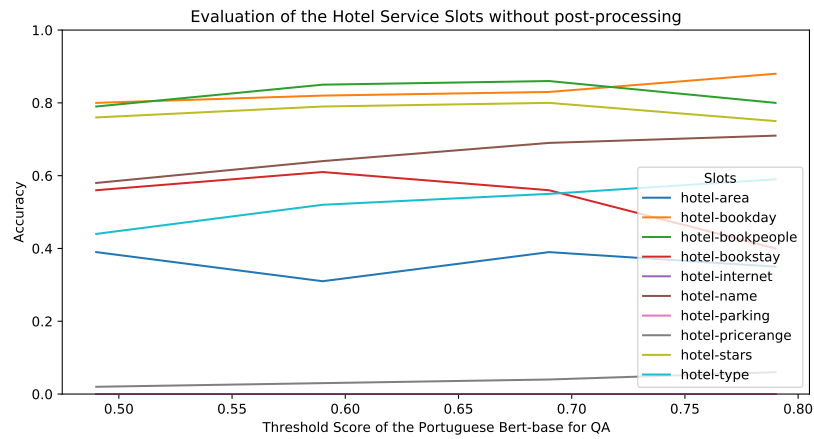


(b) Post-Processing: Levenshtein Distance

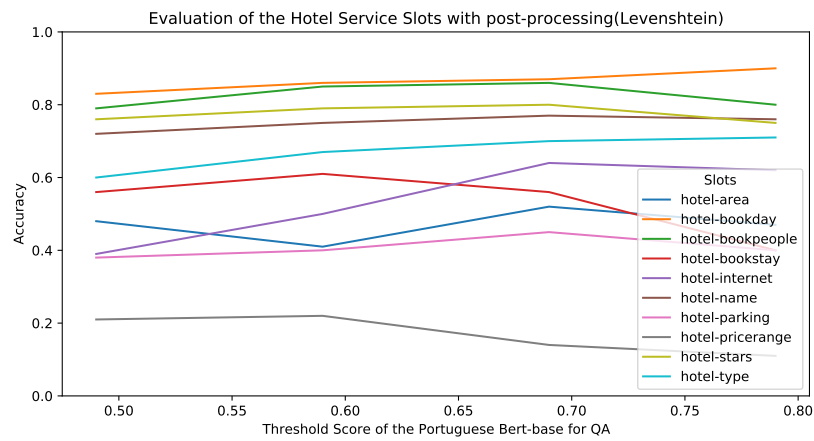


(c) Post-Processing: STS

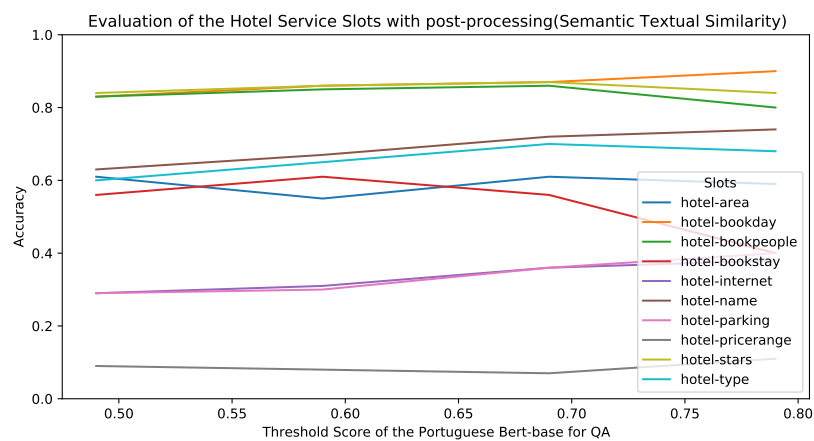
Figure 1: Evaluating the BERT-base Model with Intent Detection across different slots of the Attraction Service using various thresholds.



(a) Post-Processing: None

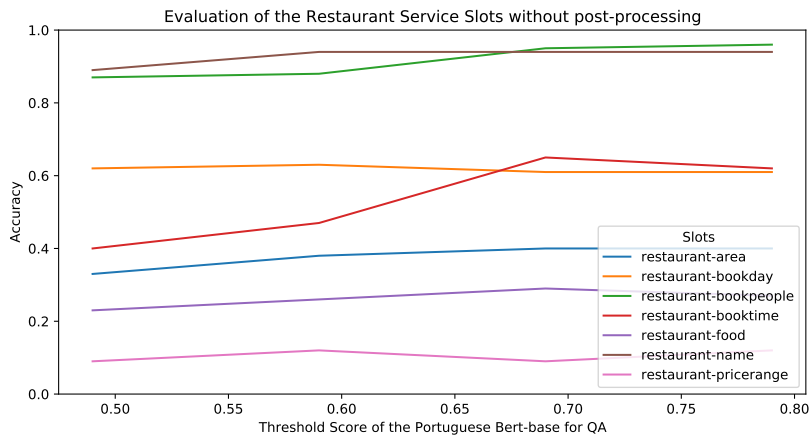


(b) Post-Processing: Levenshtein Distance

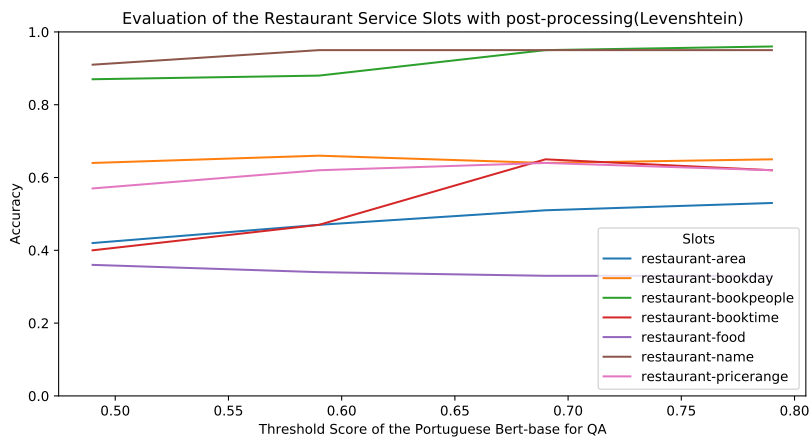


(c) Post-Processing: STS

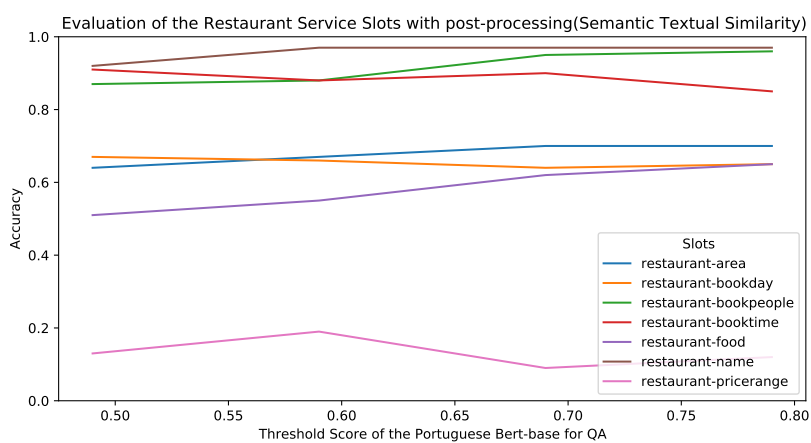
Figure 2: Evaluating the BERT-base Model with Intent Detection across different slots of the Hotel Service using various thresholds.



(a) Post-Processing: None

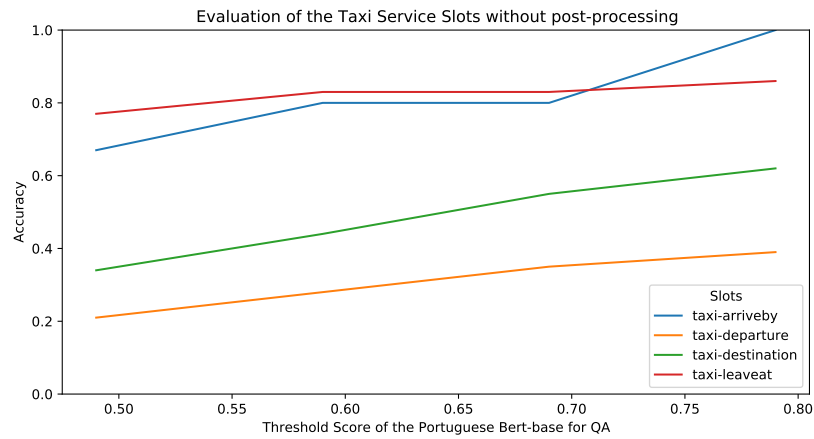


(b) Post-Processing: Levenshtein Distance

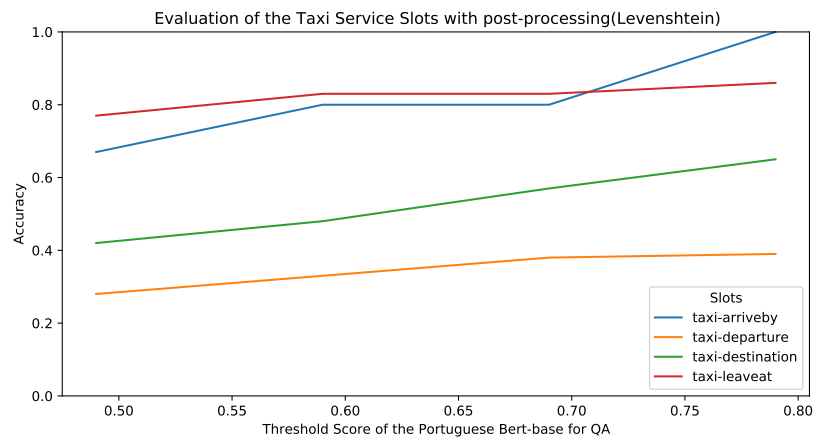


(c) Post-Processing: STS

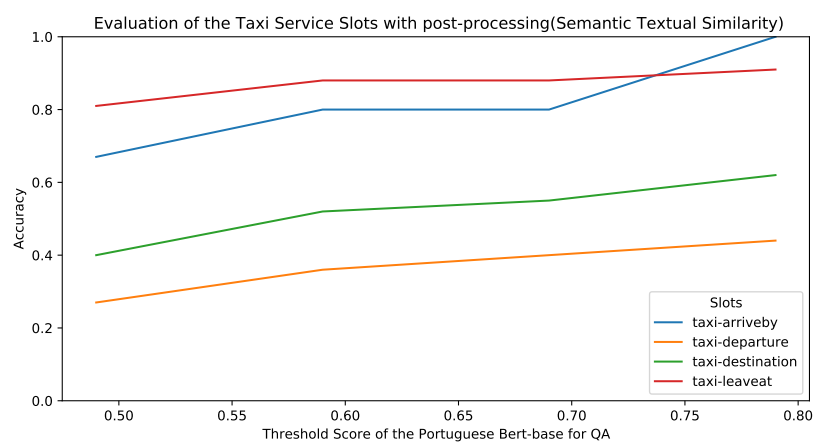
Figure 3: Evaluating the BERT-base Model with Intent Detection across different slots of the Restaurant Service using various thresholds.



(a) Post-Processing: None

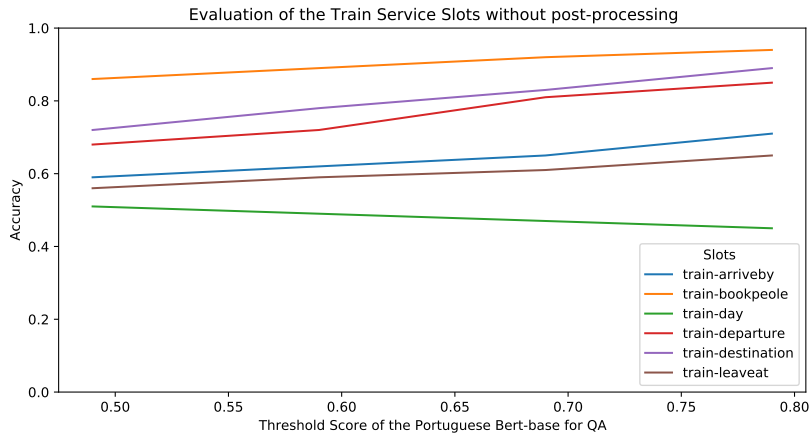


(b) Post-Processing: Levenshtein Distance

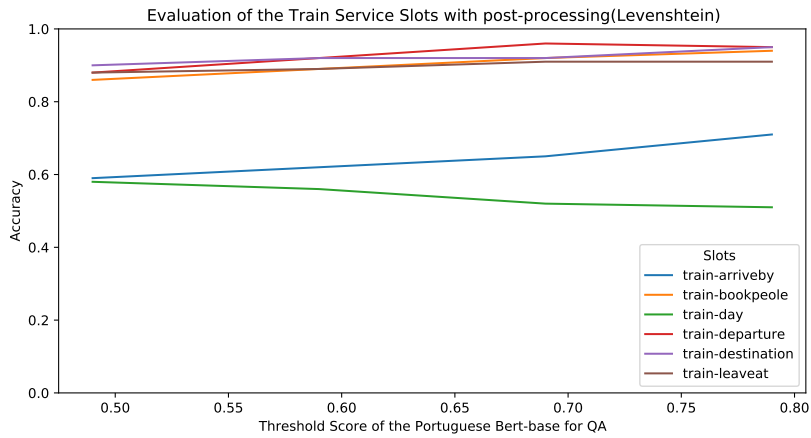


(c) Post-Processing: STS

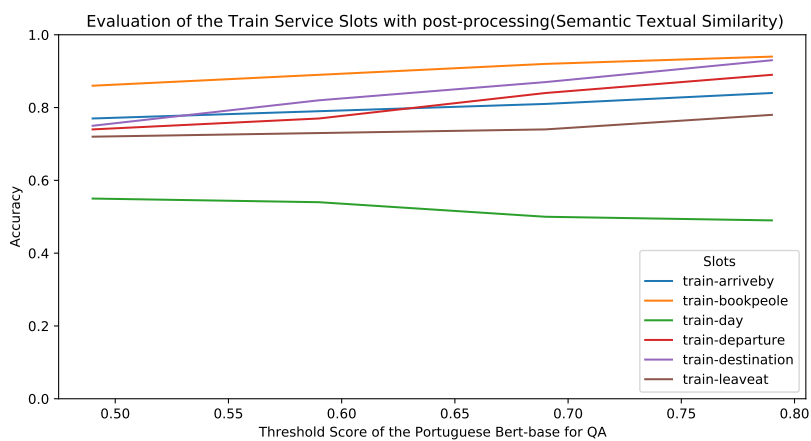
Figure 4: Evaluating the BERT-base Model with Intent Detection across different slots of the Taxi Service using various thresholds.



(a) Post-Processing: None

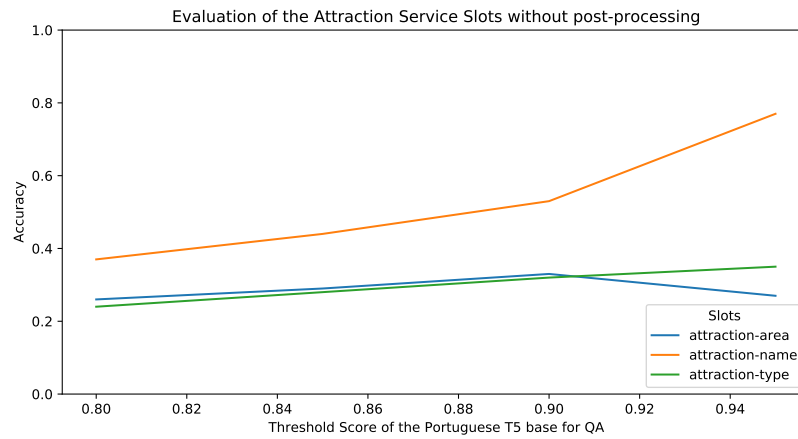


(b) Post-Processing: Levenshtein Distance

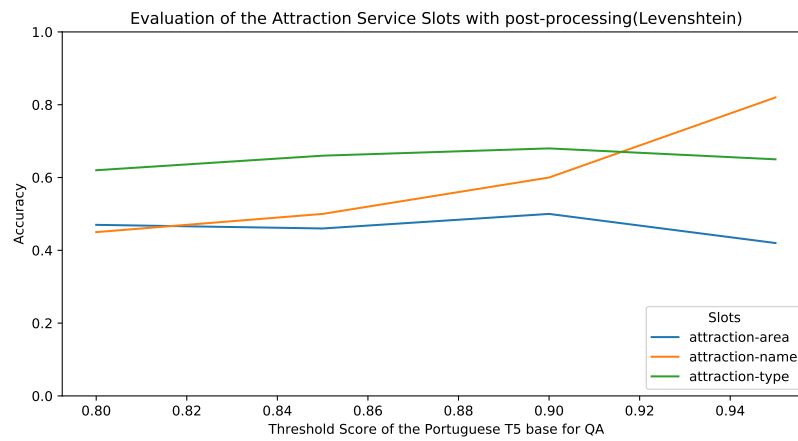


(c) Post-Processing: STS

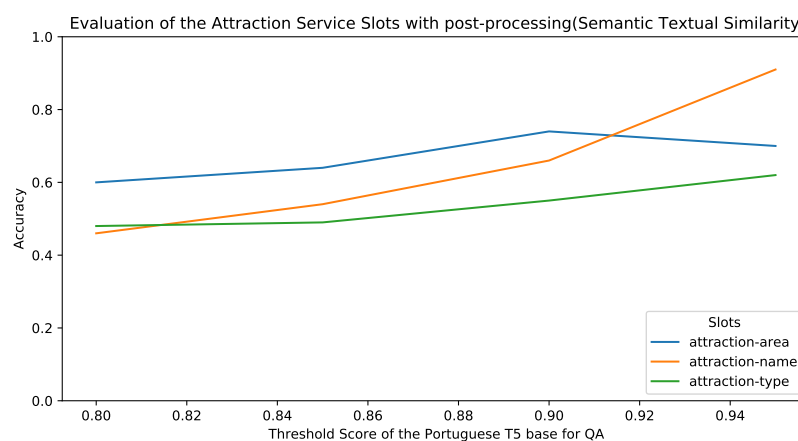
Figure 5: Evaluating the BERT-base Model with Intent Detection across different slots of the Train Service using various thresholds.



(a) Post-Processing: None

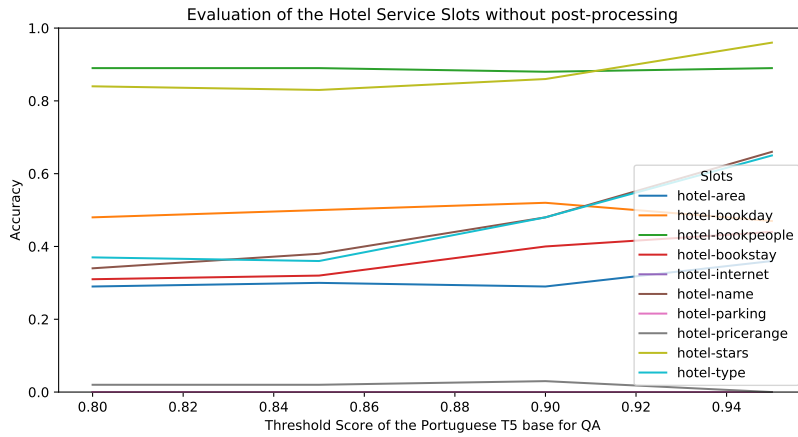


(b) Post-Processing: Levenshtein Distance

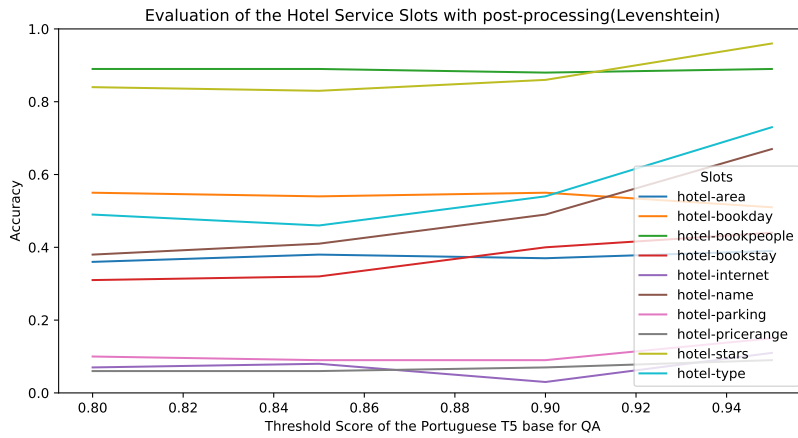


(c) Post-Processing: STS

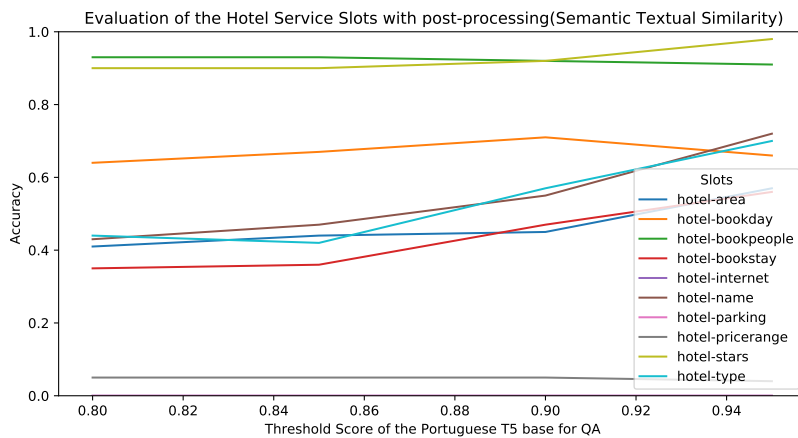
Figure 6: Evaluating the BERT-large Model with Intent Detection across different slots of the Attraction Service using various thresholds.



(a) Post-Processing: None

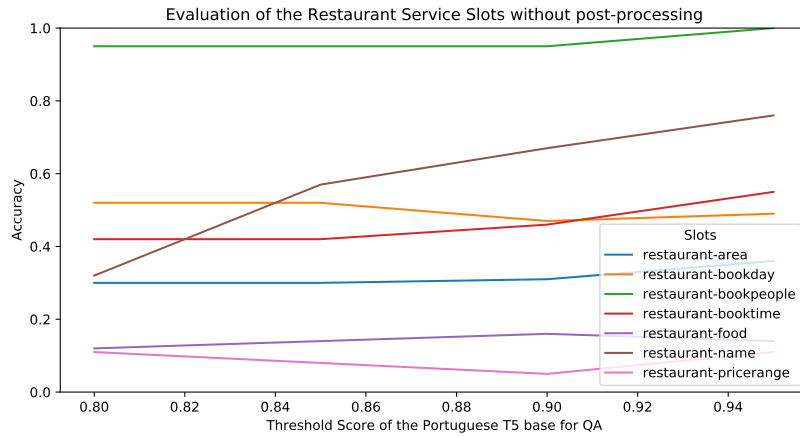


(b) Post-Processing: Levenshtein Distance

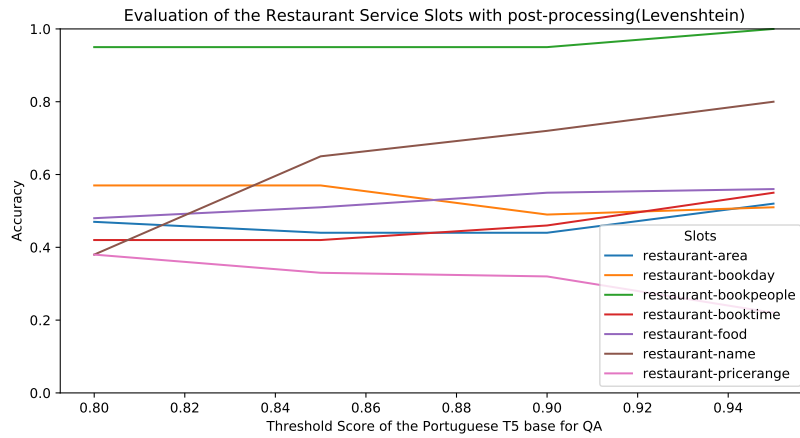


(c) Post-Processing: STS

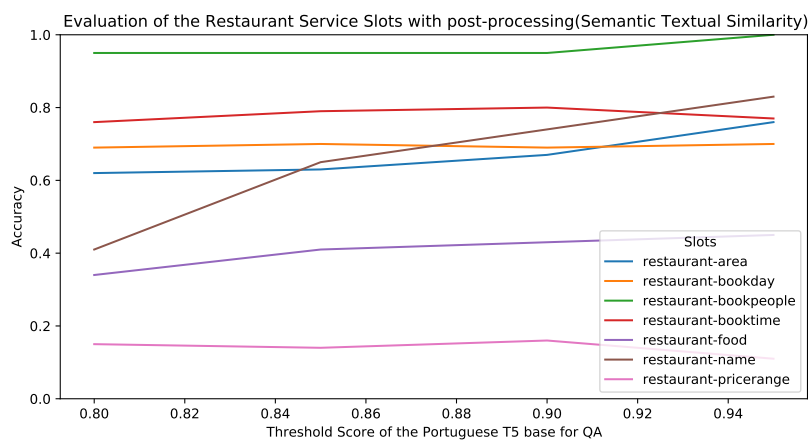
Figure 7: Evaluating the BERT-large Model with Intent Detection across different slots of the Hotel Service using various thresholds.



(a) Post-Processing: None

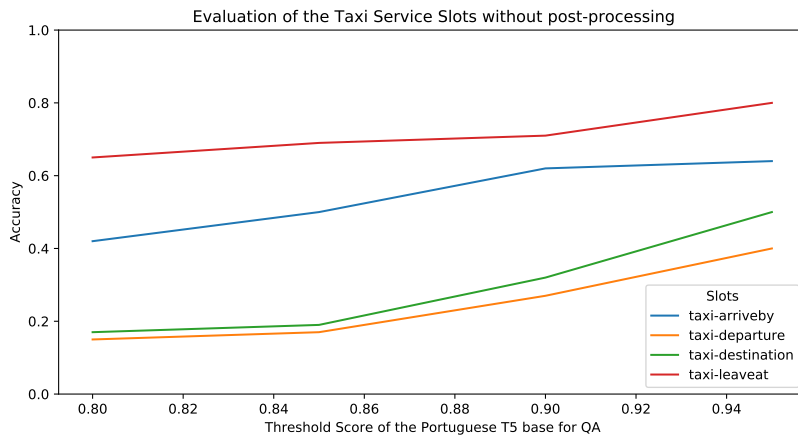


(b) Post-Processing: Levenshtein Distance

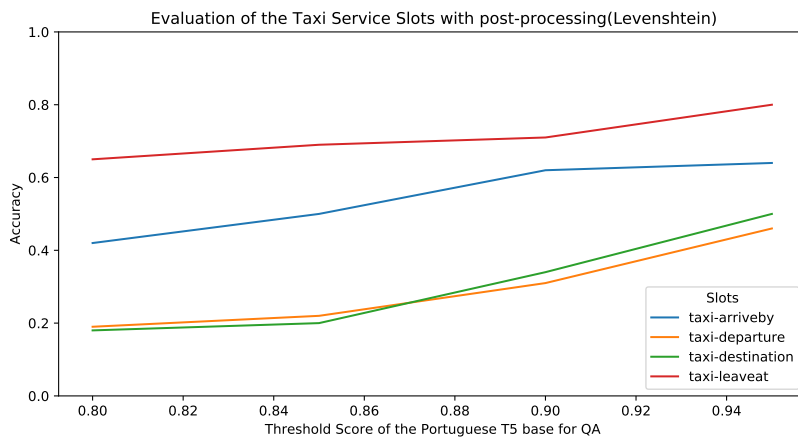


(c) Post-Processing: STS

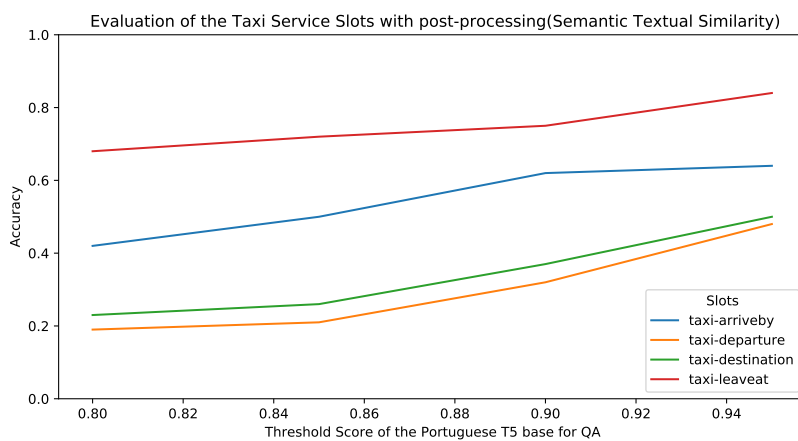
Figure 8: Evaluating the BERT-large Model with Intent Detection across different slots of the Restaurant Service using various thresholds.



(a) Post-Processing: None



(b) Post-Processing: Levenshtein Distance

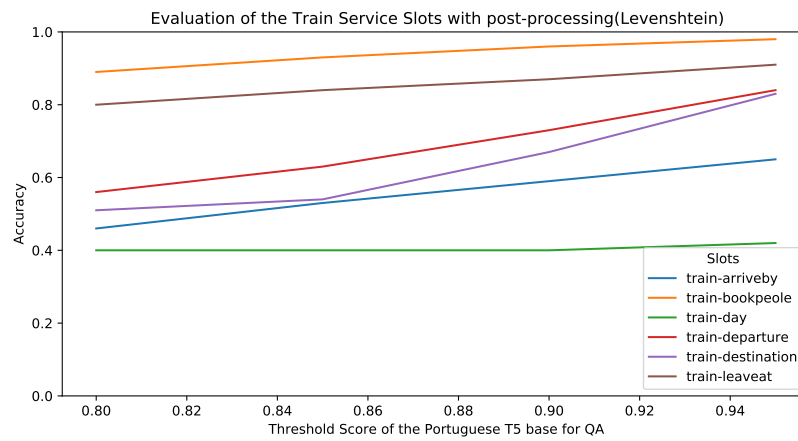


(c) Post-Processing: STS

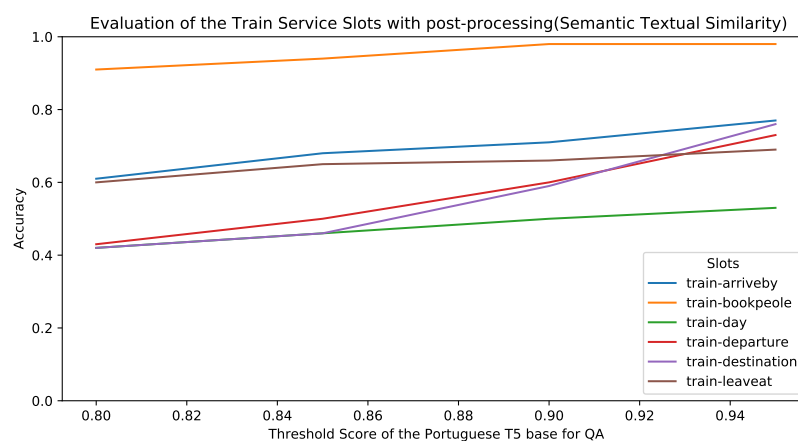
Figure 9: Evaluating the BERT-large Model with Intent Detection across different slots of the Taxi Service using various thresholds.



(a) Post-Processing: None

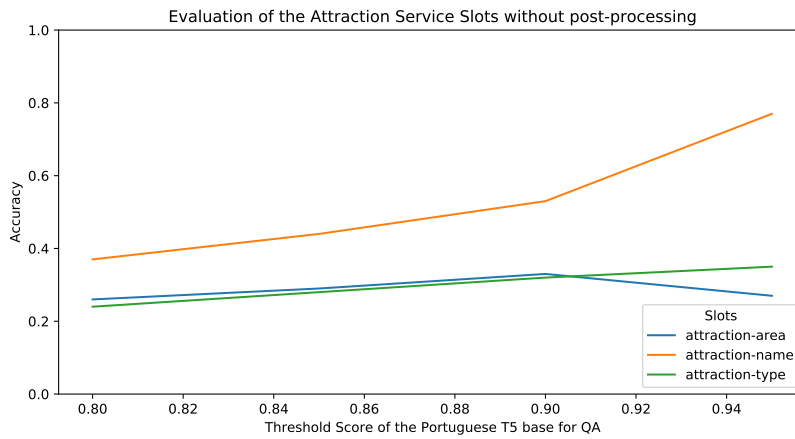


(b) Post-Processing: Levenshtein Distance

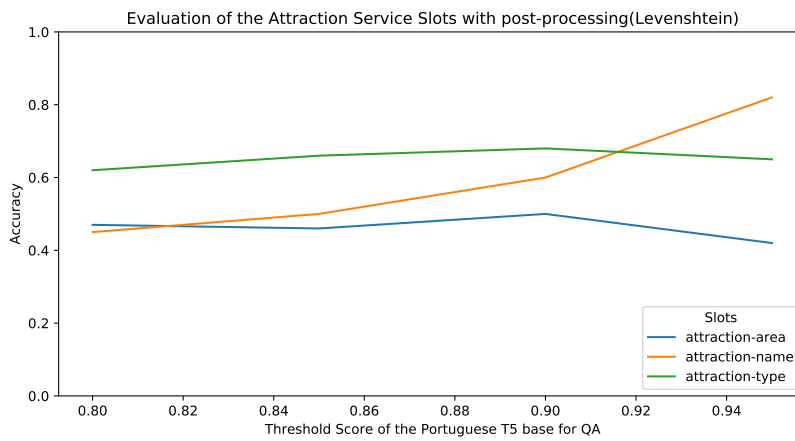


(c) Post-Processing: STS

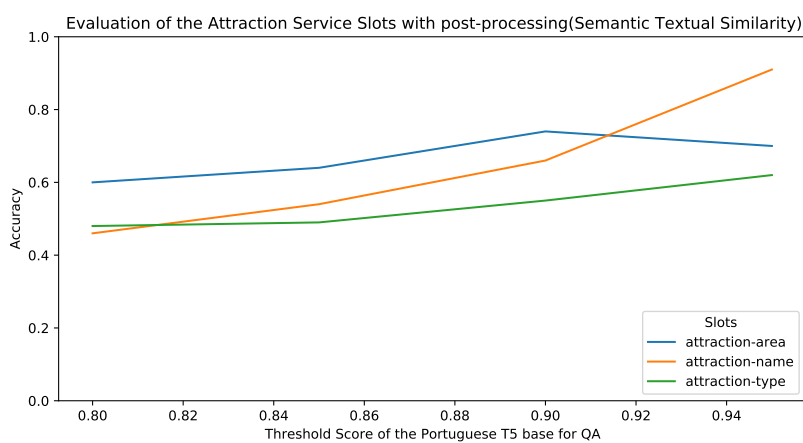
Figure 10: Evaluating the BERT-large Model with Intent Detection across different slots of the Train Service using various thresholds.



(a) Post-Processing: None

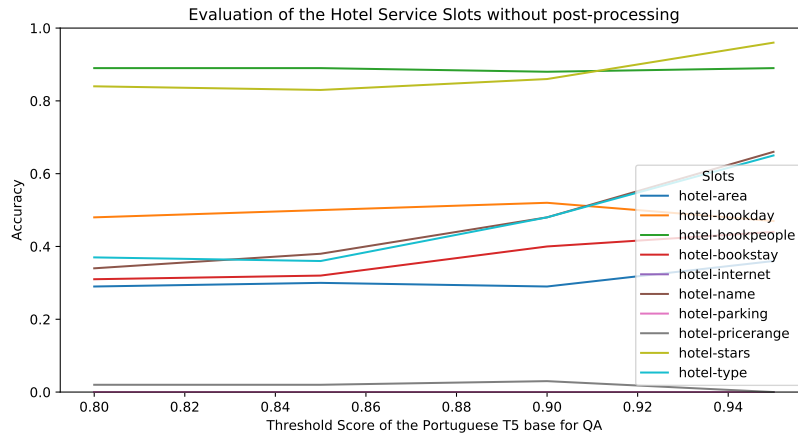


(b) Post-Processing: Levenshtein Distance

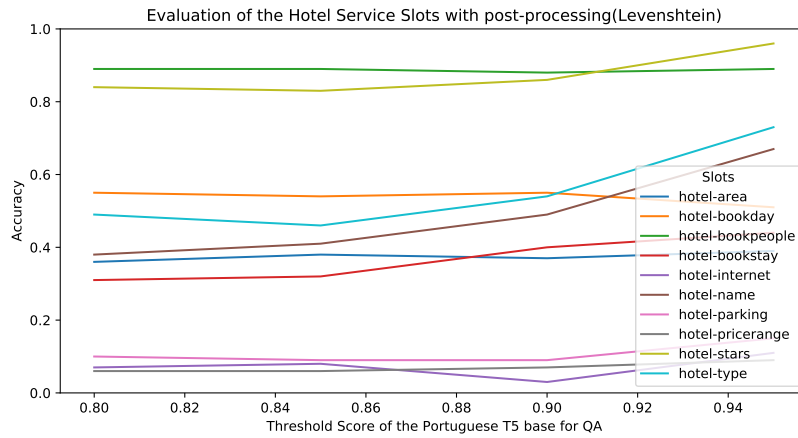


(c) Post-Processing: STS

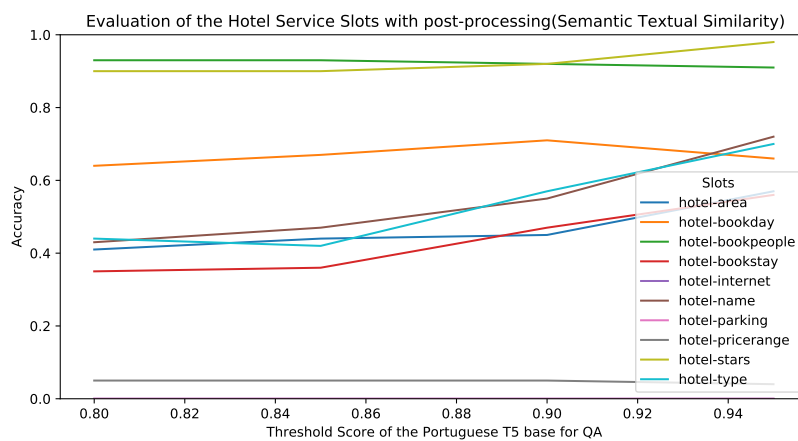
Figure 11: Evaluating the T5 Model with Intent Detection across different slots of the Attraction Service using various thresholds.



(a) Post-Processing: None

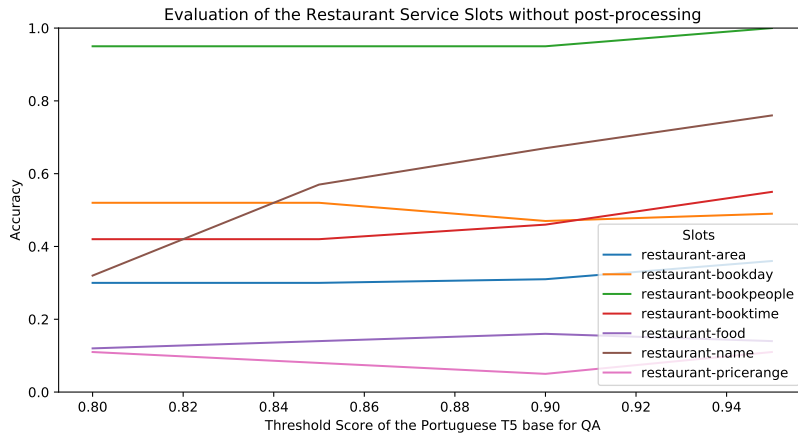


(b) Post-Processing: Levenshtein Distance

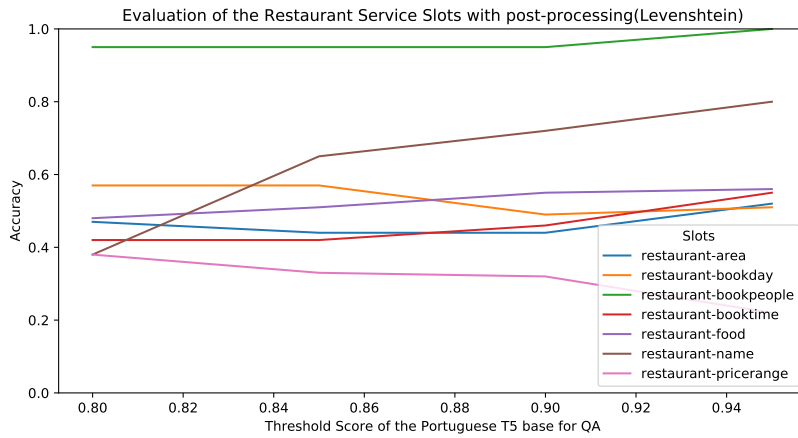


(c) Post-Processing: STS

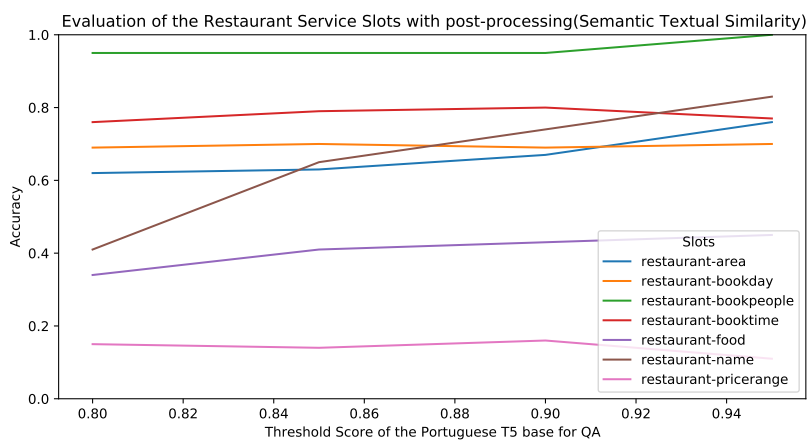
Figure 12: Evaluating the T5 Model with Intent Detection across different slots of the Hotel Service using various thresholds.



(a) Post-Processing: None

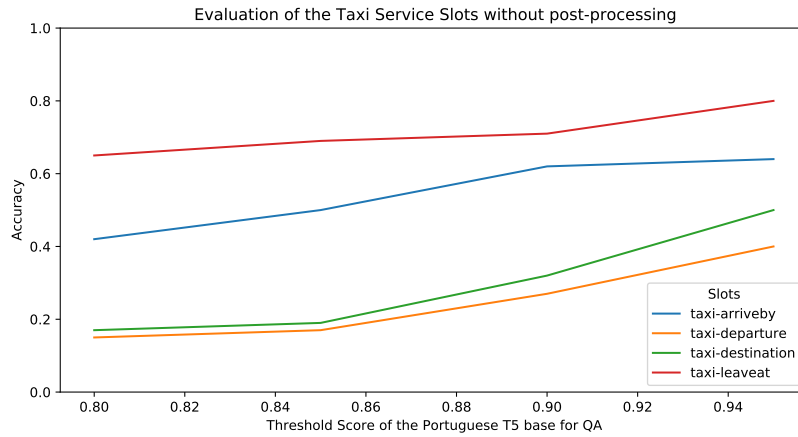


(b) Post-Processing: Levenshtein Distance

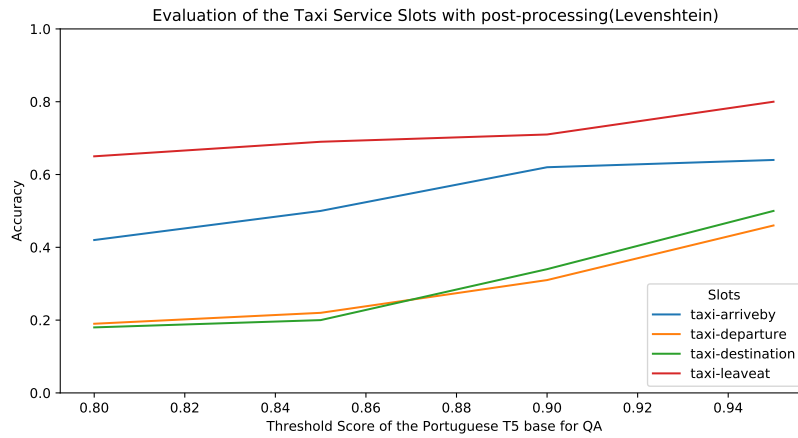


(c) Post-Processing: STS

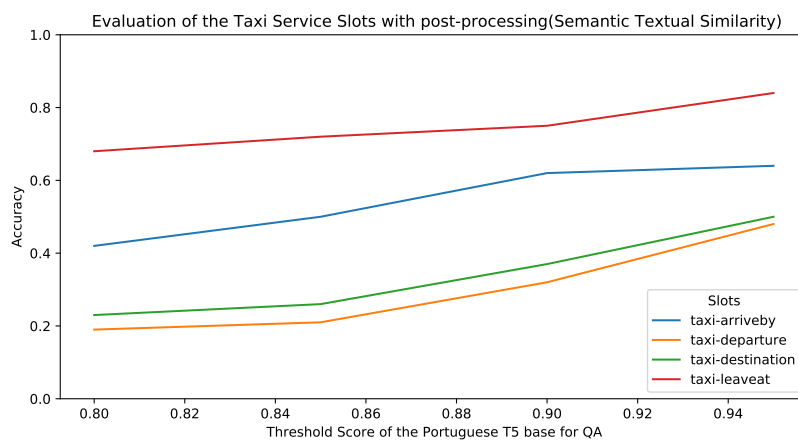
Figure 13: Evaluating the T5 Model with Intent Detection across different slots of the Restaurant Service using various thresholds.



(a) Post-Processing: None

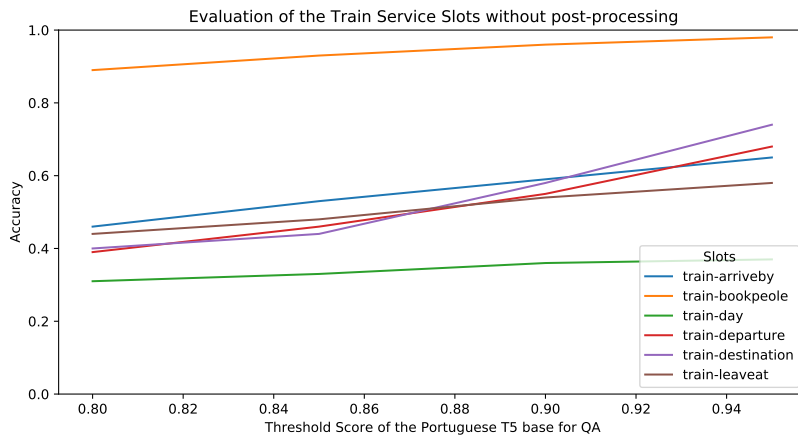


(b) Post-Processing: Levenshtein Distance

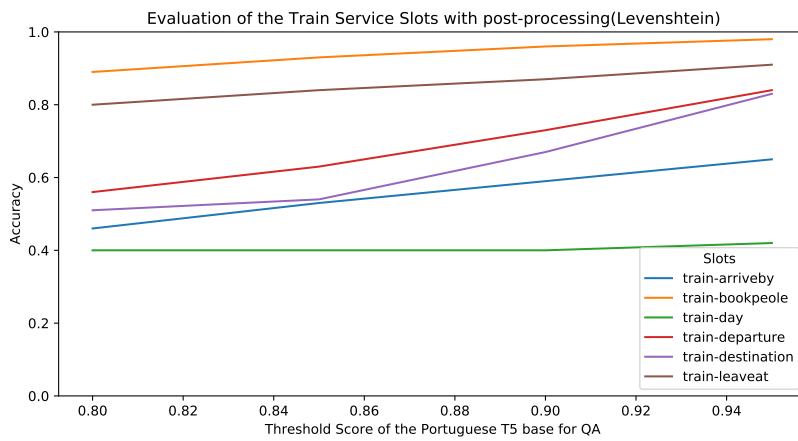


(c) Post-Processing: STS

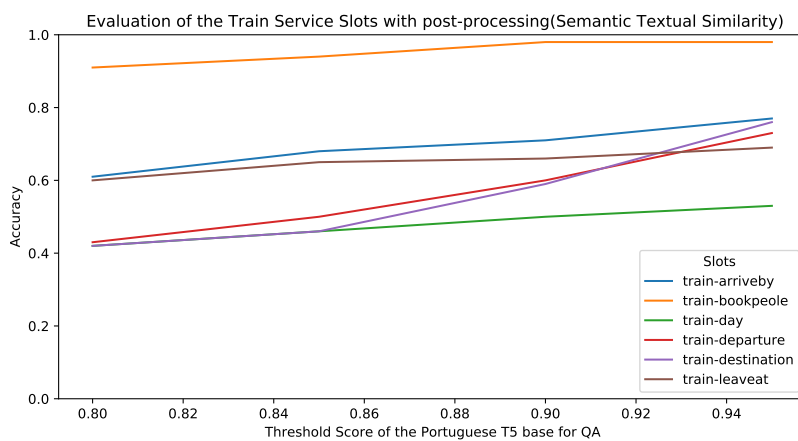
Figure 14: Evaluating the T5 Model with Intent Detection across different slots of the Taxi Service using various thresholds.



(a) Post-Processing: None

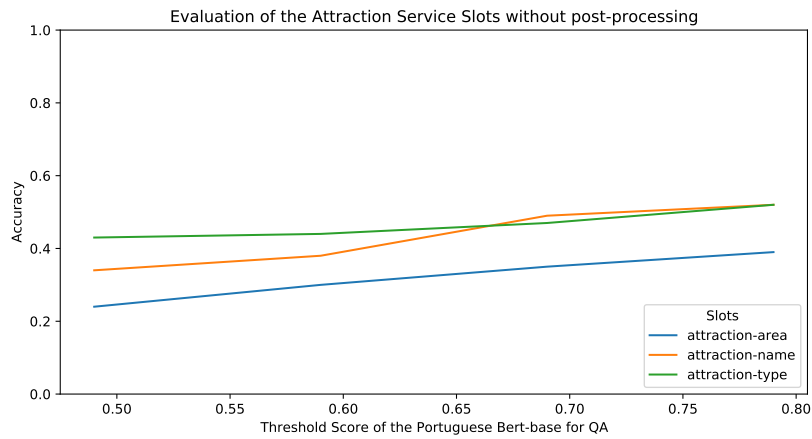


(b) Post-Processing: Levenshtein Distance

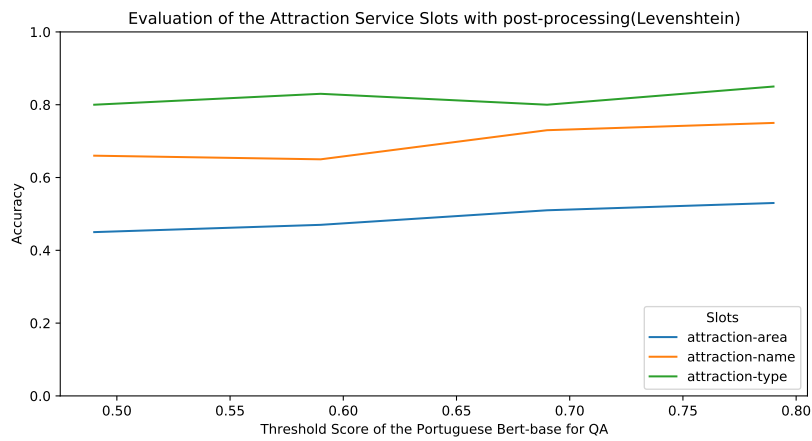


(c) Post-Processing: STS

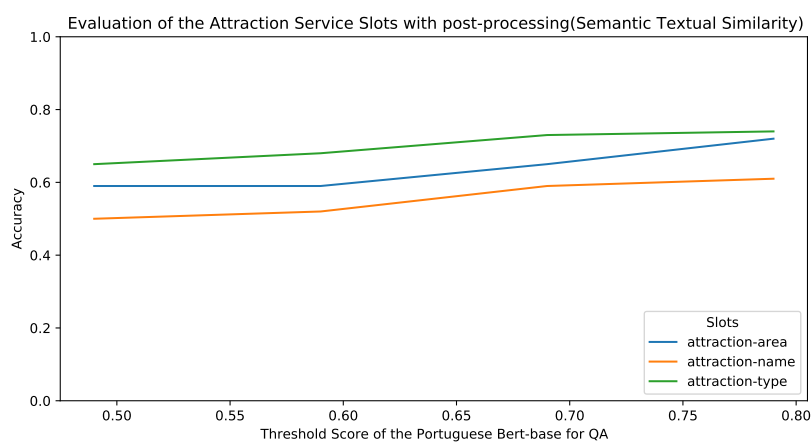
Figure 15: Evaluating the T5 Model with Intent Detection across different slots of the Train Service using various thresholds.



(a) Post-Processing: None

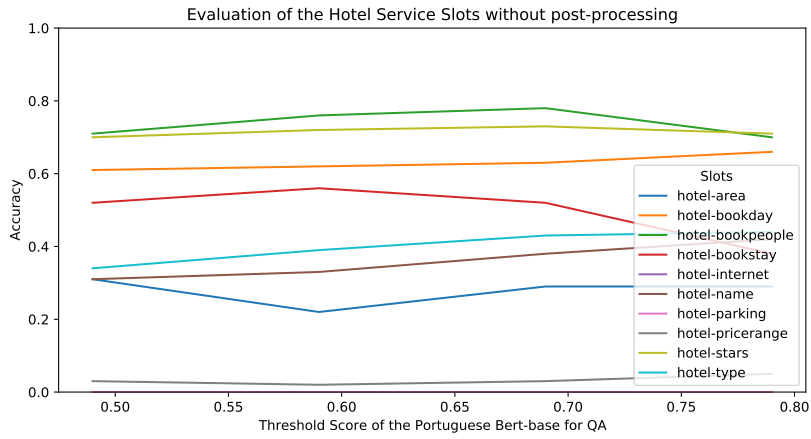


(b) Post-Processing: Levenshtein Distance

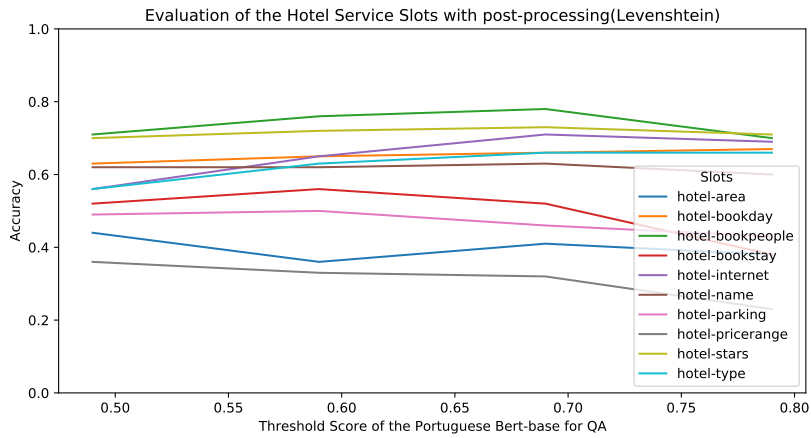


(c) Post-Processing: STS

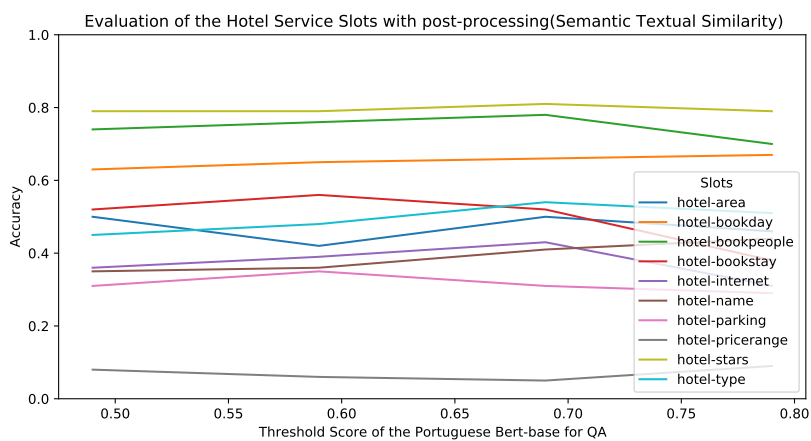
Figure 16: Evaluating the BERT-base Model without Intent Detection across different slots of the Attraction Service using various thresholds.



(a) Post-Processing: Levenshtein Distance

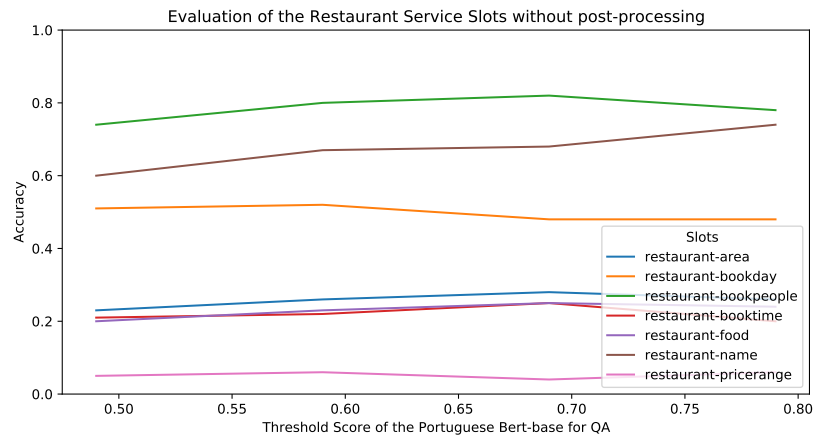


(b) Post-Processing: None

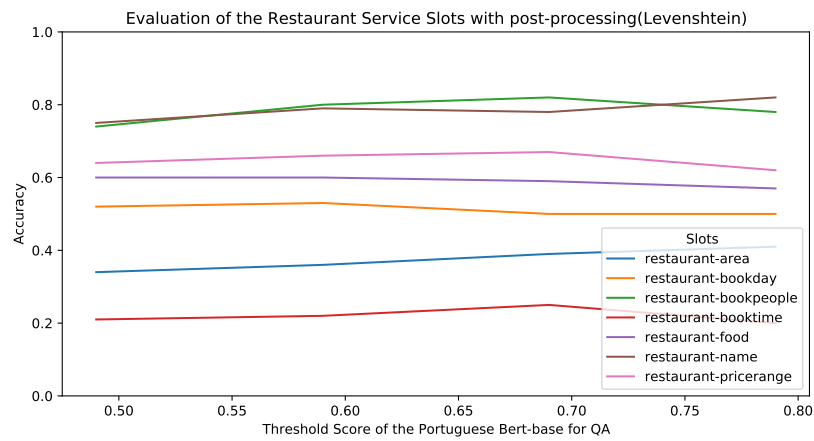


(c) Post-Processing: STS

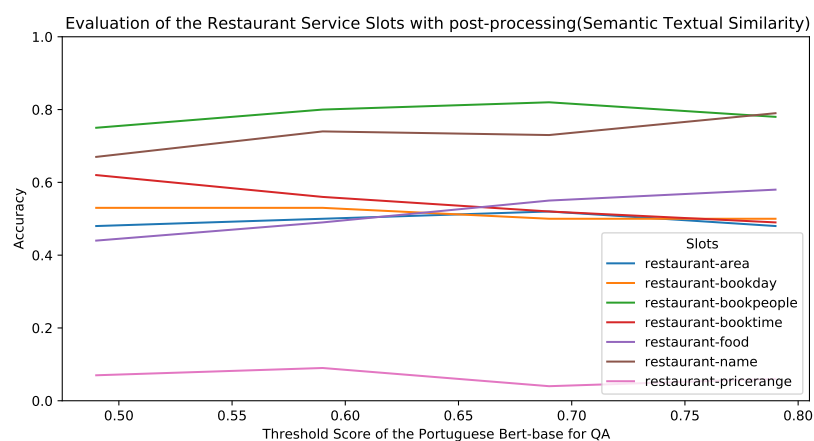
Figure 17: Evaluating the BERT-base Model without Intent Detection across different slots of the Hotel Service using various thresholds.



(a) Post-Processing: None

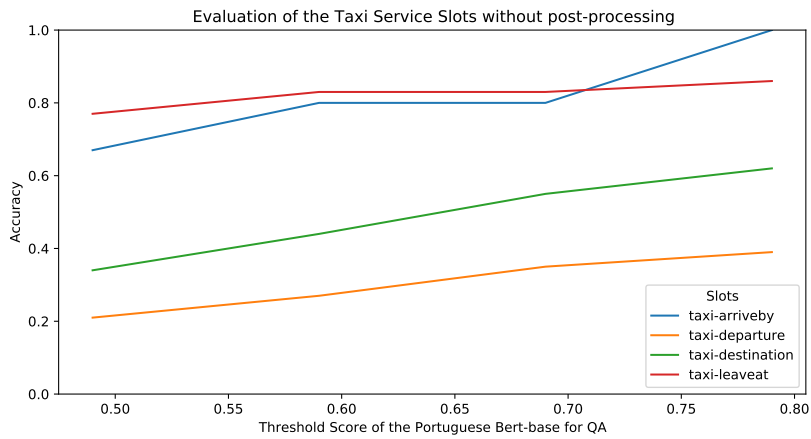


(b) Post-Processing: Levenshtein Distance

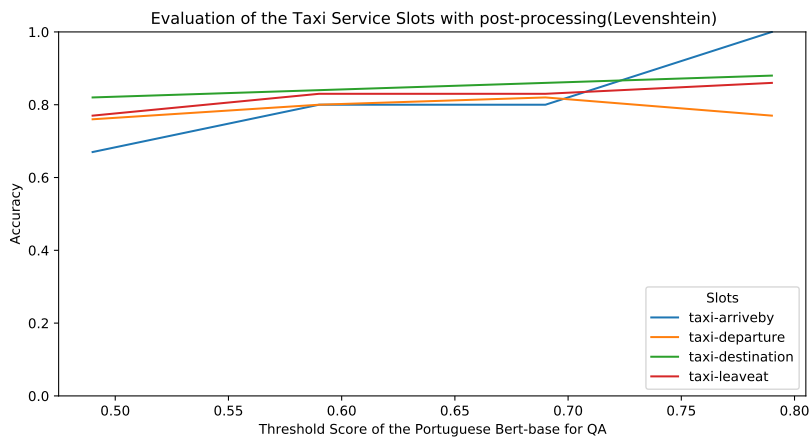


(c) Post-Processing: STS

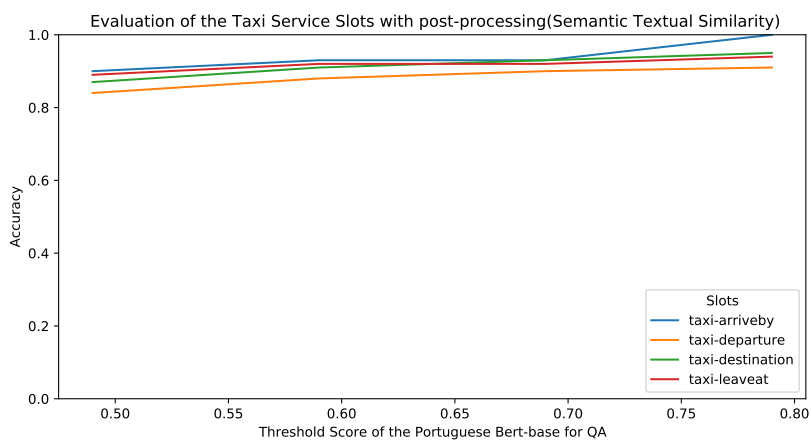
Figure 18: Evaluating the BERT-base Model without Intent Detection across different slots of the Restaurant Service using various thresholds.



(a) Post-Processing: None

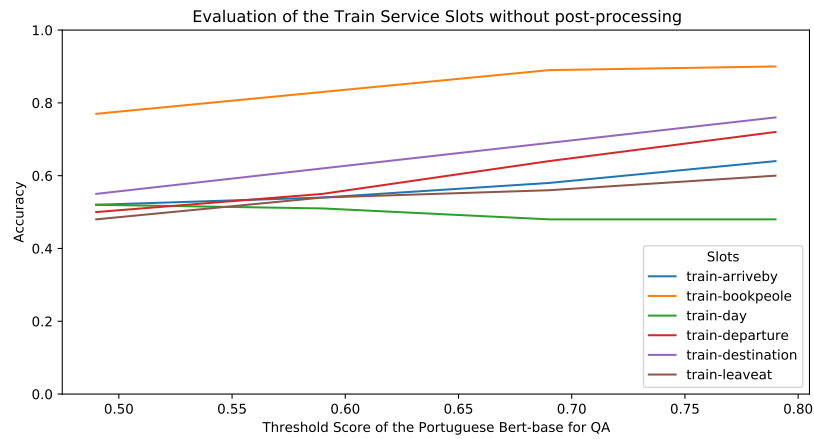


(b) Post-Processing: Levenshtein Distance

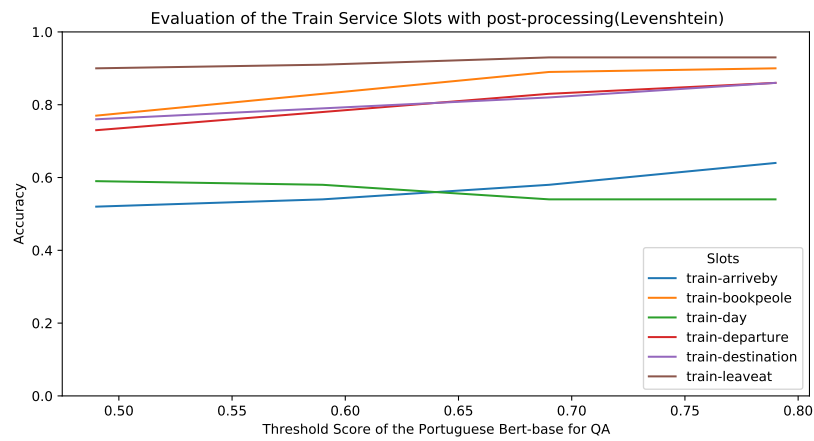


(c) Post-Processing: STS

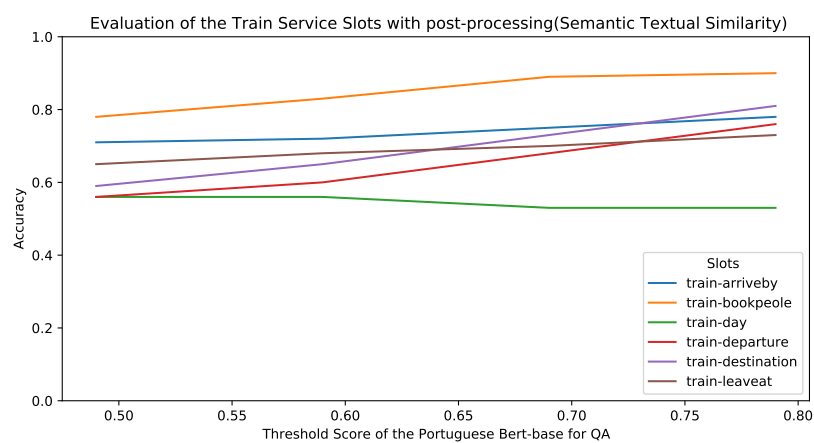
Figure 19: Evaluating the BERT-base Model without Intent Detection across different slots of the Taxi Service using various thresholds.



(a) Post-Processing: None

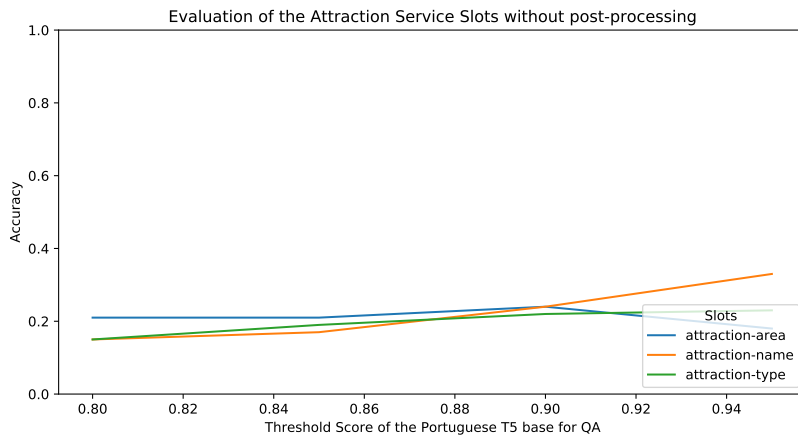


(b) Post-Processing: Levenshtein Distance

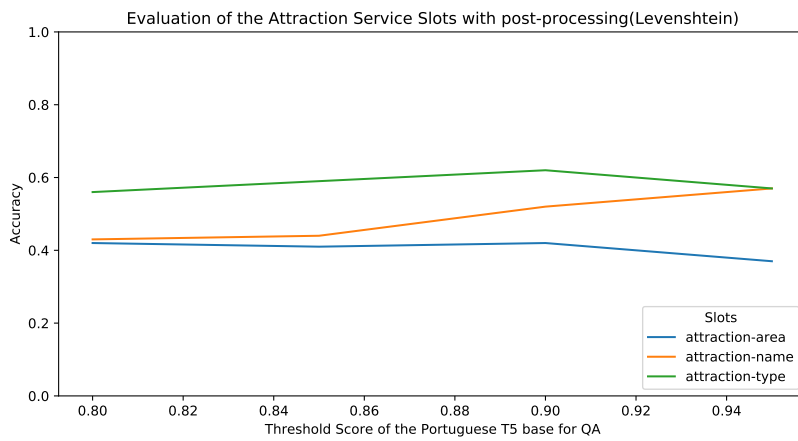


(c) Post-Processing: STS

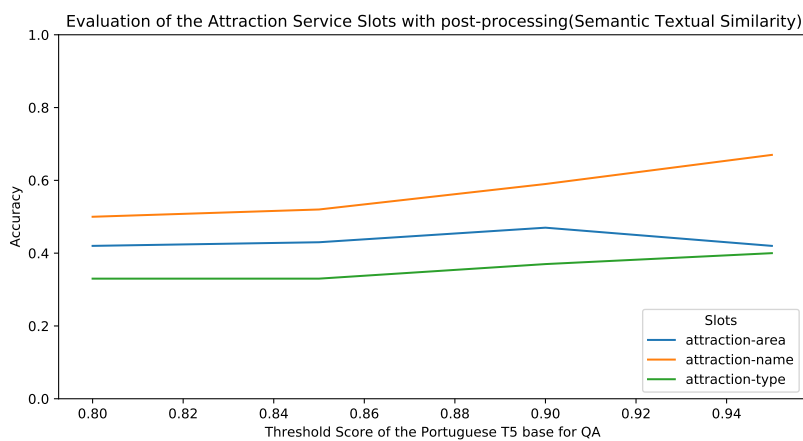
Figure 20: Evaluating the BERT-base Model without Intent Detection across different slots of the Train Service using various thresholds.



(a) Post-Processing: None

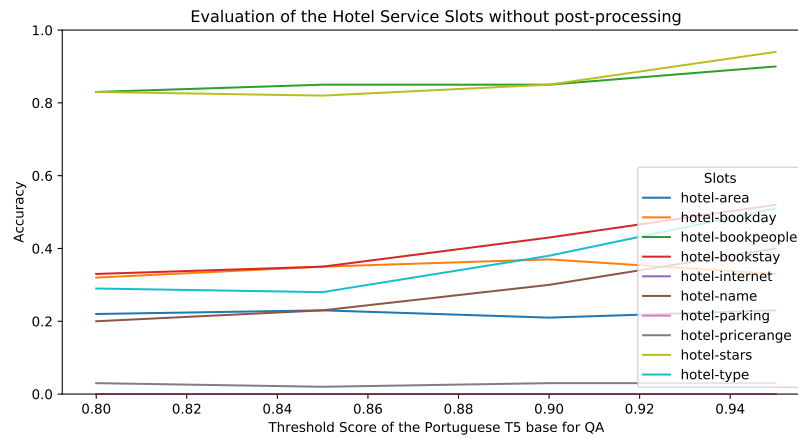


(b) Post-Processing: Levenshtein Distance

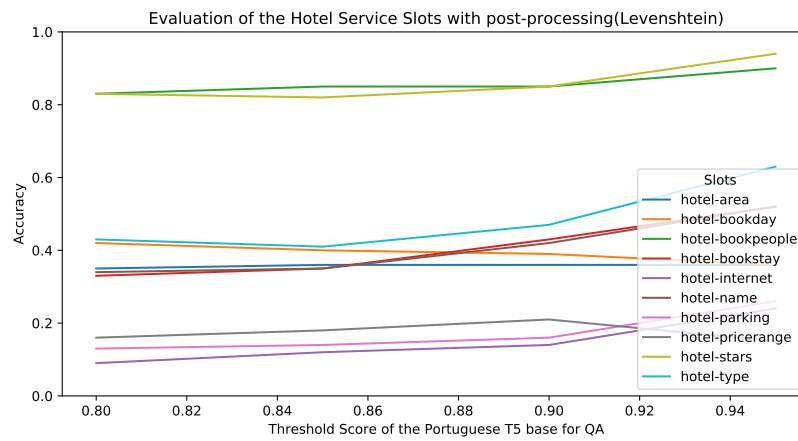


(c) Post-Processing: STS

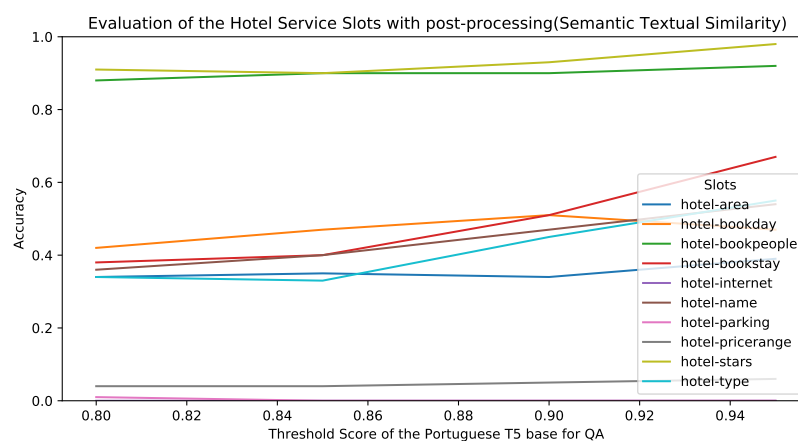
Figure 21: Evaluating the BERT-large Model without Intent Detection across different slots of the Attraction Service using various thresholds.



(a) Post-Processing: None

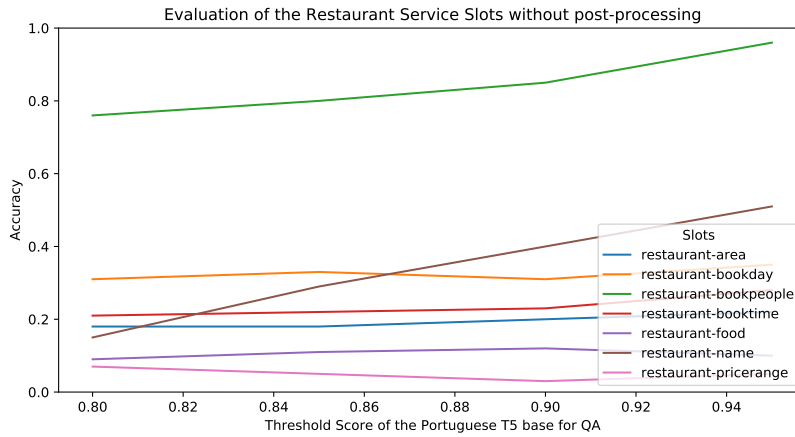


(b) Post-Processing: Levenshtein Distance

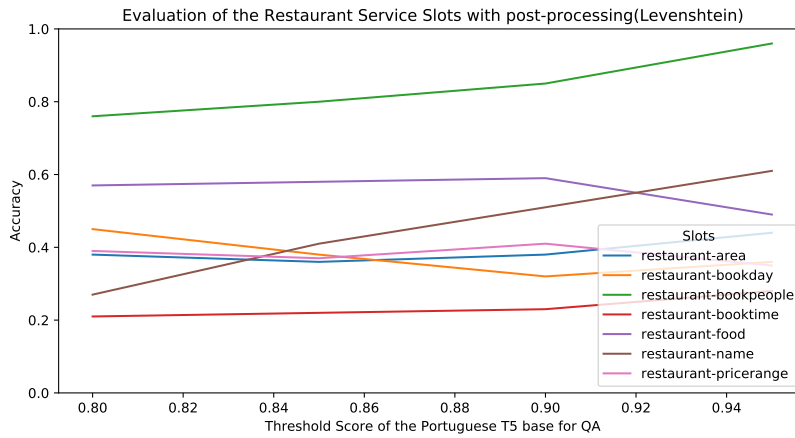


(c) Post-Processing: STS

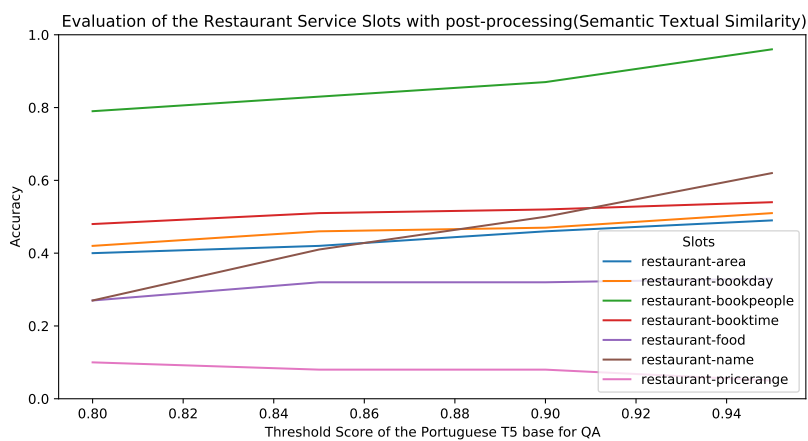
Figure 22: Evaluating the BERT-large Model without Intent Detection across different slots of the Hotel Service using various thresholds.



(a) Post-Processing: None

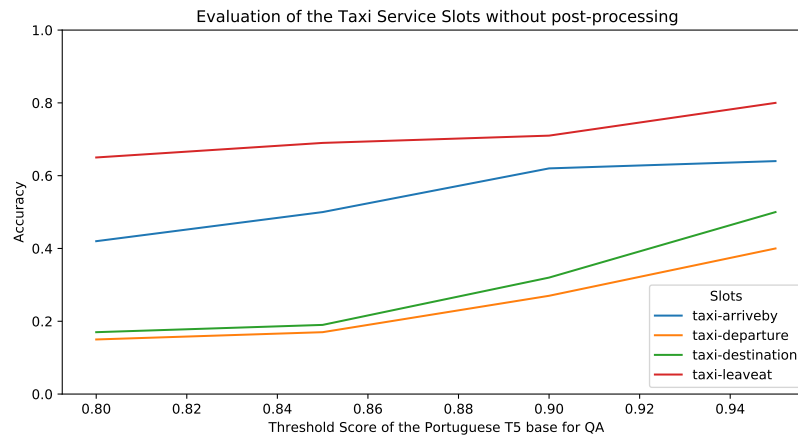


(b) Post-Processing: Levenshtein Distance

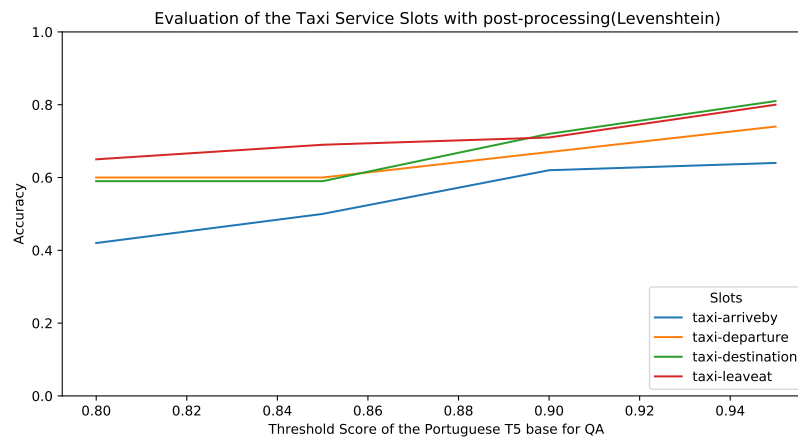


(c) Post-Processing: STS

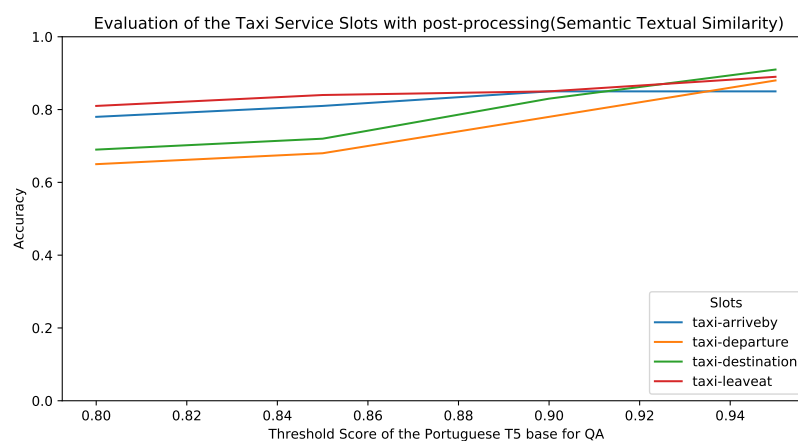
Figure 23: Evaluating the BERT-large Model without Intent Detection across different slots of the Restaurant Service using various thresholds.



(a) Post-Processing: None

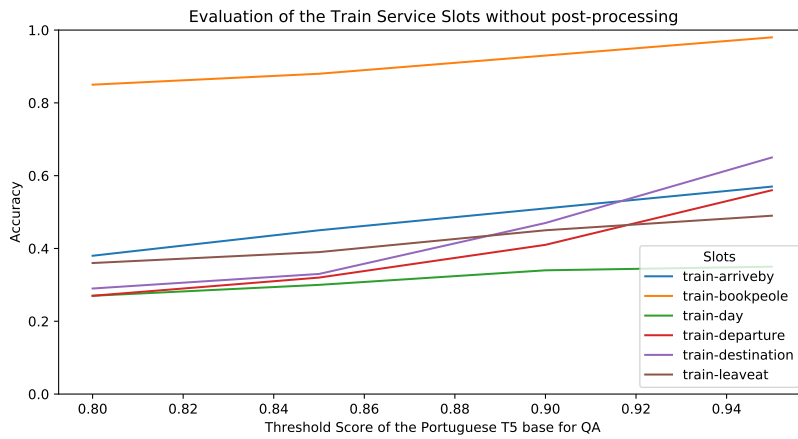


(b) Post-Processing: Levenshtein Distance

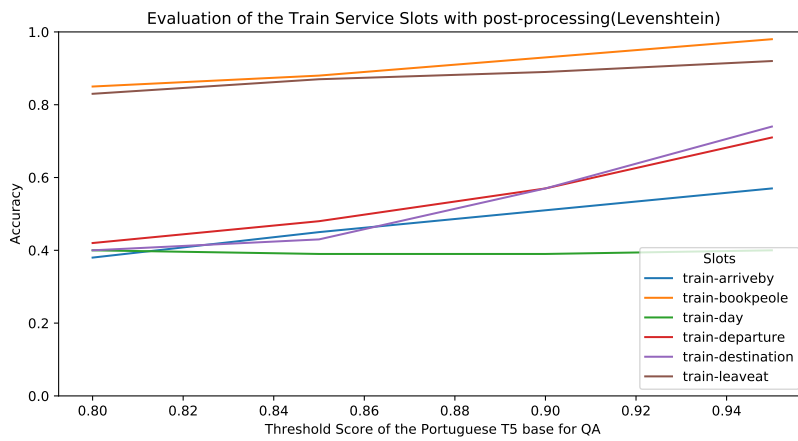


(c) Post-Processing: STS

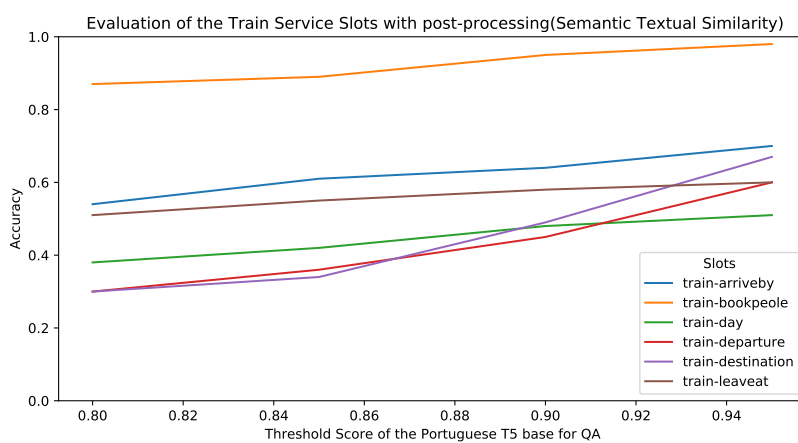
Figure 24: Evaluating the BERT-large Model without Intent Detection across different slots of the Taxi Service using various thresholds.



(a) Post-Processing: None

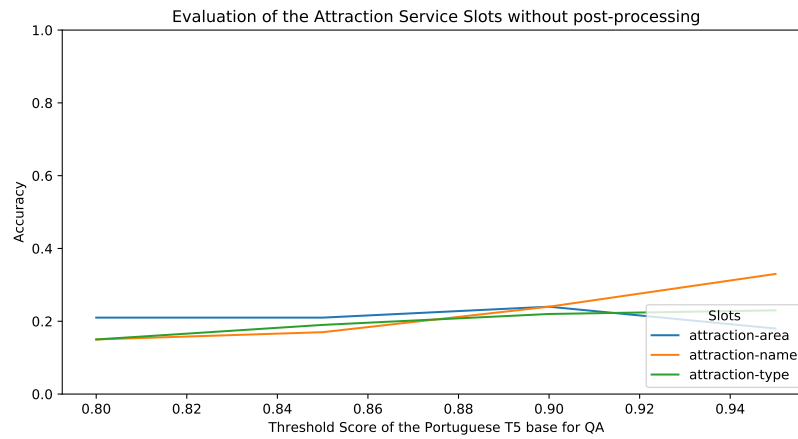


(b) Post-Processing: Levenshtein Distance

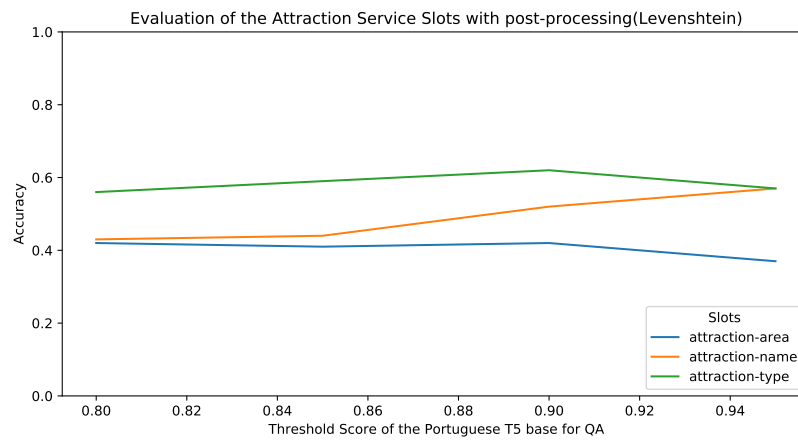


(c) Post-Processing: STS

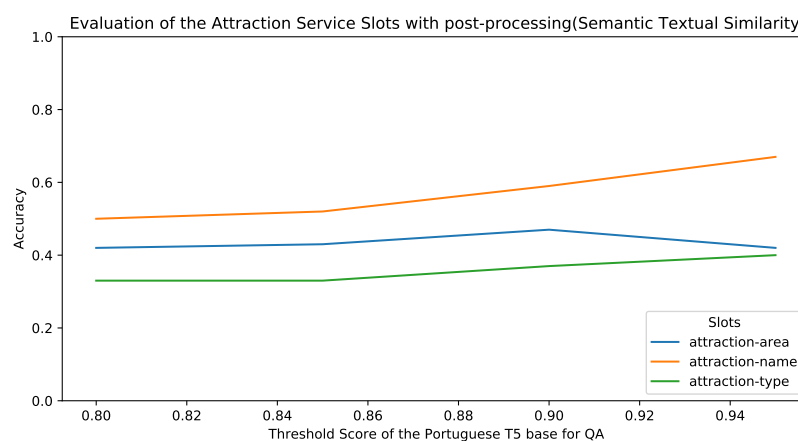
Figure 25: Evaluating the BERT-large Model without Intent Detection across different slots of the Train Service using various thresholds.



(a) Post-Processing: None

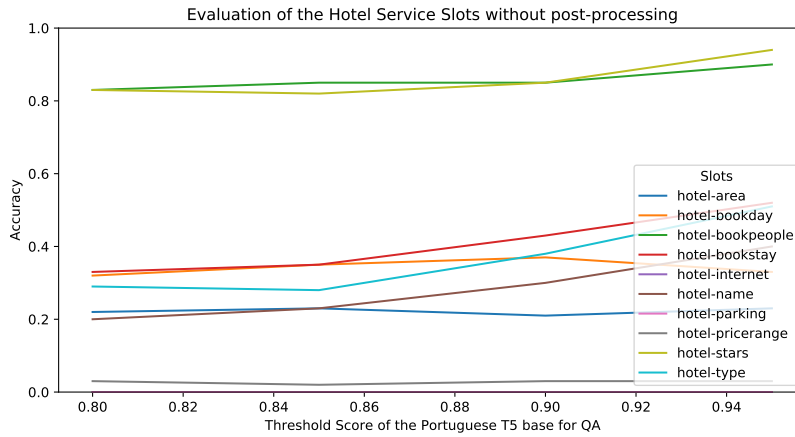


(b) Post-Processing: Levenshtein Distance

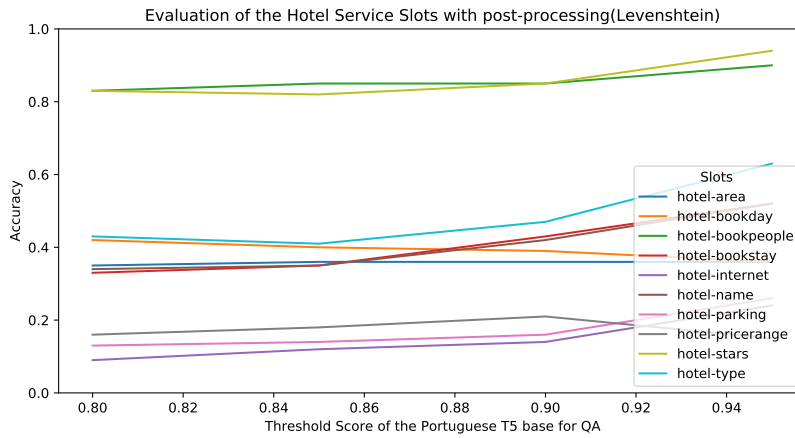


(c) Post-Processing: STS

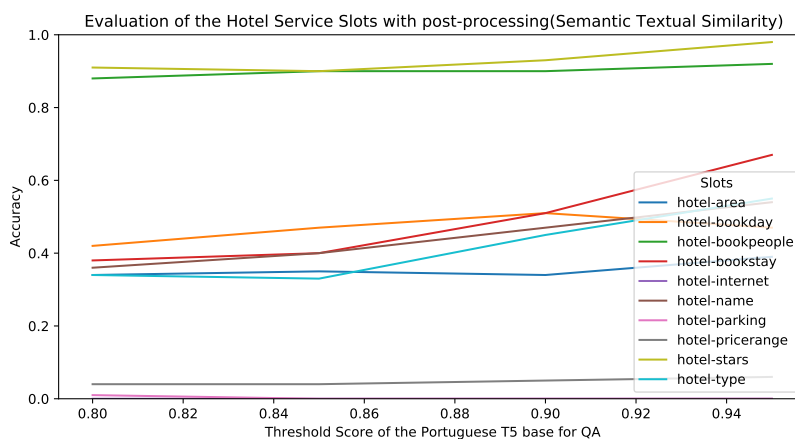
Figure 26: Evaluating the T5 Model without Intent Detection across different slots of the Attraction Service using various thresholds.



(a) Post-Processing: None

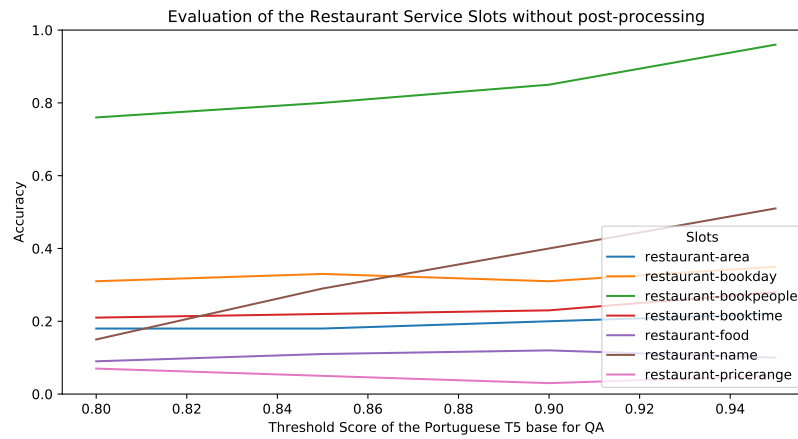


(b) Post-Processing: Levenshtein Distance

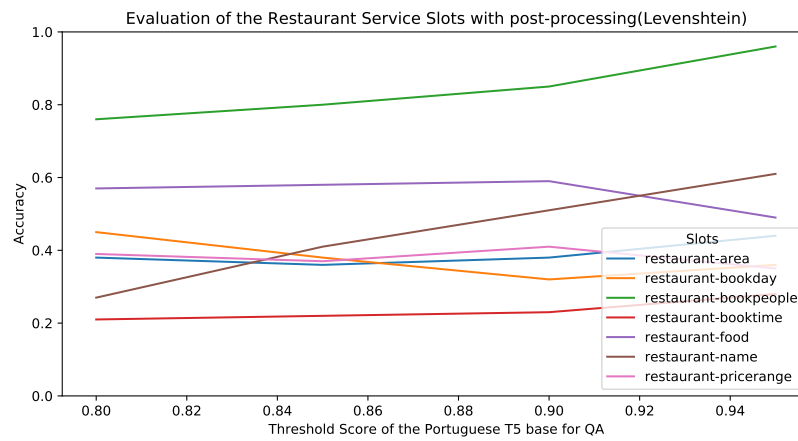


(c) Post-Processing: STS

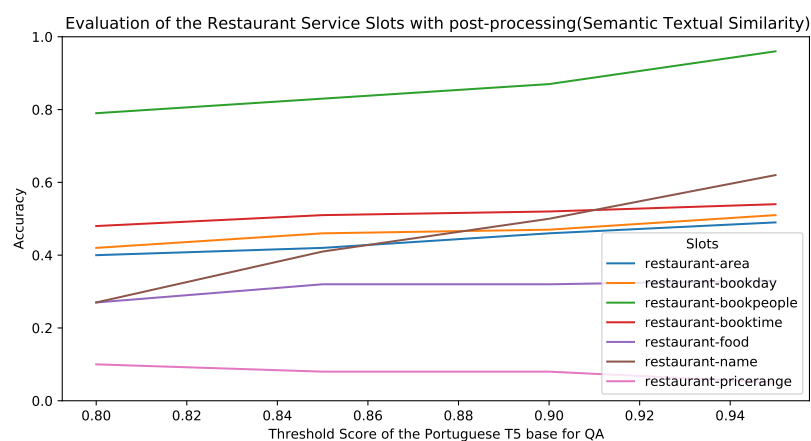
Figure 27: Evaluating the T5 Model without Intent Detection across different slots of the Hotel Service using various thresholds.



(a) Post-Processing: None

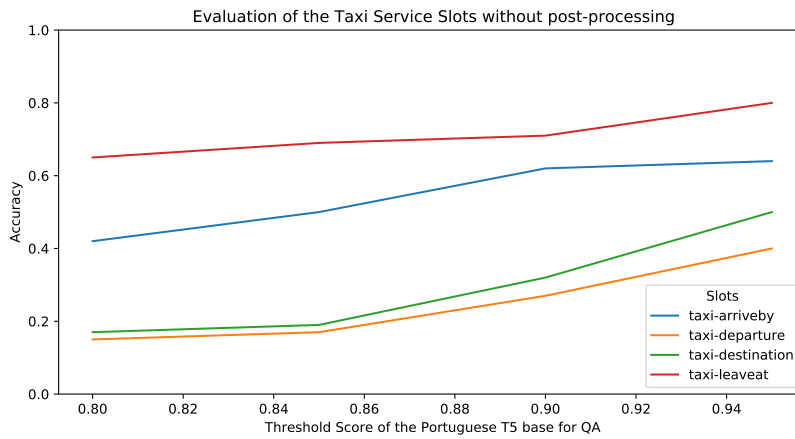


(b) Post-Processing: Levenshtein Distance

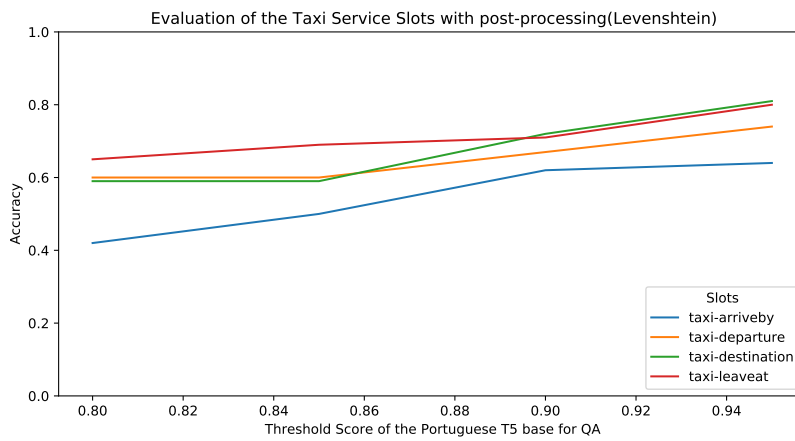


(c) Post-Processing: STS

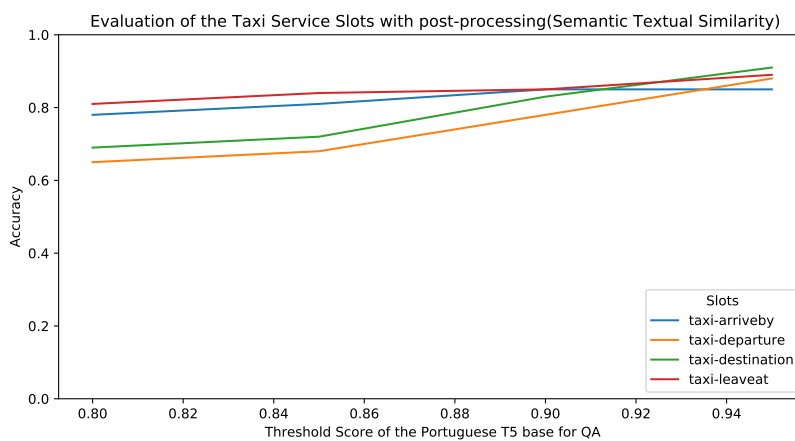
Figure 28: Evaluating the T5 Model without Intent Detection across different slots of the Restaurant Service using various thresholds.



(a) Post-Processing: None

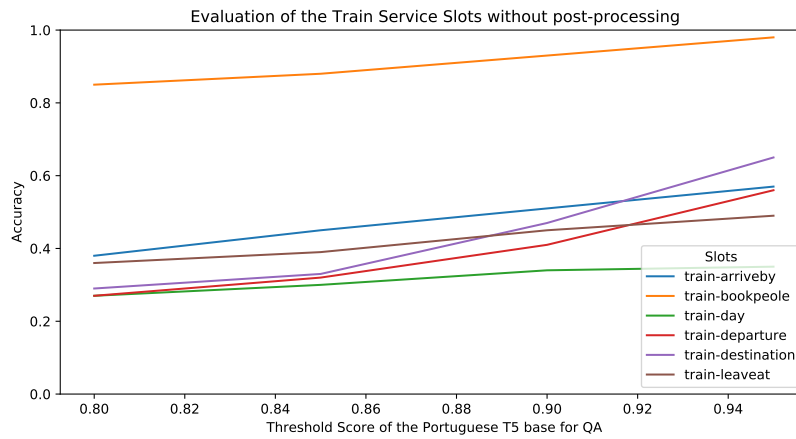


(b) Post-Processing: Levenshtein Distance

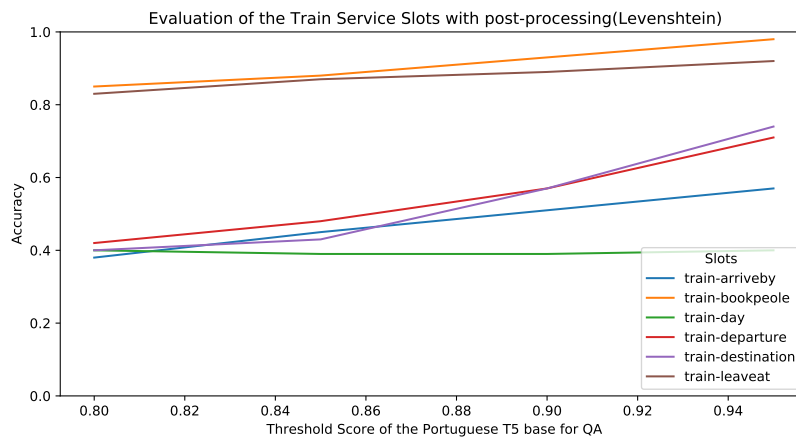


(c) Post-Processing: STS

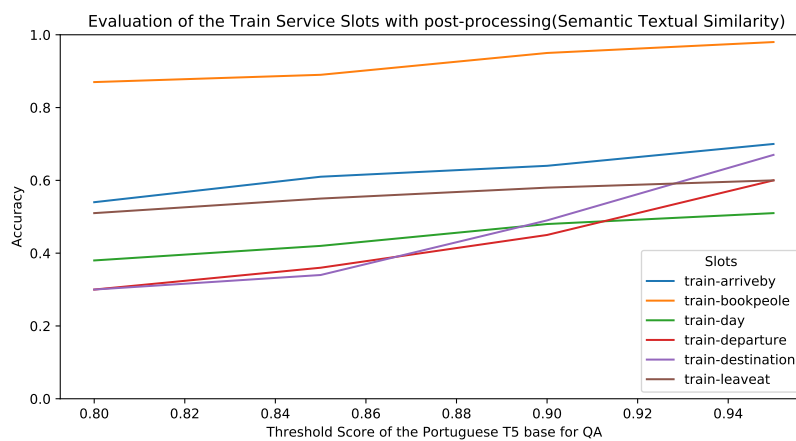
Figure 29: Evaluating the T5 Model without Intent Detection across different slots of the Taxi Service using various thresholds.



(a) Post-Processing: None



(b) Post-Processing: Levenshtein Distance



(c) Post-Processing: STS

Figure 30: Evaluating the T5 Model without Intent Detection across different slots of the Train Service using various thresholds.

Appendix C

In this appendix there are two tables which consist of the detailed evaluation of the different slots for the various variations of the models with and without the use of Intent Detection.

Slots	BERT Base			BERT Large			T5		
	None	Lev	STS	None	Lev	STS	None	Lev	STS
attraction-area	0.14	0.45	0.72	0.18	0.52	0.69	0.18	0.36	0.62
attraction-name	0.56	0.64	0.64	0.70	0.77	0.91	0.69	0.74	0.76
attraction-type	0.49	0.77	0.71	0.41	0.71	0.65	0.32	0.53	0.61
hotel-area	0.45	0.71	0.74	0.17	0.67	0.60	0.42	0.65	0.58
hotel-bookday	0.85	0.87	0.87	0.87	0.89	0.89	0.49	0.51	0.66
hotel-bookpeople	0.68	0.68	0.73	0.83	0.83	0.90	0.81	0.81	0.90
hotel-bookstay	0.40	0.40	0.43	0.40	0.41	0.42	0.71	0.71	0.86
hotel-internet	0.00	0.56	0.20	0.00	0.44	0.43	0.00	0.18	0.00
hotel-name	0.77	0.75	0.82	0.84	0.85	0.89	0.65	0.66	0.71
hotel-parking	0.00	0.15	0.00	0.00	0.50	0.67	0.00	0.43	0.00
hotel-pricerange	0.13	0.37	0.13	0.56	0.56	0.56	0.02	0.08	0.09
hotel-stars	0.66	0.66	0.71	0.82	0.82	0.86	0.91	0.91	0.93
hotel-type	0.38	0.60	0.49	0.70	0.75	0.85	0.44	0.51	0.51
restaurant-area	0.47	0.67	0.78	0.15	0.31	0.73	0.24	0.45	0.72
restaurant-bookday	0.43	0.43	0.44	0.48	0.53	0.43	0.46	0.47	0.47
restaurant-bookpeople	0.71	0.71	0.75	0.84	0.84	0.92	0.93	0.93	0.93
restaurant-booktime	0.76	0.76	0.87	0.93	0.93	0.96	0.87	0.87	0.91
restaurant-food	0.33	0.24	0.62	0.25	0.18	0.57	0.22	0.53	0.53
restaurant-name	0.72	0.80	0.81	0.81	0.91	0.94	0.67	0.73	0.75
restaurant-pricerange	0.12	0.72	0.24	0.40	0.47	0.60	0.00	0.49	0.04
taxi-arriveBy	0.40	0.40	0.80	0.29	0.29	0.57	0.29	0.29	0.57
taxi-departure	0.50	0.63	0.57	0.63	0.69	0.69	0.41	0.48	0.52
taxi-destination	0.70	0.72	0.70	0.86	0.79	0.79	0.56	0.59	0.57
taxi-leaveAt	1	1	1	0.91	0.91	0.89	0.83	0.83	0.83
train-arriveBy	0.66	0.66	0.76	0.77	0.77	0.86	0.57	0.57	0.71
train-bookpeople	0.86	0.86	0.91	0.93	0.93	0.98	0.98	0.98	0.96
train-day	0.58	0.65	0.62	0.64	0.71	0.70	0.42	0.47	0.57
train-departure	0.83	0.91	0.87	0.92	0.98	0.98	0.65	0.78	0.69
train-destination	0.87	0.98	0.94	0.90	0.98	0.97	0.71	0.86	0.76
train-leaveAt	0.58	0.84	0.66	0.47	0.86	0.68	0.46	0.83	0.63

Table 5: Results from the individual slot accuracy evaluation of the models, using Intent Detection and various types of Post-processing.

Slots	BERT Base			BERT Large			T5		
	None	Lev	STS	None	Lev	STS	None	Lev	STS
attraction-area	0.12	0.40	0.63	0.12	0.44	0.56	0.11	0.29	0.45
attraction-name	0.41	0.58	0.47	0.56	0.69	0.72	0.34	0.46	0.54
attraction-type	0.40	0.76	0.63	0.38	0.65	0.61	0.17	0.49	0.35
hotel-area	0.27	0.60	0.54	0.06	0.59	0.36	0.31	0.53	0.41
hotel-bookday	0.68	0.69	0.70	0.72	0.73	0.73	0.45	0.46	0.58
hotel-bookpeople	0.59	0.59	0.66	0.82	0.82	0.87	0.78	0.78	0.87
hotel-bookstay	0.39	0.39	0.41	0.38	0.39	0.39	0.68	0.68	0.84
hotel-internet	0.00	0.75	0.31	0.00	0.39	0.25	0.00	0.12	0.00
hotel-name	0.51	0.64	0.56	0.67	0.68	0.70	0.46	0.55	0.59
hotel-parking	0.00	0.41	0.25	0.00	0.57	0.33	0.03	0.34	0.03
hotel-pricerange	0.10	0.42	0.10	0.43	0.43	0.43	0.03	0.11	0.06
hotel-stars	0.63	0.63	0.67	0.81	0.81	0.84	0.88	0.88	0.90
hotel-type	0.32	0.64	0.39	0.60	0.70	0.70	0.40	0.49	0.45
restaurant-area	0.31	0.46	0.56	0.06	0.25	0.53	0.17	0.32	0.42
restaurant-bookday	0.57	0.61	0.57	0.61	0.66	0.62	0.47	0.53	0.54
restaurant-bookpeople	0.74	0.74	0.81	0.84	0.84	0.95	0.95	0.95	0.95
restaurant-booktime	0.39	0.39	0.61	0.49	0.49	0.67	0.49	0.49	0.64
restaurant-food	0.27	0.46	0.57	0.21	0.38	0.53	0.14	0.52	0.42
restaurant-name	0.53	0.66	0.58	0.58	0.67	0.66	0.44	0.53	0.55
restaurant-pricerange	0.07	0.68	0.14	0.18	0.45	0.27	0.00	0.34	0.03
taxi-arriveBy	0.78	0.78	0.93	0.67	0.67	0.80	0.67	0.67	0.87
taxi-departure	0.48	0.79	0.88	0.67	0.75	0.88	0.45	0.65	0.83
taxi-destination	0.66	0.83	0.91	0.80	0.84	0.94	0.54	0.72	0.86
taxi-leaveAt	0.94	0.94	0.97	0.89	0.89	0.90	0.80	0.80	0.89
train-arriveBy	0.60	0.60	0.72	0.74	0.74	0.85	0.54	0.54	0.68
train-bookpeople	0.74	0.74	0.78	0.89	0.89	0.93	0.95	0.95	0.94
train-day	0.55	0.62	0.59	0.64	0.70	0.69	0.40	0.44	0.53
train-departure	0.69	0.89	0.73	0.86	0.94	0.91	0.57	0.72	0.60
train-destination	0.73	0.90	0.78	0.85	0.94	0.91	0.64	0.80	0.68
train-leaveAt	0.52	0.87	0.59	0.46	0.91	0.65	0.44	0.88	0.60

Table 6: Results from the individual slot accuracy evaluation of the models, without Intent Detection but including various types of Post-processing.