FACULTY
OF ELECTRICAL
ENGINEERING
CTU IN PRAGUE

Natural Language Understanding in Open-Domain Dialogue Systems

by

Ing. Jan Pichl

A dissertation thesis submitted to
the Faculty of Electrical Engineering, Czech Technical University in Prague,
in partial fulfilment of the requirements for the degree of Doctor.

Dissertation degree study programme: (P2612) Electrical Engineering and Information Technology
Department of Cybernetics
Specialization: (3902V035) Artificial Intelligence and Biocybernetics

Prague, February 2024

Supervisor:

Ing. Jiří Kubalík, Ph.D.
Czech Institute of Informatics, Robotics and Cybernetics
Czech Technical University in Prague
Jugoslávských partyzánů 3
160 00 Prague 6
Czech Republic

Co-Supervisor:

Ing. Jan Šedivý, CSc.
Czech Institute of Informatics, Robotics and Cybernetics
Czech Technical University in Prague
Jugoslávských partyzánů 3
160 00 Prague 6
Czech Republic

Abstract

In recent years, conversational AI has experienced a significant increase in popularity. The conversational systems have improved significantly, especially due to the advancements of the Large Language Models. Compared to the previous systems, which were mainly built on top of the rule-based approaches, the current systems are capable of responding to various inputs. However, the cost for better response quality is the resource consumption and, consequently, the latency. Moreover, the customization of these systems at a larger scale starts to be problematic.

The main goal of the dissertation thesis is to contribute to the field of conversational system architectures and natural language understanding by leveraging hybrid approaches. We propose a novel hybrid conversational system architecture that was created as a result of a series of architectures tested during annual Alexa Prize Socialbot Grand Challenge competitions. All of the subsequent architecture iterations were awarded during the competitions, and the final one was awarded as a first-place winner. The competition allowed the system to be evaluated in a real-world setup with tens of thousands of users.

The proposed architecture is built on top of sub-dialogue units, allowing a flexible combination of these units to create a complex conversational application. Each sub-dialogue has its own set of intent recognition models that are combined variably during the runtime. The combination of the models is achieved by our proposed novel approach called hierarchical intent recognition. Additionally, the intent recognition is coupled with Out of domain detection, allowing to recognition of the limits of the conversational design and easily plugging the generative Large Language Model (LLM).

Moreover, we created a conversational platform on top of the proposed architecture, allowing the dialogue designer to create and test the conversational application easily using a minimal amount of data. The design process is done using an intuitive visual interface. The resulting applications use efficient algorithms to achieve the lowest possible latency.

Besides the intent recognition and OOD detection, we propose a pragmatic level of understanding using Dialogue Acts. The segmented approach allows fine-grained classification that can be used to modify the conversation flow, especially in the Out of domain (OOD) scenarios.

All of the proposed approaches have been tested in real-world scenarios during Alexa Prize competitions and by our additional voice-enabled applications.

Keywords:
natural language understanding, conversational systems, dialogue systems, socialbot, intent recognition, out-of-domain detection, dialogue act detection, large language models.

Abstrakt

V posledních letech jsme mohli zaznamenat zvýšenou popularitu umělé inteligence. Konverzační systémy výrazně zlepšily své dovednosti zejména díky pokroku v oblasti velkých jazykových modelů. Oproti dřívějším systémům, které byly postaveny především na přístupech založených na předem připravených pravidlech, jsou současné systémy schopny reagovat na celou škálu vstupů. Cenou za lepší kvalitu odpovědí jsou však nároky na výpočetní zdroje a s tím spojená zvýšená doba odezvy. Navíc adaptace těchto systémů v rámci složitější aplikací bývá často problematická.

Hlavním cílem této práce je přispět k rozvoji konverzačních systémových architektur a porozumění přirozenému jazyku využitím hybridních přístupů. Tato práce navrhuje novou architekturu hybridního konverzačního systému, která vznikla jako výsledek řady architektur testovaných během soutěží Alexa Prize Socialbot Grand Challenge. Všechny iterace architektury byly během soutěží oceněny, přičemž finální verze se umístila na prvním místě. Soutěž mimo jiné umožnila vyhodnotit systém v reálném prostředí s desítkami tisíc uživatelů.

Navrhovaná architektura je postavena na konverzačních jednotkách, které nazýváme sub-dialogy. Tyto sub-dialogy mohou být flexibilně kombinovány pro vytvoření komplexní konverzační aplikace. Každý dílčí sub-dialog má svou vlastní sadu modelů na rozpoznávání intentu, které jsou během běhu variabilně kombinovány. Kombinace modelů je dosažena námi navrženým novým přístupem nazvaným hierarchické rozpoznávání intentu. Rozpoznávání záměrů je navíc spojeno s detekcí Out of domain, což umožňuje rozpoznat limity konverzačního návrhu a snadno zapojit generativní jazykové modely.

Nad navrženou architekturou jsme také vytvořili konverzační platformu, která umožňuje dialogovým designerům snadno vytvořit a otestovat konverzační aplikaci s použitím minimálního množství dat. Proces návrhu probíhá pomocí intuitivního vizuálního rozhraní. Výsledné aplikace využívají efektivní algoritmy k dosažení co nejnižší latence.

Kromě rozpoznávání intentu a detekce OOD navrhujeme pragmatickou úroveň porozumění pomocí Dialogových Actů. Segmentovaný přístup umožňuje klasifikaci na úrovni nižší, než je celá promluva, kterou lze využít k úpravě průběhu konverzace, zejména ve scénářích Out of domain (OOD).

Všechny navržené přístupy byly testovány v reálných scénářích během soutěží Alexa Prize a našimi dalšími voice-enabled aplikacemi.

Klíčová slova:
rozpoznávání přirozeného jazyka, konverzační systémy, dialogové systémy, sociální chatbot, rozpoznávání intentu, detekce dialogových aktů, velké jazykové modely.

Acknowledgements

I would like to express my gratitude to Ing. Jan Šedivý CSc., for his guidance and support and for giving me a great opportunity to work on this project as the leader of the Alquist team. I would also like to thank my supervisor, Ing. Jiří Kubalík, Ph.D., for his support. Last but not least, I would like to express my thanks to my family for their support throughout my studies.

The research was supported by the Grant Agency of the Czech Technical University in Prague, grant No. SGS17/092/OHK3/1T/13, SGS18/103/OHK3/1T/37, SGS19/091 /OHK3/1T/37, SGS20/092/OHK3/1T/37, SGS21/091/OHK3/1T/37, and SGS22/082 /OHK3/1T/37. Furthermore, it was supported by Amazon company by stipends, provided infrastructure, and the opportunity to test the system on a large number of real users during the Alexa Prize Socialbot Grand Challenge 1, 2, 3, and 4.

Dedication

Dedicated to the memory of my wonderful mum for her endless love and support.

Declaration

I hereby declare that the presented thesis is my own work and that I have cited all sources of information in accordance with the Guideline for adhering to ethical principles when elaborating an academic final thesis.

I acknowledge that my thesis is subject to the rights and obligations stipulated by the Act No. 121/2000 Coll., the Copyright Act, as amended.

In Prague, 28. 2. 2024, Jan Pichl

Contents

Abbreviations **xxi**

1 Introduction **1**
1.1 Motivation 1
1.2 Problem Statement 2
1.3 Thesis Approach and Contributions 3
1.3.1 Goal and Contributions 5
1.4 Structure of the Dissertation Thesis 6

2 Conversational Systems **9**
2.1 Units of a Conversation 10
2.1.1 Token 10
2.1.2 Utterance 10
2.1.3 Conversation Turn 10
2.1.4 Contextual Window 11
2.1.5 Utterance Segment 11
2.1.6 Session 12
2.1.7 Dialogues and Sub-dialogues 12
2.2 Historical Conversational Systems 12
2.2.1 Eliza 13
2.2.2 Alice 14
2.2.3 IBM Watson 14
2.3 Natural Language Based Interfaces 14
2.3.1 Search Engines 15
2.3.2 Virtual Assistants 15
2.4 Recent Approaches in Conversational Systems 16
2.5 Conversational Systems Categorization 18
2.5.1 Goal-Oriented Systems 18
2.5.2 Open-Domain Systems 19

2.6 Conversational Systems Architectures . . . 19
2.6.1 End-to-end Systems . . . 19
2.6.2 Modular Systems . . . 20
2.6.3 Hybrid Systems . . . 20
2.7 Modular Architecture Components . . . 21
2.7.1 Natural Language Understanding . . . 21
2.7.2 Intent Recognition . . . 22
2.7.3 Entity Recognition . . . 22
2.7.4 Dialogue Acts Detection . . . 23
2.7.5 Additional Classification . . . 24
2.7.6 Dialogue Management . . . 24
2.7.7 Recommendation . . . 25
2.7.8 Natural Language Generation . . . 26
2.7.9 Retrieval Augmented Generation . . . 27

3 Natual Language Processing Algorithms 29
3.1 Text Represenation . . . 29
3.1.1 Tokenization . . . 30
3.1.2 Represeantation Methods . . . 30
3.1.3 Text Embeddings . . . 31
3.2 Semantic Text Similarity . . . 32
3.3 Traditional Machine Learning . . . 33
3.4 Convolutional Neural Networks . . . 33
3.5 Recurrent Neural Networks . . . 34
3.5.1 LSTM . . . 35
3.6 Transformers . . . 36
3.6.1 Generative Transformers . . . 37
3.6.2 GPT Models . . . 37
3.6.3 Prompt-Based Learning . . . 38

4 Effective Hybrid Architecture of a Conversational Socialbot 41
4.1 Alexa Prize Socialbot Grand Challenge . . . 41
4.1.1 Alexa Prize Socialbot Grand Challenge 1 . . . 42
4.1.2 Alexa Prize Socialbot Grand Challenge 2 . . . 42
4.1.3 Alexa Prize Socialbot Grand Challenge 3 . . . 43
4.1.4 Alexa Prize Socialbot Grand Challenge 4 . . . 43
4.1.5 Other Alexa Grand Challenges . . . 43
4.2 Alquist: A Hybrid Socialbot . . . 44
4.2.1 Information Aggregation . . . 45
4.2.2 Component Pipeline . . . 46
4.2.3 Experiments . . . 50
4.2.4 Conclusion and Next Steps . . . 51
4.3 Alquist: Towards Sub-Dialogue Models . . . 52

4.3.1 Architecture . . . 52
4.3.2 Information flow . . . 53
4.3.3 System components . . . 54
4.3.4 Experiments . . . 60
4.3.5 Error Analysis . . . 64
4.3.6 Conclusion and Next Steps . . . 64
4.4 Alquist: Utilizing Knowledge Graph . . . 65
4.4.1 Conversational Knowledge Graph . . . 66
4.4.2 Natural Language Understanding . . . 68
4.4.3 Adjacency Pairs and Dialogue Management . . . 70
4.4.4 Experiments . . . 70
4.4.5 Conclusion and Next Steps . . . 72
4.5 Alquist: Generative Models and Personalization . . . 73
4.5.1 Architecture . . . 73
4.5.2 Skimmer . . . 74
4.5.3 Intent and Out-of-Domain Classification . . . 76
4.5.4 Neural Response Generator . . . 78
4.5.5 Experiments . . . 79
4.5.6 Results and Discussion . . . 81
4.5.7 Conclusion . . . 81
4.6 Evaluation . . . 82
4.7 Conclusion and Main Contributions . . . 86

5 Modular Dialogue Architecture With Hierarchical Intent Recognition 87
5.1 Related Work . . . 88
5.1.1 Conversational Patforms . . . 88
5.1.2 Intent Recognition and Out of Domain Detection . . . 89
5.2 Architecture Description . . . 89
5.2.1 Sub-dialogue . . . 91
5.3 Natural Language Understanding . . . 92
5.3.1 Entity Recognition and Masking . . . 93
5.3.2 Hierarchical Intent Recognition . . . 94
5.3.3 Global Intents Hierarchy . . . 95
5.3.4 Out of Domain Detection . . . 95
5.3.5 Monolithical Single Model with Masking . . . 96
5.3.6 Try-Catch Mechanism . . . 98
5.3.7 Routing Mechanism . . . 98
5.4 Datasets . . . 100
5.4.1 Data Collection . . . 101
5.4.2 Datasets Comparison . . . 102
5.5 Experiments . . . 103
5.5.1 Metrics . . . 103
5.5.2 Pretrained Language Models . . . 104

5.5.3 Classification Model 105
5.5.4 Results 105
5.6 Conclusion and Main Contributions 109

6 Open-Domain Understanding Using Pragramtic Level of Conversation 111
6.1 Related Work 112
6.2 Proposed Method 113
6.3 Datasets 115
6.3.1 Switchboard 115
6.3.2 Midas 115
6.3.3 Dialogue Acts in Segmented Conversations 116
6.4 Experiments 121
6.4.1 Segmentation 122
6.4.2 Dialogue Act Classification 124
6.5 Conclusion and Main Contributions 125

7 Conclusion 127

Bibliography 129

Reviewed Publications of the Author Relevant to the Thesis 145

Remaining Publications of the Author Relevant to the Thesis 155

Awards Relevant to the Thesis 159

Other Conferences, Seminars, and Research Popularization Relevant to the Thesis 161
Conferences 161
Seminars 162
Research Popularization 162
Podcasts 163

List of Figures

2.1 The overview of modular dialogue system architecture 21

3.1 CBOW and Skip-gram architectures (Suleiman et al., 2017) 31
3.2 CNN architecture for a text classification task (Kim, 2014) 34
3.3 Diagram of an LSTM cell structure (Olah, 2015) 35
3.4 The transformer block schema (Zhang et al., 2023) 36

4.1 Alquist 1.0 architecture schema 45
4.2 Alquist 2.0 architecture schema 53
4.3 Schema of Topic graph 57
4.4 Schema of our implementation of Hybrid code networks 58
4.5 Schema of Topic Switch detector 59
4.6 Entity sentiment distribution 64
4.7 Illustration of sentence structure examples as annotation properties 67
4.8 Alquist 3.0 architecture schema 67
4.9 Alquist 4.0 Architecture schema 75
4.10 Hierarchical structure of dialogues 77
4.11 Dependency between the number of examples and the accuracy for the three most common classification models. 80
4.12 The overall performance of Alquist 1.0 throughout the competition 83
4.13 The overall performance of Alquist 2.0 throughout the competition 84
4.14 The overall performance of Alquist 3.0 throughout the competition 84
4.15 The overall performance of Alquist 4.0 throughout the competition 85

5.1 Flowstorm Web Interface 90
5.2 Sub-dialogue nodes 93
5.3 Direct referencing a sub-dialogue from another one. 94
5.4 OOD flow control 96
5.5 Single model intent recognition schema 97
5.6 Try-catch mechanism schema 100

5.7 Routing mechanism schema 101

6.1 Dialogue act action nodes 113
6.2 Counts of the 10 most frequent classes in the Switchboard dataset (after clustering). 117
6.3 Semantic request tree of MIDAS dataset (Yu and Yu, 2021). 117
6.4 Functional request tree of MIDAS dataset (Yu and Yu, 2021). 118

List of Tables

4.1 Accuracy of intent detection . . . 51
4.2 Average rating, time, and number of dialogue turns of Structured Topic Dialogues 52
4.3 Testing results of intent, entity, and combined models . . . 61
4.4 Hyperparameter tuning for NLU models . . . 62
4.5 Testing accuracy of Hybrid code networks models . . . 63
4.6 Sentiment values for entities with general negative connotations . . . 63
4.7 Error distribution in the second version of Alquist . . . 65
4.8 Results of the experiments on IWSLT 2012 English dataset. . . . 70
4.9 Results of a sequence labeling subtask . . . 71
4.10 Results of a combined sequence labeling, knowledge type classification, and ID mapping . . . 71
4.11 Experiments results on ATIS dataset . . . 72
4.12 ALQUIST 4.0 Dataset . . . 78
4.13 CLINC150 Augmented Dataset . . . 78
4.14 Dialogue snippet with generated responses . . . 79
4.15 Manual evaluation on aggregated data . . . 80
4.16 Intent classification results . . . 81
4.17 Requirements of different embedding algorithms . . . 82
4.18 Performance of Alquist in Alexa Prize Socialbot Grand Challenges . . . 85

5.1 OOD dataset comparison . . . 102
5.2 Possible combination of predicted and true intents used in the 2-layer structure 103
5.3 Results of different embeddings used in Monolithical Single Model with Masking on Flowstorm Hierarchical Intent (FHI) dataset. . . . 105
5.4 Results of different embeddings used in Try-Catch Mechanism on FHI dataset. 106
5.5 Results of different embeddings used in Routing Mechanism on FHI dataset. . 106
5.6 Classification models in Monolithical Single Model approach . . . 107
5.7 Classification models in Try-Catch Mechanism . . . 107
5.8 Classfication models in Routing Mechanism . . . 108

5.9 Hiearchical approaches summary 108

6.1 Top 10 dialogue act classes in Switchboard dataset (after clustering). 116
6.2 Sample conversation with each segment labeled with a corresponding Dialogue Act (DA) class. 119
6.3 Dataset statistic of segments per bot and user. 119
6.4 Dialogue act classification on SWDA with different context sizes. 121
6.5 Dialogue act classification on Midas with different context sizes. 121
6.6 Dialogue act classification using optimized models on SwDA dataset with a context size of 5. 122
6.7 Segmentation results on Segment-Level Contextual Dialogue Act (SLCDA) dataset 123
6.8 Two-step dialogue act classification. Context size 5. 123
6.9 Dataset act classification results on SwDA, Midas, and our SLCDA 124
6.10 Core segment classification results. 125

List of Algorithms

4.1 Filter Sentences by Token ASR Scores 48
5.1 Try-Catch Intent Recognition Algorithm 99

Abbreviations

IVA	Interactive Voice Assistant
NLP	Natural Language Processing
NLU	Natural Language Understanding
NLG	Natural Language Generation
NRG	Neural Response Generator
DM	Dialogue Management
BOW	Bag of Words
IOB	Inside, outside, begin
MDP	Markov Decision Process
POMDP	Partially Observable Markov Decision Process
DQN	Deep Q-network
AIML	Artificial Intelligence Modelling Language
APSGC	Alexa Prize Socialbot Grand Challenge
UIMA	Unstructured Information Management Architecture
ASR	Automatic Speech Recognition
TTS	Text to Speech
STD	Structured Topic Dialogue
KG	Knowledge Graph

MD	Mention Detection
CG	Candidate Generation
ED	Entity Disambiguation
NLG	Natural Language Generation
DSTC	Dialogue State Tracking Challenge
STS	Semantic Textual Similarity
BPE	Byte Pair Encoding
MLM	Masked Language Modeling
OOD	Out of domain
LLM	Large Language Model
RAG	Retrieval Augmented Generation
FHI	Flowstorm Hierarchical Intent
DA	Dialogue Act
SVM	Support Vector Machines
TF-IDF	Term frequency-inverse document frequency
CBOW	Continuous Bag of Words
GloVe	Global Vectors for Word Representation
ELMo	Embeddings from Language Models
USE	Universal Sentence Encoder
DAN	Deep Average Network
CNN	Convolutional neural network
RNN	Recurrent neural network
GRU	Gated Recurrent Units
LSTM	Long Short-Term Memory
Seq2Seq	Sequence–to–Sequence
CRF	Conditional Random Field

GPT	Generative Pre-trained Transformer
T5	Text-to-Text Transfer Transformer
NMT	Neural Machine Translation
SwDA	Switchboard Dialog Act
MIDAS	Machine Interaction Dialog Act Scheme
SLCDA	Segment-Level Contextual Dialogue Act

Chapter 1

Introduction

Conversational systems, also called chatbots in various use cases, have become increasingly popular in recent years. The technology behind the systems mainly consists of Natural Language Processing (NLP) and optional Speech Processing for the voice-enabled conversational systems. Each technology category has progressed rapidly in recent years. Over the years, we have witnessed several approaches driven primarily by technology readiness and applicable use cases. Many of the approaches have been successfully implemented in the production environment to solve a particular task. One of the essential parts of the NLP used (not only) in conversational systems is Natural Language Understanding (NLU), which is the main topic of this thesis.

1.1 Motivation

Human–machine interaction has been evolving since the early days of computing. People want to control computers in the most natural way, maximizing their effectiveness during a particular task. One of the most natural and intuitive ways for humans to communicate is through a natural language. However, it is a challenging task for computers to understand and respond to natural language. Adaptation on the human side and improvement of understanding on the technology side are two opposite approaches to a more successful human–machine interaction. However, constraining the richness of the natural language decreases the overall user experience and has a negative impact on the natural way of communication by, e.g., limiting the vocabulary or creating less complex sentences. An improved machine's ability to understand the natural language can, on the other hand, significantly improve the user experience as it does to require any significant input adjustments and modifications.

One of the first forms of interaction that is on the edge of natural language is the use of simple commands. Commands are typically words of a sequence of words from a predefined set. A user of a command-based system needs to know exactly the command wording. In this way, it is easy for the system to map the command for a specific action. Another level of interaction is the use of query language. The query language with the aspects of natural languages does not have to be strictly defined, which allows the users to construct a query freely. This approach has been used in search engines such as Google or Bing for decades. As search engine queries do not have a strict structure, people tend to try various formulations to find what they need in the search results. This leads to an adaptation on the human side.

With recent technological advancements, purely natural language human–machine interaction is becoming more and more popular. However, the origin of this type of interaction dates back to the 1960s when the first conversational system ELIZA (Weizenbaum, 1966) was introduced. The capabilities of the natural-language-based conversational systems have improved over the years as approaches evolved from simple keyword recognition and pattern matching through simple AI description languages, such as AIML (Wallace, 2003), to machine learning and deep-learning-based algorithms. Furthermore, conversational systems have also gained popularity due to advancements in Automatic Speech Recognition (ASR) algorithms. Voice-enabled systems have hit the consumer market in the form of Interactive Voice Assistants (IVAs), such as Amazon Alexa, Google Home, or Apple Siri. These assistants allow handling various use cases via voice using commands or short conversations.

Interactive voice assistants are significant, but still one of the many channels where the conversation may take place. The ultimate goal is not to replace human-to-human conversation but rather to provide an intuitive interface for various use cases where it is suitable to use conversational systems.

It brings a lot of challenges to achieve a natural, fluent, and engaging conversation. One of the steps is natural language understanding, which allows the system to understand the meaning behind the user's utterance. The understanding is crucial for further processing of the dialogue as it detects the important aspect of the current utterance in the context of a whole conversation. With a better understanding, the system can generate more suitable and specific responses, which leads to a more successful conversation. The processing requires multiple steps that will be described later in this thesis.

1.2 Problem Statement

The research of conversational systems as a subset of Natural Language Understanding has been dealing with various challenges over the last several decades. It inherited various approaches from the parent field od NLP concerning language processing from various sources, such as documents, web pages, conversations, voice transcripts, and more. Each source type has its unique language characteristics that typically differ further depending on the underlying topic. The understanding can be done on different levels, e.g., sentence-level understanding or document-level understanding. An example of the same NLU task on different levels can be a sentiment analysis—we can estimate the sentiment on a paragraph, whole document, or per sentence. When dealing with a finer-grained understanding (e.g., per sentence), we use a term *context* to describe the additional information needed for understanding a selected sequence of words.

This thesis focuses on NLU in dialogue (conversational) systems where the language may differ substantially from a language in the form of a block of text typically represented by documents or web pages. In general, the conversation may have two and more participants communicating with each other. For the purposes of the NLU in dialogue systems, let's suppose that the conversation has only two participants—a bot and a user. The bot and the user are taking turns in the conversation. At each turn, both sides respond with an utterance that may or may not be a reaction to the previous utterances in the conversation.

The problem of the NLU in open-domain dialogue systems is recognizing the underlying meaning of a user's utterance in a single turn, given the context of the previous turns. The context has an essential role in understanding as the relevant information does not have to be explicitly mentioned in the turn we are focusing on. There are multiple ways to incorporate the context into the understanding process, and they will be described in detail in Chapter 2 and 5. The understanding consists of multiple tasks that take the current utterance plus the context as an input and outputs the representation of a selected aspect in a suitable form for further dialogue processing. The tasks may, for example, include a classification into a set of classes that can be used directly in the decision logic or as input for the models in the dialogue pipeline. Alternatively, the NLU models may transform the input into to latent representation that can be more suitable for the additional models.

In the open-domain dialogue, as an opposite to a goal-oriented dialogue, there is a wide range of possible user utterances as the dialogue is not limited to a single use case. The classification algorithms used as part of the NLU pipeline cannot be easily limited to a finite set of pre-defined classes as we cannot anticipate all the possibilities of the natural language. There are multiple approaches to tackling this problem. The simplest one is to accept a certain amount of errors caused by the insufficient coverage of the NLU classes. The approach can be further coupled with a fallback strategy to handle unexpected utterances. Another approach is to use an end-to-end dialogue model where the NLU is done implicitly in the model. Dividing the whole conversation domain into smaller logical units combined with end-to-end language models fine-tuned using conversational data brings promising results and it will be further discussed in this thesis.

1.3 Thesis Approach and Contributions

The main goal of the thesis is to contribute to the field of natural language understanding in dialogue systems with a focus on multi- and open-domain system architectures. Such systems typically have no or loosely defined goal of the conversation. Recent research in the field of conversational AI can be divided into three categories—systems with a modular pipeline, end-to-end models, and a combination of both.

Modular systems are commonly used for goal-oriented dialogue systems. The pipeline modules perform various tasks from NLU, dialogue management, and database access to Natural Language Generation (NLG). Dialogue State Tracking Challenge (DSTC) is a typical benchmark of those systems (Williams et al., 2016). End-to-end models based on large language models take advantage of pre-training on large corpora of natural language text. Pre-training on unlabeled text allows the models to learn the properties of the language (or multiple languages) via language modeling tasks such as Masked Language Modeling (MLM) or next token/sentence prediction (Devlin et al., 2019). Models from the GPT family are popular representatives of these models (Brown et al., 2020). The models are then fine-tuned on the particular downstream tasks to be suitable for a certain use case. Alternatively, especially the larger ones, such as GPT-3, can be controlled using a more and more popular prompt-based learning technique (Brown et al., 2020). Recent publications also show the usage of the pre-trained language models to simulate modules in the modular pipeline of the goal-oriented systems (Budzianowski and Vulić, 2019).

Both approaches mentioned above share the requirement of conversational data for the training or fine-tuning process. The large language models are typically pre-trained using data that does not need to have a conversational nature since such data is easier to get. To use these models on conversational tasks, we need to find good quality conversational data for fine-tuning, which introduces several issues. First, obtaining a large amount of good-quality conversational data is not an easy task. The data is often biased as it is usually gathered under specific circumstances. The data from real conversations are typically private and hard to get. Second, the models are driven using the data only, and their behavior can be adjusted primarily by modifying the data or replacing it with a different set.

Since the data plays a key role (not only) in the development process of conversational applications, we see the main contribution of this thesis in addressing the situations when there is not enough good quality data for a particular use case. We see a big potential in decomposing the whole conversational application into multiple logical units (we call them sub-dialogues) in the low data setup. The decomposition comes with two main benefits: First, the reusability of the common parts of the conversation. Second, the conversational design can focus only on the application's essential parts. One of the aspects of the modular architecture is the modularity of the NLU models. Each sub-dialogue has its own set of intent recognition models using a pre-trained base. Moreover, we see the great importance of focusing on the boundaries of the decisions of understanding algorithms, i.e., detecting when the input does not correspond to any of the classes the model is supposed to recognize. This task is referred to as Out of domain (OOD) detection.

Generally, the NLU models can be divided into two categories—domain-specific and domain-independent. Intent recognition models are typical representatives of domain-specific models, while a dialogue act model is representative of a domain-independent model. The domain-independent models are a powerful base for conversational systems as they can be used to improve the performance of the domain-specific models or as a standalone discriminator of unexpected utterances.

Finally, conversational systems have a limited time for producing the response given the user utterance and the other contextual information. This makes them sensitive to using slow yet powerful components. The Large Language Models are the typical representative of the components that may consume a significant time while generating the response. Therefore, we focus on the overall efficiency of the conversational system architecture, especially on using efficient NLU algorithms.

1.3.1 Goal and Contributions

The goal of this thesis is to propose new techniques, architecture, and components that have been developed to overcome the shortcomings of (1) a simple scripted prompt–reply model that has been widely used for several decades and (2) massive and resource-demanding Large Language Models that have been popularized during last few years. To prove the new architecture's usefulness, we will introduce and discuss new data sets and use the resulting system in a production environment at scale. We will also evaluate multiple variants of the architectures and NLU approaches to select the best-performing one in the real environment.

The following list summarizes the main areas of contributions of the thesis:

- Modular architectures
  - Hybrid approach – We propose a series of conversational system architectures that combine modular approaches with a specific conversational design with end-to-end approaches handling a wide range of user inputs.
  - Resrouce effectiveness – The proposed architecture is focused on runtime latency, ensuring the conversational experience is fluent.
- Dialogue structure
  - Sub-dialogue hierarchy – We will show the decomposition of the dialogue into smaller logical units—sub-dialogues—that can be easily created, tested, and reused in various use cases. Each sub-dialogue represents a small portion of the whole dialogue (a few dialogue turns) and has its own set of intents.
  - Reusable dialogue components – Each sub-dialogue can be shared across multiple scenarios and use cases, reducing the development complexity and increasing the dialogue variability.
- Hierarchical Natural Language Understanding
  - Hierarchical Intent Recognition – We propose a flexible hierarchical intent recognition approach, which is a mandatory part of the conversational application divided into multiple sub-dialogues. As the sub-dialogues can be used in multiple scenarios, we introduce a dynamic model combination to achieve the best results.
  - Out of domain detection – Additionally, we take the Out of domain (OOD) detection as an integral part of the intent recognition. We propose a context-specific OOD handling without the need for specific data. This enables a smooth conversation design process and honors the hierarchical approach.
- Pragmatic level of Conversation – The conversational design often deals primarily with a specific level of conversation. The dialogue is then driven based on NLU

results, such as intents and entities (slot values). The pragmatic level of conversation allows driving the conversation regardless of the topic using the utterance segments and corresponding dialogue acts.

The contributions of this thesis have also been implemented and tested as part of the Alquist social bot. The social bot has been awarded several times in a row during annual Alexa Prize Socialbot Grand Challenge competitions, gaining valuable feedback from millions of Amazon Echo customers as well as professional interactors and judges. The user evaluation of the system based on the various NLU (and other) approaches is reported in Chapter 4. Besides the actual conversational system, we developed a set of tools for modular conversational design to make the design, training, and testing process more effective. Moreover, the know-how behind these tools has been incorporated into the conversational platform Flowstorm (Pichl et al., 2022).

1.4 Structure of the Dissertation Thesis

This thesis is organized into several chapters. After the introduction and motivation presented in this chapter, we continue with conversational concepts and categorizations along with the algorithms in the field of NLP. Then, we propose effective architectures of the conversational socialbot, which resulted in a novel modular conversational platform. We also present a pragmatic level of understanding as an addition to domain-specific algorithms. Finally, we conclude the thesis with a summarization of the contributions.

1. *Introduction*: Describes the motivation behind our efforts together with our goals and the problem description. There is also a list of contributions of this dissertation thesis.
2. *Conversational Systems*: Describes the key concepts of the conversational systems, their architecture variants, and their evolvement during the years. It also includes the various categories of the systems, such as goal-oriented and chit-chat systems.
3. *Natual Language Processing Algorithms*: Describes the relevant algorithm commonly used in NLP and NLU fields with focus of conversational domain.
4. *Effective Hybrid Architecture of a Conversational Socialbot*: Describes the open-domain conversational system named Alquist that was developed during the annual Amazon Alexa Prize competitions starting in 2017.
5. *Modular Dialogue Architecture With Hierarchical Intent Recognition*: Describes the sub-dialogue architecture with modular intent recognition, local and global intents logic, and out-of-domain detection.
6. *Open-Domain Understanding Using Pragramtic Level of Conversation*: Describes the NLU on the generic conversational level, which includes dialogue acts detection and utterance segmentation.

7. *Conclusion*: Summarizes the results of our research, suggests possible topics for further research, and concludes the thesis.

Chapter 2

Conversational Systems

Conversation systems have been researched for many years, and their popularity has increased significantly recently. The capabilities of the systems have also changed rapidly, as the systems based on more basic algorithms capable of solving simpler tasks in a limited domain with a limited set of actions evolved into complex systems handling various domains in a more fluent way. Regardless of the use case, the overall goal is to enable human-like communication with users through natural language. The systems allow users to interact with them using written or spoken language, and they respond with appropriate, preferably context-aware responses. They can be integrated into various applications such as web pages, virtual assistants, customer service systems, or voice-controlled devices. Over the years, they have evolved from using simple string-matching and pattern-based approaches to standard machine-learning techniques and, in recent years, to deep learning techniques, including large language models (LLM). The techniques are crucial for understanding and interpreting user input as well as for managing the dialogue and generating appropriate responses. Additionally, voice bots require automatic speech recognition (ASR) and Text-to-speech (TTS) algorithms to process voice input and provide the response via voice.

During a conversation, humans perceive more than just spoken words. They use other senses, such as body language, nuances in spoken language, and emotions. They also consider their experiences related to the location, time, and purpose of the conversation. All of these qualities have their role in human-to-human interaction, and the trend in modern systems is to model more and more of them. However, we will focus only on the language part of the conversation in this thesis, more specifically on the understanding part.

The aim of this chapter is to introduce the conversational system concepts and focus on the related research during recent decades. First, we define the important terms used in the research of conversational systems. Then, we follow with a brief history of the systems that contributed significantly to the further research. The related natural language interfaces are mentioned as a bridge between historical and modern conversational systems that taught the users to use a (modified) natural language while interacting with machines.

Additionally, we categorized the conversational systems based on their use cases and architecture type. For modular architectures, we describe the important parts and mention the related research.

2.1 Units of a Conversation

The following terms are commonly used to describe a conversation and to address individual conversational units in turn-taking conversations. The different conversation units are then used as input in the particular NLP algorithms that will be discussed later in the thesis. The decision of which type of conversational units are suitable inputs for the algorithms depends on the task type and complexity. For example, a dialogue act classification performs best when a contextual window is provided, as the previous utterances also determine the dialogue act of the current one.

2.1.1 Token

A token is a unit consisting of a sequence of contiguous characters that play a certain role in a written language (Hagiwara, 2021). The token often equals a single word. The process of splitting a text into tokens is called tokenization. However, some algorithms used for tokenization produce multiple tokens for a single word to keep a reasonable size of a vocabulary (Sennrich et al., 2016; Schuster and Nakajima, 2012). In the thesis, we will adopt the term token to represent a single word since the token classification algorithms (e.g., entity recognition) are required to perform a word-level classification.

2.1.2 Utterance

An utterance is a tuple of speaker identification and a sequence of tokens. Let's denote U to be an utterance, then

$$U = (s, t_1, \ldots, t_n)$$

where s is the speaker identification and t_i is a token. In general, the number of speakers in the conversation may be arbitrary. We are focusing on cases where there are exactly two speakers, i.e., a bot and a user.

2.1.3 Conversation Turn

A conversational turn, also called a dialogue turn, is a basic unit of the conversation. It consists of a pair of utterances, typically a user utterance (also called a message) and a bot utterance (also called a response). The reason why the bot utterance is called response and not the user utterance is that from the perspective of building a conversational system, the system is producing responses to the input utterances (messages). Generally, the formal definition of the conversational turn is following:

$$T = (U_1, U_2)$$

When speaking of conversational turn in the context of the bot–user conversation, the user utterance is the first one in the tuple, and the bot utterance is the second one.

$$T = (U_u, U_b)$$

When dealing with the turn as part of the conversation, we are speaking of the turn-taking conversation (Sacks et al., 1978). Assuming two speakers are involved in the conversation (user and bot), the turn-taking conversation proceeds in a way that only one speaker speaks at a time and the other one lines. Once the first speaker finishes, the roles are switched.

2.1.4 Contextual Window

Given a specific turn in the conversation, a contextual window is a range of multiple preceding dialogue turns. We are not considering turns succeding the current turn as the algorithms trained on the contextual windows need to be used in runtime where only past turns are available. Formally, the window W of size n is defined as follows:

$$W_k^n = (T_{k-n+1}, \ldots, T_{k-1}, T_k)$$

where T_k is a current turn. The contextual window of size 1 contains only the current turn, size 2 contains the current turn plus the previous one, etc.

Contextual windows are used in situations when the desired information cannot be classified using only the current turn as an input. For example, during entity recognition, when the bot asks *"What is your favorite movie?"* and the user responds with *"Matrix"*, the classification algorithm needs at least a previous turn to be included in the window to successfully classify *Matrix* as an entity of type movie (without entity linking or additional database of entity types).

2.1.5 Utterance Segment

Utterance segments are non-overlapping sequences of tokens that cover a whole utterance. Each utterance (both users' and bots') can be divided into one or more segments. Individual segments may have a different intent and dialogue act (see chapter 6), and they might be a reaction to a different part of the previous conversation or a newly introduced direction of the conversation. For example, a single segment utterance *"Do you want to play a game?"* can be followed by a two-segment utterance *"That sounds nice. What game do you offer?"*. It contains the segment *"That sounds nice."*—confirmation of the previous proposal—and the segment *"What game do you offer?"*—follow-up question.

Formally, we can modify a definition of the utterance to reflect the token assignment to the individual segment in the following way.

$$U = (s, (t_1, seg_1), (t_2, seg_1), \ldots, (t_k, seg_1), (t_{k+1}, seg_2), \ldots, (t_l, seg_2), \ldots, (t_n, seg_m))$$

The utterance U consists of n tokens that are partitioned into m segments. Each segment may have a different number of tokens.

2.1.6 Session

A session or a conversational session is an ordered sequence of turns, each consisting of a pair of utterances (as described above). The session is initiated by one of the speakers and goes until either of the speakers ends it using an ending utterance or by abandoning the conversation. In the bot–user scenario, we can examine the reason for the end of the conversation and categorize it, for example, into the following categories: user ended the conversation, user abandoned the conversation, the goal of the conversation is fulfilled, or a system failure occurred.

Let's denote S to be a session of length n and U_i to be an utterance, then

$$S = (U_1, U_2, \ldots, U_n)$$

2.1.7 Dialogues and Sub-dialogues

We define a dialogue to be a definition of the particular scenario that may occur during a conversation between a bot and a user. In other words, it is a description that can be directly used by the conversational system to drive the conversation to fulfill the desired goal. It is important for the description to be in a format that can be consumed by the particular system. The two most common description formats are the following:

1. **Dialogue graph** – A directed graph with various node types defining user intents, bot speeches, functions, and more. Transitions between nodes define the flow of the conversation.

2. **Prompts** – A natural language text describing the flow as the instructions for the instruction-based large language models.

We define a sub-dialogue as a component of a larger dialogue structure characterized by its nested position within the dialogue(s) that reference or embed other dialogues. In this hierarchical arrangement, a main dialogue serves as the entry point and may reference other sub-dialogues. This hierarchical structure contributes to the readability of the conversational application, reusability of the individual sub-dialogues, and better maintainability. The sub-dialogue can be in both Dialogue graph and Prompt form.

2.2 Historical Conversational Systems

In this section, we describe historical approaches to conversational AI starting in the early days of computer science and artificial intelligence research. The systems were able to simulate simple conversations and were originally based on rule-based and pattern-matching methods. They were eventually replaced by machine-learning approaches as the technology evolved.

We describe three specific conversational AI systems that are the major contributors to the field from the historical perspective. The first system is ELIZA, created by MIT computer scientist Joseph Weizenbaum (1966). ELIZA was a simple program that used pattern matching and substitution to simulate a conversation with a psychotherapist. The second one is ALICE bot (Wallace, 2003). The bot was inspired by ELIZA, and its main contribution is the Artificial Intelligence Modelling Language (AIML). The last system we want to mention is IBM Watson (Ferrucci, 2012; Ferrucci et al., 2013), which contributed especially in the field of question answering by winning the famous Jeopardy! TV show.

As the technology progressed, it started to be applied to more and more real-world use cases. In the beginning, those use cases were more focused on single-turn commands or simplified natural-language-like query language (section 2.2.3).

In recent years, the advancement of machine learning, Natural Language Processing and other technologies have made it possible for the systems to progress to multi-turn and context-aware conversational systems that can understand and respond to a wide range of inputs in a human-like manner (section 2.3.2). These systems can be integrated into various applications such as virtual assistants, customer service chatbots, and voice-controlled devices. With the rapid growth of the internet and mobile devices, conversational systems have become more widely used and accessible to the general public.

2.2.1 Eliza

ELIZA is a natural language processing computer program that simulates a conversation with a psychotherapist. It was developed by Joseph Weizenbaum (1966) at the Massachusetts Institute of Technology (MIT). The program was designed to demonstrate the potential of natural language processing and to explore the ways in which computers can be used to mimic human communication.

ELIZA's interactions with users are based on the theory of Rogerian psychotherapy (Rogers, 1957), which emphasizes the importance of active listening and the use of open-ended questions. When a user inputs a statement or question, ELIZA responds with a generic, open-ended question or statement that is intended to encourage the user to continue with the conversation. For example, if a user says *"I feel sad,"* ELIZA might respond with *"Can you tell me more about why you feel sad?"*

The program uses a set of simple pattern-matching rules to generate its responses. These rules are based on keywords and phrases in the user's utterance, and they allow ELIZA to respond in a way that is appropriate to the context of the conversation. ELIZA's pattern-matching rules are relatively simple and do not involve any deep understanding of the meaning of the user's utterance. However, the program is able to generate a wide variety of responses, which can make it seem more sophisticated than it actually is.

ELIZA was one of the first programs to demonstrate the potential of natural language processing and has inspired many other similar programs. The program was also used as a tool to study human behavior, as it revealed the extent to which people are willing to attribute human-like characteristics to computers. The program's ability to engage users in a seemingly meaningful dialogue, despite its simple rules, was seen as a demonstration of the power of human imagination.

2.2.2 Alice

ALICE is one of the systems inspired by ELIZA. It won Loebner Prize (Mauldin, 1994) three times. The bot utilizes the Artificial Intelligence Modelling Language (AIML) (Wallace, 2003) programming language. AIML allows developers to create conversational agents, or chatbots, that can understand natural language input and respond in a human-like manner. The ALICE chatbot utilizes AIML to define a set of categories that the chatbot uses to understand user input and respond appropriately. These categories consist of a pattern that the chatbot looks for in the user's utterance and a template that the chatbot uses to generate its response. The ability of the chatbot to understand and respond to user input is determined by the number and complexity of categories that have been defined in its AIML script.

One of the benefits of using AIML is that it is an open-source language that allows developers to customize and modify the chatbot's behavior easily. Additionally, the AIML community has created a large library of pre-existing AIML categories that can be used to train the chatbot as well as the interpreters written in various programming languages quickly.

2.2.3 IBM Watson

IBM Watson, a pioneering computer system, emerged from IBM's DeepQA project, led by principal investigator David Ferrucci. Initially designed for question-answering, Watson gained global attention by winning the Jeopardy! Quiz show in 2011 against the two highest-ranked players in a nationally televised two-game Jeopardy! (Ferrucci, 2012). Watson leverages IBM's DeepQA software (Ferrucci, 2011) and the Apache Unstructured Information Management Architecture (UIMA) framework (Ferrucci et al., 2009). Its success and conceptual design inspired the NLP community to start research areas that were not the primary research focus, such as information retrieval, structured knowledge representation, and automated reasoning.

Notably, Watson's capabilities expanded beyond Jeopardy! (Ferrucci et al., 2013), with its first commercial application announced in 2013 for utilization management decisions in lung cancer treatment at Memorial Sloan Kettering Cancer Center. IBM emphasized Watson's use of over 100 techniques to analyze natural language, identify sources, generate hypotheses, score evidence, and merge and rank hypotheses. In recent years, Watson's evolution has included new deployment models, enhanced machine learning capabilities, and optimized hardware, transforming it into a versatile system capable of various language-related tasks. This transformation pushed the system's abilities beyond question-answering, contributing to the broader landscape of conversational AI.

2.3 Natural Language Based Interfaces

Despite the significant milestones in conversational AI history, the technology was not mature enough for the massively used multi-turn context-ware conversational systems until recently. However, simpler natural-language-based interfaces using command–response approaches such as search engines and virtual assistants became popular at the beginning of the 21st century.

2.3.1 Search Engines

Search engines started to be popular at the end of the 20th century when the Internet started to be used on personal computers. The role of search engines as the entry point to the World Wide Web and the main tool for retrieving information has undergone significant evolution, with advancements transforming them into sophisticated natural language interfaces.

In the 1990s, the first search engines, such as Archie, Gopher, or AltaVista, primarily operated on keyword-based searches, requiring users to input queries using concise and structured language (Seymour et al., 2011). The experience was rather slow. Google Search, first launched in 1998, revolutionized the way people access information on the internet. It addressed the issues of its predecessors and started creating an extensive index of billions of web pages.

The main success (Seymour et al., 2011) of the Google search was due to their patented PageRank algorithm (Page et al., 1998). Additionally, Google Search started using advanced natural language processing algorithms to quickly and accurately match users' queries with relevant information. The NLP algorithms used for web search originally included semantic parsing intent recognition and named entity recognition. The usage of these algorithms started the shift from strict keyword-based queries to more natural-language-like and even conversational ones, which contributed to a better user experience. Another significant shift came with the concept of semantic search (Nayak, 2019). The semantic search allowed users to find the desired web page even if they used different keywords while they were semantically similar. This approach started to be massively adopted since the introduction of transformer-based models (Liu and Lane, 2016) such as BERT (Devlin et al., 2019).

Besides the understanding of context and meaning behind user queries, Google's Knowledge Graph (Singhal, 2012) was introduced in 2012, aimed to comprehend the relationships between entities, enabling more context-aware responses, especially in the factual questions such as *"When was Barack Obama born?"*

The historical progression of search engines from keyword-based systems to sophisticated natural language interfaces has played a crucial role in shaping the landscape of AI conversational systems. The integration of NLP, semantic search, and knowledge graphs has not only enhanced information retrieval but has also influenced the design and capabilities of modern conversational AI.

2.3.2 Virtual Assistants

Virtual assistants push forward the capabilities of natural language interfaces primarily by introducing an ability to interact with them via voice and a wide range of use cases.

They are rule-based and machine-learning-based applications that are designed to assist and perform tasks for users. Virtual assistants are typically accessed through a device such as a smartphone, tablet, or computer, and they can be used to perform tasks such as scheduling appointments, setting reminders, making phone calls, sending messages, and searching the internet.

They started to be popular in the early 2010s with the advancement of automatic speech recognition (ASR) and text-to-speech (TTS) technologies. Large corporations started to implement their own virtual assistants to provide their services via additional channels. The key virtual assistant providers were originally Apple with Siri (2011), Google with Google Now (2012), and Amazon with Alexa (2014). The strategy of Apple and Google was to build the assistant in the mobile operating systems while Amazon created a custom device called Echo, which is intended to be placed in households.

In the early days, virtual assistants were mainly powered by rule-based approaches with pattern matching or by simple intent recognition algorithms. The technology level allowed users to interact with the in a command–response manner. This type of interaction does not require working with a context of previous turns or sessions but only extracts the information from the current utterance. The single utterance commands then progressed to multi-turn form-filling-like conversations to naturally get additional information from the user by asking questions to fulfill the command successfully. This typically includes the combination of intent recognition and entity (slot) recognition algorithms.

One of the key features of virtual assistants is the integration of various external services such as web search, calendars, note-taking applications, reservation systems, and more. However, when the number of operatable services is high, intent recognition and service disambiguation become challenging. There are multiple ways to deal with this issue. For example, Alexa comes with the concept of invocation words for identifying a particular service. Specifying the invocation words as the name of the service is inconvenient as the user needs to remember the exact phrase. To conveniently recognize the desired service from the context, the virtual assistants need to use advanced machine-learning approaches, including the Large Language Model, for both response generation and service calls. The companies behind virtual assistants are slowly adopting these approaches, but the overall user experience still relies mostly on the more straightforward approaches. The LLMs (as in the other use cases) have a big potential to push the capabilities more towards the natural conversation (Dong et al., 2023).

2.4 Recent Approaches in Conversational Systems

In the 2010s, neural networks started to play a significant role in the field of conversational systems. The traditional approaches used before struggled to capture the complexity and nuances of natural language, limiting the development of complex and contextual dialogue systems. Neural networks, with their ability to model patterns and dependencies in data, brought new possibilities for working with context in conversational systems. Recurrent neural networks (RNNs) especially Long Short-Term Memory (LSTM) (Hochreiter and

Schmidhuber, 1997) introduced possibilities to model patterns in sequential data, even for long sequences. With the attention mechanism (Bahdanau et al., 2014), it gave birth to the Sequence–to–Sequence (Seq2Seq) model with an encoder–decoder architecture. The architecture debuted originally in machine translation task (Sutskever et al., 2014) where the input to the network is an utterance in the source language, and the output is the utterance in the target language, whereas while used in the conversation systems, the input is a user utterance and the output is the response of the system.

The baseline framework is proposed by Vinyals and Le (2015). It is a plain Seq2Seq architecture that is trained on the helpdesk chat service and Opensubtitles dataset. The drawbacks presented in the paper include short and generic responses and inconsistent personality. The responses could also be inconsistent for questions that are not identical but are semantically similar.

Wu et al. (2018) proposed an approach that tries to control the response generation by incorporating dialogue acts. The solution consists of two models: a generative network for the response generation and a policy network for the dialogue acts. The dialogue acts, as the additional input for the generative networks, improve the response quality by enabling the response to be generated differently for statements, questions, or context switch acts.

In human–human conversations, people tend to use the same words mentioned in the previous utterances to some extent. The Seq2Seq models fail to generate such words as they typically work only with the latent representation of the utterance, and the decoder part does not interact directly with the words. The copying mechanism proposed by Gu et al. (2016) tries to mitigate this problem. The task for the models is to decide when to generate the word and when to copy it from the original utterance.

The Seq2Seq models with RNN or LSTM-based encoders and decoders have typically been trained on the conversational data relevant to the specific use case as the network architecture is not suitable for pretraining on a large corpus of natural language data to learn language patterns without forgetting them while fine-tuning. The Transformer architecture introduced by Vaswani et al. (2017), with its parallelization capabilities, allowed the creation of bigger models and addressed the mentioned limitations of traditional Seq2Seq models. One of the biggest advantages of the transformer models is the possibility of effectively using transfer learning instead of learning the desired task from scratch. The transformer architecture has been a standard for NLP models in various tasks, including the sub-task for conversational systems (intent recognition, named entity recognition, ...) and end-to-end conversational systems. The pre-trained models benefit from already learned language patterns, allowing the downstream tasks to be trained using smaller data sets.

The trend with the reduction in the data amount needed for model training eventually resulted in few-shot and zero-shot models. Popularized by OpenAI's Generative Pre-trained Transformer (GPT) 3 by Brown et al. (2020), the approach represents a significant shift in the capability of models to perform tasks without explicit training for them. This approach introduces a new dimension to conversational systems, allowing them to generalize and adapt to a wide range of tasks with minimal supervision. A model can take a prompt as an input where the desired input is described in a natural language (zero-shot learning). Additionally, the prompt may contain a few examples of mapping the input to the output (few-shot learning).

2.5 Conversational Systems Categorization

The conversational systems and their underlying research are commonly divided into two categories based on the purpose of the system: goal-oriented systems and open-domain systems. A goal-oriented system typically follows a scenario to fulfill a predefined goal or multiple goals. As it typically requires gathering certain information, additional algorithms, such as slot-filling, must be implemented compared to the open-domain systems. On the other hand, open-domain systems require a significant amount of conversational data to train a system that is robust enough.

2.5.1 Goal-Oriented Systems

Goal-oriented systems refer to conversational systems that are designed to achieve a specific task or objective, such as a room or table reservation (Pateras et al., 1999), technical support (Vinyals and Le, 2015), or providing tourist guidance (Mrkšić et al., 2016). These systems usually require users to provide specific information to the chatbot, which is then used to fill predefined slots with specific values. This process is commonly referred to as "slot-filling." The main objective of a goal-oriented dialogue system is to fill all the required slots using the lowest number of dialogue turns possible.

The system has a specific goal or a predefined set of goals. Each goal has a corresponding set of values that need to be obtained from the user to achieve the goal. Therefore, the key aspect of the system is its ability to understand and interpret user input and extract desired values. Moreover, for a fluent conversation, it is critical for the system to recognize multiple values from a single utterance instead of asking redundant questions.

Once the required values are extracted, they are typically used to create a query in a database or call a service that performs a particular action. The database or service usually returns a value or values that need to be presented to the user. The information is processed by the Natural Language Generation (NLG) part of the system. The NLP typically uses sentence templates or the encoder–decoder models described earlier. Additionally, Large Language Models (LLMs) play a significant role in response generation, as they are strong in considering the context as well as the actual variable values.

The traditional architecture described above, consisting of natural language understanding for value extraction, dialogue management for action selection (e.g., database query), and natural language generation for presenting the output to the user, has been used for years. With the rise of LLMs, it still remains relevant, with the only difference being that each part of the framework may be handled by the same LLMs with an appropriate prompt. The LLMs allow faster development of these systems as they don't require large sets of training data. However, the downside of these models is the hardware requirements that are significantly higher than using simpler models.

2.5.2 Open-Domain Systems

In contrast to goal-oriented systems, open-domain systems are designed to handle a wide range of topics and user queries without a specific predefined task. The goal is to keep an engaging conversation as long as possible (Ram et al., 2018). There is no success or failure at the end of the conversation, and there is a focus on natural responses. Users can only give a rating based on their subjective impression. The systems have been historically based on end-to-end architecture (described later) using a single model trained on a large corpus of conversation data. With the recent advancement of capabilities of the end-to-end systems mostly driven by LLMs, the open-domain system gained more popularity. Various mobile apps offering virtual friends or even a relationship partner have emerged on popular platforms.

Additional techniques allowing the model to be enriched by additional information started to be used frequently recently. The techniques are mostly based on Retrieval Augmented Generation (RAG) (Gao et al., 2023a).

2.6 Conversational Systems Architectures

2.6.1 End-to-end Systems

An end-to-end dialogue system is a type of conversational AI that is designed to handle the entire process of understanding and generating natural language for the purpose of carrying out a conversation. This process includes understanding the user's intent, generating an appropriate response, and delivering that response in a natural and conversational manner.

End-to-end dialogue systems are typically based on neural network architectures, such as Recurrent neural networks (RNNs) (Wen et al., 2017) and transformer models (Vaswani et al., 2017; Radford et al., 2018, 2019; Brown et al., 2020). These architectures are trained on large datasets of conversational data to learn patterns in human language and how to generate appropriate responses.

The original research on end-to-end dialogue systems was focused on encoder-decoder architecture (Cho et al., 2014). The encoder takes in the input sentence and encodes it into a fixed-length vector representation that captures the meaning of the sentence. The decoder then takes this vector representation and generates an appropriate response. Recent LLMs such as GPTs simplify the architecture by using only the decoder part (Radford et al., 2018, 2019; Brown et al., 2020).

End-to-end dialogue systems can be used in a variety of applications, including chatbots, virtual assistants, and customer service systems. They can be integrated into websites, mobile apps, and other platforms to provide users with a natural and convenient way to interact with the system. Their main advantage is that they can handle a wide range of inputs and generate natural and appropriate responses without the need for rule-based systems or manual scripting. However, these systems still have some limitations, such as the need for large amounts of training data and the difficulty of handling more complex tasks and scenarios.

Overall, end-to-end dialogue systems are an advanced form of conversational AI that has the potential to greatly improve the user experience in a variety of applications. With the continued advancement of neural network architectures and the availability of large amounts of training data, it is likely that end-to-end dialogue systems will continue to improve and become an increasingly important part of our daily interactions with technology.

2.6.2 Modular Systems

Modular dialogue systems are designed with a modular architecture, where different components or modules handle specific aspects of the dialogue process. The components are typically trained separately on a domain-specific datasets. A single dialogue turn is processed sequentially by all of the components. The typical flow is shown in Figure 2.1, and the individual components will be described in detail in section 2.7. Sequential processing is often required by certain components requiring the outputs from the previous ones. When the dependency is not necessary, parallelization may be implemented to speed up the process.

The modular approach often comes along with a set of lightweight components compared to a monolithic end-to-end model. It allows the individual components to be trained on a smaller amount of specific data. This approach is more suitable for goal-oriented systems where a large amount of in-domain conversational data is not typically available. The datasets for the individual components can be created by domain experts to bootstrap the initial versions of the system. The modularity also allows the inclusion of the additional optional components required for specific domains only. The typical examples are domain-specific classifiers, e.g., emotion detection for psychologically oriented dialogues.

2.6.3 Hybrid Systems

In hybrid dialogue systems, the architecture is designed to take advantage of the strengths of both end-to-end and modular approaches. The end-to-end approach ensures the ability to respond to a wide range of utterances by leveraging pretraining on large dialogue datasets without relying on hand-crafted rules or separate components. On the other hand, modular approaches divide the dialogue process into separate components, each designed to handle a specific task in an easily modifiable way. This allows for more explicit control over individual components and a more structured approach to development.

The critical point of the hybrid system is the logic that combines the end-to-end model with the individual components. Switching the approaches comes with the challenges of sharing the context, i.e., the information extracted using the other component. Moreover, the modular system may use templated responses, and the fluent combination of these responses with the ones generated by the end-to-end approach may not be straightforward.

The strengths of this approach are that it is used in scenarios where the end-to-end approach grants the ability to respond to unexpected utterances in a reasonable way while the modular approach precisely drives the conversation in a designed way. We focus on this scenario in more detail in the following chapters of this thesis.

2.7 Modular Architecture Components

As outlined above, the modular architecture consists of various modules that we want to describe in more detail in this section. The modules work together to perform specific tasks that are divided into the following high-level categories: Natural Language Understanding (NLU), Dialogue Management (DM), and Natural Language Generation (NLG). These modules can be combined in different ways, but the canonical information flow is illustrated in Figure 2.1. Additionally, short-term storage—usually called context—long-term storage—usually called profile—and knowledge base are used for storing the information and using it in the conversation. In the case of voice-enabled applications, the Automatic Speech Recognition (ASR) and Text to Speech (TTS) components are at the beginning or at the end of the flow, respectively.

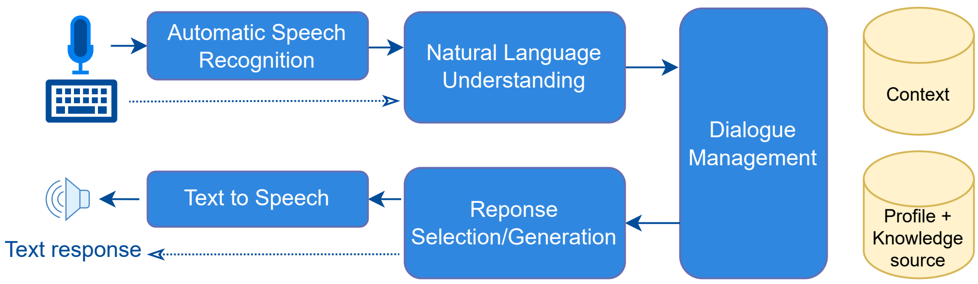


Figure 2.1: The overview of modular dialogue system architecture

2.7.1 Natural Language Understanding

Natural Language Understanding (NLU) is the family of tasks related not only to the conversational system but also to other parts of NLP. In the conversational systems, the NLU algorithms process the utterance at the beginning of the pipeline. Their results are critical for further processing. In voice-enabled systems, the algorithms often work with a text transcript produced by ASR or multiple ASR hypotheses if available. Additionally, the understanding can be done using the audio of recorded voice or a combination of both. This is useful for tasks such as emotion recognition (Trinh Van et al., 2022) as the information about the emotion lies in the voice tone more than in the actual content of the speech.

The most essential component of NLU is the intent recognition. It categorizes the utterance into one of the intent classes, which influences further processing. Additional components may include:

- Entity Recognition and Slot Filling – Extracting relevant entities and filling semantic slots from user utterances to capture important pieces of information necessary for task completion or context understanding within the conversation.

- Dialogue Act Detection – Recognizing the communicative actions or speech acts performed by speakers in a conversation. This includes categorizing utterances into different dialogue acts, such as questions, statements, requests, acknowledgments, etc., to better understand the structure and flow of the dialogue.

- Sentiment Analysis – Analyzing the sentiment expressed in user utterances to detect emotions, attitudes, or opinions, which can influence the tone and style of the conversational system's responses.

We describe the most important component in more detail in the following subsections.

2.7.2 Intent Recognition

Intent recognition in dialogue systems is the process of identifying the intent or goal of a user's spoken or written input. It is a critical component of natural language understanding in dialogue systems, as it allows the system to determine what the user is trying to accomplish and respond appropriately.

Since intent recognition is the text classification task, it leverages the algorithms and model architecture for this task type. Text classification relies on two main aspects: text representation and the actual classification model. The first systems used Bag of Words (BOW) or N-grams (Lee et al., 2013) as the text classification method. Coupled with similarity metrics or classification models such as Logistic regression or Support Vector Machines (SVM), it forms a baseline approach. Additional features such as part-of-speech tags (Voutilainen, 2003), sentiment, topics, or dependency parses (Nivre, 2010) were often used to improve the overall performance. Moreover, the preprocessing of the input, such as removing stop words, has been a highly leveraged step (Uysal and Gunal, 2014).

With the introduction of a text representation capturing the semantics (Mikolov et al., 2013b; Pennington et al., 2014; Bojanowski et al., 2017; Wu et al., 2017), the need for feature engineering and preprocessing became lower. This type of representation is suitable to be processed by neural networks to learn the downstream task. Additionally, the amount of data needed for the model training is lower thanks to the pertaining of the representations on a large unlabeled text. The neural architectures used for text classification often leveraged convolutional (Kim, 2014) or recurrent units (Ravuri and Stolcke, 2015).

Recent advances in neural architectures, especially the introduction of a transformer architecture (Vaswani et al., 2017), have led to more effective language representation. The models such as BERT, Albert, DistilBert (Devlin et al., 2019; Lan et al., 2019; Sanh et al., 2019) and GPT-3 (Brown et al., 2020) are some of the most popular nowadays and achieve the state-of-the-art performance on various text classification tasks.

2.7.3 Entity Recognition

Entity recognition in dialogue systems is the process of identifying and extracting specific pieces of information, known as entities, from natural language text. These entities can include proper nouns, such as person or location names, as well as numerical values, such as dates or monetary amounts. The goal of entity recognition is to enable a dialogue system to understand the meaning hidden in the text spans representing the entities.

Entity recognition is a token classification task where each token is tagged as an entity or not. More specifically, each token is tagged with either *Inside, outside, begin* (IOB) (Ramshaw and Marcus, 1999) tag to distinguish non-entity tokens, a first token belonging to a particular entity, and additional entity tokens. Additionally, there could be an entity type tag for each entity token resulting, for example, in *"B-city, I-city, O"* tags.

The original approaches leveraged dependency parsing (Nivre, 2010) to identify nouns and noun phrases that could be later matched with the list of anticipated entity tokens. However, this approach requires the list of entities to be known, which is often not the case, or they cannot be easily enumerated. Similarly to intent recognition, entity recognition leverages the text embedding representation, capturing the semantics. With the popularization of the LSTM-based (Hochreiter and Schmidhuber, 1997) neural networks, these models started to be used for entity recognition as their strength is the ability to capture dependencies in longer sequences. The bidirectional LSTM layer is used to handle the dependencies in both directions in the utterance. Additionally, the Conditional Random Field (CRF) layer (Huang et al., 2015) is used to model dependencies among the output classes. Another approach (Kurata et al., 2016) takes advantage of encoder–decoder architecture to encode the input sentence into a fixed-length vector that is used as the input for the decoder layer. The decoder layer then operates with the information from the whole sentence. The most recent approach also leverages transformer-based architectures. In comparison to the text classification, the model has been modified head to label each token individually. Few-shot approaches (Fritzler et al., 2019) aim to lower the amount of needed training data.

As intent detection and entity recognition depend on each other, several combined approaches were proposed. A similar approach to the entity recognition itself can be applied to the combined recognition. The only modification is that the end-of-sentence token is tagged with the intent (Hakkani-Tür et al., 2016). Another approach is to modify the encoder–decoder architecture by adding a second decoder for the intent, which shares the same encoder with the entity-tagging decoder (Liu and Lane, 2016).

2.7.4 Dialogue Acts Detection

Dialogue act detection is the task of identifying the communicative intention behind an utterance in a dialogue. It involves categorizing the utterance into one of several predefined dialogue acts, such as informing, confirming, or requesting. Dialogue act detection is an important component of dialogue systems, as it helps the system understand the user's goals and intentions and respond accordingly.

Dialogue act detection can be performed using various techniques, including rule-based methods, machine learning algorithms, and neural networks similar to intent recognition. In rule-based methods, hand-crafted rules are used to identify the dialogue act based on features such as word frequency, part-of-speech, and syntax (Yeh, 2016). Machine learning algorithms use labeled training data to learn the mapping between utterances and dialogue acts. The state-of-the-art approaches are described later in chapter 6.

Dialogue act detection is often used in addition to other natural language processing tasks, such as intent classification, entity recognition, and sentiment analysis, to provide a more complete understanding of the user's inputs. This information can be used to inform the system's response generation process, allowing for more context-aware and human-like responses.

2.7.5 Additional Classification

The additional classification is not essential for the processing of the modular conversational system. However, for some use cases, the selected classifiers may improve the overall NLU accuracy or be beneficial for further processing. The additional classification tasks are, for example:

- Emotion classification – the task of recognizing and categorizing the emotional state of a speaker from their speech or text. This involves detecting the presence of emotions such as happiness, sadness, anger, fear, surprise, and others. The results of emotion classification can be used to improve the user experience in conversational systems, analyze public opinion, and develop applications that can respond to the emotional state of users.

- Topic classification – the task of identifying the topic or subject matter of a text or speech. This involves assigning the text to one or more predefined categories, such as sports, politics, entertainment, and others.

- Topic-specific classification – classification of topic-dependent aspects. This is closely connected to intent recognition, which is also dependent on the current topic. However, topic-specific classification is relevant throughout the discussion of the particular topic, while specific intent is relevant only in a part of the topic. The example is a product classification for a company-specific dialogue system.

2.7.6 Dialogue Management

Dialogue management is the process of controlling the flow of a conversation between a dialogue system and a user. It involves deciding what action the system should take based on the user utterance and consequent NLU annotations. It can be performed in different ways, including rule-based systems, finite-state machines, and machine learning-based systems.

In rule-based systems, the dialogue management process is defined by a set of handcrafted rules (Allen, 1995) that specify how the system should respond to different inputs. This is a straightforward approach considering the reasonable amount of intent classes, entity categories, and additional classifications. The mentioned tasks represent the NLU annotations combinations that can be directly mapped to the dialogue actions. Finite state machines extend rule-based systems, where the dialogue is modeled as a series of states and transitions (Larionov et al., 2018). These systems allowed for more dynamic and context-sensitive responses by encoding dialogue policies as state transitions.

Machine learning-based approaches take advantage of both supervised learning and reinforcement learning. An approach called Neural Belief Tracker is proposed by Mrkšić et al. (2016). It assumes each intent–entity pair determines the action of the bot unambiguously and learns the policy that maps the pairs to the output actions. Reinforcement learning algorithms such as Markov Decision Processes (MDPs) and Partially Observable Markov Decision Processes (POMDPs) were applied to learn optimal dialogue strategies through interaction with users (Young et al., 2013). These approaches enabled more adaptive and data-driven dialogue management but required significant amounts of training data and computational resources. Additionally, Li et al. (2017) proposed a system leveraging end-to-end reinforcement learning approach for both DM and NLU. The system uses a combined intent-entity model, as described earlier in this chapter. Next, DM uses the output from NLU to construct a query to the knowledge base and to determine the next action. NLG is a part of the user simulator, (Schatzmann et al., 2007) which uses both template-based and sequence-to-sequence generated utterances. The actual DM is represented as Deep Q-network (DQN) (Mnih et al., 2015). Instead of the user simulator, the labeled data for supervised learning can be used as proposed by Williams et al. (2017).

Recent advancements in neural networks driven mainly by transformer-based architecture have also impacted dialogue management. End-to-end neural dialogue systems use neural networks to directly map user inputs to system responses without explicitly modeling dialogue states (Serban et al., 2017). Sequence-to-sequence models and attention mechanisms have been employed to learn effective dialogue policies from raw conversational data. These approaches offer improved performance and scalability compared to traditional methods, especially in handling complex and diverse dialogue scenarios (Bordes et al., 2016).

2.7.7 Recommendation

The recommendation in dialogue systems refers to the process of suggesting items or actions to a user based on their past behavior or preferences. This can be used to personalize the user experience and make it easier for them to find a suitable topic of discussion. It is very important in open-domain dialogue systems as the users may not be aware of the system's capabilities and, therefore, struggle with proactive suggesting of new topics. In a dialogue system, the recommendation can be integrated as a component that provides suggestions at the beginning of the conversation and during the conversation when a particular topic ends and a new one needs to be suggested.

There are several types of recommendation algorithms that can be used in dialogue systems. Content-based (Lops et al., 2011) is one of the types that is based on the characteristics of the items being recommended. It recommends items or actions that are similar to those that a user has liked or interacted with in the past. In dialogue systems, this approach can be used to recommend responses or topics based on the content of the current conversation and the user's long-term profile.

Another approach is Collaborative filtering (Resnick and Varian, 1997) that recommends items or actions based on the preferences of similar users. In dialogue systems, this approach can be applied to recommend responses or topics based on the users with similar values storied in their profile or with similar conversational histories.

The additional approaches include a hybrid recommendation (Burke, 2002) where both content-based and collaborative filtering are leveraged. Also, additional values that are relevant for the current context (Adomavicius and Tuzhilin, 2010) rather than for the long-term user profile are useful in the dialogue systems since this contextual information may change rapidly over time. This includes the emotional state of the user, current topic, sentiment of the utterances, or even location.

2.7.8 Natural Language Generation

Natural language generation is one of the final components in a modular dialogue system pipeline. It is responsible for forming a natural language response given the output of previous components, especially the dialogue management output. Given the finite number of the possible DM actions, the straightforward approach is to use a predefined set of templates for each action. Each template contains placeholders, which are eventually replaced by entity or slot values. There can be multiple templates for a single action to mitigate repetitiveness that can be chosen randomly or based on the similarity with the previous context.

To avoid handcrafted responses, the Sequence–to–Sequence (Seq2Seq) models (Cho et al., 2014) were proposed. The input sequence consists of an action name and a list of entity type and entity value (slot name, slot value) pairs. The input is decoded using an LSTM decoder (or another neural network architecture such as transformer-based), which outputs the hidden representation of the whole utterance. The decoder is then initialized with the last hidden state of the encoder, and based on the previously generated token and the previous hidden state, it generates the output. To incorporate the information about the whole input in every decoding step, the attention mechanism (Bahdanau et al., 2014) is used.

The improvements proposed by Dušek and Jurčíček (2016) use the contextual utterances to avoid repetitiveness and generate the sentences in a way that reflects the previous statements. The first approach is to simply concatenate previous user utterances with the intent-slot-value representation. The second approach requires a different decoder for the contextual utterance. The results of both of the decoders are then concatenated. The last proposed approach is to use a ranker for the k-best-generated results. The ranker gives a higher rank to the utterances based on the common n-gram with the preceding user utterance. This approach selects more natural responses as they use a similar structure to the previous sentence.

The transformer architecture (Vaswani et al., 2017) contributes to the NLG significantly and more impactfully compared to the other fields of NLP. The extensive pre-training allowed the models to produce meaningful responses only with data-efficient fine-tuning. One of the models is Text-to-Text Transfer Transformer (T5) (Raffel et al., 2020). It is a versatile transformer-based encoder–decoder model that frames all NLP tasks as text-to-text tasks. In the NLG segment, T5 can be trained in a text-to-text manner where the input text includes the dialogue history and a special token indicating the system's turn, and the output text is the generated response.

Generative Pre-trained Transformer (GPT) architecture (Radford et al., 2018) is undoubtedly one of the most popular model architectures used by a wide public. It used decoder-only transformer architecture, and its later versions (Brown et al., 2020; OpenAI, 2022) have shown an impressive performance in response generation tasks. However, the response generation is not focused on conversational systems in plain GPT models. Therefore, focused models such as (Zhang et al., 2019) have emerged. It is trained on conversational data and optimized for generating human-like responses in dialogue scenarios and fine-tuned with additional dialogue-specific objectives to improve response quality and coherence.

2.7.9 Retrieval Augmented Generation

Even though Neural response generation based on the machine learning model has progressed significantly, as described in the previous subsection, they tend to generate non-specific and sometimes non-relevant responses. The models themselves can only use the information included in the training data. It also depends on the model's capacity and whether it can hold all the relevant information.

Retrieval Augmented Generation (RAG) (Gao et al., 2023b) combines the retrieval approach with response generation in a way the input for the generative model is enriched via retrieved information. The information may be retrieved using various methods such as keyword matching or full-text search, but the most popular is a semantic search method (Guha et al., 2003). The semantic search is strong in finding semantically similar chunks of text but cannot extract the exact information. Additionally, it may fail to distinguish between semantically close chunks of text. To mitigate the issue, multiple matched candidates are propagated into to generation step. The generative model then extracts the relevant information. This preselection is important as the information source may be large and cannot be fitted into the input of the model.

Chapter 3

Natual Language Processing Algorithms

In this chapter, we describe the machine learning algorithms from the Natural Language Processing (NLP) field that are relevant to the conversational AI described in this thesis. As the thesis is focused on Natural Language Understanding, we will describe the algorithms from that part. However, many of the modern model architectures can be effectively used across NLP fields, and especially models primarily used for Natural Language Generation (NLG) can be used for understanding as well with an appropriate mapping of the results. Even the Large Language Models (LLMs) are among the most performing model types recently, we also focus on the other model architectures and approaches since the balance between the model's resource consumption and the actual performance is critical for this thesis and for the production use of the conversational systems.

The overview outline of this chapter is as follows:

1. *Text Representation* – a method for converting text data into sparse or dense vector formats. Typically used for similarity measuring or as an input for the bigger model.

2. *Traditional Machine Learning* – non-neural models such as logistic regression or SVM that typically take the vector representation of the text as an input. Suitable for use cases where training and inference time are critical.

3. *Neural Networks* – feed-forward, recurrent, convolutional, or transformer-based model, using the text representation or representing the text inside the model.

4. *Large Language Models* – transformed-based models with millions or billions of parameters with extensive pretraining.

3.1 Text Represenation

Text representation in machine learning involves converting raw text data into numerical vectors or matrices that can be processed by machine learning algorithms. This process, also known as feature extraction or vectorization, aims to capture the semantic and syntactic information present in the text while preserving its underlying structure. However, some of the original methods are not designed to capture the semantics but work with the representation only on a vocabulary level. Common techniques are described in subsection 3.1.2.

3.1.1 Tokenization

Tokenization is a intial task in NLP that involves breaking down a text (a sequence of character) into smaller units, typically words, subwords, or individual characters, known as tokens. The set of tokens then for a vocabulary and each token can be represented as an index in the vocabulary.

In word-level tokenization, the input text is segmented into individual words based on whitespace or punctuation boundaries. However, word-level tokenization may not be sufficient for inflection languages as the the resulted vocabulary size would be to high. Subword tokenization addresses this limitation by breaking down words into smaller linguistic units, such as prefixes, suffixes, or stems. This approach enables more effective handling of out-of-vocabulary words and improves the generalization capability of NLP models. Techniques like Byte Pair Encoding (BPE) (Gage, 1994) and WordPiece (Wu et al., 2016) are commonly used for subword tokenization. Character-level tokenization, on the other hand, segments the input text into individual characters. The word or sub-word representation need to be learn as part of the model leveraging this input type.

3.1.2 Represeantation Methods

1. One-hot encoding: This method creates a binary vector for each unique word in the text data, with a ‘1’ in the position of the corresponding word in the vocabulary and a ‘0’ in all other positions. This method does not capture the relationships between words or the semantic meaning of the text.

2. Bag of Words: Similarly to the one-hot encoding, BOW is a vector with a length of the size of the vocabulary. Each position contains a number of occurrences of the corresponding word in the input text.

3. Term frequency-inverse document frequency (TF-IDF): This method represents text as a numerical vector based on the frequency of each word in the text data. The TF-IDF value for a word is calculated as the product of its term frequency (TF) and the inverse document frequency (IDF), which measures the rarity of the word across the entire text corpus.

4. Word embeddings: This method represents words as dense, continuous-valued vectors in a high-dimensional space. The vectors are trained on large text corpora using techniques such as neural language models, and they capture the semantic relationships between words and their context in the text data.

5. Sentence embeddings: This method represents entire sentences or documents as a single vector, capturing the overall meaning and context of the text data. Sentence embeddings are often created by averaging the word embeddings of the words in the sentence or by using advanced techniques such as transformers.

In summary, text representation in machine learning is an important step in converting raw text data into a format that can be used for training and evaluating machine learning models. The choice of text representation method depends on the specific requirements of the machine learning task and the resources available for processing the text data.

3.1.3 Text Embeddings

Word embeddings are a type of representation for words used in natural language processing tasks. They map words to dense, continuous vectors of numbers, allowing them to capture the meaning and semantic relationships between words. In contrast to one-hot encoded representations, which assign a unique binary vector to each word, word embeddings provide a compact, dense representation of words that can capture both their syntactic and semantic relationships. Sentence embeddings have the same goal but on the scope of a whole sentence.

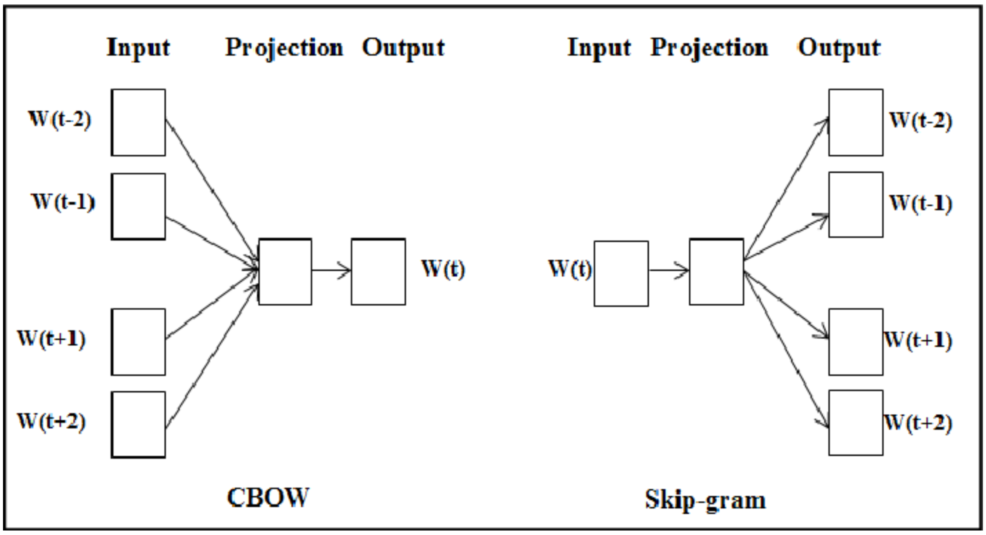


Figure 3.1: CBOW and Skip-gram architectures (Suleiman et al., 2017)

Word2Vec is a popular word embedding technique that was introduced by Mikolov et al. (2013a). It uses a shallow neural network to learn the relationships between words in a large corpus of text. The network takes a word as input and predicts the surrounding context words, allowing it to capture the meaning and context of a word. Word2Vec can be trained using two methods, Continuous Bag of Words (CBOW) and Skip-Gram, each with its own strengths and weaknesses.

Global Vectors for Word Representation (GloVe) is another popular word embedding technique that was introduced by researchers at Stanford University (Pennington et al., 2014). It combines the advantages of Word2Vec with a global log-bilinear regression model that considers both word-word co-occurrence statistics and word context information. This results in embeddings that capture not just local context but also the global semantic relationships between words.

FastText is a word embedding technique introduced by Mikolov et al. (2018). It is designed to handle out-of-vocabulary words, which are words that are not present in the training data, by breaking words down into subword units or characters. This allows FastText to handle morphologically rich languages, such as those that have many affixes, and also handle rare or unknown words. FastText operates by training a simple neural network to predict the probability of a word given its surrounding context words, much like Word2Vec. The main difference is that FastText uses character n-grams in addition to whole words as input to the network, allowing it to better handle rare or unseen words.

Embeddings from Language Models (ELMo) is a more recent embedding technique that was introduced by researchers at the Allen Institute for Artificial Intelligence (Peters et al., 2018). ELMo uses deep bidirectional language models to generate context-sensitive representations of words. The approach involves fine-tuning a pre-trained language model on a specific task, allowing the embeddings to be tailored to the task at hand and to capture task-specific information.

Universal Sentence Encoder (USE) (Cer et al., 2018) is a sentence embedding with a focus on transfer learning to other NLP tasks. It is introduced in two variants: Deep Average Network (DAN) and transformer-based (TRAN). Two variants of the encoding models allow for trade-offs between accuracy and compute resources.

In addition to the native sentence embeddings such as USE, the popular method for representing a whole sentence is to use an average or TF-IDF-weighted average of individual word vectors.

3.2 Semantic Text Similarity

Semantic text similarity is the task of determining the degree of similarity between two pieces of text based on their meaning, as opposed to their surface-level features such as word count or word overlap. There are different approaches to measure semantic similarity based on the selection of text representation and a similarity function. The typical text representation for semantic text similarity is the usage of text embeddings, as described in the previous section.

One of the popular similarity functions used along text embeddings is the cosine similarity.

$$\cos(\theta) = \frac{\mathbf{w_1} \cdot \mathbf{w_2}}{||\mathbf{w_1}||\,||\mathbf{w_2}||}$$

Alternatively, neural network-based models such as Siamese networks or Transformer models can also be used to measure semantic similarity by comparing the representations of the two texts generated by the models. The neural network can be used to compute just the similarity using the text embeddings or combine the learning of the text representation with the actual similarity estimation. When using a neural network to encode a text input, we can use two approaches: Bi-encoder and Cross-encoder. The former encodes each text in a pair by an individual or shared encoder and produces two representations. The latter encodes the pair as a whole, resulting in a single representation. While the cross-encoder may achieve better accuracy in certain scenarios, the bi-encoder is suitable for scenarios where the set of representatives should be encoded during the system initialization, and then the already encoded representations can be compared to the query at the runtime. This saves the computation resources for creating the representations of a rarely changing set of representatives.

3.3 Traditional Machine Learning

Traditional machine learning in NLP typically involves the use of statistical and probabilistic models, as well as feature engineering techniques, to process and analyze text data. These methods often rely on handcrafted features extracted from text, such as one-hot encodings, Bag of Words, n-grams, part-of-speech tags, or syntactic patterns. Common algorithms used in traditional NLP include Support Vector Machines (SVM), Naive Bayes classifiers, decision trees, and logistic regression. While these approaches have been successful in many NLP tasks, they often require extensive feature engineering and domain-specific knowledge and may struggle to capture complex linguistic patterns and semantics present in natural language. However, when combined with a robust enough text representation using pre-trained models on large corpora of a language, they may form a strong baseline that is significantly more computationally effective compared to the complex neural networks. The resource effectiveness is beneficial for real-time applications such as chatbots or voice bots. Additionally, the training time is significantly shorter, allowing the dialogue creators to prototype rapidly.

3.4 Convolutional Neural Networks

Convolutional neural networks (CNNs) are a type of artificial neural network that were originally developed for image processing tasks. However, they have also been adapted and applied to a wide range of NLP tasks (Kim, 2014).

In NLP, CNNs are used to process and analyze text data by treating it as a 2D grid of words, where each word is represented by a vector. The convolutional layer applies a convolution operation to the input data, which detects local patterns or features in the text data. The pooling layer reduces the dimensionality of the data by aggregating the output

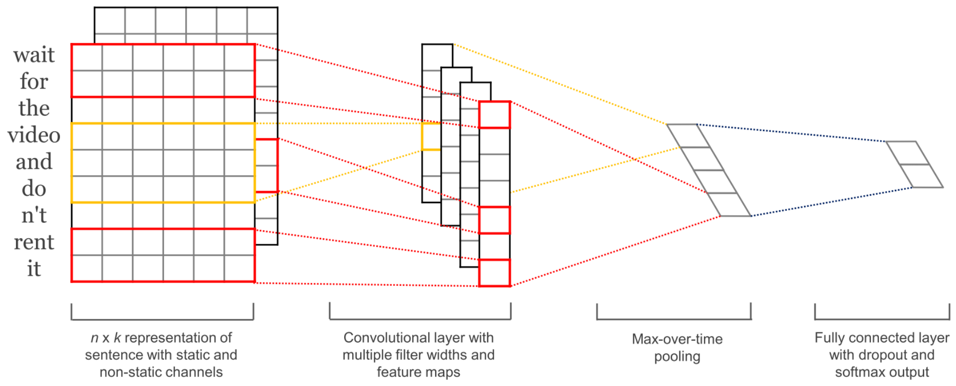


Figure 3.2: CNN architecture for a text classification task (Kim, 2014)

of the convolutional layer over a small window. The fully connected layer then combines the output of the pooling layer and produces the final output of the CNN.

One of the key advantages of CNNs in NLP is their ability to learn hierarchical representations of text data. The convolutional layers learn local patterns or features in the text, while the deeper layers learn more global patterns or features. This allows CNNs to achieve high accuracy in tasks such as text classification, sentiment analysis, and named entity recognition.

Overall, CNNs have been successfully applied to a wide range of NLP tasks and have shown to be a powerful tool for understanding and analyzing text data. However, they have recently been replaced by more effective transformer-based models.

3.5 Recurrent Neural Networks

Recurrent neural networks (RNNs) are a type of artificial neural network that are particularly well-suited for processing sequential data, such as text or speech. They are able to handle variable-length input sequences and maintain a hidden state that captures information about the past elements of the sequence.

In NLP, RNNs are used to process and analyze text data by treating it as a sequence of words, where each word is represented by a vector. The RNN processes each word in the sequence one at a time, updating its hidden state at each step. The hidden state captures information about the past words in the sequence and allows the RNN to understand the context of the current word.

One of the key advantages of RNNs is their ability to handle variable-length input text and maintain context across the entire sequence. This makes them well-suited for tasks such as language modeling, understanding, and text generation, where the input text can vary greatly in length, and context is important to understand the meaning of the text.

Additionally, their ability to handle sequential data and maintain the order of the words is also important to understanding the meaning of the text.

There are different types of RNNs that have been used in NLP such as the standard RNN, Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) (Cho et al., 2014), which are designed to overcome the vanishing and exploding gradients problem of the standard RNNs.

3.5.1 LSTM

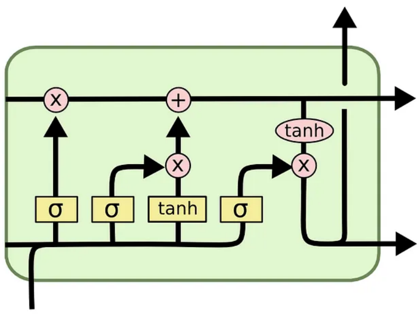


Figure 3.3: Diagram of an LSTM cell structure (Olah, 2015)

Long Short-Term Memory (LSTM) is a type of Recurrent neural network (RNN) used in deep learning, which was designed to address the vanishing gradients problem that occurs in traditional RNNs (Hochreiter and Schmidhuber, 1997). This problem occurs when the network tries to propagate gradients over a long sequence, causing the gradients to become very small and disappear. LSTM solves this problem by using a memory cell that can store information over a long period of time and gates that control the flow of information into and out of the memory cell.

LSTMs are widely used in various tasks (Ravuri and Stolcke, 2015; Huang et al., 2015; Kurata et al., 2016; Hakkani-Tür et al., 2016). They can handle input sequences of varying lengths and learn to recognize patterns and dependencies in sequential data. They have also been used in dialogue systems, where the LSTM network is used to encode the user's inputs, generate context-aware responses, or perform a classification.

Their sequential nature, however, does not allow a proper parallelization during the training and inference time in multi-GPU environments. Besides other improvements, this shortcoming is addressed in the recent transformer architecture.

3.6 Transformers

Transformers are a very popular type of neural network architecture that was introduced in the paper "Attention is All You Need" by Vaswani et al. (2017). They have been widely adopted in NLP tasks due to their ability to handle long-range dependencies and process sequential data efficiently.

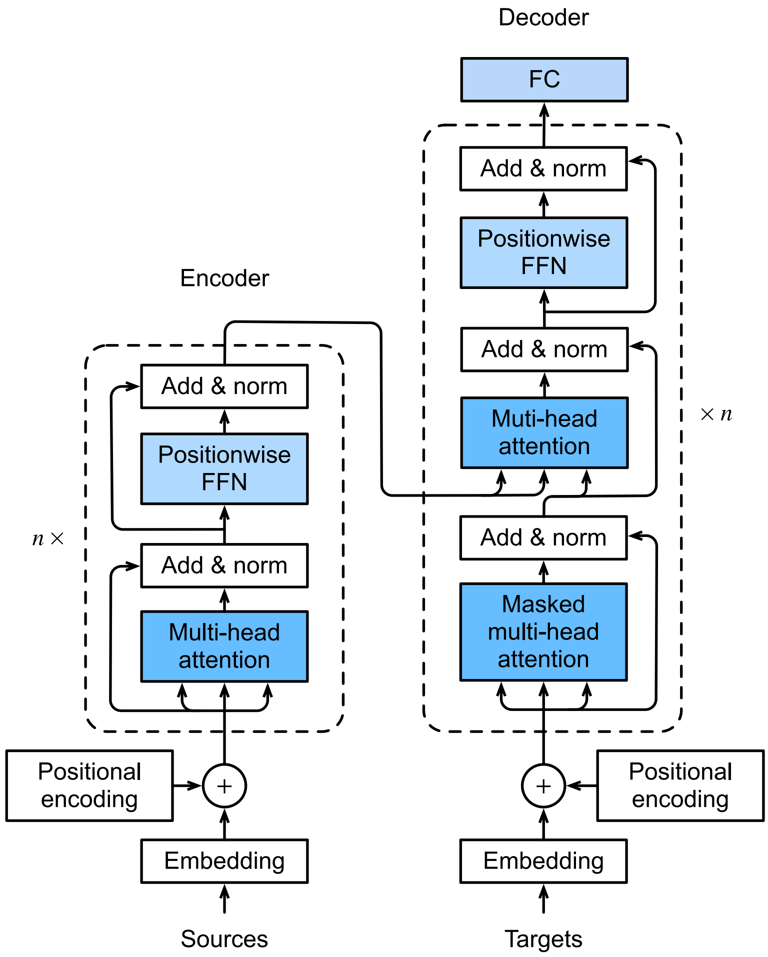


Figure 3.4: The transformer block schema (Zhang et al., 2023)

Transformers are based on the concept of self-attention, which allows the model to weigh different parts of the input sequence differently when making predictions. This allows the model to focus on the most relevant parts of the input for a given task.

An original transformer model is composed of an encoder and a decoder, where the encoder processes the input sequence, and the decoder generates the output sequence. The encoder is composed of multiple layers, each containing a multi-head self-attention mechanism and a feedforward neural network. The decoder is similar to the encoder but also includes a masked self-attention mechanism to prevent it from seeing future tokens in the output sequence.

One of the key advantages of transformers in NLP is their ability to handle long-range dependencies and process sequential data efficiently. Another advantage of transformers is their ability to handle large amounts of data and can be easily parallelized across multiple GPUs, which allows them to achieve state-of-the-art results on a wide range of NLP tasks.

Popular transformer models include BERT (Devlin et al., 2019), GPT-2 (Radford et al., 2019), and GPT-3 (Brown et al., 2020), which have been applied to a wide range of NLP tasks and have shown to be a powerful tool for understanding and generating text.

3.6.1 Generative Transformers

Generative transformers, as a subset of all transformer-based models, are models focused on generating natural language text as their output. They are often considered zero-shot learners because they can perform language tasks without any explicit supervision or training data for the specific task at hand. The typical representatives are Generative Pre-trained Transformers (GPTs) models.

Generative transformers are trained on large amounts of unlabeled text data. This pre-training allows them to learn the patterns and structure of language, which can be fine-tuned for specific NLU tasks, such as question answering, text classification, and sentiment analysis.

In NLP tasks, generative transformers can be used to make predictions or generate output without having seen any examples of the specific task during training Brown et al. (2020). This is because they have learned a rich representation of the patterns and structure of language through pre-training. As a result, they can be used in zero-shot learning settings, where the task and its associated outputs are unknown during training.

3.6.2 GPT Models

Generative Pre-trained Transformer (GPT) models are a series of advanced NLP models developed by OpenAI (Radford et al., 2018, 2019; Brown et al., 2020; OpenAI, 2022). These models are built upon the transformer architecture using only the decoder part and are designed to generate human-like text based on the context provided to them.

They use the unmodified transformer decoder blocks, including self-attention mechanisms, to capture long-range dependencies in sequential data efficiently. In the GPT architecture, the model consists of a stack of multiple layers of transformer decoder blocks, where each block includes a multi-head self-attention mechanism, followed by position-wise feedforward networks. However, unlike the transformer used in tasks like machine translation, GPT is an autoregressive model, meaning it generates text sequentially, one token at a time, from left to right.

During the pre-training phase, GPT models are trained on vast amounts of unlabeled text data. During this phase, the model learns to predict the next word in a sentence, capturing the structure and semantics of the language. After pre-training, the model can be fine-tuned on specific tasks or domains with labeled data to adapt its learned representations for more targeted applications such as conversational systems. Additionally, as introduced in ChatGPT-3.5 (OpenAI, 2022), reinforcement learning is used to fine-tune the models using the rewarded responses.

GPT models have evolved throughout time, and they have changed, especially in the number of parameters:

- GPT-1 (Radford et al., 2018) – The first iteration had 117 million parameters and laid the foundation for subsequent models.
- GPT-2 (Radford et al., 2019) – Significantly larger with 1.5 billion parameters. It introduced the ability for fine-tuning and demonstrated impressive language generation capabilities.
- GPT-3 (Brown et al., 2020) – The largest and most powerful model in the series with a staggering 175 billion parameters. GPT-3 showcased unprecedented language understanding, zero-shot learning, and versatile application across various NLP tasks.
- ChatGPT-3.5 – It is assumed to have the same number of parameters as GPT-3. It has introduced reinforcement learning fine-tuning. Thanks to the public availability, it extremely popularized the whole GPT model family.
- GPT-4 – The most powerful GPT model. The number of parameters has not been revealed. However, the community estimates it to be over 1 trillion.

GPT-3 introduced the concept of zero-shot learning, enabling the model to perform certain tasks without explicit training for them. This capability demonstrates the generalization and adaptability of the model to diverse tasks. This contributes to the versatility of the model usage in various NLP applications, including language translation, summarization, question-answering, text completion, and more. Its ability to generate coherent and contextually relevant text has found applications in creative writing, content generation, and conversational agents.

OpenAI has adopted a strategy of progressively increasing model sizes, leveraging larger datasets, and refining training techniques to enhance the performance and capabilities of each GPT iteration. In summary, GPT models represent a groundbreaking advancement in the field of NLP, showcasing the effectiveness of pre-training on massive datasets and transformer architecture.

3.6.3 Prompt-Based Learning

Prompt-based learning is a machine learning paradigm that involves training models to perform specific tasks using structured prompts or instructions provided as input. In this approach, instead of training models on large amounts of labeled data, models are trained to generate outputs conditioned on predefined prompts or cues. The prompts serve as guidance for the model to produce desired outputs, enabling targeted learning for specific tasks. The GPTs are typical representatives of such models.

Prompt-based learning is particularly effective for tasks where labeled data is scarce or expensive to obtain. By providing explicit instructions or examples through prompts, models can learn to generalize to new inputs and produce accurate outputs even with limited training data. Additionally, prompt-based learning allows for fine-grained control over model behavior and output, as the prompts can be tailored to emphasize certain aspects or criteria of the task.

This approach has been successfully applied in various natural language processing tasks, such as text classification, question answering, and language generation. For example, in text classification, models can be trained to classify documents into predefined categories based on prompts that specify the desired classification criteria or features of interest. Similarly, in question answering, models can be trained to generate answers to questions by providing prompts that guide the model to focus on relevant context or information.

Chapter **4**

Effective Hybrid Architecture of a Conversational Socialbot

This chapter describes a novel approach to an open-domain conversational system with an overall goal to conduct a coherent and engaging conversation with a user. The main non-functional requirement is the resource efficiency of the system's runtime. The system is not oriented on a particular domain, and it should be able to respond to various user inputs for an arbitrary topic. This type of conversational system has been called *socialbots* since the first annual Alexa Prize Socialbot competition (Ram et al., 2017). The name *socialbot* stands for a *social chatbot* as an illustration of the open-domain "social" conversation between a user and the bot.

The novel approaches described in this chapter include the evolution of four versions of the system called Alquist from the first four years of the Alexa Prize competition. Each version of the system leverages strong parts of the previous ones and builds on top of it. All the systems shared the main aspect of modular architecture instead of a monolithic single model. The modular architecture allowed the usage of external information sources as well as the rapid development of the conversational strategy and content.

The chapter is organized as follows. First, we describe the concept of socialbots as part of the Alexa Prize competitions. Then, we describe the versions of the Alquist socialbot with an emphasis on understanding natural language. We will evaluate the relevant individual parts of the system on the corresponding datasets. Moreover, the overall system performance will be human-evaluated using a large number of evaluators. Finally, we will summarize the main contributions of the evolved system architecture.

4.1 Alexa Prize Socialbot Grand Challenge

The Alexa Prize (Ram et al., 2017) is an annual competition organized by Amazon to encourage research in the field of conversational AI and to advance the development of socialbots. The term socialbot, first introduced during the first year of the competition, became widely used for conversational systems that can conduct an engaging, coherent, and human-like conversation with users on an arbitrary topic. Coherence and engagingness are two critical points of the conversation. The recent models trained on a large amount of data are capable of a coherent conversation without a low effort. The engagingness, on the other hand, typically requires proper dialogue management and access to the up-to-date information that needs to be incorporated into the conversation based on the context of the conversation.

Despite the competitive character of the Alexa Prize and awards for the first three places plus the Grand prize for the average time of the final conversations being higher than 20 minutes, the main benefit of developing the system as part of the competition is the huge number of users evaluating the system. The evaluation corresponds with the competition assignment, i.e., the users evaluate the relevance, coherence, and engagingness of the conversation by giving one to five stars at the end of the conversation.

The Alexa Prize Socialbot Grand Challenge is a multi-stage competition where a different group of users evaluate the competing bot during each stage. The socialbots are tested in real-world interactions, first with a limited set of selected users, then with various Amazon Echo users, and finally with a handful of trained judges.

The Alexa Prize has become a major event in the field of conversational AI, attracting participants from around the world and inspiring new research and innovation in the development of socialbots. The competition has contributed to the advancement of the field and has helped to create more natural and engaging conversational experiences for users. This thesis focuses on the first four years of the competition. First, we describe the differences over the years, and then we focus on the evolution of the underlying Alquist technology.

4.1.1 Alexa Prize Socialbot Grand Challenge 1

In late 2016, conversational AI started gaining popularity among technological companies as the research had been pushing the capabilities further. However, the limitations were still significant, especially in the field of context-aware or longer conversations. The first Alexa Prize competition (Ram et al., 2017) aimed to address this challenge and advance conversational AI, gaining the university research teams access to large-scale data and real-world feedback, which is crucial to the effective development and evaluation of a socialbot. Teams were also provided with live news feeds to enable their socialbots to stay current with popular topics and news events that users might want to talk about.

4.1.2 Alexa Prize Socialbot Grand Challenge 2

The second year of the competition introduced several modifications to help the research teams focus on the relevant research. The Alexa Prize team introduced a CoBot toolkit (Khatri et al., 2018b,a), allowing both new teams as well as teams participating in the previous year the seamless integration to the Amazon Skills Kit (ASK) infrastructure. Additionally, the toolkit included several out-of-box models such as Topic and Dialog Act Classifiers, Conversation Evaluators, and Sensitive Content Detector. As Automatic Speech Recognition (ASR) is provided via the ASK infrastructure to the teams, the significant improvement was the release of the new ASR model with a 25% relative improvement in Word Error Rate (WER). Additionally, the Common Alexa Prize Chats (CAPC) dataset, containing turns from the Alexa Prize conversations, which was introduced during the first year of the competition, was now available from the beginning, allowing the teams to start training their models based on the data gathered from multiple socialbots.

4.1.3 Alexa Prize Socialbot Grand Challenge 3

Besides the improved CoBot (v2), the direction of the third year of the Alexa Prize competition (Gabriel et al., 2019) was driven by the rising popularity of the generative models GPT (Radford et al., 2018) and GPT-2 (Radford et al., 2019). Amazon provided teams with a model based first on GPT and then on GPT-2 and fine-tuned on the Topical chat dataset created by Gopalakrishnan et al. (2019). The dataset itself was also available to the teams. The model was called Neural Response Generator (NRG) and was available in two versions, the first vanilla fine-tuned models and the second enhanced policy-driven (PD). The response generation process of the PD model can be controlled using dialogue acts, topics, and knowledge to create the most suitable response. Additional contributions included enhanced Topic classification, the Dialogue Act classification models, and the enhanced Topical chat with a simulated ASR error.

4.1.4 Alexa Prize Socialbot Grand Challenge 4

The focus on generative models was the primary topic in the fourth year of the Alexa Prize competition as well (Hu et al., 2021). The improved NRG models were trained on the conversation of the top 5 bots from the previous year. The controlled version of NRG was focused on empathy and topical control. The new NLU models included both Hierarchical Recurrent Neural Network (HCNN) (Gabriel et al., 2019) and BERT (Devlin et al., 2019) models for topic classification, basic intent recognition, and named entity recognition. The evaluation started to be more focused on A/B testing by including experiment traffic that was not impacting teams' ranking.

4.1.5 Other Alexa Grand Challenges

Besides the Alexa Prize Socialbot Grand Challenge, Amazon recently introduced two additional conversational AI competitions, the Taskbot Grand Challenge and Simbot Grand Challenge.

The *Alexa Prize Taskbot Grand Challenge* (Gottardi et al., 2022) encourages research to advance the development of task-oriented dialogue systems. The competition's goal is to build a task-oriented dialogue system that assists humans with real-world Cooking and Do-It-Yourself tasks. The created systems use a multimodal interface that combines voice interaction with visual elements such as lists, text, and images on supported devices.

The multimodal approach was then adopted by Alexa Prize Socialbot Grand Challenge 5, allowing the teams to support the conversation about popular topics via relevant text and images and create a visual identity for the socialbot.

The main criterion of the bot performance evaluation was user satisfaction rating, which was equivalent to the rating in the Socialbot Grand Challenge. Additionally, the completion metric was used that indicate whether a user was able to complete the task.

The *Alexa Prize Simbot Grand Challenge* (Shi et al., 2023) is aimed at building conversational robots that also need to interact with and complete tasks in a (simulated) physical environment. The users interact with a bot via voice, making the bot navigate the environment and manipulate objects.

The evaluation of the bots was inspired by the Taskbot Challenge, including the user satisfaction rating and a mission success rate, which are equivalent to the completion metric. The mission was defined as a set of tasks that the bot needed to accomplish in the environment.

Taskbot and Simbot Challenge, along with the original Socialbot Challenge, have a goal to push boundaries in conversational AI with different goals and in different environments. All three competitions eventually concluded that users benefit from a multimodal interface regardless of the use case. However, the multimodality opens additional research topics, especially the ability to be aware of the other modalities.

4.2 Alquist: A Hybrid Socialbot

The core concept of the Alquist architecture, as we published in Pichl et al. (2017), is to leverage a hybrid design. The hybrid system combining the end-to-end system and tree-based approach allows the system to pursue the engagigness of the most popular topics, leveraging the precise conversational design while allowing the system to respond to various utterances from a wide range of topics. Moreover, the data sources containing the current information about real-world entities can be easily integrated.

We designed a system that is split into smaller tree-based topic-specific dialogues, such as sports, movies, or books. An entertaining dialogue needs to contain specific information about the conversational topic. A tree-based approach and constrained dialogue allow us to work with this information easily. On the other hand, we use the generative and retrieval dialogue implementation once the dialogue diverges from the tree-based topics. The information provided using this approach is not that rich, but the topic coverage is significant.

The high-level system architecture adopts the modular system concepts described in section 2.7. Automatic Speech Recognition (ASR) as an input and Text to Speech (TTS) as an output (shown in Figure 2.1) are provided by the Alexa infrastructure, allowing us to work directly with text input and output. The architecture is divided into two categories: Information Aggregation (offline) and Alquist Pipeline (online). The information Aggregation part gathers information from various sources for later use. It is described in section 4.2.1. Alquist Pipeline is the actual system's core, which accepts the message from the user and generates a response. Alquist Pipeline and its components are described in section 4.2.2.

The high-level information flow through the pipeline is as follows.

1. A user utterance along with the ASR hypotheses are processed by the analysis components (section 4.2.2, left part of the Figure 4.1). In case of a low hypothesis score, the bot may produce a response asking the user to repeat.

2. Dialogue management (middle part of the Figure 4.1) selects one of the following modules: chit-chat, opinion, question answering, or one of the Structured Topic Dialogues (STD) based on the information from the analysis part. The STDs are preferred multi-turn units. If the previous STD is not finished, the DM prioritizes the finishing of the dialogue.

3. In case an STD is triggered, a sub-dialogue manager relevant to the selected STD comes into play. Unless the STD is interrupted by the global DM, the sub-DM selects the most suitable flow within the dialogue.

4. Once the specific module is selected and produces the answer, the answer is normalized by a set of rules and checked for profanities (important for generative and retrieval approaches). Finally, it is passed to the TTS service (external).

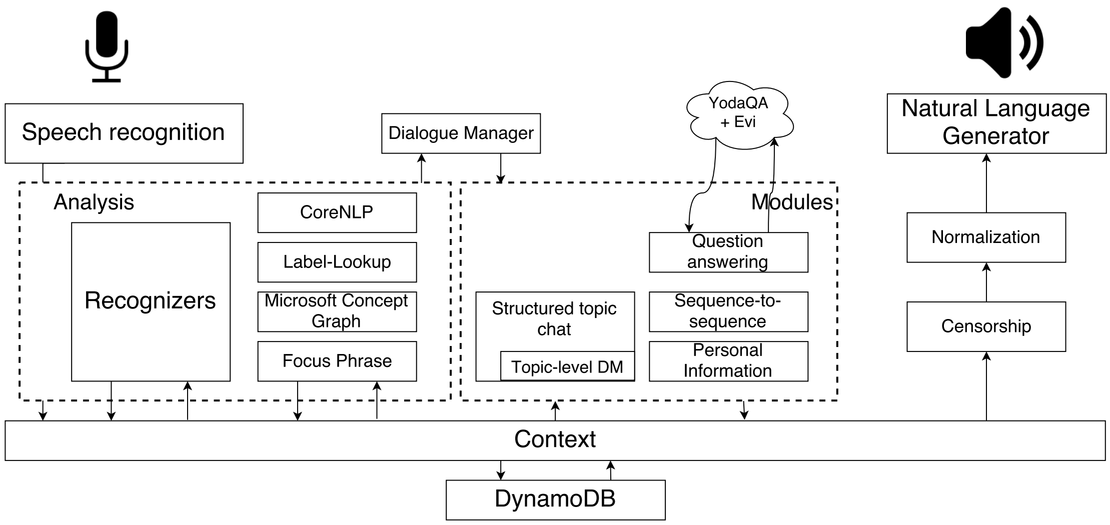


Figure 4.1: The architecture schema of the Alquist: A Hybrid Socialbot

4.2.1 Information Aggregation

One of the important aspects of an engaging conversation is to have access to up-to-date information. For example, referencing the performance of an actor in a movie or discussing the sports match result could be used in respective topics. For that reason, we implemented two main information sources. We take advantage of several types of information sources based on the type of information stored:

- **Knowledge Bases** include the facts that do not change over time, or their values change only occasionally. We obtain this type of information from a Freebase RDF database (Bollacker et al., 2008b).

  We also use the Microsoft Concept graph (Roach, 2016) to recognize concepts belonging to the individual entities in the user's message. We expanded the Concept graph data with additional entries containing movies and games. These entries were obtained from The Movie Database and Internet Game Database, respectively.

  Additionally, we use The Movie Database, Internet Game Database, Goodreads, and Last FM APIs to obtain other movie, game, book, and music-related information respectively[1].

- **Regularly updated information** usually depends on the current events or other time-related aspects. As we want to deliver up-to-date content, we regularly save data from several services. We download daily news articles from The Washington Post and headlines from Today I Learned[2] subreddit, which we present as fun facts in conversations.

The information aggregation pipeline is not part of the runtime component pipeline and runs separately therefore, it is not included in the Component Pipeline section. However, the results of the Information Aggregation are persisted and used by individual pipeline components.

4.2.2 Component Pipeline

The pipeline consists of several interconnected components. It was designed to be highly modular, allowing us to swap individual components and test different approaches quickly.

The core Alquist pipeline was inspired by Apache UIMA (Ferrucci and Lally, 2004). We adopted several design concepts, such as central annotation storage (context), where every component stores its results. We describe the context in detail later in this section.

There are two main types of components in the pipeline: analysis components and response-producing components. The purpose of the analysis components is to extract intent and entities from the user's input. The response-producing components are either generative or retrieval-based neural networks and finite state automata.

The response-producing components are grouped into modules according to the types of responses they produce. We currently employ the following modules: Chit-chat, Question Answering, Structured Topic Dialogues, and Opinion. These modules are selected by the top-level DM based on information extracted during message analysis.

[1] The URLs of the services: www.themoviedb.org, www.igdb.com, www.goodreads.com, www.last.fm

[2] https://www.reddit.com/r/todayilearned/

Context

The context contains all the information relevant to the current session. All modules of the pipeline are storing and accessing information in the context. It holds the ASR hypotheses, NLP annotations, recognized entities, and eventually the final response. The information that needs to be stored for whole sessions or all following sessions can be marked as a session context or long-term context, respectively. It can be used, for example, for storing a user's name or user's favorite topics. The information about a topic and the last state of ongoing Structured Topic Dialogue is also stored in the context. Additionally, for each topic, the last state and discussed entity are stored in the session context.

The context is regularly (after each response) stored in a persistent Dynamo Database. This allows the bot to retrieve the context even if the session was interrupted and to continue with the dialogue.

NLP Analysis

The NLP analysis part of the pipeline uses its components to extract the necessary information from the user's utterance. This information is then used by the dialogue manager and the following parts of the pipeline to produce a response. The analysis components can be generic or topic-specific. Generic analysis components are used for all user's messages. These components tokenize the input, annotate with part-of-speech (POS) tags, and detect entities from the message. Topic-specific analysis components are used only in certain states of Structured Topic Dialogues. An example of such a topic-specific analysis component is specialized entity recognition of book or movie titles.

General analysis components Stanford CoreNLP (Manning et al., 2014) was selected for standard natural language processing (NLP) tasks. The tool provides several annotators. Currently, we use annotators for sentence splitting, tokenization, part-of-speech tagging, dependency parsing, lemmatization, sentiment analysis, and named entity recognition (NER). Additionally, we take advantage of TrueCase annotation, which reconstructs the letter case that was lost by ASR. This is helpful for NER, which has a model trained on sentences with original case.

The entity recognition is implemented using two approaches. The first one is Microsoft Concept Graph (Cheng et al., 2015; Wu et al., 2012), which provides a list of possible concepts for a given entity (for example "Frozen" has the following concepts: film, feature, processed food, ...). We have a snapshot of the Concept graph[3] stored in the Elasticsearch instance. This allows us to query for a specific entity name and retrieve a concept quickly. The second tool is based on Label-lookup[4] which is used in question answering system YodaQA (Baudiš, 2015). Label-lookup uses the cross-wikis (Spitkovsky and Chang, 2012) dataset to match strings together with the nearest Wikipedia concept. Our version of the tool has two modifications. First, we include the Freebase ID in the database and thus

[3] Snapshot was downloaded in March 2017.

[4] `https://github.com/brmson/label-lookup`

eliminate a SPARQL query to DBpedia Lehmann et al. (2015) to retrieve it. Secondly, we deleted all cross-wiki labels with the Levenshtein distance greater than 3 from the canonical label.

Topic-specific Analysis Components The group of topic-specific analysis components consists of the recognizers that are suitable for a specific Structured Topic Dialogue. Each topic has a fixed list of patterns. If the pattern is found in the message, it is stored in the context. They are used, for example, for recognition of sports teams, subjects of news articles, or movie genres.

We also extract a focus phrase from the user's message as candidate words for entity recognition. This procedure is achieved by a combination of several approaches. The first approach selects all words from the input sentence that are marked as a named entity (NE) by the CoreNLP tool. The second approach selects every sequence of consecutive words marked as a noun phrase (NNP) by the POS tagger as a focus phrase, too. Additionally, a heuristic method based on the dependency parser output selects the lowest noun modification (`nmod`) of the sentence root and its corresponding adjective modification (`amod`). We also trained Conditional Random Fields (CRF) sequence labeling using data that were generated using described heuristics and manually corrected. The results of all approaches are combined, and the duplicates are removed.

ASR Confidence and Profanity Filtering We compute a speech recognition confidence as follows. We take all ASR hypotheses (up to 5 hypotheses are obtained from the Conversational ASR model provided by ASK) and their corresponding scores and eliminate some of them based on the following procedure.

Algorithm 4.1 Filter Sentences by Token ASR Scores

```
function FilterSentences(sentenceList)
    tokenProbabilities ← GetTokenScores(sentenceList)
    averageProbability ← ComputeAverageScores(tokenProbabilities)
    for each sentence in sentenceList do
        if averageScore(sentence) < threshold then
            Remove sentence from sentenceList
        end if
    end for
    if no sentences remain in sentenceList then
        Ask user to repeat the message
    end if
    return sentenceList
end function
```

The profanity filtering is done simply by searching for banned words from a list. If a word from the list is found in a user's message, the bot refuses to talk about that topic.

Dialogue manager

Dialogue management is divided into two levels.

1. **Top-level DM** decides which module should be executed (chit-chat, question answering, Structured Topic Dialogue, etc.) and additionally which topic should be executed in Structured Topic Dialogue (sports, movies, etc.). We experimented with several approaches (as described in section 4.2.3). In addition to a predicted topic, we implemented a feature that allows the bot to select a topic according to a mentioned entity. Whenever the Structured Topic Dialogue module is recognized, the bot checks if there is an entity matched to some entry in MS Concept Graph. If there is such an entity, the bot compares its concept to a predefined list of concepts corresponding to each topic. For each topic, the bot sums up the popularity of the entity given each concept from a list. We select a topic with the highest accumulated popularity count.

2. **Topic-level DM** is used in individual Structured Topic Dialogue. It uses an intent recognition-based approach with a mapping of each intent to a particular action. The actions are, for example, selecting a response, saving information into context, or accessing the API. If no action is selected, the system responds with a response generated by the generative network.

The system controls the dialogue flow using a state graph. However, we allow the user to take the initiative. We detect an additional set of intents specific to a given dialogue, and if one of these intents is detected, the system switches to a defined state. This approach allows us to handle a turn-taking conversation with an initiative on both sides. Intent specific to dialogue is detected by embedding a similarity-based method.

Response Generation

Structured Topic Dialogues cover the main interaction with the user. They are designed to cover the most frequent topics and to provide more in-depth interaction and engaging conversation to the user. The system additionally contains generic dialogue about entities from Freebase and dialogues providing easier and more fluent interaction with the bot (help dialogue, initial dialogue, exit dialogue, recommendation dialogue). Since our system contains multiple dialogues, it has to decide which one should be executed. This decision is made by the topic-level DM.

Generative Neural Networks are used for a chit-chat dialogue. Chit-chat does not necessarily require knowledge about a set of entities. This is, for example conversation about the user's (bot's) mood, a question about the day, etc. It is triggered when the chit-chat module is recognized or if no other module generates a response.

We use sequence-to-sequence (Sutskever et al., 2014) network architecture to generate the responses. This framework is commonly used for machine translation but was recently proposed as a conversational model as well (Vinyals and Le, 2015). Both the encoder and decoder are based on the LSTM blocks. The parameters were empirically selected as follows. We set the size of both encoding and decoding vocabulary to 50,000, the number of LSTM cells is 1024 in each layer, and the number of layers to 3. Additionally, we set the batch size to 64 and the learning rate to 0.8. As a training data, we used the discussions from Reddit[5]. We removed messages with special characters and those longer than 20 words from the dataset. The preprocessing step resulted in the dataset of 3,735,209 message-response pairs, and we created a training split consisting of 3M samples. The rest of the samples were used as validation data.

Question answering module is used whenever the user asks a factoid question. These questions are recognized by the module recognizer. The actual process of generating answers is delegated to three systems: YodaQA (Baudiš, 2015; Pichl, 2016), Evi (provided for the competing teams), and a system for answering questions about a news article.

4.2.3 Experiments

With the thesis focus on the NLU part, we report the results of two main experiments. The first one is intent recognition accuracy since it is the most critical NLU component of the system that directly impacts the decision logic in Top-level DM. The second one is the average rating and conversation duration of the individual topics, as these two values are the main criteria of socialbot evaluation.

Intent Recognition

The goal of the intent recognition part of the system is to both maximize the accuracy and minimize the training and inference time.

Results are shown in the table 4.1. Tested methods of intent detection are:

- **TF-IDF:** We use the TF-IDF implementation from the scikit-learn library (Pedregosa et al., 2011). We found the hyper-parameters of TF-IDF by grid search. They are following, analyzer: word, ngram_range: (1,2), max_df: 0.9, min_df: 0.0, norm: l1, smooth_idf: False, sublinear_tf: True. The rest of the parameters have default values.

- **Embeddings:** We use pre-trained GloVe (Pennington et al., 2014) vectors. We convert training examples to vector representation by calculating the average of embeddings of the example's words. We normalize vector representations of training examples to unit length. We do the same for the user's input. We calculate the cosine similarity of the user's input to each training example and use the label of the most similar training example as the recognized intent. We achieved the best accuracy with the 300-dimensional variant of vectors.

[5] https://www.reddit.com/r/datasets/comments/3bxlg7/i_have_every_publicly_available_reddit_comment/

- **Logistic regression:** We use the scikit-learn implementation with default parameters and balanced classes. The input is the vector of word uni- and -bi grams concatenated with the vector of POS uni- and bi-grams.
- **Neural network:** We use GloVe embeddings of dimension $N = 300$ for input words (both message and focus) and a multi-channel convolution (Kim, 2014) with channels of size 1 to 5 for message input and 1 to 3 for focus phrase input. The number of convolutions is set to $N/2$, the activation function is tanh, and max-pooling is over the whole sentence. Since the convolutions of message and focus input are separate layers, the resulting vectors are then concatenated. Additionally, the label predicted for the previous message is concatenated to the vector as well. The concatenated output is then fed into a dense layer with an output dimension equal to 300, followed by dropout with a rate of 0.5, and finally with a softmax-activated dense layer with an output dimension equal to a number of classes.

Table 4.1: Accuracy of intent detection

Text representation	**Classifier**	**Accuracy**
TF-IDF	Cosine similarity	88.3
Glove	Cosine similarity	89.7
Word+POS Ngrams	Logistic regression	92.6
Glove	CNN	**92.7**

Average Rating and Conversation Duration

We selected the average user's rating and average time spent as a quality metric for our Structured Topic Dialogues. The average rating is calculated as follows. We collect user's ratings (one to five stars) of the whole conversation. The rating of the conversation is assigned to all Structured Topic Dialogues that were used in the conversation. We calculate the average of assigned ratings. Time spent in the dialogue and the number of dialogue turns are measured from the start of Structured Topic Dialogue until a different module is recognized by the DM. The average ratings, times, and dialogue turns are presented in the table 4.2.

4.2.4 Conclusion and Next Steps

As the first-in-the-row of the proposed conversational systems, this version of the Alquist socialbot forms a strong baseline for the following versions to be compared with. The proposed hybrid architecture proved to be an effective approach for the rapid development of the social bot. The proposed system implemented as part of the Alexa Prize Socialbot Grand Challenge Competition was awarded as the second-best socialbot. The detailed results are shown in section 4.6. We evaluated the most important part of the NLU

Table 4.2: Average rating, time, and number of dialogue turns of Structured Topic Dialogues

S. T. dialogue	Rating	Time	Turns	S. T. dialogue	Rating	Time	Turns
Books	3.774	83 s	6.151	Jokes	3.931	50 s	4.131
Fun facts	**3.972**	64 s	3.823	Movies	3.669	69 s	6.199
Games	3.828	93 s	**6.979**	Music	3.823	35 s	2.947
General chat	3.616	40 s	2.509	News	3.763	**102 s**	5.655
Gossip	3.761	66 s	4.315	Recommendation	3.575	15 s	1.403
Help	3.509	24 s	1.025	Repeat	2.429	14 s	2.392
Holidays	3.740	26 s	2.495	Sports	3.767	50 s	4.007
Initial chat	3.437	44 s	3.959	Stop	3.682	44 s	2.745

pipeline—intent recognition—and showed the competitiveness of the algorithms that do not demand a lot of computational power. The other parts of the NLU pipeline are subject to the research of the following versions of the system and will be described in the following sections.

4.3 Alquist: Towards Sub-Dialogue Models

The previous version of the system was built using a hybrid approach. The structured part of the hybrid system was divided into individual topics, each represented by a designed dialogue structure. As a next step, we propose a sub-dialogue structure, described in Pichl et al. (2018) and in this section, addressing several major issues of the previous approach. First, the monolithic dialogue structures caused repetitiveness due to limited flow variability, while sub-dialogue structures allow a unique experience in every session. Second, the maintenance of the large structure is demanding with the increasing size of the dialogue. The last important aspect is the reusability of the individual parts of the sub-dialogue structure.

The dialogue structure changes come along with the changes in NLU components. Intent recognition, even though it is still being divided into top-level and sub-dialogue level classification, is significantly improved on the sub-dialogue level side. Additional classifications, such as sentiment detection, dialogue act detection, entity recognition, entity sentiment detection, and topic switch detection, are introduced and discussed later in this section.

4.3.1 Architecture

The system consists of multiple components, which are visualized in Figure 4.2. The architecture focuses on effective information persistence with an emphasis on low-latency access. Moreover, the components orchestrated by the Core are developed with effectiveness in mind while keeping state-of-the-art results. Each component is described in detail in Section 4.3.3.

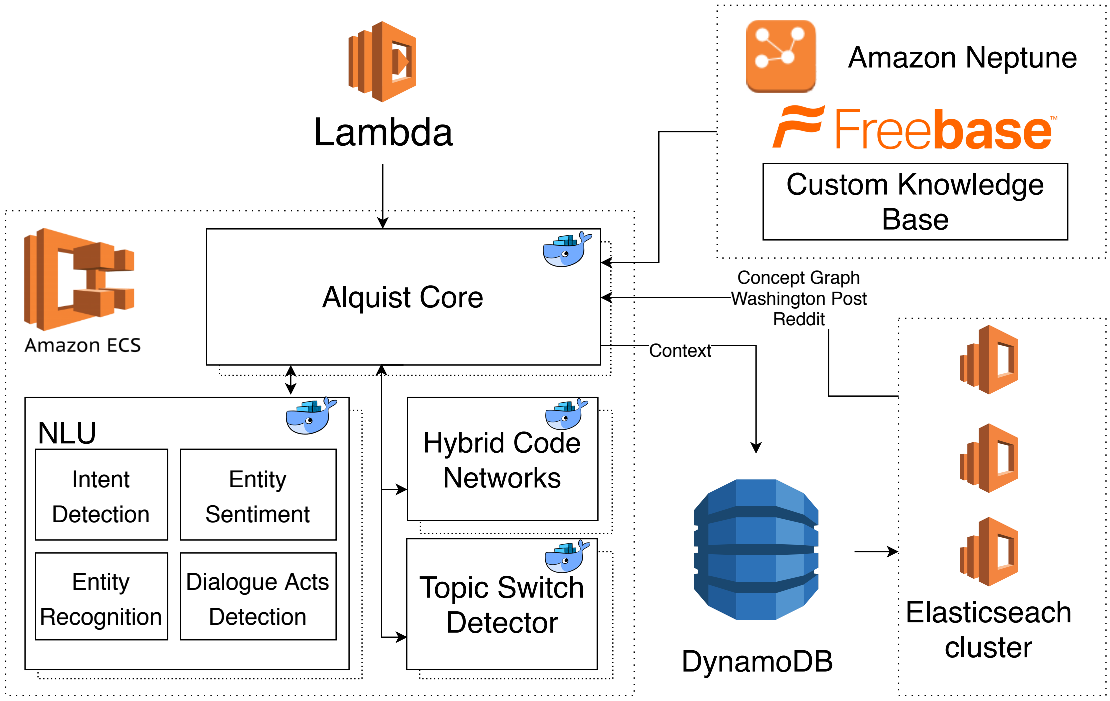


Figure 4.2: Alquist 2.0 architecture schema

The core component serves as an integration layer managing various NLU and other modules distributed across services. The NLU module encapsulates multiple understanding-related models. A hybrid code network is a module on the border of NLU and DM dedicated to each sub-dialogue. Topic switch detector allows seamless transition among individual topics. All of these four modules are standalone containers.

All the information (context) gathered during each dialogue turn is persistently saved in the database. The information includes all of the annotations, messages, generated responses, current topics, current states, etc. Crawled data that are required to be queried using full-text queries are stored in the Elasticsearch database. The crawling sources include Washington Post articles and Reddit content. Additionally, we have a snapshot of Microsoft Concept Graph (Cheng et al., 2015), which is used to determine a type of a given entity.

That last component is an RDF knowledge graph. It contains the last dump of Freebase (Bollacker et al., 2008a) enriched by our custom data.

4.3.2 Information flow

The following list describes the high-level processing logic and the flow of the user utterance through the individual system components.

1. The request from a client in a given format is passed to the Alquist Core.
2. The **Alquist Core** receives the request and based on the session ID, it loads the context (up to 20 previous turns) from the database. If there is no previous turn (the user just started the conversation), a welcome dialogue is triggered with an empty context history.
3. The user message is annotated by all of the models in the **NLU module** plus the Topic Switch Detector is triggered.
4. If there is a **Topic Switch** detected, a new sub-dialogue is selected based on the assigned topic nodes, which are assigned to the current entity and intent combination.
5. If there is not the **Topic Switch** detected, the bot continues with the previous sub-dialogue, or it suggests a new one when the previous one is over.
6. The user message is sent to **Dialogue manager**, which produces a response according to the selected sub-dialogue.
7. Before the pipeline is finished, the bot stores the context in the database.

4.3.3 System components

Knowledge base

To obtain content for the dialogues, we scrape data from various sources. Two examples of those are the Washington Post and Reddit. Reddit posts from the subreddit "Today I learned" are used as trivia about people and other entities. We designed our knowledge base in order to add structure to this data. We chose the RDF-triples format to expand it with information already converted to it, namely a dump of Freebase. Data integration between Freebase and our data (articles from now on) is done as follows: First, each article is searched for named entities. Articles from certain sources already contain this information, but the rest need to be processed with a named entity recognition tool. Once the entities are obtained, they are linked to their Freebase IDs using a query to our implementation of a Freebase fuzzy label lookup[6]. Finally, the articles and Freebase entities are linked using a simple ontology of our own design. The knowledge is periodically updated with new articles, and it can be queried with SPARQL query language. One improvement over Elasticsearch, where we stored the data initially, is the fact that when we look for an article about an entity, we can be sure that it is mentioned there.

Entity recognition

Entity recognition is a task that gives a label for every single word from a given utterance. We use a standard *inside, outside, begin* (IOB) (Ramshaw and Marcus, 1999) schema

[6] https://github.com/brmson/label-lookup

commonly used in Named Entity Recognition (NER) task with the addition of a type of entity (e.g. *B-movie*). The only difference to the traditional NER task is that we do not require the entity to be a strictly named entity that can be mapped, e.g., to a Wikipedia article. For example, in the sentence *I want to talk to you about my life*, the word sequence *"my life"* is marked as an entity. This approach allows the bot to start a dialogue about abstract topics such as *"my life"*. Additionally, we want to recognize a correct entity type from the utterances where it is possible. This type-inference is based on the current utterance thus, it should be possible for the model to recognize it. For example, in the sentence: *I want to talk about Matrix*, our model should label *"Matrix"* as a Generic Entity since it is not possible to know that Matrix is a movie without external knowledge. On the other hand, in the sentence *Let's chat about Matrix movie*, it should label *"Matrix"* as a movie.

For the sequence tagging task, we use the BI-LSTM-CRF (Huang et al., 2015) model. The input is a user utterance where each word is represented by word embedding. The embeddings are followed by a bidirectional LSTM layer, which is then connected to a fully connected layer. The sentence features extracted by the previous layers are then fed into the CRF layer. The number of predicted classes is two times the number of types (one for B and one for I) plus one (for O). As training data, we use a manually labeled dataset of utterances gathered during conversations with real users. We use our own annotation tool to make the process as fast as possible.

Intent detection

Intent detection classifies each utterance into one of the predefined top-level classes. These classes are related to the sub-dialogues that the bot is capable of talking about. The classes are, for example: "tell topic", "change name", "tell news", etc. Detected intent combined with recognized entities is used in deciding which sub-dialogue should be triggered. There is not a strict boundary between entity and intent during the dialogue design. For example, the sentence *Let's talk about rock music* can be labeled with *tell_topic* intent and *rock music* entity or the intent can be *tell_about_music* and entity can be *rock*. Those two approaches are equal as long as we stay consistent. Note that a new sub-dialogue is triggered based on the entity and intent only if the topic switch is detected (based on the contextual data) as described in subsection 4.3.3.

We use multi-channel convolutional neural network (Kim, 2014) as a model for intent detection. The words from input utterances are represented as word embeddings. Embeddings are followed by five channels of convolutions, followed by a max pooling layer. The last two layers are fully connected layers. The output of the model is the probability of each intent class. The model is trained on a dataset, which is a combination of the utterances from real conversations and utterances generated from templates.

Dialogue Act Detection

Dialogue act detection is also the utterance classification task. Unlike intent detection, the dialogue act classes are not related to the specific dialogues, but they can be used across various NLP tasks. The classes describe whether the utterance is a statement, question, acknowledgment, etc. We use the commonly used Switchboard dataset (Godfrey et al., 1992), which is a phone call transcription annotated by dialogue acts. The original dataset contains over 200 classes, which are clustered into 43 classes based on the predefined rules[7]. We trained our model using these 43 classes.

We use the same model architecture as for the intent detection. We do not use predicted dialogue acts directly during the conversations, but we rather use a feature vector, which is an output of the second-to-last fully connected layer. This feature vector is input to the dialogue manager as described in section 4.3.3. However, as reported by Pichl (2018), there is an improvement in dialogue act detection accuracy when the contextual information is incorporated.

Entity sentiment

The entity sentiment is an important task for forming the bot's personality and opinion base. For each detected entity, we search X social network (formerly Twitter) via Twitter-API for recent tweets containing the mentioned entity. We perform sentiment analysis on gathered tweets and compute a mean value from received results. We use this value as a basis for the socialbot's initial opinion of the given entity.

For sentiment detection, we first clean the tweet of non-word strings and tokenize it, then convert the individual tokens to their embedding representation. We use them as an input to a bidirectional recurrent (LSTM) neural network layer. Our model utilizes a dense layer from which we obtain estimated sentiment. The model then determines the sentiment of the tweet on a scale from 0 (negative) to 1 (positive). We have trained our model on two separate datasets: the IMDB sentiment dataset (Maas et al., 2011) and the Twitter sentiment classification dataset (Go et al., 2009).

Topic graph

The topic graph presented in this section reflects the goal of the system, which is to have a conversation about popular topics such as movies, books, sports, etc. However, arbitrary topic node structures can be implemented.

The Topic graph contains the graph structure of topics, sub-dialogues, and their interconnections. We show the structure of topics in our Topic graph in Figure 4.3. It consists of topic nodes and sub-dialogues. Each topic node has assigned one or more sub-dialogues. For example, "Movies" topic node has assigned sub-dialogues about movies in general, like *"Where do you watch movies"*, *"What is your the most favorite movie"* or *"Are you a big movie fan"*. The Movie topic node has assigned sub-dialogues about some specific movie

[7] `https://web.stanford.edu/~jurafsky/ws97/manual.august1.html`

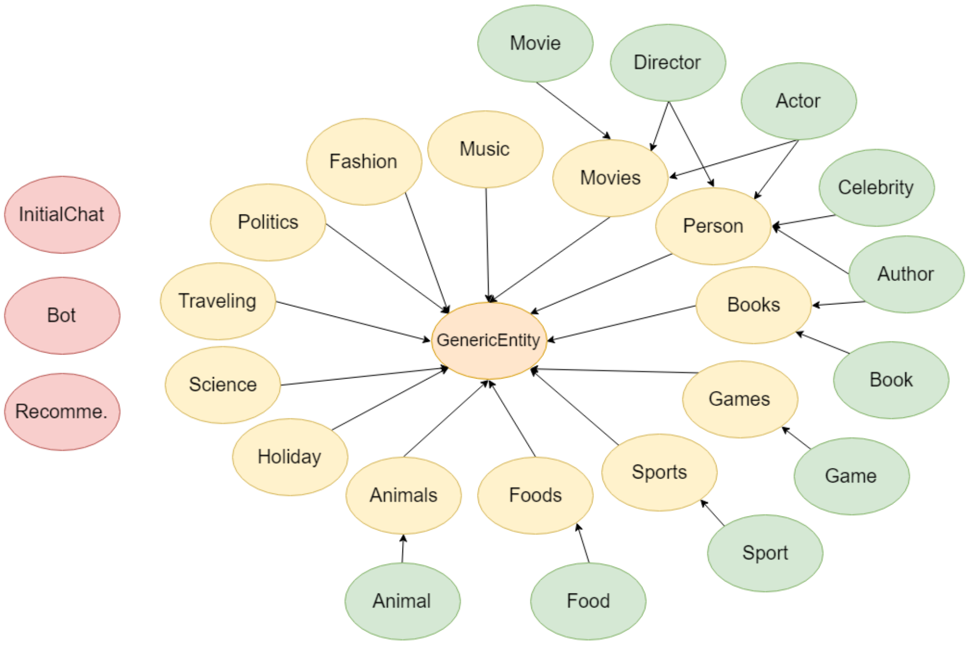


Figure 4.3: Schema of Topic graph. Yellow nodes contain dialogues about a topic. Green nodes contain dialogues about an entity of its type. The orange node GenericEntity contains dialogues about entities of unknown types. Red nodes are special topic nodes containing dialogues about the bot, user, and Initial chat. These nodes are not connected to the rest of the Topic graph.

like *"Actor starring in the movie"*, *"Director of the movie"* or *"What is your favorite part of the movie"*. Each of these sub-dialogues is implemented as a model in Hybrid code networks.

Topic nodes are connected by oriented edges, which point from more specific to less specific topic nodes. There is, e.g., an edge from Movie to Movies, or from Director to Movies and Person. If we detect that the user wants to talk about Steven Spielberg, and we know that he is a director, we can select any sub-dialogue from the Director node and all nodes that we can reach from the Director node (Person, Movies). Although we prefer the sub-dialogues of the Director node or the sub-dialogues close to it. If the selected sub-dialogue ends, the topic graph automatically selects a new sub-dialogue from the Director node or close to it. This method of active selection of new sub-dialogues keeps the user engaged. We have a special node we call "GenericEntity" for all entities of unknown type. It is the least specific topic node. It means that there is no edge from the GenericEntity node to any other node. It contains three sub-dialogues: *"Funfact"*, *"Shower-thought"* and *"News"*. These sub-dialogues do not require any specific knowledge about the entity. They select the content of sub-dialogues by a text search of the entity name. This method allows us to maintain a conversation about any existing entity.

There is also a Recommendation topic node. Its purpose is to suggest new topics of conversation to the users. This node is used only if users themselves do not specify the topic of the conversation or we run out of sub-dialogues for the topic that they requested.

Topic-Level Intent Recognition

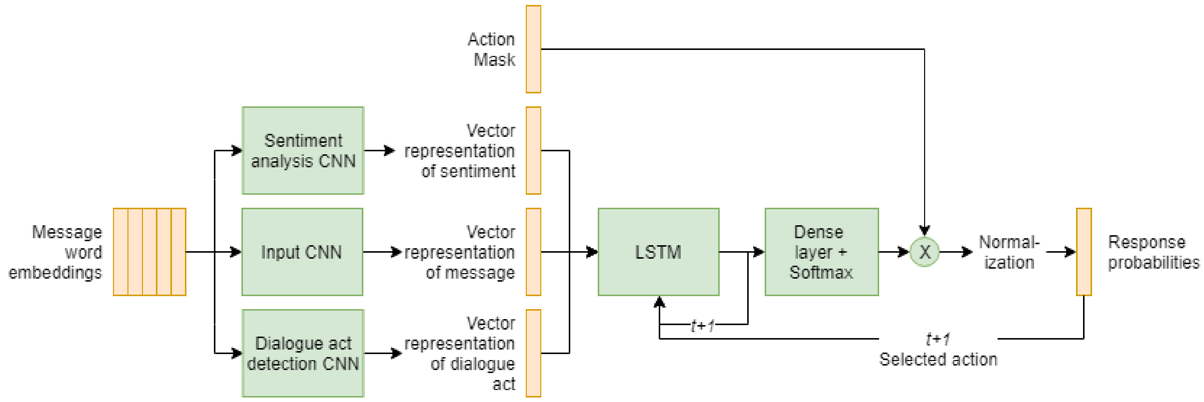


Figure 4.4: Schema of our implementation of Hybrid code networks

We use modified Hybrid code networks (HCN) described in Williams et al. (2017) as a topic-level intent recognition model that is tightly connected to the dialogue manager. HCN combines an RNN with domain-specific knowledge encoded as software and system action templates. The task of the model is to select the best response based on the input message and the context of the dialogue. HCN requires fewer training examples compared to other end-to-end approaches thanks to domain-specific code but also retains the benefits of end-to-end learning. The model obtained state-of-the-art performance on the bAbI Dialogue tasks (Bordes et al., 2016). These three properties (ranking of handmade responses, low data requirements, and end-to-end learning) are the main reasons why we decided to use HCN as our topic-level intent recognition model.

Our modified implementation of Hybrid Code Networks consists of the following components: input convolutional neural networks, recurrent neural network, and domain-specific code implementing text actions, functions, action masks, and *can start* methods.

The model obtains the input message, which can be represented in the following three ways. The first way uses an input convolutional neural network described in (Kim, 2014) with pre-trained fastText embeddings (Bojanowski et al., 2017). We obtain the output from the second last layer of CNN. The CNN is trained on the training data of the sub-dialogue. The second way uses the architecture of (Kim, 2014) too, but the weights are pre-trained on the sentiment analysis task. The pre-trained weights are frozen during training on the sub-dialogue. We again obtain the input message from the second last layer of the CNN. The last way uses the pre-trained model for dialogue act detection, from which we extract the output of the second last layer. The weights of the model are frozen during training.

These three input representations are concatenated into a single vector. Furthermore, we concatenate the response class predicted in the previous step. The resulting vector is fed into RNN. The timesteps of the RNN are constructed as follows: Instead of going across the values of the single input vector, the timesteps go across the input messages. This allows the model to learn the representation of the dialogue's context. Our RNN layer consists of LSTM (Hochreiter and Schmidhuber, 1997) cells.

The vector output of RNN is element-wise multiplied by the action mask vector. The action mask vector consists of zeros and ones. Its purpose is to prohibit some actions by assigning them a zero probability. The action mask vector is produced by the action mask code. It consists of a set of rules. They prohibit the usage of responses that don't directly follow the last used response in the dialogue. The element-wise multiplied vector is fed to the softmax layer that computes probabilities of responses. We select the response with the highest probability.

There are two types of responses: text responses and functions. Text responses are directly returned as the response. No additional processing is required. The functions are represented as some code that needs to be executed. The result of the code must be the class of the following response, which is the text response or next function. Meanwhile, the function code can arbitrarily modify values saved in the context.

The text response may contain text actions. Such response can be *"Movie was directed by {say_director}"*, where the text action is *{say_director}*. The text actions are replaced by text actions code before they are presented to the user. The text action code can also modify values saved in the context.

The last part of our HCN implementation is *can start* code. It is responsible for determining whether the dialogue can be started based on the values saved in the context. If the dialogue requires the director's name, but it is unknown, for example, the *can start* code flags the dialogue as not able to start. Another reason for this flag can be that the sub-dialogue has been executed previously, and we do not want to repeat it. The topic graph has to select different dialogue to execute if such a case happens.

Topic Switch detector

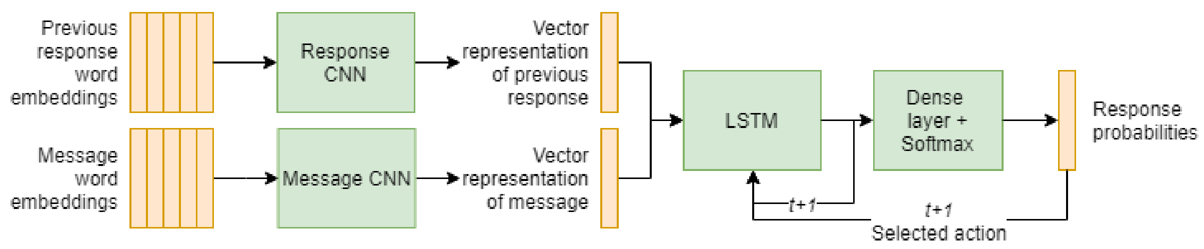


Figure 4.5: Schema of Topic Switch detector

Topic Switch detector is a component that determines whether the user wants to switch the topic of the conversation. If such a request is recognized, we switch the topic of the conversation according to the detected intent and recognized entity. We use this model on top of the intent detection because a decision to switch the topic must be conditioned on the context of the conversation. Intent detection works only with the most recent message, which is not sufficient because message *"I like pop music."* can have different topic switch labels in different contexts. If socialbot asks user *"What do you like?"*, we want to switch the topic to sub-dialogues about music. However, if we are in the middle of the dialogue about favorite music genre and the socialbot asks user *"Which music genre is your favorite?"*, we do not want to switch the topic because it leads to a restart of the conversation about music. But the intent detector returns intent *"Music"* for message *"I like pop music."* in both contexts. This is the reason why we use a Topic Switch detector that is trained to make decisions not only on a recent message but also on the context of the conversation.

The Topic Switch Detector model uses an architecture similar to the HCN model. It consists of two input CNNs Kim (2014) and an RNN. The model's inputs are the last dialogue turn's response and the current user's message. The input CNNs create vector representations of both inputs that are concatenated and passed to the LSTM and the softmax layer. Two output classes of the softmax layer correspond to the probability of the user wanting to switch the topic and not wanting to switch the topic of dialogue.

The topic switch is trained on sub-dialogues from the Topic graph. We generate artificial conversations from training data for individual dialogues, in which we mix training examples of intent detection. We mark the turns into which we mix the intent example by class one, which indicates the topic switch, and the rest of the turns by class zero. The model learns to predict these labels.

4.3.4 Experiments

Top-level Intent and Entity Recognition

We have experimented with several models for the top-level intent and entity recognition tasks. We trained separate models for each task and compared them with the combined model that has outputs for both intent and entity. We had 8,052 samples for intent detection and 3,494 samples for entity recognition. We are continuously annotating new samples during the bot development. Most of the samples are gathered from dialogue logs. At the beginning of the development, a small portion of samples was generated from templates.

The models for entity and intent are described in section 4.3.3 and section 4.3.3, respectively. The combined model is a modification of the entity model. It has two stacked BI-LSTM layers. The last state of the first layer is the intent output and the sequence output of the second layer is fed into a dense layer that is followed by CRF.

We tried three different embeddings for each task, and each test was triggered ten times. We used GloVe (Pennington et al., 2014) with 50 and 300 dimensions and custom fastText embeddings with 100 dimensions. The results are shown in Table 4.3.

Table 4.3: Testing results of intent, entity, and combined models. There are three models for each task that share the same architecture, but they use different embeddings. The accuracy and sentence error rate fields contain mean value and standard deviation across 10 measurements. The results for the combined task contain values for the intent task/values for the entity task. The sentence error rate is the metric applied only for entity recognition and shows how many of the sentences have at least one misclassified token.

	Model	**Accuracy**	**Sentence error rate**	**Training time**
Intent	GloVe50	91.6% ± 0.9	-	**37 sec**
	GloVe300	**94.8% ± 0.4**	-	2 min 10 sec
	FastText	94.7% ± 0.4	-	47 sec
Entity	GloVe50	98.6% ± 0.1	17.0% ± 1.7	**1 min 14 sec**
	GloVe300	98.7% ± 0.2	17.6% ± 1.7	1 min 53 sec
	FastText	**98.8% ± 0.2**	**14.9% ± 1.8**	1 min 18 sec
Combined	GloVe50	93.0% ± 0.5 / 98.2% ± 0.2	- / 21.1% ± 2.1	**3 min 28 sec**
	GloVe300	**95.0% ± 0.4** / 98.3% ± 0.2	- / 21.3% ± 2.5	5 min 7 sec
	FastText	93.5% ± 0.4 / **98.9% ± 0.2**	- / **14.2% ± 2.4**	3 min 42 sec

It is not a big surprise that the models with smaller embeddings are trained faster. On the other hand, there is no such difference in training duration between GloVe300 and fastText. GloVe300 vectors are slightly better for intent detection, whereas Fasttext seems to be better for entity recognition.

Topic-Level Intent Recognition

We evaluate three architectures of topic-level intent recognition that are inspired by hybrid code networks. Moreover, we compared the impact of using word2vec and fastText embeddings for each architecture. We evaluate the turn accuracy and dialogue accuracy of the models on bAbI Task 6 dataset and our in-house Alquist conversational dataset.

Architecture Evaluation The first tested architecture is the same as an architecture of HCN (Marek, 2018). It uses an average of word embeddings, bag-of-words vectors, and additional features as inputs. We concatenate these features and pass them to the LSTM layer. We do the element-wise multiplication of the output of the LSTM layer with the vector of the action mask. We pass this result to the softmax function and select the response with the highest probability. The second architecture uses the LSTM input layer instead of the average of word embeddings and bag-of-words vectors. The third architecture uses a convolutional input layer inspired by (Kim, 2014).

Datasets The Dialog bAbI Task Data (Weston et al., 2015) is a dataset of conversations from the restaurant reservation domain. It is used to test end-to-end dialog systems in a way that favors reproducibility and comparisons and is lightweight and easy to use. The dataset is divided into six tasks with increasing difficulty. We use task six because it is the only task that contains records of real-world conversations between humans and chatbots. It contains noisy and hard-to-learn dialogues due to voice recognition errors and non-deterministic human behavior. It is an ideal benchmark because we faced the same challenges in the Alexa Prize. The dataset contains 56 response classes, and it is split into 3,249 training dialogues, 403 validation dialogues, and 402 testing dialogues.

The Alquist conversational dataset is our private dataset collected from our previous version of Socilabot competing in the first Alexa Prize Socialbot Grand Challenge. The dataset consists of 37,805 dialogues between the user and the socialbot from the domain of books. There are 344,464 message-response pairs in total. The average length of dialogues is 9.11 pairs, the median is 7 pairs, and there are 23,633 unique responses. The dataset is also noisy and hard to learn because it contains voice recognition errors, and some of the messages come from uncooperative users. All 23,633 responses can be clustered into 30 semantically unique responses. This reduction can be achieved thanks to the fact that dialogues were represented as a state graph. Each node in the state graph corresponds to one of 30 semantically unique responses.

Metrics The main two metrics for topic-level intent recognition are turn accuracy and dialogue accuracy. The turn accuracy shows the percentual number of dialogue turns where the topic-level intent was recognized correctly. The dialogue accuracy shows the percentual number of a dialogue where each turn had the topic-level intent recognized correctly.

Table 4.4: Set of best hyperparameters for each model founded by Bayesian hyperparameter optimization on the validation set of bAbI Dialogue Task 6 and achieved Turn accuracy

Hyperparameter	**Model**					
	Word2vec	Word2vec +CNN	Word2vec +RNN	fastText	fastText +CNN	fastText +RNN
LSTM size	85	109	219	55	245	505
Convolutional filters	-	6	-	-	21	-
LSTM dropout	0.92	0.79	0.74	0.85	0.80	0.94
Input LSTM dropout	-	-	0.91	-	-	0.97
Convolutional dropout	-	0.84	-	-	0.72	-
Fully connected dropout	0.59	0.93	0.98	0.82	0.79	0.76
Learning rate	0.001	0.005	0.00005	0.008	0.0001	0.0003
Activation function	tanh	tanh	relu	relu	relu	relu
Input activation function	-	-	tanh	-	-	tanh
Adam epsilon	1E-8	0.1	1E-8	1E-8	1E-8	1E-8
Adam beta1	0.5	0.5	0.9	0.9	0.5	0.5
Turn accuracy	71.3%	70.4%	65.5%	69.4%	**71.5%**	68.0%

Table 4.5: Testing accuracy of Hybrid code networks models

Model	bAbI6		Alquist	
	Turn Acc.	Dialogue Acc.	Turn Acc.	Dialogue Acc.
Bordes and Weston (2017) Bordes et al. (2016)	41.1%	0.0%	-	-
Liu and Perez (2016) Liu and Perez (2017)	48.7%	1.4%	-	-
Eric and Manning (2017) Eric and Manning (2017)	48.0%	1.5%	-	-
Seo et al. (2016) Seo et al. (2016)	51.1%	-	-	-
Williams, Asadi and Zweig (2017) Williams et al. (2017)	55.6%	**1.9%**	-	-
fastText	57.6%	0.8%	86.9%	51.7%
fastText+CNN	**58.9%**	0.5%	90.6%	63.0%
fastText+RNN	54.9%	0.3%	80.6%	40.5%
word2vec	57.4%	0.4%	92.2%	**68.0%**
word2vec+CNN	56.3%	0.1%	**92.6%**	67.8%
word2vec+RNN	54.6%	0.1%	83.9%	45.2%

Results We found the best set of hyperparameters for each architecture on the validation set of bAbI Dialogue Task 6 by Bayesian hyperparameter optimization. They are presented in Table 4.4. We trained the models with the best set of hyperparameters for 12 epochs on both datasets. The best model regarding turn accuracy on the bAbI Task 6 dataset is a model using a convolutional input layer and fastText embedding vectors, which outperformed the baseline reported by Williams et al. (2017). This model achieved a turn accuracy of 58.9%. The best model regarding turn accuracy on the Alquist conversational dataset is the model using convolutional input layer and word2vec embedding vectors, which achieved a turn accuracy of 92.6%. The complete results are presented in Table 4.5.

Sentiment

We have trained our sentiment model separately on two different datasets. The IMDB movie review dataset (Maas et al., 2011) and the sentiment140 dataset (Go et al., 2009). The IMDB dataset is a dataset containing long-form movie reviews with star ratings, whereas the sentiment140 dataset contains tweets that have been annotated as positive or negative based on the emotes used.

The model reached 0.88 on the validation set for the IMDB dataset, and 0.83 accuracy on the test set for the sentiment140 dataset.

Table 4.6: Sentiment values for entities with general negative connotations

Entity	IMDB data	sentiment140 data
terrorism	0.38	0.53
Hitler	0.28	0.64
murder	0.24	0.38

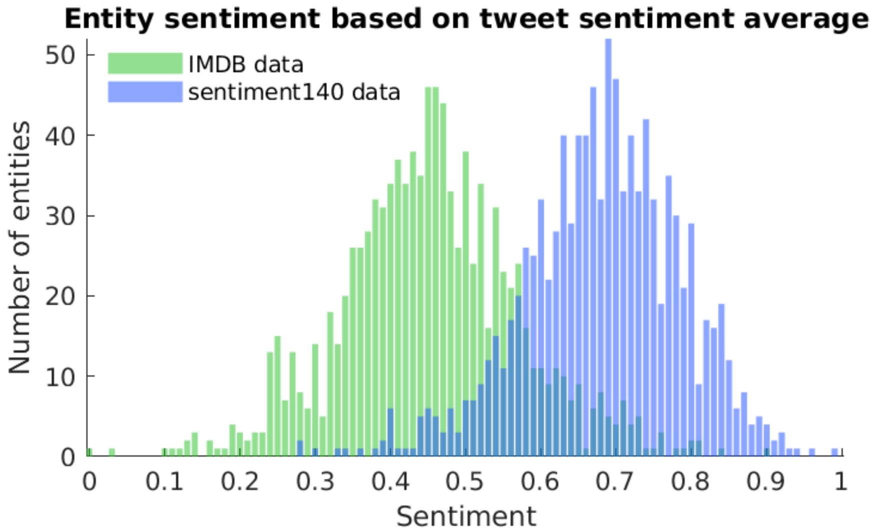


Figure 4.6: The figure shows the histogram of detected entity sentiments for both trained models. Sentiment value 0 translates to the most negative and sentiment value 1 is the most positive.

We compared the models to see how the detected sentiments differ for tweets containing entities recognized by our system. From the Figure 4.6, it is clear that the model trained on sentiment140 data heavily skews towards positive sentiments for various entities. While we believe that could be helpful in order to keep interactions with users positive, the model often fails to recognize desired negative sentiments for serious and generally negative topics (see Table 4.6). Due to this, we are currently using the model trained on the IMDB dataset.

4.3.5 Error Analysis

We run analyses of the errors produced by the system. We manually went through 100 randomly selected conversations and marked the turns that we found incoherent (the engagement of the system was not evaluated). Note that the evaluation was done on a subjective basis. For each error turn, we marked which of the system components caused the erroneous response (some turns can contain multiple errors). The distribution of the errors is shown in Table 4.7. An error is marked only for those turns that produced a response violating the coherence of the dialogue. There can be, for example, a turn with a punctuation error, but the responses are still generated correctly—such a turn is not marked as an error.

4.3.6 Conclusion and Next Steps

We designed a second version of the Alquist socialbot that addressed the shortcomings of the previous version, specifically the lack of flexibility caused by large topical dialogue structures. We divided the NLU into top-level and top-level parts. The primary focus on topic-level NLU, along with topic-switch detection, allows us to smoothly change topics when appropriate while precisely navigating in a selected topical dialogue. We show a competitive performance of the proposed topic-level intent recognition. Considering the dialogue accuracy metric, there is, however, still room for improvement (Table 4.5) as the longer dialogues tend to accumulate the error over the turns easily. Fortunately, from the use experience perspective, this does not necessarily cause the dialog to fail, as the failure still keeps the user within the discussed topic.

As shown in Table 4.7, most of the errors are caused by wrong topic switch detection. In comparison to the topic-level intent recognition, this is a more significant error category as it may result in disruption of the conversation by switching to the wrong topic or switching when it is not appropriate. Therefore, as the next steps, we need to focus on smoother transitions by decomposing the dialogues into smaller structures.

Table 4.7: Error distribution in the second version of Alquist. The analysis was done manually on 100 conversations. Some of the analyzed turns contained multiple errors. The most typical combination was an error in the Topic Switch module and in the Intent module. Conversation examples are synthesized by our team based on observed patterns in the data.

Type	%	Example
Topic Switch	44 %	**Bot:** . . . How is it going? **User:** it's going good what happened today in history **Bot:** Well, I'm always busy as people keep chatting with me.
Intent	28 %	**User:** how old are you (intent: how_are_you)
Punctuation	19 %	**User:** good I'm ⟨SEP⟩ good you're really good
Dialogue Act	6 %	**User:** who's boyfriend (DA: Other, correct Wh-question)
Dialogue Manager	4 %	**Bot:** . . . Who do I have the pleasure of speaking with? **User:** why are fire trucks red **Bot:** Hey Red, I'm happy to meet you!

4.4 Alquist: Utilizing Knowledge Graph

We continue to leverage the modular architecture of the conversational system. The shift from a rich dialogue structure, which can present all the relevant information to the users as they traverse through it, focused on a single topic as proposed by Pichl et al. (2017) to a more modularized approach proposed by Pichl et al. (2018) contributed to a more flexible conversational experience and easy content management. This eventually contributed to the user experience as highlighted in section 4.6. Even though the flexibility was higher, there is still an inefficiency remaining within each sub-dialogue unit and while switching between individual units.

To mitigate the errors related to the fixed dialogue structures, we propose a system, described in Pichl et al. (2020) and in this section, using minimal conversational units—adjacency pairs (Boyer et al., 2009). Adjacency pairs are conversational units that can be chained flexibly to create a complex conversational structure. Thanks to the fact that dialogues built using adjacency pairs are short, there is no need for strict topic switching, which caused most of the errors as shown in Table 4.7. Moreover, we use a conversational knowledge graph that stores the information available for use as well as the information expressed by the users in previous sessions. The previous versions of Alquist were also able to store users' preferences; however, only in predetermined parts of dialogues. With the new approach, the system can remember facts expressed at any point. The utilization of the knowledge graph allows us to remember a significantly large number of facts extracted during the dialogue, which can be subsequently used in the interaction by Alquist or asked about by users.

4.4.1 Conversational Knowledge Graph

To be able to handle more complex interactions with the user, it is necessary to have access to real-world knowledge and objective factual information as well as personalized (subjective) knowledge, such as opinions, preferences, and others. To represent the knowledge and opinions of our system about the world, we have created an RDF graph database where we used a dump of the Wikidata database[8] as its base.

We have also designed a custom ontology that is to be partially mapped to the ontology used by Wikidata. Our ontology is inspired by other common ontologies such as Schema.org[9] and EVI. However, we expanded it with objects representing the user and the bot. We also added properties modeling personalized relationships between the user or bot and various other objects in the database. Examples of such properties are *likes, hates, hasFavorite, hasOpinion*.

As shown in Figure 4.7, we also created custom annotation properties in the ontology (e.g. *doYou*). These properties are tied to object properties. Their range is commonly a string (e.g. *"Do #DOMAIN# like #RANGE#"*), and they represent delexicalized examples of sentence structures that represent the domain object property. This allows us to do two things. We can automatically generate and update datasets for dialogue acts detection directly from the knowledge base (KB). Furthermore, we can use the sentence structures assigned to properties in adjacency pairs templates to generate system responses regarding relationships between objects in the database.

Additional examples of sentence structures, such as:

- *ProvideInformation_Negative* - #DOM#'s name is not #RAN#.

[8]https://www.wikidata.org/wiki/Wikidata:Database_download
[9]https://schema.org/

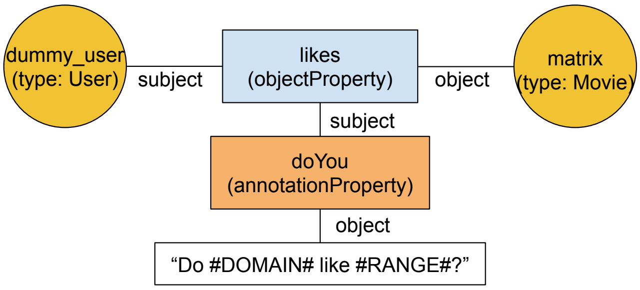


Figure 4.7: Illustration of sentence structure examples as annotation properties

- *OpenQuestion_Object_Positive* - What is #DOM#'s name?
- *YesNoQuestion_Positive* - Is #DOM#'s name #RAN#?

The new approach to representing knowledge allows the system to create a profile of the user directly in the knowledge database based on the information the user shares with the system. The system is then able to reference the profile during the conversation, and this leads to more variability and higher personalization. However, this personalized engagement needs to work in both directions—and so the profile of the bot is represented in the knowledge base in the same way. The database contains a representation of the bot, complete with its preferences, personal profile, likes, and dislikes. The profile is adjusted and anonymized for the purposes of the competition, but the system can draw information from it in real-time and include it in the conversation.

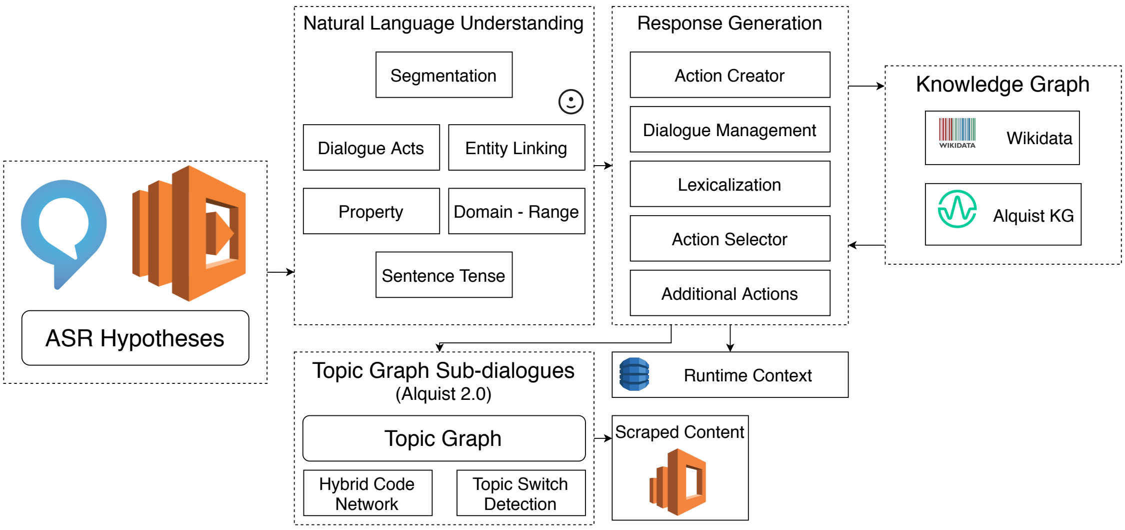


Figure 4.8: Alquist 3.0 architecture schema

4.4.2 Natural Language Understanding

As the system architecture is now highly focused on the adjacency pairs and knowledge graph, we need to introduce additional NLU models to work with smaller language units and structures from the knowledge graph effectively. The first model is a sentence segmentation that predicts punctuation and is used as a segment separator. Knowledge-graph-related models are entities linking a property detection that recognize the respective aspects in the sentence and map them to the knowledge graph nodes and edges.

Sentence Segmentation

The goal of this model is to split the sentences into clauses. The standard way how to tackle this task is to decide based on punctuation marks. However, the transcripts from the ASR modules typically consist of speech hypotheses without punctuation. Therefore, we start by using punctuation restoration, and afterward, we split the sentences based on the restored punctuation marks.

Compared to the former Alquist system, where we used a bidirectional neural network—TBRNN (Tilk and Alumäe, 2016)—we decided to use a BERT-based architecture (Devlin et al., 2019). Our model consists of a pre-trained BERT model (on a masked language model task) layer with a linear classification layer that predicts the punctuation mark after the middle word of a fixed-size input sequence.

Since the model predicts only the punctuation after the word in the middle, we need to add padding to the beginning and the end of the input speech hypothesis. Afterward, we feed the model with sequences of a fixed segment size based on the moving window, and the model returns one of the possible punctuation characters for each word in the original speech hypothesis.

Dialogue Act Detection

The key concept of the system is the processing of adjacency pairs. It is highly connected to the identification of the dialogue acts in the message of a user. We created a custom hierarchical structure for the dialogue acts. We treat this problem as a simple text classification task where each class is defined by the combination of the tags on the path from the tree root to a leaf node. The description of the classes is reported in Pichl et al. (2020). We defined in total 37 leaf classes, each of the classes can be mapped to one or more dialogue act classes presented in the Switchboard dataset[10] Godfrey et al. (1992).

The data for the dialogue act detection model were created manually through a template mechanism which we have been using since the previous version of our system. The created data are stored in the KG. We adopt the same model architecture that we have been using for intent detection and dialogue act detection described earlier in the previous sections.

[10]`https://web.stanford.edu/~jurafsky/ws97/manual.august1.html`

Entity Linking

As our new approach is based on working with a Knowledge Graph (KG), we need to identify each entity with its unique KG ID to be able to generate a proper response. This task is commonly referred to as entity linking. Entity linking, as stated in Broscheit (2019), consists of three parts: Mention Detection (MD), Candidate Generation (CG), and Entity Disambiguation (ED).

To generate mentions (text spans of potential entity occurrence), we used a neural network based on a bidirectional LSTM and BERT model, where the training set was manually labeled from a small subset of the user's messages. We test our model on the sequence labeling task. See the result in Table 4.9.

After retrieving the possible text spans, we combine the candidate generation with entity disambiguation. First, we have to decide if the entity is from our "private" knowledge graph or from the "general" knowledge graph (for example, Wikidata). We trained our neural network model to classify into two label types. If the model returns the label classifying the entity candidate as one from our "private" KG, then we have a set of manually designed rules to decide which possible entity should be assigned to that specific text span. The other case is that the candidate is from the "general" KG: then we query an Evi knowledge graph service to make the mention resolution. We collect all candidates for possible text span that the system returns and query the Evi knowledge graph service again to get their IDs (more specifically, Wikidata ID).

Further processing is coupled with property detection (see section 4.4.2), where we map the text span to two classes – domain or range. We have to match predictions from property detection and entity linking together to be able to decide which entity (linked to the knowledge graph) is the domain and which one is the range.

Property detection

The knowledge graphs store facts as triples, with each triple containing two entities and one relation. The user utterance can be mapped to this triplet—the entities are called domain and range; the relation between them is called property. To properly handle the response, we need to retrieve the property out of the user's message.

We tackle the problem of property detection as a multi-task model. These models are showing promising results on tasks like joint slot-filling and intent classification Liu and Lane (2016).

Based on Yu Wang (2018), we design our neural network with the approach based on the multi-task loss function. The model is trained on two tasks—sequence labeling and sentence classification. The sequence labeling identifies the domain and range entities in the text. This is a similar approach to the proposed entity linking task. However, the domain and range entities are a subset of all entities that might occur in the text. To align the domain and range entities with their linked KG IDs, we match the overlapping entities. The classification part of the models classifies the utterance into one of the property classes.

We safely assume each input to contain only one property since we use segmented results as an input for this task.

4.4.3 Adjacency Pairs and Dialogue Management

The goal of the dialogue management part of the proposed system is to map the user utterance to a subgraph from the KG using the NLU annotations described above. The NLU annotations are done for each ASR hypothesis. Moreover, sentence segmentation may create multiple segments for a single user utterance. In that case, the subsequent annotations are done for each segment individually.

Once the entities and properties are linked to the corresponding KG IDs, the system ranks each linking using the ASR hypotheses score and individual annotations scores. For example, for the question "What is your name" the entity type is a *chatbot*, the property is a *name*, and the dialogue act is an *open question*. For each dialogue act class, there are several classes that may follow the original one while not breaking the coherence of the conversation. This can be expressed as an adjacency pair matrix. Based on the matrix, we select candidate edges from the KG. The candidate edges are then ranked based on the entity relevancy consisting of entity popularity and semantic similarity with the previous utterances of the conversation. Each entity's popularity is updated when used in conversations that were rated with four or five stars.

4.4.4 Experiments

Sentence Segmentation

We illustrate the results of the sentence segmentation models in Table 4.8. There is a significant improvement in all metrics compared to the previous model (TBRNN). Moreover, since the BERT model is more computationally demanding, we also analyzed the performance of various BERT alternatives that use distillation technique (Sanh et al., 2019) or a reduction of the number of parameters (Lan et al., 2019). Based on the observed results, we can conclude that all tested models outperform the former bidirectional one.

Table 4.8: Results of the experiments on IWSLT 2012 English dataset.

Model	*Comma*			*Period*			*Question*			*Overall*		
	Pr.	*Re.*	F_1	*Pr.*	*Re.*	F_1	*Pr.*	*Re.*	F_1	*Pr.*	*Re.*	F_1
TBRNN	65.5	47.1	54.8	73.3	72.5	72.9	70.7	63.0	66.7	70.0	59.0	64.4
BERT	**74.9**	**66.8**	**70.7**	**81.8**	85.5	**83.6**	66.7	**73.9**	**70.1**	74.4	**75.4**	**74.8**
DISTILBERT	70.3	58.6	63.9	81.4	76.3	78.8	**78.4**	63.0	69.9	**76.7**	66.0	70.9
ALBERT	84.0	44.1	57.8	77.0	**87.6**	82.0	64.6	67.4	66.0	75.2	66.4	68.6

Entity Linking

We experimented with separate sequence labeling tasks to identify the entity mentioned in a text. We use the standard CONLLL 2003 dataset (Tjong Kim Sang and De Meulder, 2003) and an in-house dataset (Alquist 3.0) of manually labeled conversations. Our dataset contains 2,486 utterances. We split the data in an 80:20 ratio for the train and test splits, respectively. The Bi-LSTM model uses Fasttext embeddings as an input (see embedding selection in section 4.4.4. The Table 4.9 shows the results of the selected models on the described datasets.

Table 4.9: Results of a sequence labeling subtask

Model	**Dataset**	**Precision**	**Recall**	**F1**
Bi-LSTM	CoNLL 2003	90.0	89.4	89.8
Bi-LSTM	Alquist 3.0 data	93.6	92.0	92.6
BERT	CoNLL 2003	91.3	92.1	91.7
BERT	Alquist 3.0 data	94.4	95.0	94.7

Moreover, we evaluated the overall performance of the entity linking pipeline, which consists of sequence labeling, knowledge type classification, and rule-based ID mapping. For a knowledge type classification, we use the same Bi-LSTM and BERT model plus an additional simple logistic regression classifier on top of the Fasttext embeddings. We use our Alquist 3.0 dataset, where each entity mention is also labeled with the corresponding KG ID. Some of the entities in the text may not correspond to any of the KG IDs. These entities are omitted from the evaluation. Table 4.10 shows the result of the full entity linking pipeline.

Table 4.10: Results of a combined sequence labeling, knowledge type classification, and ID mapping

Labeling+Type Model	**Precision**	**Recall**	**F1**
Bi-LSTM+LR	82.8	79.5	81.1
Bi-LSTM+Bi-LSTM	84.2	81.3	82.7
Bi-LSTM+BERT	84.9	83.7	84.3
BERT+LR	83.4	80.6	82.0
BERT+Bi-LSTM	84.3	81.9	83.1
BERT+BERT	84.8	84.2	84.5

Property Detection

To find the most suitable word embeddings in our proposed system, we selected the following models for our experiments:

- GloVe (Pennington et al., 2014) - 300d
- Fasttext (Bojanowski et al., 2017; Mikolov et al., 2018) - 300d
- Bert (Devlin et al., 2019) - 768d - without fine-tuning

We selected the publicly available ATIS dataset for embedding selection and property detection experiments. The data contain spoken utterances classified into one of 26 intents. Each token in a query utterance is aligned with IOB labels representing the slot values. Primarily, the dataset is used for intent recognition and slot filling, but it is sufficient for our testing purposes with minor modifications. The results on both the original and modified dataset are shown in Table 4.11. All embeddings are used as an input for the Bi-LSTM model. From the results, we can see that the best-performing embedding type is Fasttext. We adopted it for further experiments, including the previously mentioned entity linking.

To modify the data for our property selection task, we map each intent to the list of properties from the Knowledge Graphs. Additionally, we label each entity, whether it is a domain or range.

The final model of property detection has two outputs. The sequence labeling output will classify each token into three categories – domain, range, or outside. The text classification output predicts the label of the property (mapped to our KG), which is mentioned in the user's message.

Table 4.11: Experiments results on ATIS dataset. Intent F1 and Slot-filling (SF) F1 are measured on an unmodified dataset. SF D&R F1 is a metric measured after each entity tag is divided into Domain and Range categories. The intent class is directly mapped to a KG property.

Embeddings	**Intent F1**	**SF F1**	**SF D&R F1**
BERT	92.4	95.7	93.3
GloVe	90.7	96.8	94.1
FastText	**95.7**	**98.4**	**95.4**

4.4.5 Conclusion and Next Steps

We evaluated the individual parts of the architecture leveraging utterance to Knowledge Graphs mapping with an adjacency-pair-based strategy for constructing the responses. The segmented utterances are transformed into triplets that may correspond to a subgraph from the KG. We evaluated individual NLU algorithms related to the architecture, which show promising results on each task. However, the approach suffers from a lack of relevant knowledge in existing KGs and from the increased complexity of dialogue creation.

4.5 Alquist: Generative Models and Personalization

The adaptability on both language and content sides introduced by adjacency pairs and knowledge graphs contributed to the flexible flow of the conversation. One of the prerequisites to bring the flexibility to a more engaging level is to have proper content stored in the knowledge graph. When initialized with a standard knowledge source such as Wikidata, we obtain a large set of factual information. However, information about current events, trending news, and fun facts are typically not present in such a knowledge base. The original aim of the system was to obtain this additional information and store it in the knowledge graph. However, such a process is rather slow and requires a significant amount of traffic to be successful.

As a minor improvement to the described issues, we added several domain-specific sources and trivia sources for more engaging conversation. However, a more robust approach is needed to handle various and mostly unexpected inputs. Therefore, we propose a system leveraging out-of-domain detection and generative models to handle inputs that cannot be effectively responded to using dialogue scenarios or knowledge base information. Additionally, the personalization of the conversation is introduced by a continuous exploration and exploitation of the information about a user. The system is built using principles we described in Pichl et al. (2022) and the actual system for the competition we described in Konrád et al. (2021).

4.5.1 Architecture

We built the system on top of our multi-purpose conversational platform called Flowstorm (Pichl et al., 2022). The platform allows to rapidly create conversational flows. We enriched it with generative model integration that can be dynamically triggered in various stages of the conversation. Unlike in the previous versions of the system, the decision to use a generative model is not made at the top level but is tightly connected to the proposed out-of-domain mechanism.

The out-of-domain (OOD) detection depends on the specific dialogue structure, i.e., the system needs to recognize that the dialogue is not able to handle the input. For that reason, it is designed as a part of the intent classifier that recognizes that a given input is unexpected. The OOD as part of a hierarchical intent recognition will be described in chapter 5. Once the OOD input is recognized, the system uses the Neural Response Generator—neural generation model trained on large dialogue corpora that produces a response based on the context of the dialogue. A combination of dialogues as originally described by Pichl et al. (2018), OOD recognition (subsection 4.5.3), and the Neural Response Generator (Konrád et al., 2021) allows us to utilize high-quality hand-designed dialogues while adding the necessary resilience to unexpected user inputs that is a crucial component for coherence in open-domain dialogue systems.

The proposed innovation of an engagement uses the idea that in order to entertain the conversational partner, one has to learn what entertains them first and then utilize the knowledge in the following conversation. This might remind us of a famous problem of computer science: the problem of exploration and exploitation. The socialbot is in the role of an entertainer who has zero prior knowledge about the user. Therefore, it has to explore the user's preferences first. However, it can't stay in a pure exploration mode for the rest of the conversation. Gradually, it has to proceed into an exploitation phase after some time to maximize its engagement score. This philosophy is reflected in the design of the components that the fourth version of Alquist is made of.

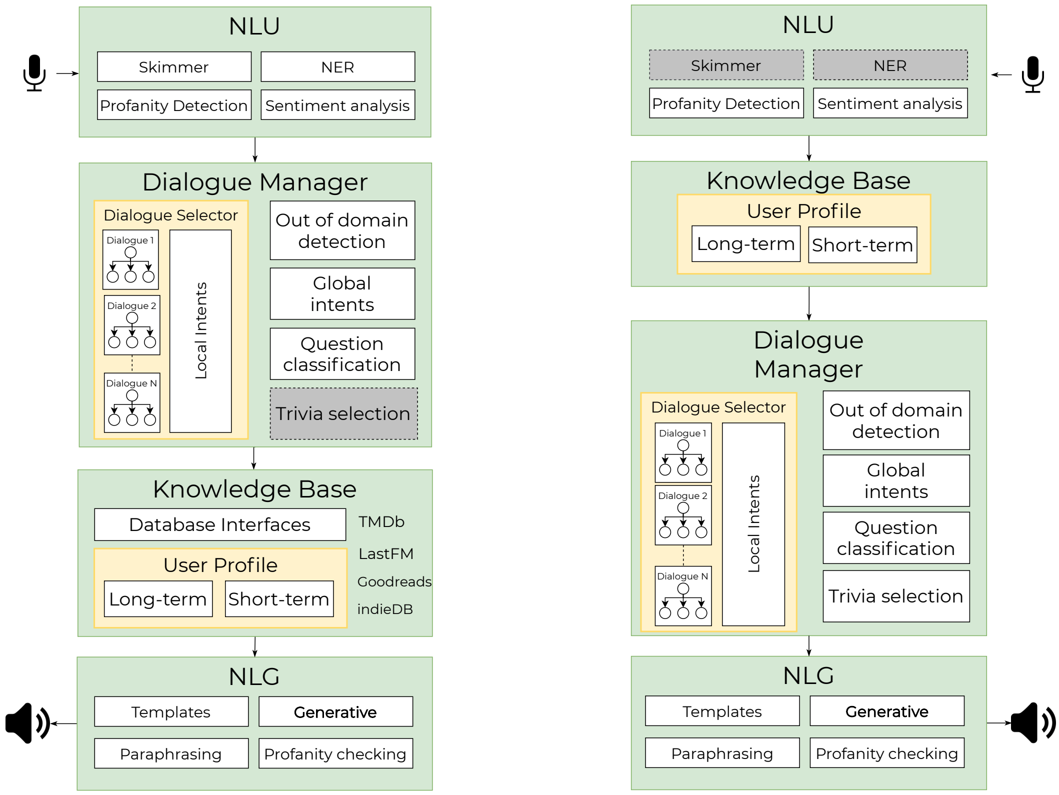
For the exploration part, in which Alquist learns the preferences of the user, the main research and development emphasis was put on Skimmer (subsection 4.5.2; a component that extracts information the user mentions without the bot explicitly asking for it), User Profile, and Knowledge Utilization. The mentioned components collect and organize the pieces of information mentioned by the user that are utilized in the following dialogue.

For the exploitation part, in which Alquist utilizes the knowledge about the user, the main emphasis was put on the Dialogue Manager, Trivia Selection, Intent and Out-of-Domain classification (subsection 4.5.3), and the Neural Response Generator. Those components are responsible for selecting the next action in the dialogue and response production that utilizes the knowledge about the user.

Figure 4.9 presents the organization of all of Alquist's components. We took Pichl et al. (2017, 2018, 2020) as our starting point. The unchanged components are described in the previous sections. The flow of the components is as follows. First, the Skimmer analyses the user input for the mentioned pieces of information. The pieces of information are stored in the User Profile. Based on the values stored in the user profile, Dialogue Management selects the next dialogue to start or selects and presents some trivia related to the actual topic of a conversation. The dialogue is directed according to the Intent classification of the user input. Finally, if the Out-of-domain classification recognizes an unexpected user input, the Neural Response Generator produces a coherent response based on the context of the conversation.

4.5.2 Skimmer

The Skimmer component is intended to extract as much information about a communication partner as possible. Such information can be expressed in two basic scenarios. The first scenario is when the bot asks a user a direct question (e.g., *"Do you have a brother?"*) and stores the answer to the question. In this scenario, the bot is aware of the dialogue context, it knows what type of answer is expected, and it can store the response in the User Profile accordingly. Using the stored value, the bot can carry out a highly personalized conversation and ask relevant questions such as *"How is your brother today?"*. This strategy leads to a more personalized conversation.



(a) Exploration flow

(b) Exploitation flow

Figure 4.9: The system components are shown in two different orders based on the dialogue strategy—exploration and exploitation. In the exploration strategy, our bot uses dialogues to extract information about the user. In the exploitation strategy, our bot utilizes the information stored in the User Profile and selects dialogues accordingly. The grey components are not essential for the corresponding strategy, but they are not disabled completely.

Since we do not want to disrupt the fluency of the dialogue by asking too many personal questions to gather information about a user, we want the bot to have the ability to extract the information from each user utterance. For example, the information about the user having a brother can be mentioned "by the way" in the conversation (e.g., in a movie-related conversation, the user may mention *"I was with my brother at the cinema yesterday."*). We want to extract the information from the sentence regardless of the topic being discussed. For this purpose, we implemented the component called *Skimmer*. It skims through each utterance and saves the values in the User Profile based on a list of rules. Each of the rules contains the following attributes:

- *Regular expression* - a set of patterns which must or must not (negative patterns) be contained in the utterance.
- *User Profile attribute* - the name of the attribute where the value will be stored.

◦ *Value* - the value stored in the attribute, typically *true*, *false*, or a matched group of the regular expression.

The component processes the user utterance in the following way. It takes each rule from the list and tries to match it with the corresponding regular expression. If it is matched, the value is stored in the specific attribute of the User Profile.

4.5.3 Intent and Out-of-Domain Classification

Following the concept of dialogue presented in the second version of Alquist (Pichl et al., 2018) using the Flowstorm (Pichl et al., 2022) platform as a base, we design each dialogue as a tree structure. A crucial point in the conversation structure is where we expect the user input/user utterance. Each user utterance is then classified into a specific intent for which the dialogue designer manually writes training utterances. However, the dialogue designer cannot incorporate each possible intent because of the complexity of language and the open-world assumption (Keet, 2013). Based on that, these user utterances for which the dialogue is not prepared are called out-of-domain input utterances. On the other hand, the in-domain intent is a user utterance for which the dialogue designers have prepared a response. Such a response is designed in a coherent and engaging conversational style.

We have also incorporated the concept of intent hierarchy inherited from the Flowstorm dialogue architecture into our dialogue design. It comes with two types of intents— global and local (contextual) intents plus Out of domain (OOD). However, despite the fact that OOD detection has been receiving more attention lately (Marek et al., 2021; Tan et al., 2019a), the current datasets for evaluating the OOD performance (Larson et al., 2019; Gangal et al., 2020; Lee and Shalyminov, 2019) are not designed for testing the hierarchical structure of our dialogues and contain mainly explicit commands. To solve this issue, we created our testing data from anonymized queries of real users. The following subsection describes our initial experiments with the hierarchical intent structure used in the Alquist socialbot during the Alexa Prize Socialbot Grand Challenge 4. The extended approaches and experiments are shown in chapter 5.

Model

We experimented with various approaches to intent recognition focused on usage as part of the hierarchical dialogue structure (shown in Figure 4.10). The hierarchical structure provides the dialogue designer with modularity in creating the flow of the conversation but puts a significant emphasis on the effectiveness of the algorithms for intent classification.

To maintain effectiveness, we train a separate model for each level of the hierarchy as described in chapter 5. We show the whole classification system in Figure 5.7. The system works in two steps. First, we need to determine which intent model in the hierarchy is appropriate—local or global. A brief description of the approach is as follows: We utilize cosine similarity over sentence embeddings between user queries and train examples of each intent to classify utterances as local or global intents while prioritizing local intents.

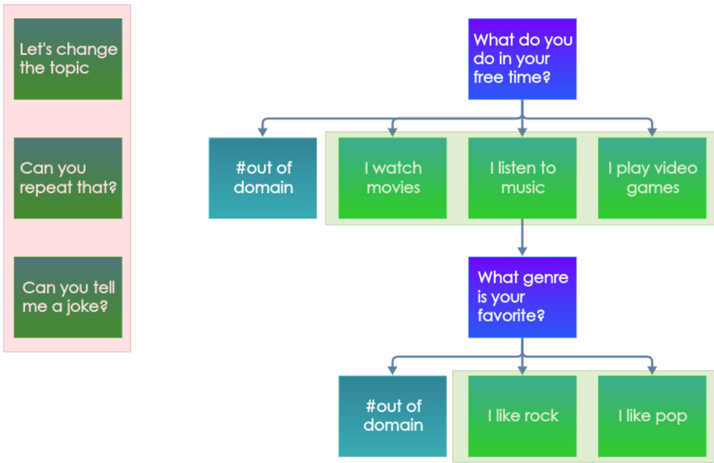


Figure 4.10: Hierarchical structure of dialogues. The dialogue consists of the nodes representing user inputs (green nodes) and bot responses (blue nodes). The chained nodes create a diverging flow of conversation. We omitted some of the nodes for clarity. The user input nodes represent intents. There are two types of intents. The user utterance can be classified into one of the local intents (green boxes) specific to a certain turn of the dialogue, plus any global intent (red box) that can be applied in any turn of the dialogue. Moreover, the user utterance can be classified as out-of-domain.

The priority is based on using a stable sorting algorithm over cosine similarities between sentence embedding of the user utterance and the examples of each intent. We also allow manual setting of the threshold for local and mainly for global intents, leading to the filtering out of the intents if the cosine similarity is not high enough. In the second step, the utterance is classified into a specific intent by the corresponding logistic regression selected in the previous step. We use logistic regression because of the speed of its training and the proven performance in low-resource scenarios (see Figure 4.11). Thus, the final intent classification is performed by logistic regression. More detailed descriptions and experiments are shown in chapter 5.

Datasets

We performed our analysis on a publicly available dataset as well as on manually labeled anonymized queries:

- CLINC150 Dataset (Larson et al., 2019)
- ALQUIST 4.0 Dataset

A summary of the datasets is shown in Table 4.12 and Table 4.13. All samples in the ALQUIST 4.0 Dataset were carefully checked for consistency and drawn from aggregated and anonymized queries. The datasets generated during this study are not made available for privacy reasons. We artificially partitioned the CLINC150 Dataset to support the hierarchical structure. The partitioning was performed as a random selection of 16 intents from the dataset and then divided into two sets, one set representing local intents and the other representing global intents. The ratio was set to 1:3 to represent a typical situation in a dialogue. The split results in 4 local intents and 12 global intents. The process was repeated 15 times. The local intents drawn in a step are excluded from the next selection. Global intent selection allows the selection of a single intent multiple times. This leads to the resulting dataset containing some of the samples duplicated. The resulting dataset statistics are shown in Table 4.13, and the related experiments results are shown in the subsection 4.5.5.

Table 4.12: ALQUIST 4.0 Dataset

Type of utterances	**Average Per Intent**	**Total Utterances**
Local - Train	564	3952
Global - Train	399	2793
Local - Test	27	193
Global - Test	49	344
OOD - Test	61	429

Table 4.13: CLINC150 Augmented Dataset

Type of utterances	**Average Per Intent**	**Total Utterances**
Local - Train	400	7200
Global - Train	100	21600
Local - Test	120	1800
Global - Test	330	5400
OOD - Test	1200	1200

4.5.4 Neural Response Generator

A Neural Response Generator (NRG) is a neural conversational response generation model trained on large conversational corpora. It generates a response based on the most recent turns of dialogue. We use such a model in Alquist in two settings. The neural response generator creates a response for out-of-domain user inputs, and it generates follow-up questions about trivia.

The motivation to use a NRG for the Out of domain (OOD) inputs is the following. We put the main content emphasis in Alquist 4.0 on hand-designed dialogues. Dialogues are represented as graphs (Figure 4.10). They consist of nodes representing the user inputs and the bot responses structured in diverging dialogue flows. Although they allow a human

Table 4.14: A hypothetical dialogue situation, in which a statement–question control mechanism of the NRG is used. The first user input is identified as out-of-domain. The Neural Response Generator produces a question that is followed by the user's utterance. That is followed by a generated statement that transits into a hand-designed dialogue.

Alquist (Hand-designed dialogue):	*What do you do in your free time?*
User (Out-of-domain):	*I draw pictures.*
Alquist (NRG-Question):	*What kind of pictures do you like to draw the most?*
User:	*I usually draw portraits*
Alquist (NRG-Statement):	*I see. Portraits are hard to make.*
Alquist (Hand-designed dialogue):	*Do you visit galleries?*

designer to create high-quality conversations with maximum control over the dialogue flow, the designer can not predict all possible flows the conversation can go through. We can detect the situations in which the user diverts from the predesigned flow by the OOD detection. However, the problem of how to continue the conversation in a meaningful way emerges. The fact that we can't predict all possible flows of the conversation also means that we can't design them. Neural response generators that can create a response on the fly based on the context of the dialogue and the user input are the way we solve the problem.

4.5.5 Experiments

Our analysis includes an inspection of the input features (embeddings). We include the following sentence embeddings:

- Average of word embeddings FastText (Mikolov et al., 2018)
- Universal Sentence Encoder - Deep Average Network (USE-DAN) (Cer et al., 2018)
- Universal Sentence Encoder - Transformed-encoder (USE-TRAN) (Cer et al., 2018)

The model described in section 4.5.3 is tested in two ways — automatically and manually. The automatic evaluation is performed over our ALQUIST 4.0 Dataset (shown in Table 4.12) and the artificially hierarchical augmentation of the CLINC150 Dataset (shown in Table 4.13). The manual evaluation was performed on aggregated data selected from anonymized user conversations with the socialbot. We collected all user utterances from these parts of dialogues and performed the human evaluation. The results can be seen in Table 4.15. Besides evaluating the performance solely for the OOD detection, we looked at the performance of the local and global intents. The results are discussed in the following section.

To select the most suitable model for the final intent classification, we measure the difference between the three most common classification models—Logistic Regression, Support Vector Machine, and the 2-layer Neural Network. We focus on the necessary number of

Table 4.15: Manual evaluation on aggregated data

		True intent		
		Local intent	**Global intent**	**OOD**
Predicted intent	**Local intent**	3451	23	35
	Global intent	0	472	0
	OOD	0	15	238

needed examples to achieve sufficient accuracy. The evaluation is shown in Figure 4.11 and highlights the problem of the neural network when dealing with low-resource scenarios. The measurement was performed over CLINC150, randomly choosing five classes (our average number of intent classes for the intent model) and randomly choosing N examples. This procedure was repeated 25 times, and the shown values are averaged. We selected Logistic Regression as a model performing well in the low-resource scenario.

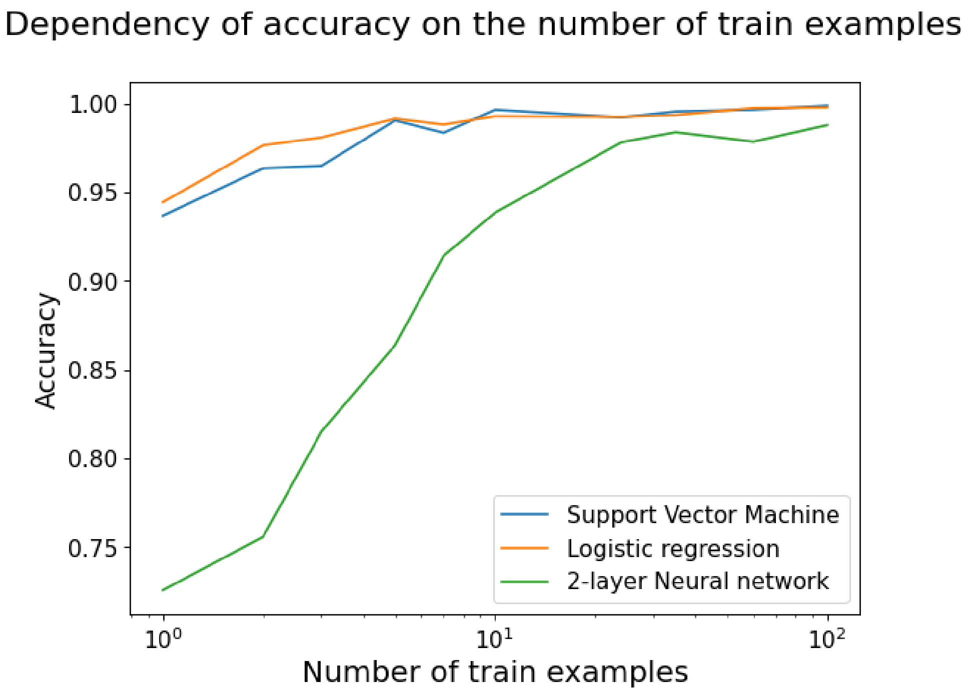


Figure 4.11: Dependency between the number of examples and the accuracy for the three most common classification models. Measurement was performed on the CLINC150 Dataset. The number of examples is on a logarithmic scale and was measured on a range from 1 to 100 train examples.

4.5.6 Results and Discussion

Taken together, our results (shown in Table 4.16) suggest a relationship between the type of embedding and the performance of the classification model. The difference in results can be explained by the word-order sensitivity of advanced embedding techniques. Another important aspect is the memory requirements and the speed of obtaining the embedding (shown in Table 4.17). The results suggest the usage of USE-DAN as an appropriate embedding layer.

It should be mentioned that the performance on each level of the hierarchy remarkably differs between our two datasets. We believe that it is caused by the artificial augmentation of the CLINC150 Dataset and its unrealistic representation of real-world use cases. We should also notice the higher performance for the local intent classification than for the global intent classification (notable mainly on the ALQUIST 4.0 Dataset). It is caused by the hierarchical structure of our model, which emphasizes the local intent over the global as was described in 4.5.3. This is aligned with our experience that staying in the local context of the dialogue is beneficial for coherence.

The performance of the OOD detection needs to be evaluated with respect to precision and recall. The high precision and lower recall indicate that the algorithm is suitable for classifying OOD in the conversational domain because we prefer false negative over false positive—it supports the consistency of the dialogue. In addition, we performed a human evaluation (shown in Table 4.15), which demonstrates the performance on real-world data. It supports previously stated conclusions.

Table 4.16: Intent classification results

		CLINC150			**ALQUIST 4.0 DATA**		
Embeddings	*Intent type*	*Precision*	*Recall*	*F1*	*Precision*	*Recall*	*F1*
FastText	Local	99.6%	91.6%	95.2%	85.5%	61.6%	68.7%
	Global	98.9%	89.3%	92.8%	67.8%	42.7%	48.2%
	OOD	99.8%	44.1%	61.2%	99.8%	81.3%	89.6%
USE - DAN	Local	99.9%	96.5%	98.1%	90.8%	69.5%	74.3%
	Global	99.6%	96.6%	98.0%	77.0%	54.7%	60.9%
	OOD	99.9%	89.0%	94.2%	99.9%	83.8%	91.2%
USE - TRAN	Local	99.5%	97.7%	98.5%	80.0%	60.9%	65.5%
	Global	99.5%	97.7%	98.5%	85.4%	59.6%	67.2%
	OOD	99.9%	91.1%	95.3%	99.9%	85.6%	92.2%

4.5.7 Conclusion

In order to actively engage the user in the conversation, we have developed Skimmer, a component that learns about the user from their messages and fills in information in their

Table 4.17: Requirements of different embedding algorithms

Embeddings	Speed	RAM usage
FastText	200 it/s	190 MB
USE - DAN	120 it/s	1765 MB
USE - TRAN	25 it/s	1650 MB

User Profile. We are then able to utilize what we have learned about the user's interests and personality further in the conversation, making it evident that the socialbot is invested in learning more about the user, remembers their preferences, and takes them into account during the conversation.

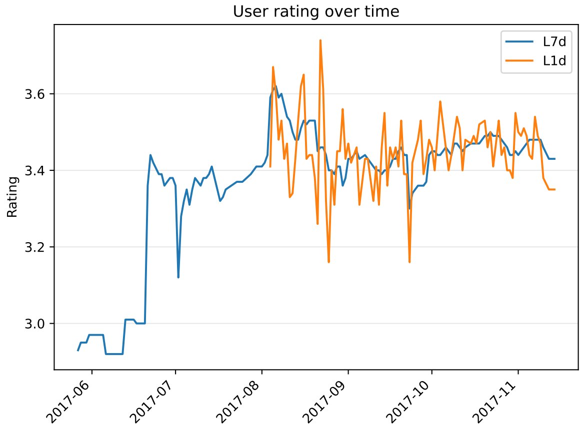
We have introduced Out of domain query detection as a core functionality of our system. This allows us to hand over more control of the conversation to the user, which makes the socialbot seem more responsive to the user queries. However, it also introduces a problem of how to handle the OOD responses.

In order to keep the conversation engaging even when moving in an unexpected direction, we have also integrated two generative Neural Response Generators. Firstly, the Neural Response Generator triggers when the user input is classified as OOD. Secondly, we utilize the NRG to generate a follow-up prompt when presenting the user with trivia relevant to the currently discussed topics. However, the actual approaches to response generation are outside the scope of this thesis. We provide only the overview to provide a complete view on OOD handling.

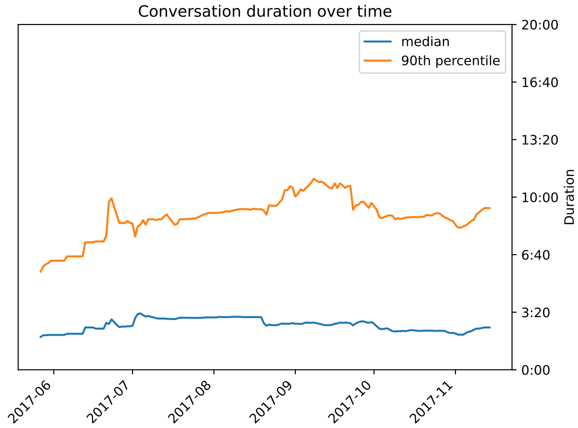
4.6 Evaluation

In this section, we show the overall evaluation of all described versions of the Alquist social bot. The objective evaluation of conversational systems is not a straightforward task. Besides the evaluation of individual tasks mentioned earlier in this chapter, we focus on the main metrics of the Alexa Prize Socialbot Grand Challenge, namely user rating and conversation duration. During each year of the competition, the average user ratings were provided on a daily basis, reporting the one-day average (L1d) and the last 7 days' average (L7d). The L1d values have a high fluctuation over time, while the L7d values provide a more stable view of the socialbot performance. Time conversation duration values were provided as the median duration over the last 7 days and 90th percentile duration over the last 7 days. The number of conversations during the middle phases of the competition was, on average, 6,000 per week and 11,000 per week during the final phases. The users provided ratings for approximately 15% of the conversations.

The results of the first version of the system called Alquist 1.0, are shown in Figure 4.12. The L1d values were not available during the first phase of the competition therefore, there are missing values at the beginning of the chart. As the first system in the row, it shows the lowest values at the beginning when not all of the components were ready. The system built on top of the Structured Topic Dialogues (STDs) allowed us to continuously improve the content coverage to create more engaging conversation. However, improving the content coverage using the monolithic approach stopped increasing both the rating and conversation duration in the second half of the competition, as shown in Figure 4.12a and Figure 4.12b. At the end of the competition, the Alquist 1.0 was a second-place winner with an average rating of 3.44 and a median conversation duration of 2 minutes and 6 seconds Table 4.18.



(a) Alquist 1.0 rating

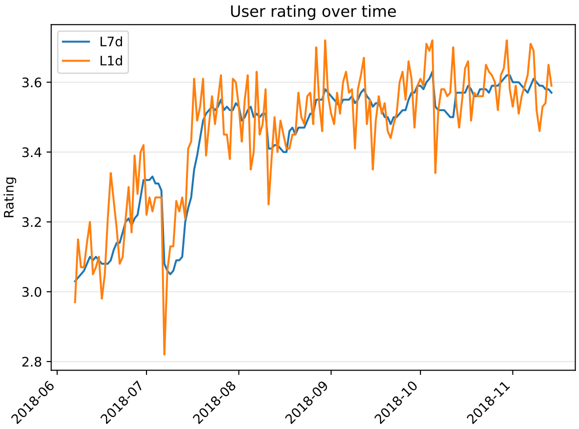


(b) Alquist 1.0 conversation duration

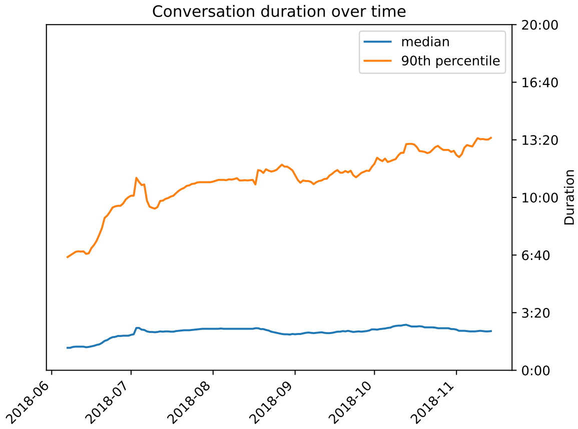
Figure 4.12: The overall performance of Alquist 1.0 throughout the competition

Redesigning the monolithic approach in Alquist 2.0 also resulted in a cold start (Figure 4.13) during the second year of Alexa Prize Socialbot Grand Challenge (APSGC) as the previously created dialogue structures needed to be redesigned in order to respect new NLU and DM approaches. Compared to the previous version, the sub-dialogue structures allowed more rapid development and, combined with separated topic-level and top-level intent recognition along with topic switch detection, contributed to longer and more engaging conversations. We managed to keep increasing the rating and duration throughout the entire duration of the competition. This was thanks to the variable flow allowed by dynamic topic switching, compared to the previous approach, where the topic switch could be executed only by matching a corresponding intent class. At the end of the competition, the Alquist 2.0 was once again a second-place winner with an increased average rating of 3.6, a median conversation duration of 2 minutes and 13 seconds, and a 90th percentile conversation duration of 12 minutes and 49 seconds (Table 4.18).

Alquist 3.0 leveraged the structure from its predecessor, resulting in high ratings and durations from the beginning. Additionally, the KG was initialized with multi-domain content using Wikidata. The adjacency-pair-based content was iteratively added on top of the sub-dialogues. The main contribution of the architecture was better flexibility, which led to ratings above 3.7 and 90th percentile durations above 16 minutes and 40 seconds in the middle of the conversation. However, the lack of handling a wide range of utterances eventually led to a decrease in the metrics. At the end of the competition, the Alquist 3.0



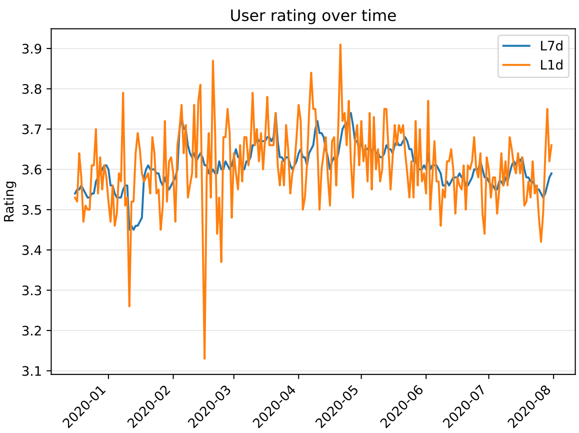
(a) Alquist 2.0 rating



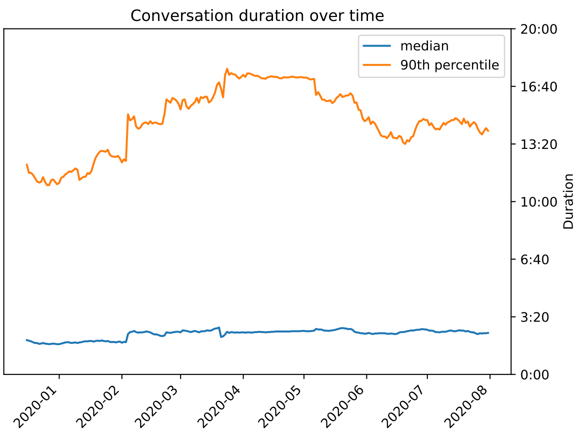
(b) Alquist 2.0 conversation duration

Figure 4.13: The overall performance of Alquist 2.0 throughout the competition

was a third-place winner with an average rating of 3.59, a median conversation duration of 2 minutes and 6 seconds, and a 90th percentile conversation duration of 12 minutes and 31 seconds.



(a) Alquist 3.0 rating

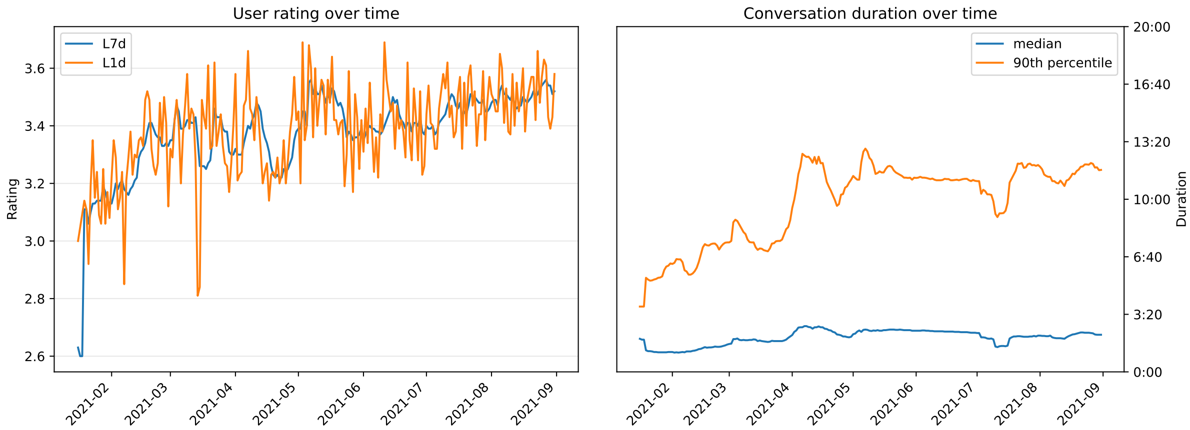


(b) Alquist 3.0 conversation duration

Figure 4.14: The overall performance of Alquist 3.0 throughout the competition

The hierarchical approach proposed as the architecture for Alquist 4.0 was compatible with the previous approaches, resulting in easy resuing of the existing content. The incorporation of OOD detection in combination with the generative models resulted in increasing rating and conversation duration. However, if we compare the absolute values of both ratings and duration with the previous year, we can see the values are lower overall. This trend was common for all of the competitors in APSGC and is also described by Hu et al. (2021). They report a 9.7% decrease in rating compared to the previous year. This was the first time when the ratings and durations decreased compared to the previous year of the competition. The possible explanation is that due to the rising popularity of the generative models, user expectations grew as well.

However, the variability in the discussed topics achieved by novel OOD detection coupled with NRG contributed to the coherence and engagingness significantly, resulting in the Alquist 4.0 winning first place with an average rating of 3.51, a median conversation duration of 1 minute and 30 seconds, and a 90th percentile conversation duration of 9 minutes and 11 seconds.



(a) Alquist 4.0 rating (b) Alquist 4.0 conversation duration

Figure 4.15: The overall performance of Alquist 4.0 throughout the competition

User ratings and associated conversation duration metrics are influenced by various factors, many of which are subjective in nature. Considering the difficulty of the task of objective conversation evaluation and the competition setup, where the goal is to compare the socialbots, these metrics represent a reasonable estimation of the socialbot quality. Additionally, some of the users provided us with text feedback. We use the feedback along with individual component benchmarks to estimate the contribution of individual parts of the proposed architectures. The summarized results of all socialbot versions are outlined in Table 4.18.

Table 4.18: The final performance of Alquist socialbot in Alexa Prize Socialbot Grand Challenges and their corresponding NLU approaches.

System	NLU	# Dialogues	Avg. Rating	Median Time	90th Percentile Time	Alexa Prize
Alquist 1.0	Single-level intents	16	3.44	2:06	8:24	2nd
Alquist 2.0	Topic-level+Topic Switch	164	**3.6**	**2:13**	**12:49**	2nd
Alquist 3.0	KG mapping	164	3.59	2:06	12:31	3rd
Alquist 4.0	Hierarchical+OOD	**273**	3.51	1:30	9:11	**1st**

4.7 Conclusion and Main Contributions

We introduced four versions of a socialbot architecture. The architectures in the respective competition concern every component relevant to the conversational system as described in chapter 2. This thesis contributed to the Natural Language Understanding (NLU) part of these architectures. The architecture designs in all versions of the socialbot focus on a hybrid approach, combining the end-to-end approach with a controllable dialogue design. This approach requires modifications on the NLU level in comparison to the standard approaches.

First, we introduced a monolithic approach with traditional single-level intent detection and entity recognition. The intent recognition operated across the Structured Topic Dialogue (STD) units. The lower-level NLU analysis was done using only a set of predefined rules. This approach formed a baseline for a socialbot, showing the strengths and weaknesses of the hybrid approach. The possibility to switch between retrieval, generated, or templated content is one of the biggest strengths. However, the balance between disruptive topic switching and unnatural continuation of the current flow is the challenge we identified for the upcoming versions. Overall, Alquist 1.0 was awarded as a second-place winner during the APSGC.

The second version contributed with a separation of top-level and topic-level intent recognition. Additionally, using a topic graph capturing hierarchical relations among individual topics with topic switch components allowed the users to navigate seamlessly through the conversational topics. The entity sentiment enhanced the conversation with a socialbot opinion about the highly discussed entities. Overall, Alquist 2.0 was also awarded as a second-place winner during the APSGC.

The still-challenging topic-switch detection task resulted in the architecture based on the elementary adjacent-pair-based units in Alquist 3.0. The NLU was focused on utterance segmentation and mapping each segment to a corresponding triple in the KG. It resulted in a very flexible dialogue design. Overall, Alquist 3.0 was also awarded as a third-place winner during the APSGC.

Alquist 4.0 was built on top of the Flowstorm platform, which was created based on the experience from the previous socialbot versions. The NLU contributions were inherited from the platform and are described in more detail in the following chapter. In summary, it includes the OOD detection that significantly contributes to the seamless inclusion of the generative models into to the hand-designed conversational flows. Overall, Alquist 4.0 was the first-place winner of the Alexa Prize Socialbot Grand Challenge (APSGC).

The contributions on the side of the architecture designs cover multiple parts of the system. The goal was not to focus on a single component but rather on the system as a whole, emphasizing the overall performance. As the systems were utilized in production and handled over 10 million conversational turns in total, we examined the performance considering both the key metrics and computational resources.

Chapter **5**

Modular Dialogue Architecture With Hierarchical Intent Recognition

As described in the previous chapter, we have explored various architectural modifications of a conversational system mainly oriented to an open-domain conversation. All of the described architectures have modularity and a hybrid approach as the key point. Additionally, we see the importance of effective dialogue design supported by a proper software tool. As outlined in chapter 2, minimizing manual work during the dialogue creation has always been one of the most critical focus points within the NLP research community. The very first systems required a lot of manual rules to establish a reasonable conversation on a very limited domain (Weizenbaum, 1966; Wallace, 2003). In contrast, the recent progress in Large Language Model (LLM) capabilities shows more than promising results of using such models controlled using simple prompts.

LLMs excel in completing provided input, in other words, generating the most probable sequence of tokens given the input sequence. Zhang et al. (2019) show the potential of using these models as an eng-to-end conversational model. The model is capable of generating grammatically correct and relevant sentences but has difficulties reflecting the user goal or taking a proactive role. Additionally, to enable the model to discuss a new domain, one must fine-tune it on the domain-specific data to reflect its specifics.

The concept of instruction-based models was introduced to minimize the data required for the model adaptation. The pioneers of the instruction-based model family are the later versions of Generative Pretrained Transformers by OpenAI (2022). The instructions for the model are provided in the form of natural language prompts, explaining the model what is its goal and what to reflect while generating the responses. Using the natural language prompts to explain the model's task is a few-shot learning approach (Brown et al., 2020). Besides the highly discussed problems such as model hallucination (Yao et al., 2023), the related problems include not reflecting parts of the prompts, especially forgetting the information provided at the beginning of the prompt sequence when the prompt has many tokens. The prompt decomposition, the introduction of conversational memory, and the introduction of external knowledge aim to mitigate the mentioned problems. Many tools have emerged to streamline the development processing using LLM with Langchain (Chase, 2022). All of these prompt-chaining tools and approaches rely on multiple calls of LLMs, which results in a significant latency on final response generation.

In this chapter, we propose a novel dialogue architecture that addresses the LLM-related problems above. The architecture was created as an evolution of hybrid approaches described in chapter 4. We created a platform called Flowstorm (Pichl et al., 2022) that uses the proposed dialogue architecture and allows developers to prototype and create new conversational content with minimal effort rapidly. The platform was successfully used as a base for the winning version of Alquist during the Alexa Prize Socialbot Grand Challenge 4 and for the Elysai application (Jakimiv et al., 2021).

The main idea of the architecture is a simplified dialogue design process by decomposing the large dialogue structure into small logical units (called sub-dialogues) as accented in Pichl et al. (2018) and Konrád et al. (2021). Besides the simple creation and maintenance of the sub-dialogue as part of the complex conversational application, it allows content reusability among various use cases without the need for retraining of the corresponding NLU models. Moreover, each of the sub-dialogue units has its own set of NLU (intent) models that can be easily combined with the additional NLU models to form a complex understanding throughout the application. The novelty lies in two major aspects. The first one is a modular intent recognition that allows a dynamic combination of multiple intent models based on the sub-dialogue hierarchy. The second one is a built-in out-of-domain (OOD) detection that handles situations beyond the scope of the designed dialogue. OOD is crucial in connection to the generative (LLM) models. It allows us to adaptively recognize the boundaries of the dialogue and trigger a relevant LLM model.

To maintain a reasonable response latency that is critical for the conversational system, especially in voice-enabled systems, we focus on models and algorithms that are less demanding on the computational resources.

Our main goal is to create an architecture that is primarily designed in a way that can be used to create complex and modular open-domain conversational systems. Thanks to its modularity, it can also be easily adapted for more narrow-focused, goal-oriented systems.

5.1 Related Work

5.1.1 Conversational Patforms

There are two categories of related conversational platforms: open-source frameworks or libraries and online platforms. Rasa (Bocklisch et al., 2017) and DeepPavlov (Burtsev et al., 2018) are the most famous representatives of the framework category. They provide the developer with a set of programmatic and declarative interfaces for defining and training natural language understanding (NLU) and dialogue management (DM) models. They also allow the creation of a pipeline of the components where the developer can create a sequence of models or components for utterance processing. Both libraries require a local installation following a non-negligible effort in integrating it with the rest of the conversational application.

Alexa Skills Kit (ASK) (Amazon.com, 2015), Dialogflow (Google, 2017), and Voiceflow (Voiceflow, 2019) are the representatives of the category of the online platforms. ASK and Dialogflow offer an online application for the definition of NLU models (intents and slots/entities). ASK introduced Alexa Conversations for dialogue management, allowing the developer to specify the conversation flow using sample conversations. Dialogflow CX (part of the Dialogflow providing a new way of working with the conversational agents) offers a visual view of the conversation agents to see the possible flows of the conversation. Voiceflow uses a visual dialogue editor as an approach to building a conversational application. Dialogue blocks can be connected via transitions to create a flow of blocks where each block represents a specific functionality, such as keyword matching or API calls.

5.1.2 Intent Recognition and Out of Domain Detection

We described the related work of intent recognition in subsection 2.7.2 in detail. Overall, it follows the progress of text classification algorithms that have evolved from simple approaches using BOW or one-hot representation with classifiers such as SVM or logistic regression to more complex CNN, RNN, and transformer-based neural networks. Most recently, the zero- or few-shot learning approaches using LLMs are increasingly popular.

The Out of domain (OOD) detection can be divided into two main streams. The first stream uses data representing OOD utterances, and the second stream estimates thresholds or boundaries of in-domain data and treats the rest as OOD. Larson et al. (2019) show several approaches, including training an intent classifier with an additional class representing the OOD or finding a threshold of the intent classifier using the OOD data. Additionally, Zhang et al. (2021) propose learning an adaptive boundary around in-domain classes to reflect different in-class variances.

5.2 Architecture Description

We design a conversational application as a set of connected dialogue flows. We call each dialogue flow a sub-dialogue (see subsection 5.2.1). Each sub-dialogue is focused on a small subset of a domain (e.g., in the movie domain, one sub-dialogue can be focused on a user's favorite movie). Each application has at least one sub-dialogue but typically consists of a large number of sub-dialogues. Let's denote the sub-dialogue that is triggered when the application is launched as "main". The other sub-dialogues are triggered based on the conversational design and current context. The sub-dialogues can be statically linked as parts of other sub-dialogues or can be dynamically selected based on the specific selection logic (Marek, 2023).

Formally, let's denote an application A and a set of n sub-dialogues $\mathcal{S} = (S_0, S_1, \dots, S_n)$. The application A is then defined as:

$$A = (\mathcal{S}, S_m),$$

where $S_m \in \mathcal{S}$ is the main sub-dialogue. A sub-dialogue may be directly referenced by one or many other sub-dialogues which may be denoted as an adjacency matrix $I^{n \times n}$:

$$I_{i,j} = \begin{cases} 1 & S_j \text{ is a sub-dialogue referenced from } S_i \\ 0 & \text{otherwise} \end{cases}$$

When a sub-dialogue S_j is referenced from S_i, it means that there is a node (see below) in the graf structure of S_i directly referencing S_j. The other option to include sub-dialogues in the application is to add them to a set $\mathcal{S}_{\text{sel}} \subseteq \mathcal{S}$ from which they can be selected using a selection logic. Please note that the selection logic is not in the scope of this thesis and, therefore, will not be discussed in detail.

One of the key ideas is the reusability of individual sub-dialogues. The architecture, as well as the platform itself, allows multiple references to the same sub-dialogues. It results in an adaptive conversational flow while keeping the application resources easily maintainable. However, you can launch each sub-dialogue independently and test it without connecting to the rest of the application. Due to the intent hierarchy as described in subsection 5.3.2, a different set of global intents may come into play in each situation.

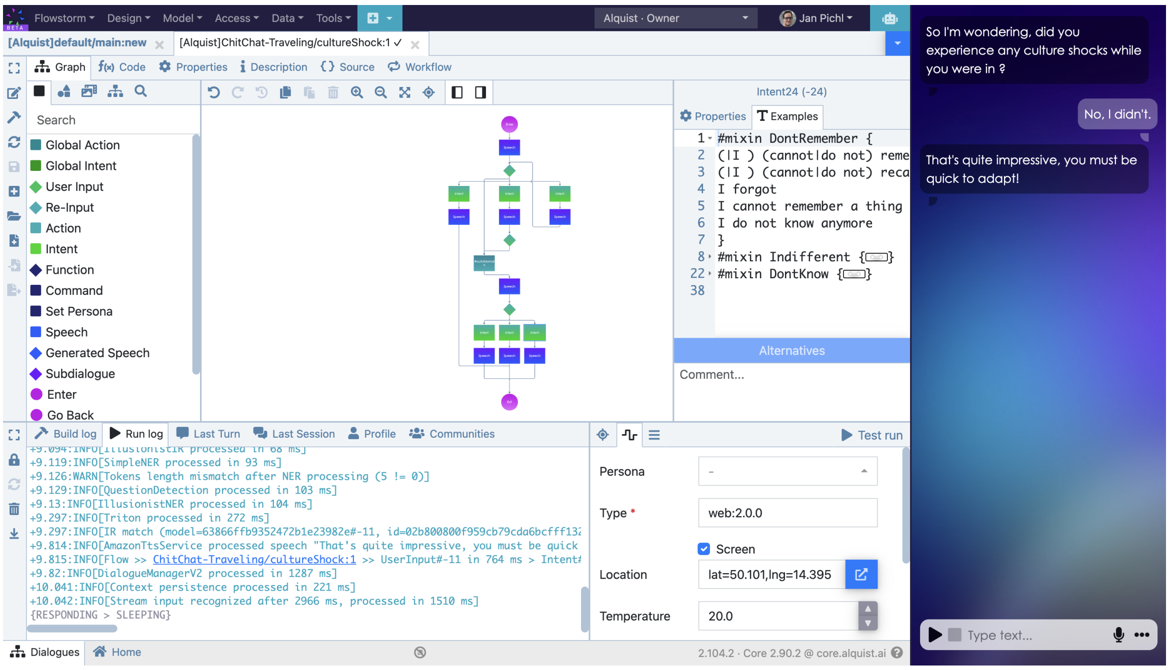


Figure 5.1: Web UI of Flowstorm platform showing the main visual flow editor. The conversational editor (in the middle) shows a particular sub-dialogue consisting of several nodes. Additionally, there is a client for instant testing in the right panel and a conversational log for debugging at the bottom.

5.2.1 Sub-dialogue

Sub-dialogue, as the main unit forming the conversational application, is a graph structure consisting of connected nodes representing the flow of the conversation. Formally, it is an oriented graph defined as follows:

$$S_i = (\mathcal{N}, \mathcal{E}),$$

where $\mathcal{N}$ is a set of nodes and $\mathcal{E}$ is a set of oriented edges. Each node $N \in \mathcal{N}$ has its type t. Each node type plays a different role in the conversation. The key node types are described in the following list. The basic structure and node types are shown in Figure 5.2.

- **Enter** – Entry point of the sub-dialogue. Each sub-dialogue must have exactly one Enter.
- **Speech** – Speech node contains one or more natural language responses that are presented to a user. If there are multiple responses, one is randomly selected. It can also contain slots that are eventually filled with variables based on the context.
- **Generated Speech** – Generated Speech node references an LLM. It contains a prompt and model parameters. Once reached in the conversational flow, the LLM is queried with the given prompt, and the generated response is streamed to the output.
- **User Input** – Point in a conversation where the bot waits for the user utterance. It is typically connected to multiple intent nodes and optionally to action nodes. Each User Input node has its own underlying intent recognition model trained separately. It has as many intent classes as there are intent nodes connected to it.
- **(Global) Intent** – Intent node contains examples of the users' possible utterances, which serve as training data for the intent recognition model. Each intent represents a single class in the intent recognition model. There are two types of intent nodes: contextual Intent and Global Intent. Contextual intents can be recognized only at a specific point of the conversation—when the conversation flow reaches the User Input node they are connected to. Global Intents, however, can be recognized at any point in the conversation.
- **Function** – Function node contains code with a custom logic. The code can contain arbitrary logic (usually working with attributes — see the attributes subsection). The function returns the transition to the next dialogue node. It is suitable for branching the dialogue based on the attribute values, API results, etc.
- **(Global) Action** – Actions represent specific situations in the conversation flow. There are several predefined situations: *Silence*, *Error* and *Out of domain*. *Silence* action is triggered when no speech is recognized. *Error* action is triggered when the logic written in the Function nodes fails (connection errors or bad design). *Out of domain* action represents the utterance that does not belong to any of the intent classes. This will be described in detail in the following sections.

- **Sub-dialogue** – Sub-dialogue node introduces the ability to reference another sub-dialogue. When the conversation flow reaches this node, it triggers the referenced sub-dialogue, processing its logic from the Enter node to the Exit.
- **Exit** – Exit node represents the end of the sub-dialogue flow. If the sub-dialogue is the "main" dialogue, the session is ended. If it is a lower-level sub-dialogue, the conversation flow jumps to the parent sub-dialogue (i.e., the one where the current sub-dialogue was referenced).

Additionally, each subdialogue has the following properties that can be referenced from the nodes.

Init code – Each sub-dialogue has a declarative code part that typically serves as a place to define attributes (see below) and methods that can be called from the function nodes.

Attributes – There are four scopes of the attributes that can be defined in the Init code. The attribute stores values that can be used to modify the conversation flow, or they can be presented directly as part of a response. Each attribute must have its default value which is used to initialize the attribute based on its scope.

1. *Turn* – the value is reset to default at the beginning of each turn.
2. *Session* – the value is reset to default at the beginning of each session.
3. *User* – the default value is used for each new user. Once the value of a user attribute is set, it is valid for all sessions of the user.
4. *Community* – the default value is used for each community namespace. Once the value of a community attribute is set, it is valid for all users using the same community namespace.

Figure 5.3 shows the scenario when a sub-dialogue B is directly referenced from a sub-dialogue A. Once the flow reaches the *Intent 2* node, the next edge enters the sub-dialogue B, starting with the Enter node. Analogically, after exiting through the *Exit* node, the flow continues with the outgoing edge from *Subdialogue* node in sub-dialogue A.

For the intent recognition described in the following sections, it is important to introduce the Dialogue Stack mechanism when dealing with multiple sub-dialogues. The Dialogue Stack is initialized with a reference to the main sub-dialogue once the conversation is started. When another sub-dialogue is entered, its reference is pushed to the top of the stack. Analogically, when the sub-dialogue is exited, the top reference from the stack is removed.

5.3 Natural Language Understanding

This thesis focuses on Intent Recognition as the critical part of Natural Language Understanding. The novel Intent recognition approach is based on our proposed conversational

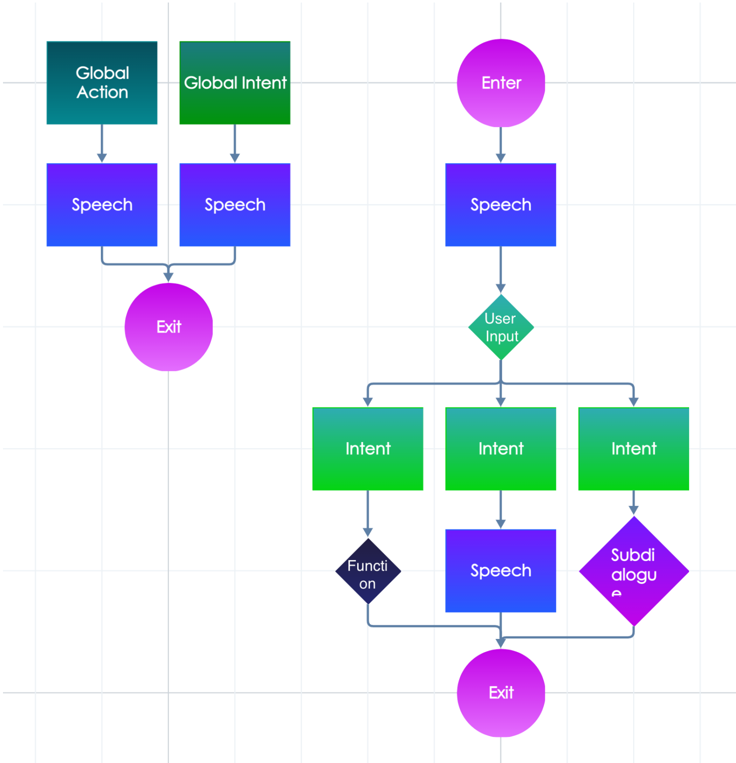


Figure 5.2: Nodes connected to the sub-dialogue structure. The names of the node types are shown in the corresponding boxes in the picture.

system architecture. The essential part of the proposed approach is an out-of-domain detection. Additionally, we briefly mention the Entity Recognition task as we apply entity masking as a prerequisite for Intent Recognition.

5.3.1 Entity Recognition and Masking

We use a combination of state-of-the-art methods with simpler algorithms to achieve the best accuracy while maintaining reasonable training and inference times. These methods include BERT (Devlin et al., 2019) model (with a token classification head), Bi-LSTM-CRF architecture (Huang et al., 2015) and a Regex-based tool Duckling[1] (used for entities with specific structure which can be easily described using regular expressions—date, time, numbers, URLs, amount of money, etc.).

We created a custom annotation interface focusing on dataset creation and augmentation rather than on the annotation of existing samples. The data format consists of the context sentence, i.e., the sentence where the entity may occur and the actual entity value. For example, the contextual sentence is `My favorite movie is {movie}`, and the entity values for type `movie` are `Matrix, Gladiator, Forest Gump`. Before training the model,

[1] https://github.com/facebook/duckling

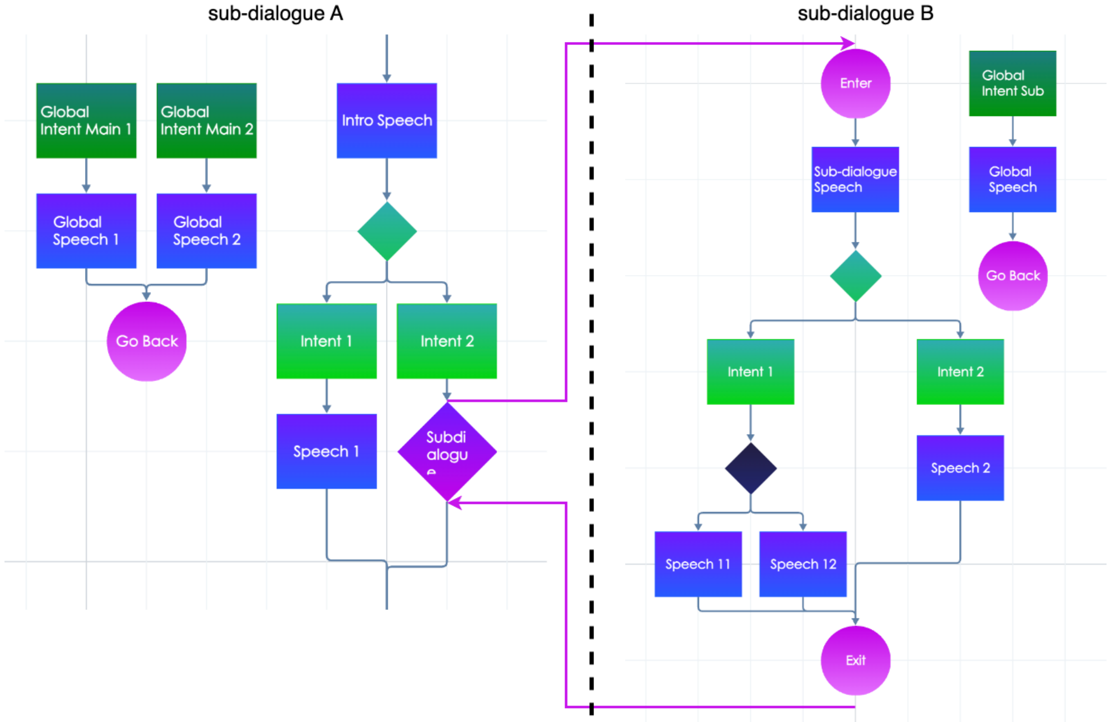


Figure 5.3: Direct referencing a sub-dialogue from another one.

the examples are expanded in a way that each contextual sentence is used with each entity value, e.g., `My favorite movie is Matrix`, `My favorite movie is Gladiator`, etc. This may lead to a poor dataset as the samples differ only in the entity, and the rest of the sentence is the same. Therefore, we apply a query rewrite (Ye et al., 2023) of the original sentence. Additionally, we may specify the contextual sentence directly with the entity value using the following form: `My favorite movie is [Matrix]{movie}`.

The recognized entities are masked out in the user utterance prior to the intent recognition phase. This step allows the intent recognition model to focus only on the types of entities rather than the actual values. The intent recognition module receives, for example, the sentence `My favorite movie is {movie}` instead of `My favorite movie is Matrix`. Only the entity types included in the intent examples are masked out.

5.3.2 Hierarchical Intent Recognition

The Hierarchical Intent Recognition is a novel approach highly connected to our modular architecture. We introduced the concept of sub-dialogues, each consisting of local and global intents. Each sub-dialogue may contain multiple global intents and multiple user input nodes with various numbers of local intents connected to it. For a sub-dialogue S_d, let the

$$\mathcal{G}_d = (g_1, \ldots, g_n)$$

be a set of global intents of the sub-dialogue S and let the

$$\mathcal{U}_d = (u_{d,1}, \ldots, u_{d,m})$$

be the set of User Input nodes where each node $u_{d,i}$ has its own set of local intents

$$\mathcal{L}_{d,i} = (l_1^{d,i}, \ldots, l_{k_i}^{d,i}).$$

There is one model trained for each $u_{d,i} \in \mathcal{U}_d$ that classifies the user utterance into $|\mathcal{L}_{d,i}|$ classes. Additionally, one extra model is trained for global intents with $|\mathcal{G}_d|$ classes.

5.3.3 Global Intents Hierarchy

The intent recognition step is triggered whenever a user reaches the User Input node, i.e., during each conversational turn. Different intent classes are considered during each turn, and the selection depends on two factors: the set of contextual intents connected to the current User Input node and sub-dialogues referenced in the dialogue stack.

Assume the following situation. Dialogue stack $DS = (S_n, S_{n-1}, \ldots, S_1)$, where S_n is on the top of the stack and S_1 is the main sub-dialogue. Then, the conversation flow reached the User Input node $u_{n,k}$ in the sub-dialogue S_n. Given the dialogue stack and the user input node, the relevant intent classes for this situation are the following:

$$\mathcal{I}(DS, u_{n,k}) = \mathcal{L}_{n,k} \cup \bigcup_{i=1}^{n} \mathcal{G}_i$$

In other words, we take all intent classes relevant to the current user input node and global intent classes from all sub/dialogue from the dialogue stack. The intuition behind that is the following. The contextual (local) intents are relevant only for a specific point in the conversation, e.g., yes or no for a question asked in the previous turn. However, global intents are relevant for each point in the conversation. Moreover, the inherited global intents from dialogues on the dialogue stack reflect the global capabilities of the entire application. For example, let's assume a conversation about sports. It contains a main dialogue with several global intents, e.g., "help" and "sports rules". This sub-dialogue also references another sub-dialogue focused on football. It contains global intent about "football rules". When this sub-dialogue is reached through the main, the dialogue stack contains references to both of these sub-dialogues. Therefore, both "help", "sports rules" and "football rules" are active global intents. Since it is likely for sports rules and football rules to have similar training examples, football rules are prioritized since they are closer to the top of the dialogue stack, which is the most relevant for the current scope.

5.3.4 Out of Domain Detection

During each dialogue turn, the utterance is classified into one of the intent classes as described above. However, the utterance does not have to correspond to either of the

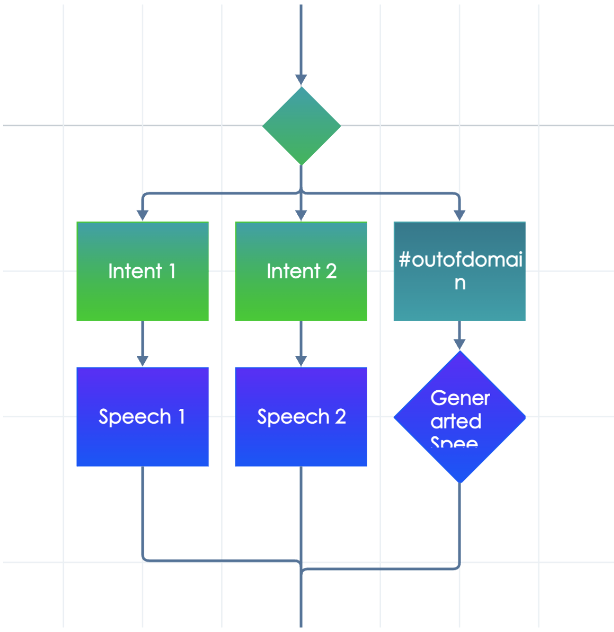


Figure 5.4: Typical flow control when a Generated Speech is triggered after the out-of-domain is detected.

classes, i.e., it is an Out of domain (OOD) input. In the context of the proposed dialogue architecture, the utterance is considered to be OOD if none of the active local and global intents classes are relevant. In other words, the utterance may be from the same topical domain, and it is also considered OOD if a suitable intent class does not exist. This approach is useful to be used with the combination of generative LLMs. The dialogue design is then focused only on the desired situations, and the rest is handled using LLM.

We design the actual detection as part of our intent recognition approach. Therefore, the relevant intent classes for the given dialogue stack and user input nodes are extended with an additional OOD class:

$$\mathcal{I}(DS, u_{n,k}) = \mathcal{L}_{n,k} \cup \bigcup_{i=1}^{n} \mathcal{G}_i \cup \{I_{OOD}\}$$

From the perspective of the dialogue design procedure, it is important for the detection approach not to require additional training samples for the OOD class, and we rely purely on the in-domain samples.

When the OOD is recognized, the dialogue flow continues with a local OOD action (if connected to the current User Input node) or with a global OOD action. The flow snippet, when the local action is used, is shown in Figure 5.4.

5.3.5 Monolithical Single Model with Masking

To set a baseline for hierarchical intent recognition, we start with a single model per application. The application consists of multiple sub-dialogues. Each sub-dialogue has its own set of global intents and user input nodes with corresponding local intents. However, only a subset of all intent classes are relevant in each dialogue turn, as described earlier. Although there is a finite number of user input nodes and possible dialogue stack combinations given the application, training a separate model per combination would result in an extremely large number of models. Therefore, we need to apply a masking mechanism to dynamically turn off the intents that are not relevant to the current state of the conversation.

Given the application and set of sub-dialogues, $A = (\mathcal{S}, S_m), \mathcal{S} = (S_0, S_1, \ldots, S_n), S_m \in \mathcal{S}$ and global intents $\mathcal{G}_i$ and local intents $\mathcal{L}_{i,j}$ per each sub-dialogue S_i. The set of all intent classes is the following:

$$\mathcal{I}(A) = \bigcup_{i=0}^{n} \left(\mathcal{G}_i \cup \bigcup_{\mathcal{L}_{i,j} \in u_{i,j}, u_{i,j} \in \mathcal{U}_i} \mathcal{L}_{i,j} \right) \cup \{I_{OOD}\}$$

Then the masked intents are:

$$\mathcal{I}(A) \setminus \mathcal{I}(DS, u_{n,k}) = \bigcup_{i=0}^{n} \left(\mathcal{G}_i \cup \bigcup_{\mathcal{L}_{i,j} \in u_{i,j}, u_{i,j} \in \mathcal{U}_i} \mathcal{L}_{i,j} \right) \setminus \left(\mathcal{L}_{n,k} \cup \bigcup_{\mathcal{G}_j \in S_j, S_j \in DS} \mathcal{G}_j \right)$$

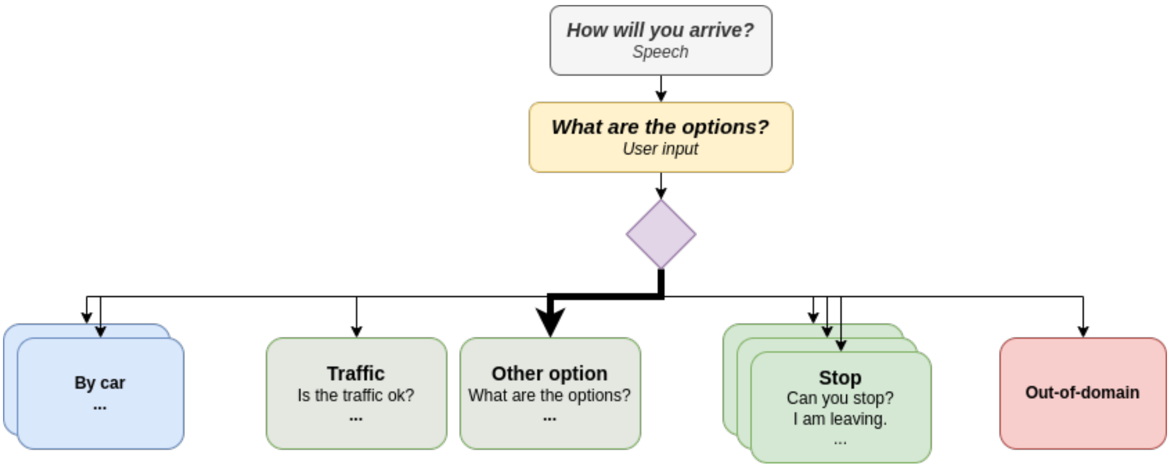


Figure 5.5: Single model intent recognition using a combination of all global and local intent per application plus the OOD class. Blue nodes represent the local intents, dark green nodes are global intents from the sub-dialogue on top of the stack, light green are the remaining global intents, and red is the OOD class.

The decision schema with intent examples is shown in Figure 5.5. This approach does not require an additional classification combination since we classify all relevant classes using a single model. However, we need to train a single model per application, which results in the following inefficiencies from the dialogue architecture point of view:

1. Classifying into classes that are apriori irrelevant and masked out

2. User inputs with the same (or similar) class sets may be used in different sub-dialogues. The model does not have the information on which one of them is relevant for the current turn.

3. Different models and, therefore, different results for a classification in particular sub-dialogues used in different applications.

5.3.6 Try-Catch Mechanism

We propose the try-catch mechanism to address the issues mentioned in the previous sections. The main goal is to create a model or multiple models per sub-dialogue that can then be combined with other models to cover all relevant classes in the current turn. This approach is based on intuition about human reasoning during a conversation. When we evaluate an utterance of our communication partner, we typically expect the utterance to be relevant to the previous context of the conversation. For example, when talking about weather and we ask *"What is the weather outside"*, we probably expect the response to be something like *"It is sunny"* or *"It is raining outside"*. However, the response can be *"Why are you asking?"* or even *"Hm, I am hungry"*. Even though the last response is not probable, given the previous context, it still may occur. When categorizing the utterance of our communication partner, we tend to prioritize the "classes" that are closely related to the previous context. This fact is related to the adjacency pairs discussed in subsection 4.4.3 and to the topic relevancy.

The approach uses a bottom-up strategy. When we start at the local intent level and when no class is matched, we continue with the global intent classes per each sub-dialogue on the dialogue stack. In this scenario, there is an individual model $m_{u_{n,k}}$ for each user input node $u_{n,k}$, where S_n is the sub-dialogue on the top of the dialogue stac DS and k is the index of user input node in the sub-dialogue. Additionally, model $m_{\mathcal{G}_i}$ classifies into the global intent classes $\mathcal{G}_i$.

The OOD class is returned only if none of the classification layers returns an in-domain class. The diagram analogical to the previous approach is shown in Figure 5.6.

5.3.7 Routing Mechanism

The routing, also used in our system by Konrád et al. (2021) described in chapter 4, mechanism approach preserves the usage of individual models per user input node and per global intents in each sub-dialogue. Additionally, we propose to use one high-level model that decides which of the lower-level models is used for the final classification. The important requirement for the high-level model is to be generic enough and does not require retraining when the underlying models change. Therefore, we selected a similarity function that can be used to compute the similarity between the input and variable set of representatives. The classification is then two-step. First, we use the similarity function to select a layer and its corresponding lower-level model. Additionally, we take the selected lower-level model to classify the final intent class.

Algorithm 5.1 Try-Catch Intent Recognition Algorithm

```
function CLASSIFY_INTENT(utterance, u_{n,k})
    model ← get_model(u_{n,k})
    intent_class ← classify(model, utterance)
    if is_ood(intent_class) then
        return CLASSIFY_GLOBAL(utterance)
    else
        return intent_class
    end if
end function
function CLASSIFY_GLOBAL(utterance)
    DS ← get_dialogue_stack
    for S_i in DS do
        model ← global_model(S_i)
        intent_class ← classify(model, utterance)
        if not is_ood(intent_class) then
            return intent_class
        end if
    end for
    return OOD
end function
```

Given the dialogue stack DS of size n, the high-level model needs to classify into $n+1$ layers, i.e., one layer for each intent set $\mathcal{L}_{n,k}, \mathcal{G}_0, \ldots, \mathcal{G}_n$. Then we use one of the corresponding models $m_{u_{n,k}}, m_{\mathcal{G}_0}, \ldots, m_{\mathcal{G}_n}$. The high-level classifications use the intent examples from the intent classes from all underlying layers. Each example is then embedded into a vector space (see section 5.5). We calculate a centroids $\boldsymbol{c}_{\mathcal{L}_{n,k}}, \boldsymbol{c}_{\mathcal{G}_0}, \ldots, \boldsymbol{c}_{\mathcal{G}_n}$ for each layer by averaging the vector representations of intent samples corresponding to each layer. We take all training samples of all corresponding intent classes and create their vector representations $\boldsymbol{e}_j \in \mathbb{R}^m$, where m is the dimension of the embedding vector and calculate the centroids:

$$\boldsymbol{c}_{\mathcal{L}_{n,k}} = \frac{1}{n} \sum_{j=1}^{n} \boldsymbol{e}_j, \text{ where } n = |\mathcal{L}_{n,k}|$$

$$\boldsymbol{c}_{\mathcal{G}_i} = \frac{1}{n} \sum_{i=j}^{n} \boldsymbol{e}_i, \text{ where } n = |\mathcal{G}_i|, \forall \mathcal{G}_i \in S_i, S_i \in DS$$

During the classification phase, the similarity between the input utterance and each centroid is estimated. Based on the most similar centroid, the lower-level model is selected and used for the classification.

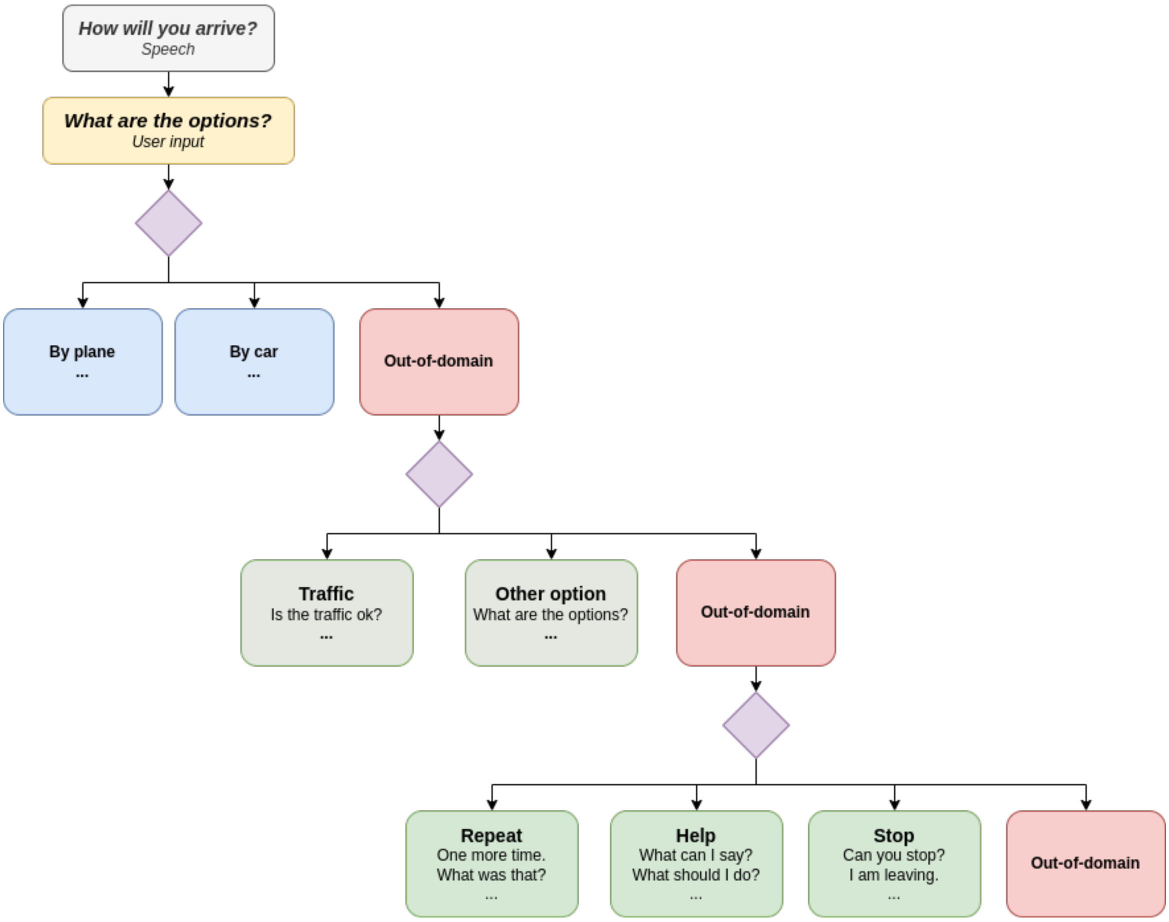


Figure 5.6: Hierarchical try-catch approach starting with a local intent model. Each purple diamond is a different model corresponding to a given layer.

5.4 Datasets

We prepared our in-house dataset for the experiments with the hierarchical intent recognition approaches described earlier. We identified the need for a custom dataset while analyzing the publicly available intent recognition datasets that also include out-of-domain samples. The reasons are summarized in the subsection 5.4.2. Additionally, we needed to create a dataset that did not contain proprietary user input queries collected during the Amazon Alexa Prize Socialbot Grand Challenge 4. The overall goal of our dataset is to make it challenging for our use-case of hierarchical intent recognition. As you can see in Table 4.16, the F1 scores on the CLINC150 dataset are in most cases above 95% indicating that the dataset is not challenging enough thus not reflecting the real-world scenarios.

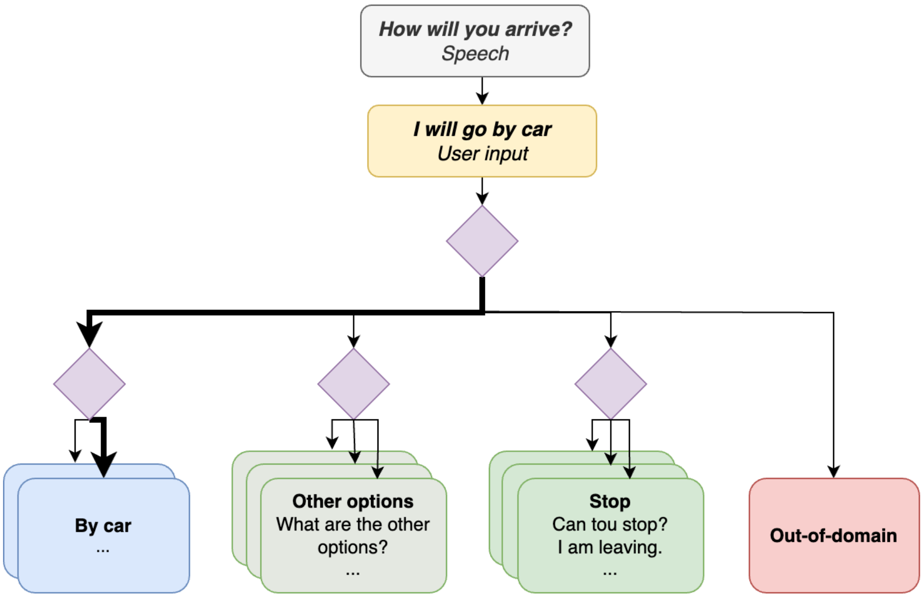


Figure 5.7: Routing mechanism schema

5.4.1 Data Collection

Data collection of the dataset was performed as a multiple-step process leveraging content created in the Flowstorm platform (Pichl et al., 2022) using its web UI and additional data collection as part of the Wizard-of-Oz Kelley (1984) experiment:

- We explored the content created using the Flowstorm platform. Each content contains intent examples assigned directly to the corresponding intent classes. We used content created for the Elysai application (Jakimiv et al., 2021) and additional chit-chat-oriented content. We filter out duplicate intent classes and classes containing less than ten examples.

- We performed the Wizard-of-Oz (Kelley, 1984) experiment with 15 participants. Participants were asked to answer all collected questions. We collect all given answers (like "I am going shopping" or "I have no idea"). We encourage the participants to provide four types of answers: (1) related to the question, (2) related to the given topic but not to the question, (3) related to all covered topics, and (4) not related to the question or covered topics—Out-of-Domain. Please note that this process is the only source of OOD examples as the design process in Flowstorm does not require explicit OOD specification.

- We semi-automatically cluster each type of provided answers—grouping by semantic similarity (using cosine similarity) and then manually stabilizing data

- Lastly, we shuffle and divide the examples into a train, valid, and test split in a ratio of 6:2:2.

Despite the fact that the hierarchical dialogue may have an arbitrary number of layers, we created the dataset to have three layers: local intents, global intents and OOD. Global intents and OOD is always specific for the given set of local intents.

Since the data collected comes from the Flowstorm conversation and the dataset reflects the hierarchical intent structure, we call it the Flowstorm Hierarchical Intent (FHI) dataset.

5.4.2 Datasets Comparison

As outlined at the beginning of this section, we struggled with the usage of the existing dataset for the purpose of testing the proposed novel hierarchical intent architecture. Although we showed preliminary hierarchical intent recognition experiments and results in the final part of the chapter 4 on the proprietary ALQUIST 4.0 (Konrád et al., 2021) and artificially augmented CLICNC150 (Larson et al., 2019) datasets, we created a novel dataset suitable for hierarchical intent recognition task focus on multi-level hierarchy and OOD detection.

One of the strongest points of our dataset is the context-aware OOD approach. In contrast, the dataset proposed by Larson et al. (2019) and Gangal et al. (2020) contains only OOD examples that are not specific to the given topic. On the other hand, the dataset by Arora et al. (2020) contains OOD examples specific to each domain, but our dataset is superior regarding the number of examples and granularity of topics, making our dataset more challenging. Moreover, all of the mentioned datasets have the intents on the same level, which does not fit into the hierarchical scenario.

We created a dataset containing 11,807 intent samples in total. Compared to the other datasets, CLINC150 has 23,700 samples, ROSTD has 47,590 in total for newly created out-of-domain and previously created in-domain samples, and HINT3 has 5,242 samples. To reflect the real-world scenario, we tried to keep the sample per intent class on a minimal size, which is 52 on average. Additional statistics are shown in Table 5.1.

Table 5.1: Comparison of our dataset with previously proposed OOD datasets. Decision points correspond to the user input nodes in our dialogue architecture. For the other datasets, the decision point corresponds to a single domain.

Dataset	**Decision points**	**Context-aware OOD**	**# Intents per decision point**	**# Intents total**	**# Topics**	**Hierarchical structure**
FHI (ours)	**49**	**Yes**	Avg. 18	**227**	**12**	**Yes**
CLINC150	10	No	Avg. 15	150	10	Yes
ROSTD	3	No	Avg. 4	12	3	No
HINT3	3	No	**Avg. 36**	108	3	No

5.5 Experiments

In this section, we run the experiments on the three proposed hierarchical intent recognition approaches. We utilize the combination of pre-trained language models for text embeddings together with a classification model. We apply a threshold mechanism with minor adjustments for each proposed method for the OOD classification. Generally, we set the threshold for each intent class individually. Let's define $F(u)$ as an intent classification function of user utterance u, intent class $I_i \in \mathcal{L} \cup \mathcal{G}$, and a probability $p^{method}(u, I_i)$ of u belonging to the class I_i using one of the *method* (single model, try-catch, routing). Then:

$$F(u) = \begin{cases} I_i \text{ if } \exists I_i \in \mathcal{L} \cup \mathcal{G} : p^{method}(u, I_i) \geq T_i \\ OOD \text{ otherwise} \end{cases}$$

For the single model, $p^{method}(u, I_i)$ is estimated as a softmax probability of the corresponding class. For the try-catch approach, we estimate the probability and compare it with the threshold at each level individually. Finally, for the routing mechanism, we compute the similarity between the utterance and all the centroids.

5.5.1 Metrics

To find a significant difference in the performance of the OOD detection, we used standard evaluation metrics (Larson et al., 2019; Tan et al., 2019b): Accuracy, Recall, and False rejection rate (FRR). More specifically, for our three-layer structure, we define Local In-domain Accuracy, Global In-domain Accuracy, Out-of-domain recall, Local False Rejection Rate, and Global False Rejection Rate. The suggested division of metrics can be easily extended to more layers.

Table 5.2: Possible combination of predicted and true intents used in the 2-layer structure

		Predicted intent		
		Local intent	*Global Intent*	*Out-of-Domain*
True intent	*Local Intent*	L-TP	L-FP	L-OOD
	Global Intent	G-FP	G-TP	G-OOD
	Out-of-Domain	OOD-FP	OOD-FP	OOD-TP

Out-of-domain Recall is computed as:

$$\frac{\textbf{OOD-TP}}{\textbf{OOD-TP} + \textbf{OOD-FP}}$$

Local False Rejection Rate is defined as :

$$\frac{\textbf{L-FP} + \textbf{L-OOD}}{\textbf{L-TP} + \textbf{L-FP} + \textbf{L-OOD}}$$

and finally Global Ralse Rejection Rate as:

$$\frac{\textbf{G-FP + G-OOD}}{\textbf{G-TP + G-FP + G-OOD}}$$

Similar to Tan et al. (2019b), we used the Local False Rejection Rate and Global False Rejection Rate as critical metrics suitable for coherent conversation (Konrád et al., 2021). In general, both metrics reflect what portion of the In-domain examples system misclassifies as the Out-of-domain class. The low false rejection rate shows the desired performance for the user of conversational agent because, as Candello and Pinhanez (2018) shown, misclassification of the In-domain class into the Out-of-domain class is more problematic to coherency.

5.5.2 Pretrained Language Models

Vector representation of a sentence shows promising results in different text classification tasks (Cer et al., 2018). As shown by Reimers and Gurevych (2019), the possibility to precompute sentence embedding beforehand is crucial in real-world systems, where we need to decrease the response time to a minimum. After following this recommendation, we use the following vector representations of a sentence:

- Universal Sentence Encoder (USE) Cer et al. (2018)
- ParaphraseBERT (Ko and Choi, 2020) implemented by Reimers and Gurevych (2019)
- MPNet (Song et al., 2020) implemented by Reimers and Gurevych (2019)
- MiniLM (Wang et al., 2020)
- DistilBert (Sanh et al., 2019)
- Albert (Lan et al., 2019)
- BERT (Devlin et al., 2019)

All algorithms show promising results in sentence similarity tasks (Reimers and Gurevych, 2019; Cer et al., 2018) and allow precompute vector representation of sentences beforehand. For USE, we used two different versions, one based on Deep average networks (DAN)[2] and the other trained with a Transformer encoder (TRAN)[3]. ParaphraseBERT was used in version with 3 encoders (ParaphraseBERT-Small)[4] and 12 encoders (ParaphraseBERT-Big)[5]

[2] https://tfhub.dev/google/universal-sentence-encoder/4
[3] https://tfhub.dev/google/universal-sentence-encoder-large/5
[4] https://huggingface.co/sentence-transformers/paraphrase-MiniLM-L3-v2
[5] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

5.5.3 Classification Model

In addition to the embedding representation, we experiment with several classification models. We did the experiments with the models on all of the mentioned hierarchical approaches with one exception. The routing mechanism uses cosine similarity in all cases on the top level, where it computes the similarity between the input utterance and all of the centroids representing individual levels. The lower uses one of the mentioned models:

- 2-layer Neural Network – feed-forward network, both hidden layers have the size of 128, ReLu activation
- 1-layer Neural Network – feed-forward network, hidden layer have the size of 128, ReLu activation
- Random Forest – 128 estimators, Gini impurity criterion
- Logistic Regression – L2 penalty, inverse regularization strength 1
- Cosine Similarity

5.5.4 Results

We provide results of different embeddings in all of the three proposed approaches. We measure the OOD recall, false rejection rate of local intents, and false rejection rate of global intents. In other words, the embedding selection is focused on the OOD performance, and the results are shown in Tables 5.3, 5.4, and 5.5. The inference time are measured on AMD EPYC 7V12 CPU. The in-domain classification accuracy is shown in the other half of this subsection.

Table 5.3: Results of different embeddings used in Monolithical Single Model with Masking on FHI dataset.

	OOD Recall ↑	**Local FRR ↓**	**Global FRR ↓**	**Inference time ↓**
USE-DAN	59.0	**7.8**	9.2	**0.75ms**
USE-TRAN	61.4	9.5	9.7	2.8ms
MPNET	**63.8**	9.4	**7.1**	10ms
DistilRoberta	62.2	8.6	11.3	5ms
MiniLM-L12	58.9	9.9	13.8	4.2ms
MiniLM-L6	57.7	10.2	14.2	1.6ms
qa-distilbert	59.1	10.4	14.4	5.2ms
par. MPNET	56.3	9.6	12.5	11ms
par. albert-small	55.8	8.9	15.2	21ms
par. MiniLM-L12	56.2	11.0	16.1	3.2ms
par. MiniLM-L3	59.2	10.8	12.7	1.4ms
BERT	47.9	9.6	22.1	32ms

Table 5.4: Results of different embeddings used in Try-Catch Mechanism on FHI dataset.

	OOD Recall ↑	Local FRR ↓	Global FRR ↓	Inference time ↓
USE-DAN	72.5	**7.7**	12.4	**0.75ms**
USE-TRAN	**75.4**	8.9	13.1	2.8ms
MPNET	75.3	10.5	**11.3**	10ms
DistilRoberta	70.2	8.4	15.6	5ms
MiniLM-L12	69.2	9.2	16.4	4.2ms
MiniLM-L6	67.0	7.7	17.1	1.6ms
qa-distilbert	72.5	8.5	14.5	5.2ms
par. MPNET	71.7	10.2	13.9	11ms
par. albert-small	64.6	8.6	14.8	21ms
par. MiniLM-L12	64.7	10.3	19.4	3.2ms
par. MiniLM-L3	72.3	10.4	14.7	1.4ms
BERT	51.2	9.9	23.5	32ms

Table 5.5: Results of different embeddings used in Routing Mechanism on FHI dataset.

	OOD Recall ↑	Local FRR ↓	Global FRR ↓	Inference time ↓
USE-DAN	74.7	**7.3**	7.1	**0.75ms**
USE-TRAN	77.8	8.8	8.4	2.8ms
MPNET	**78.3**	8.4	**5.7**	10ms
DistilRoberta	71.1	7.7	9.6	5ms
MiniLM-L12	73.6	8.7	11.3	4.2ms
MiniLM-L6	71.9	7.9	12.8	1.6ms
qa-distilbert	74.8	8.3	11.3	5.2ms
par. MPNET	72.7	8.2	9.9	11ms
par. albert-small	68.4	7.7	13.2	21ms
par. MiniLM-L12	68.8	9.7	15.2	3.2ms
par. MiniLM-L3	73.4	9.7	9.1	1.4ms
BERT	51.9	8.7	21	32ms

Furthermore, we show the results of the local and global intent accuracies given the embedding and classifier combination in the Tables 5.3, 5.4, and 5.5. These results represent the ability of the model to select the correct intent layer and classify the input utterance into a corresponding class in the layer.

The table Table 5.9 shows the best results of each of the approaches. We can see that the best-performing approach is the routing mechanism, which uses the high-level decision using the cosine similarity and the low-level classification using logistic regression. The intuition behind the low performance of the try-catch mechanism is that it cumulates the error as it passes the utterance to the other levels. On the other hand, the routing mechanism performs well thanks to the intuitive usage of the threshold in combination with the cosine similarity on the higher level.

Table 5.6: Comparison of results on Monolithical Single Model with Masking using different embeddings (rows) with different similarities (columns): 2-layer neural network, 1-layer neural network, random forest, logistic regression, and cosine similarity. The first value represents local intent accuracy and the second value represents global intent accuracy.

	2-layer NN	**1-layer NN**	**RF**	**LR**	**CS**
USE-DAN	83.4 / 84	82.6 / 82.5	84.5 / 82.8	**87.9** / **85**	82.3 / 79.5
USE-TRAN	81.4 / 83	79.8 / 83.4	82.4 / 83.5	81.4 / 84.6	83.1 / 84.6
MPNET	76.4 / 71.2	73.7 / 62.9	82.6 / 84.3	82.7 / 86.9	82.5 / 84.9
DistilRoberta	75.8 / 67.1	74.4 / 58.1	81.4 / 83.8	79.9 / 84.8	79 / 84.8
MiniLM-L12	71.3 / 52.1	71.5 / 49.4	82 / 82.5	81.9 / 79.7	82.5 / 83.2
MiniLM-L6	70.1 / 48.7	72.9 / 48.2	83.2 / 79.4	79.2 / 78.2	81.4 / 78.2
qa-distilbert	73.5 / 66.8	71.4 / 54.9	81.6 / 82.6	77.8 / 81.5	80.8 / 81.5
par. MPNET	78.7 / 71.6	78.3 / 60.8	85.9 / 81.9	83 / 82.4	85.9 / 81.2
par. albert-small	78.6 / 72.3	77 / 67.4	84.1 / 80.1	82.6 / 78.2	84.4 / 80.8
par. MiniLM-L12	75.6 / 58.7	75.8 / 56.5	83.2 / 77.5	80.2 / 75.5	83 / 75.5
par. MiniLM-L3	71.8 / 50.5	69.9 / 50.4	80.4 / 82.2	78.3 / 75.2	79.8 / 82.5
BERT	72.8 / 53.5	71.9 / 40.5	83.5 / 70.3	78.9 / 64.2	83.5 / 70.9

Table 5.7: Comparison of results on Try-Catch Mechanism using different embeddings (rows) with different similarities (columns): 2-layer neural network, 1-layer neural network, random forest, logistic regression, and cosine similarity. The first value represents local intent accuracy and the second value represents global intent accuracy.

	2-layer NN	**1-layer NN**	**RF**	**LR**	**CS**
USE-DAN	80.5 / 55.9	77.7 / 54.6	78.1 / 55.3	**81.7** / 59.5	78.2 / 51.4
USE-TRAN	78.2 / 56.4	77.8 / 55.7	79.3 / 56	78.2 / 57.7	76 / 57.7
MPNET	71 / 44.1	68.6 / 35.1	78 / 58.6	77.6 / **60.8**	75.6 / **60.8**
DistilRoberta	71.9 / 40.3	67.6 / 30.9	77.5 / 58.3	75.3 / 58.2	76.8 / 58.2
MiniLM-L12	68.1 / 25.3	64.6 / 21.6	76.1 / 56.2	74 / 52.5	77.5 / 56.8
MiniLM-L6	66.9 / 21.9	65.1 / 21.2	76.4 / 52.8	75.6 / 50	76.8 / 50
qa-distilbert	70.6 / 39.2	68.4 / 28.3	77.7 / 55.5	74.5 / 53.9	77.3 / 53.9
par. MPNET	73.3 / 45.4	73.5 / 34.3	81.4 / 54.7	79.2 / 55	80.5 / 53.1
par. albert-small	73.1 / 45.1	72.5 / 39.1	81.4 / 52.2	77.9 / 49.6	79.5 / 53.6
par. MiniLM-L12	71.4 / 30.8	68.9 / 28.9	78.1 / 49.5	75.5 / 49.6	78.3 / 49.6
par. MiniLM-L3	67.1 / 23.3	66.8 / 23.6	75.9 / 55.2	76.3 / 48.1	77.2 / 56.1
BERT	69.5 / 27.4	70.2 / 22.1	79.6 / 42.9	75.9 / 38.1	79.6 / 44.5

Table 5.8: Comparison of results on Routing Mechanism using different embeddings (rows) with different similarities (columns): 2-layer neural network, 1-layer neural network, random forest, logistic regression, and cosine similarity. The first value represents local intent accuracy and the second value represents global intent accuracy.

	2-layer NN	**1-layer NN**	**RF**	**LR**	**CS**
USE-DAN	86.7 / 89.8	86.1 / 88.4	86.5 / 88.3	**88.1** / 90	87 / 85.4
USE-TRAN	86.2 / 89.1	84.8 / 89.2	85.9 / 89.4	86.2 / 90.1	85.1 / 90.1
MPNET	79.9 / 77.3	77.1 / 68.6	85 / 92.7	84.4 / **92.8**	84.6 / **92.8**
DistilRoberta	78.3 / 73.3	75.7 / 63.8	84.9 / 89.9	83.8 / 90.4	83.7 / 90.4
MiniLM-L12	74.6 / 58.3	73.5 / 55.5	84.9 / 88.3	83.1 / 85.8	84.2 / 88.7
MiniLM-L6	75 / 54.4	74 / 54	84.9 / 85.5	83.2 / 84.1	85.2 / 84.1
qa-distilbert	77.7 / 72.5	74.7 / 60.6	84.8 / 88.5	81.5 / 87.2	84.6 / 87.2
par. MPNET	81.9 / 77.2	80 / 66.5	87.6 / 87.3	86.3 / 88.5	88.1 / 87
par. albert-small	81.1 / 78.4	78.8 / 73.1	87.5 / 86.3	86 / 83.8	87 / 86.8
par. MiniLM-L12	77.8 / 64.3	77.2 / 62	85.5 / 83.3	84.3 / 81.7	86.2 / 81.7
par. MiniLM-L3	74.9 / 56.4	74.8 / 56.2	84.3 / 87.6	83.1 / 81.1	84.4 / 87.9
BERT	78 / 59	77 / 45.9	86.5 / 75.7	82.8 / 69.7	86.5 / 76.3

Table 5.9: Summary of OOD detection methods and their performance metrics. We selected the best models based on the local intent accuracy.

Method	**Threshold**	**Local Acc**	**Global Acc**	**OOD Recall**
Single model	Max Softmax Prob.	87.9	85.0	59.0
Try-catch	Max Softmax Prob.	81.7	59.5	72.5
Routing	Avg Cos. Sim.	**88.1**	**90.0**	**74.7**

5.6 Conclusion and Main Contributions

In this chapter, we proposed a novel hybrid conversation system architecture. Inspired by the series of the architecture used in Alexa Prize Socialbot Grand Challenge, we created a modular architecture capable of running the conversation in real-time while achieving high performance in the NLU part. The main idea of the architecture is the decomposition of the entire conversational application into smaller units called sub-dialogues. The sub-dialogue may be arbitrarily nested into each other by a static design or using a dynamic decision logic.

This concept is also part of our Flowstorm platform (Pichl et al., 2022), allowing the dialogue designer to easily create various sub-dialogue structures using a visual flow editor. The sub-dialogues are created using different node types and edges defining the flow. The built-in Out of domain mechanism allows the designers to handle the OOD explicitly or by implicit logic. The platform was used as a foundation of the winning socialbot Alquist in the Alexa Prize Socialbot Grand Challenge 4. Also, a popular conversational application Elysai (Jakimiv et al., 2021) was created using our platform.

The proposed architecture based on the sub-dialogues introduces a novel concept of hierarchical intent recognition with Out of domain detection. The hierarchical approach is proposed and evaluated in three modifications: A single Model with Masking, a Try-Catch Mechanism, and a Routing Mechanism. We evaluated the Routing Mechanism to outperform the other two and promoted it as the main approach used on the platform.

Overall, the proposed architecture with a hierarchical intent recognition and Out of domain detection allows variability in dialogue design. The created dialogue structure can be easily combined and allows rapid development and fast runtime. The OOD detection is an important step in leveraging powerful Large Language Models for handling various utterances.

Chapter 6

Open-Domain Understanding Using Pragramtic Level of Conversation

In the previous chapter, we focused on the part of the Natural Language Understanding (NLU), which is tightly connected to the discussed domain. The decomposition into multiple layers represents the intents' relevancy and probability of occurrence. Local intents are typically direct answers to the previous question or reactions to the previous input. Global intents, on the other hand, are typically relevant to the discussed domain but may interrupt the expected flow. Finally, Out of domain (OOD) represents the utterances that, given the context, are irrelevant or simply not covered by the actual dialogue design. The proposed approach of OOD detection is suitable for handling using a single method such as a LLM capable of handling various domains as outlined in section 4.5. However, the variability of the responses that may be classified as OOD is high, so dividing them into multiple categories might be beneficial. Each category may then be processed by a different logic, a model fine-tuned on different data, or the category class may be used as an input for such a model.

One of the critical properties of the mentioned categories is that they can be used across various domains. This leads to linguistic categorization, such as dialogue acts. Dialogue acts represent various categories that may be further used to determine the suiting dialogue act of the response either by a set of rules or by instructing the model. Additionally, the dialogue acts can be used to identify different segments of a single utterance.

In this chapter, we would like to propose a pragmatic level of NLU based on the dialogue acts detection in conversational systems. The main goal is to establish a topic- and use-case-independent set of aspects of the natural language that can be further used in the dialogue manager or other parts of the system. We propose an approach to dialogue acts detection focusing on two major aspects: contextual detection and segment-level detection. Unlike single-utterance detection, contextual detection improves accuracy as the dialogue acts often depend on the previous context. Segment-level detection is beneficial for complex utterances where each segment may be handled differently or even decided not to be handled at all.

The segmentation and the actual dialogue act recognition may be executed as two separate steps. However, we propose a single model recognizing the multiple dialogue acts in a single utterance, resulting in a segmented utterance. Together with the intent recognition and OOD handling using LLMs, it brings better variability to the conversations, improving the user experience. Additionally, we introduce the concept of the core segment, indicating the most relevant segment for further conversation.

6.1 Related Work

Dialogue Act (DA) recognition, associating semantic labels with utterances in a conversation, has been addressed through various approaches in recent research. The research has been done using several datasets with Switchboard (Godfrey et al., 1992) as the most common one. The task is typically treated as an utterance classification task. Various neural networks have been utilized, specifically a Recurrent neural network (RNN) architecture leveraging bidirectional Long Short-Term Memory (LSTM) units forms the base model, while a Conditional Random Field (CRF) layer serves as the top layer to classify utterances into their respective dialogue acts (Chen et al., 2018). With the introduction of transformer architecture (Vaswani et al., 2017), the popular architectures such as BERT (Devlin et al., 2019), Albert (Lan et al., 2019) or DistilBERT (Sanh et al., 2019) have been used.

Another research by Wan et al. (2018) has explored DA classification from a question-answering perspective, proposing an enhanced dynamic memory network with a hierarchical pyramidal utterance encoder. Along with adversarial training, they try to tackle limitations such as handcrafted feature reliance or using traditional methods with adversarial examples. Besides Switchboard dialogue act corpus, they use MapTask corpus (Thompson et al., 1993) for the evaluations.

He et al. (2021) proposed integrating speaker turn changes when modeling dialogue acts to differentiate from the written text. These models incorporate conversation-invariant speaker turn embeddings, enhancing the capture of dialogue semantics while considering different speaker turns. Evaluation is done on multiple benchmark datasets, including the Switchboard corpus.

Moreover, leveraging Sequence–to–Sequence (Seq2Seq) approaches, inspired by Neural Machine Translation (NMT), has been proposed by Colombo et al. (2020) to improve the modeling of tag sequentiality in DA classification. A Seq2Seq model tailored for DA classification, consisting of a hierarchical encoder, guided attention mechanism, and beam search, was proposed, and the authors show the comparison with traditional classification approaches.

6.2 Proposed Method

We propose a contextual dialogue act detection model focused on open-domain conversations. The model is intended to be used on both user utterances and bot responses. The bot responses annotated with dialogue acts may be then used along with user utterances annotations to control the responses of generative models (Marek, 2023). Additionally, we use the user utterance annotations directly as a signal that can be used to control logic created as part of the dialogue design process in the Flowstorm platform. The typical situation is illustrated in Figure 6.1 and shows the OOD Global action followed by a few additional actions.

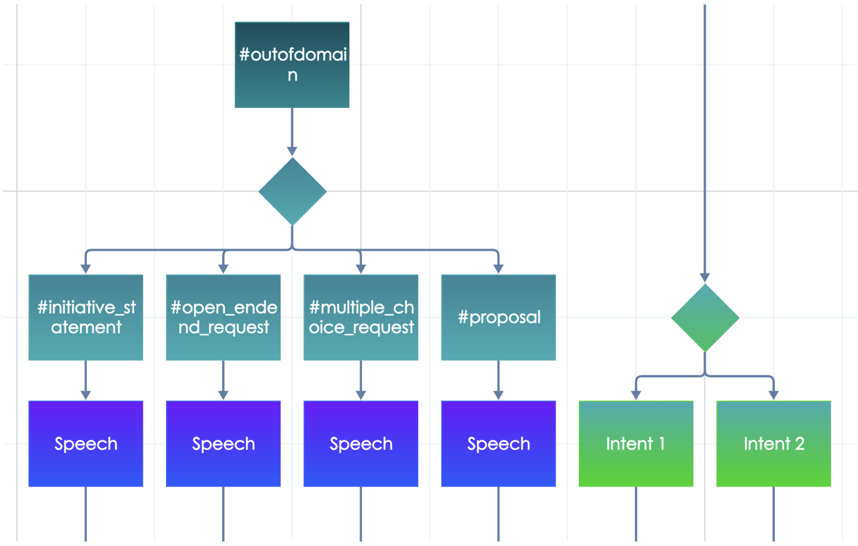


Figure 6.1: Action nodes following the global action Out of domain in Flowstorm platform corresponding to the selected dialogue classes (Initiative statement, Open-ended request, Multiple-choice request, Proposal).

In this scenario, an out-of-domain utterance is processed with a different branch based on the listed dialogue act classes. Please note that for illustration purposes, not all classes are shown. The Dialogue Act class corresponds to the whole utterance if the utterance is not segmented. Otherwise, we identify a "core" segment and return its class.

We propose two approaches for the actual dialogue act classification. The first one combines two tasks, sentence segmentation, with a classification of each segment separately, similarly as described by (Yu and Yu, 2021). The main limitation of this approach is that the actual classification relies on successful segmentation and takes only the context of the preceding segments. The second approach combines the two steps by modeling the task as the token classification. We take each token from the given utterance and classify it into one of the classes. When two adjacent tokens have a different dialogue act label, they form a segment boundary.

The formal description of the first approach is following: Let

$$S = seg_1, seg_2, \ldots, seg_m, B = b_1, b_2, \ldots, b_{m-1}$$

where S is a set of segments in an utterance U, where each segment seg_i is a sequence of tokens and B is the set of predicted segment boundaries such that $b_i \in 0, 1$, where 0 represents no segment boundary and 1 represents a boundary. The classifier $f(U)$ diven the utterance U predicts the probability

$$f(U) = p(b_i = 1|s_i), \forall i \in \{0, \ldots, m-1\},$$

i.e., the probability of a boundary between segments s_i and s_{i+1}.

The classification $g(seg_i)$ then estimates the probability y_i^{da} of a $i-th$ segment corresponding to the dialogue act class da given the segment, all its predecessors, and the contextual window W (see subsection 2.1.4) of size $n-1$ (current utterance is not included):

$$g(seg_i, W^{n-1}) = p(y_i^{da}|seg_j, W^{n-1}, j \leq i)$$

On the other hand, the token classification approach $h(U, W^{n-1})$ takes a sequence of m tokens from the utterance U and estimates the probability of each token t_i corresponding to a dialogue act class.

$$h(U, W^{n-1}) = p(y_i^{da}|t_j, W^{n-1}, j \leq i), \forall i \in \{0, \ldots, m\}$$

The segment boundary b_i between segments s_i and s_{i+1} is the estimated as follows:

$$b_i = \begin{cases} 1 \text{ if } \operatorname{argmax}_{da \in DA} y_i^{da} \neq \operatorname{argmax}_{da \in DA} y_{i+1}^{da} \\ 0 \text{ otherwise} \end{cases}$$

For both approaches mentioned above, we use an additional classifier that estimates whether the segment is "core" or not. The core segment is the one that the conversational system needs to react to. For example, when the system asks *How are you?* and the user responds *I am fine, thank you. What about you?* we can divide the utterance into three segments: *I am fine—thank you—What about you?* with corresponding classes: `Specific answer`, `Thanks`, and `Open-ended request`. In this example, the last segment is the core one, and the system should react to this one.

The model is a binary classifier that estimates for each segment whether it is the core one or not. The input *inp* is encoded to contain the context (= previous turns), current utterance, and the actual segment. Please note that the context is optional, and we run experiments with various context lengths. Given the turn T_i and the context length n, we classify encode the segment seg_j form the user utterance U_u^i as follows:

$$inp_j = U_u^{n-1}\texttt{<SEP>}U_b^{n-1}\texttt{<SEP>}\ldots\texttt{<SEP>}U_u^{i-1}\texttt{<SEP>}U_b^{i-1}\texttt{<SEP>}U_u^i\texttt{<CORE>}seg_j$$

where `<SEP>` is a separator token of individual utterances and `<CORE>` separates the current segment from the utterances. Then, we estimate the probability of each segment from the current utterance being the core one. Each utterance contains exactly one segment. Therefore, we select the segment with the highest probability as the core one.

$$core = \operatorname*{argmax}_{seg_j \in U_u^i} h(inp_j)$$

6.3 Datasets

6.3.1 Switchboard

One of the most popular dialogue act datasets is The Switchboard Dialog Act (SwDA) Corpus (Jurafsky and Shriberg, 1997). The corpus is based on the dataset of the manually transcribed human-to-human telephone conversation corpus created by Godfrey et al. (1992). The dataset comprises 1,155 telephone conversations that each last for five minutes and involve two participants. The conversations are limited to 70 pre-defined topics, such as child care, recycling, and news media, with callers posing questions to receivers. The dataset features 440 speakers who contribute to a total of 221,616 utterances, which can be reduced to 122,646 utterances by merging consecutive utterances made by the same speaker. One of the advantages of the dataset is the presence of the context of the whole conversation. Contextual information is often important for a successful dialogue act classification.

The dataset contains two-sided telephone conversations, which are transcribed into the text. Most importantly, each sentence (or sentence segment) is labeled with one dialogue act class. The original dataset contains over 200 classes. The Switchboard Coder's Manual[1], however, contains rules to cluster those classes into a total number of 43. The corresponding names and examples for each dialogue act class are shown in Table 6.1.

The distribution of the top 10 classes is shown in Figure 6.2. Most of the sentences from the dataset are statements followed by acknowledgments and opinions. Some of the classes, such as questions, are divided into several sub-classes (Yes-No-Question, Wh-Question, Declarative Yes-No-Question, Open-Question, Rhetorical-Questions). This fact should be considered before building a dialogue system since multiple similar classes can reduce the overall performance, and they could be unnecessary for the particular system.

Overall, the dataset contains 213,543 training samples and 4,514 testing samples. The average length of the sentence is 8 tokens.

6.3.2 Midas

Yu and Yu (2021) proposed a dataset called Machine Interaction Dialog Act Scheme (MIDAS) focused on open-domain human–machine interactions. The motivation is that the interaction between a human and a machine is, in its structure, different than a human–human interaction. For example, users tend to say more commands when speaking with a machine. Additionally, the set of dialogue act classes reflects the need for additional decision algorithms, such as dialogue management. The data were collected using the Alexa Prize Socialbot Grand Challenge 2 winning social bot Gunrock (Yu et al., 2019) and annotated by human annotators.

The annotation scheme introduces a hierarchical structure of the dialog acts of human-machine conversations. The structure is divided into two main sub-trees: semantic request type (Figure 6.3) and functional request type (Figure 6.4). The semantic request type

[1] `https://web.stanford.edu/~jurafsky/ws97/manual.august1.html`

Table 6.1: Top 10 dialogue act classes in Switchboard dataset (after clustering).

Name	Tag	Example
Statement-non-opinion	sd	Me, I'm in the legal department.
Acknowledge (Backchannel)	b	Uh-huh.
Statement-opinion	sv	I think it's great
Abandoned or Turn-Exit	%	So, -
Agree/Accept	aa	That's exactly it.
Appreciation	ba	I can imagine.
Yes-No-Question	qy	Do you have to have any special training?
Non-verbal	x	[Laughter], [Throat_clearing]
Yes answers	ny	Yes.
Conventional-closing	fc	Well, it's been nice talking to you.

captures dialog content and is divided into the initiative and responsive classes. This is critical from the dialogue design perspective as the initiative branch typically leads to a change in the flow. Next, the initiative class is separated into two categories, question and command, and the responsive class is further divided into opinion, statement non-opinion, and answer.

Functional request type helps dialog systems achieve discourse level coherence and is divided into incomplete, social convention, and other classes. Incomplete utterances are common in human–machine conversations, especially due to ASR errors and the situations when the user's utterance is cut prematurely. Social convention is typically used to reply to the utterance in any context properly.

The dataset was created by annotating socialbot conversations. It resulted in 24k sentences from 468 conversations. 12.9k sentences were from a user, and the rest from the system. The sentences were automatically pre-segmented before the annotation.

6.3.3 Dialogue Acts in Segmented Conversations

We present a Segment-Level Contextual Dialogue Act (SLCDA) dataset with a novel class structure optimized for conversational systems. We address the following limitations identified in the existing datasets: (1) Either low language richness in humam–machine data or annotated human–human conversation. (2) The segment boundaries are not explicitly

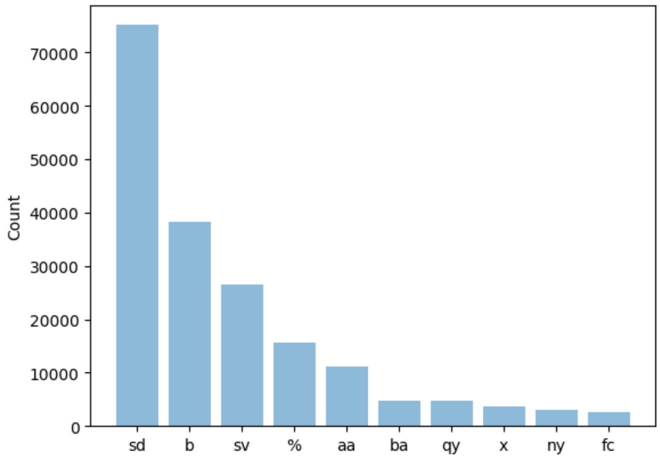


Figure 6.2: Counts of the 10 most frequent classes in the Switchboard dataset (after clustering).

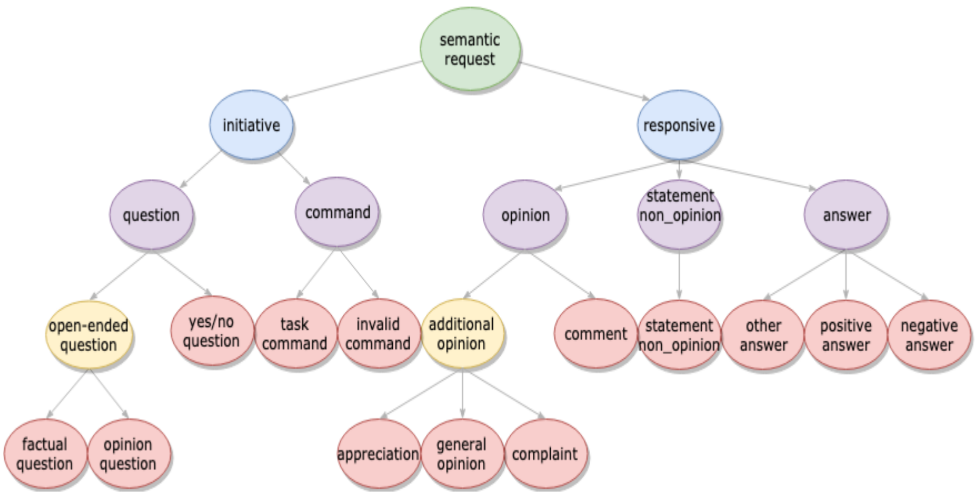


Figure 6.3: Semantic request tree of MIDAS dataset (Yu and Yu, 2021).

annotated, but the approaches rely on punctuation-based segmentation, which may not be aligned with DA-based segmentation.

The data are gathered from anonymized conversations from the selected applications created on the Flowstorm platform (Pichl et al., 2022). As a result of dealing with the mentioned limitations, we created the dataset with the following contributions:

1. Dialogue acts class set suitable for both manual dialogue design and generative models. Additional tags describe specific situations in conversations.

2. Sentence level dialogue acts allowing utterance segmentation and fine-grained dia-

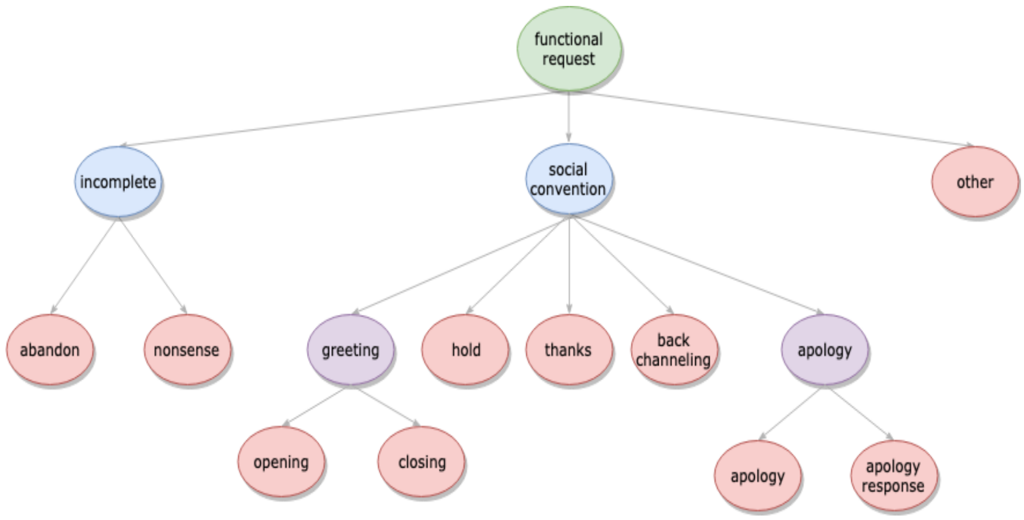


Figure 6.4: Functional request tree of MIDAS dataset (Yu and Yu, 2021).

logue acts detection.

3. Both user utterances and bot responses are annotated. The annotated user utterances are suitable for training classification models. The annotated bot responses are suitable for training dialogue-act-controllable generative models.

4. In multi-segment utterances, the core segment is labeled.

Annotation Process

We randomly selected 7,558 of the conversations that were more than 10 turns long. The conversations were selected from a database of conversation logs containing template-based dialogues and dialogues using the generative models. The intention behind this decision was to primarily ensure the wide range of both user utterances and bot responses. From the set of selected conversations, three annotators annotated 253 conversations.

We used Label Studio (Tkachenko et al., 2020-2022) as an annotation tool. We created a labeling interface by modifying the template for conversational data. The whole conversation was shown on a single screen, allowing the annotators to consider the full context. To annotate the dialogue act classes and tags segment-wise, each annotator had to select the range inside the utterance. Then, the proper class could be assigned to the selected segment.

Segmentation Principles

Unlike a traditional segmentation using a punctuation restoration (Tilk and Alumäe, 2016; Nagy et al., 2021), we created the segments based on the sentence's meaning, e.g., the

Table 6.2: Sample conversation with each segment labeled with a corresponding DA class.

Speaker	Utterance
Bot:	Welcome! I am Poppy. How do you want me to call you? It's easier if you type it but you can also say it out loud.
User:	How do you want to call me?
Bot:	You tell me.
User:	Call me Princess.
Bot:	Nice to meet you, Princess. Tell me, what brings you here?
User:	I just checked you out.
Bot:	I see. I would like to give you a short introduction to this app. It will take about 5 minutes. Do you have the time now?
User:	Can you take only two minutes? My phone's low in percentage.
Bot:	Well, I'd say let's try it! . . .
Legend:	Opening, Initiative Statement, Open-Ended Request, Task Command, Specific Answer, Opening – Responsive, Opening – Responsive

sentence *"Hi, I'm Anthony."* is segmented into *"Hi"* and *"I'm Anthony"* as it consists of `Opening` and `Initiative statement`. On the other hand, *"Don't worry, it's not so bad"* is considered a single segment labeled as `Acknowledgement`. The number of segments for user and bot utterances is shown in Table 6.3.

Table 6.3: Dataset statistic of segments per bot and user.

Participant	Turns	Segments	Avg. # seg.
Bot	4,158	16,251	3.91
User	3,777	1,490	1.18

Dialogue Act Classes

The class set consists of 23 individual dialogue act classes. The classes are divided into six groups: **Greetings/Politeness**, **Interaction Particles**, **Feedback**, **Requests/Proposals**, **Statements**, and **Unrecognized**. Each segment in the dataset is labeled by exactly one of the 23 classes. Additionally, there are 7 tags that can be optionally assigned to the sentence as additional information. Both the dialogue act classes and the additional tags are designed to cover typical situations in the human-bot conversation for both human and bot utterances.

The **Greetings/Politeness** group contains conversational openings and closings, apologies, and thanks. When recognized from the segmented user utterance, it is a good practice to address it with a proper response before providing the reaction to the other segments that may push the conversation further.

The **Interaction Particles** group contains responses to the questions and proposals such as "yes", "no" and other types of acknowledgments and refusals.

The **Feedback** group contains positive and negative feedback that typically refers to the quality of the conversation.

The **Requests/Proposals** group contains requests/questions and proposals. These utterances typically indicate that the communication partner is supposed to provide a response addressing the request or proposal. The requests are further divided into open-ended, multiple-choice and yes-no.

The **Statements** group is the most represented one. From the dialogue design perspective, the important disambiguation within the group is between initiative/non-initiative and specific/non-specific statements (answers). The initiative statements or task commands often mean a request for a change in the current flow of the dialogue that should be handled by the dialogue manager. Additionally, the non-specific answer indicates the situations when the user provided an answer from the linguistic point of view but the content of the answer is not specific enough.

The **Unrecognized** group contains utterances that cannot be assigned to a specific class that is either unfinished, contains speech recognition errors, or makes no sense. Note, there is a `Incomplete` tag (see below) for unfinished utterances. Not all unfinished utterances are unrecognized. When the utterance contains enough words to determine a dialogue act, we assign a specific class along with an `Incomplete` tag.

The additional **Tags** group allows an extra label for each annotated segment. It is used along with the dialogue act label to capture additional information. The tags are the following:

- Repair – a user or bot asks for a clarification of the previous utterance or when a segment is rephrasing the previous segment in a single utterance.
- Abrubt shift – non-coherent change in a topic
- Meta – comments about the previous speech, typically misunderstandings or confusion. Typically coupled with negative feedback.
- Obscene – the inappropriate, rude, or vulgar utterance
- Incomplete – incomplete utterances due to speech recognition errors, long pauses, or any other failures.
- Hold – filling sentences without a specific meaning. It often indicates an initiative statement within the next segment.

6.4 Experiments

We run several experiments using the publicly available datasets SwDA and Midas and our custom dataset SLCDA. First, we experimented with different context sizes using BERT and Albert models. Besides the classification accuracy as the main metric, we also measure train time, inference time, and CPU memory at the model's runtime. The reason behind the additional metrics is that the focus of this thesis is also the resource effectiveness of the proposed solutions. Unlike the models selected in the previous chapter, BERT and Albert have millions of parameters (108M and 12M, respectively), so their training and inference time and memory consumption are significant.

The train time is measured while training on Nvidia A100 GPU with a batch size of 256 for the BERT model and 64 for the Albert model. The inference time is measured on the AMD EPYC 7V12 CPU using a batch size of 1, and the reported value is an average of 100 inference requests.

Additionally, we provide experiments on optimized models. The model optimization is done by converting the trained model format (PyTorch (Paszke et al., 2019)) to ONNX[2]. Additionally, the models are optimized and quantized using the HuggingFace Optimum library[3] with default parameters.

Table 6.4: Dialogue act classification on SWDA with different context sizes.

	Context	**Eval acc**	**Train time**	**Inference time**	**CPU mem**
BERT	1	78.9	4m12s	87ms	855MB
BERT	2	81.0	4m43s	88ms	855MB
BERT	3	81.5	5m12s	87ms	855MB
BERT	5	**82.0**	5m55s	91ms	855MB
Albert	3	80.6	6m27s	138ms	523MB
Albert	5	81.5	6m53s	140ms	523MB

Table 6.5: Dialogue act classification on Midas with different context sizes.

	Context	**Eval acc**	**Train time**	**Inference time**	**Model size**
BERT	1	74.0	2m15s	88ms	855MB
BERT	2	79.4	2m17s	95ms	855MB
Albert	1	73.1	1m48s	125ms	523MB
Albert	2	**78.5**	1m48s	131ms	523MB

As we can see in Table 6.4 and 6.5, the context size plays a significant role in the dialogue act detection. Context size 1 means that we classify only given the current

[2] `https://onnx.ai/`
[3] `https://huggingface.co/docs/optimum/index`

utterance. Therefore, the most significant accuracy increase is when we included at least the previous turn (context size 2). Additional context turns slightly increase the accuracy as shown at the SwDA dataset. Midas dataset does not provide a context longer than 2 turns.

Table 6.6 shows the results using ONNX, ONNX optimized, and ONNX quantized BERT and Albert models. The optimized version uses 16-bit floating-point precision instead of the default 32-bit, and the quantized version uses an 8-bit integer for the model weights and activations.

The results show that the optimization reduces the memory using both CPU and GPU significantly. However, the inference time is higher rapidly for the CPU inference because the CPUs typically do not have direct floating point 16 support. GPU inference time is 1.75 times faster, while the accuracy is on the same level. On the other hand, the quantization is slower on GPU, as the GPUs are not suitable for integer operations. However, the memory requirement is reduced rapidly, but also the accuracy is reduced from 82 % to 77 % for the BERT model and from 81.5 % to 38 % for the Albert model.

Table 6.6: Dialogue act classification using optimized models on SwDA dataset with a context size of 5.

	Acc	**CPU mem**	**Inference time (CPU)**	**GPU mem**	**Inference time (GPU)**
BERT onnx	**82.0**	827MB	15ms	1090MB	3.5ms
BERT onnx-opt	**82.0**	372MB	218ms	289MB	**2ms**
BERT onnx-quant	77.0	203MB	**7ms**	86MB	15ms
Albert onnx	81.5	653MB	14ms	553MB	3.5ms
Albert onnx-quant	38.0	**170MB**	7ms	**27MB**	15ms

Using the proposed dataset, we present the results of two experiments conducted to explore two main properties of the dataset. The first experiment is segmentation, and the second is the actual dialogue act classification on already recognized segments. The total number of 253 conversations is divided into train and test splits containing 202 and 51 conversations, respectively. The splits are the same for both experiments.

6.4.1 Segmentation

The segmentation task is focused on dividing long utterances possibly into smaller units—segments. Each segment is then intended to be an atomic part of the initial utterance from the dialogue act perspective. We selected a token classification approach as a baseline model for this task. The input for the model is a user or bot utterance from a single turn. The important preprocessing step is to remove the punctuation characters "`.,!?`" as we do not want the model to use these characters as the reference for the segmentation boundary. Additionally, we convert the text into lowercase. Since the segments are not overlapping and a new segment starts right where the previous ends, each token can be labeled either

'O' (outside) or 'END' (indicating that the segment ends after this token). The task of the model is then to label each token to find the boundaries of the segments.

We selected TBRNN (Tilk and Alumäe, 2016), BERT (Devlin et al., 2019), ALBERT (Lan et al., 2019), and DistilBERT (Sanh et al., 2019) as the baseline models (the same models as in section 4.4.4). The implementation of the models was used from the Huggingface Transformers[4] library. Both models were trained for 15 epochs with batch size 32. The results are shown in Table 6.7.

Table 6.7: Segmentation results on SLCDA dataset

Model	**Precision**	**Recall**	**F1 score**
TBRNN	86.8	81.2	84.0
Albert	88.4	**95.5**	92.1
DistilBERT	88.9	94.6	91.8
BERT	**90.7**	94.5	**92.5**

Additionally, we run the sentence classification of each segment to estimate the corresponding dialogue act classes. We combine the segmentation models with BERT and Albert as dialogue act classification models. Then, we compare the estimated dialogue act classes per each ground truth segment. In other words, we label each token from the classified segment with the inferred dialogue act label and compare it with the ground truth. If all of the tokens from the true segment have the correct label, we mark the segment as correctly classified. E.g., if the segmentation does not identify a boundary between two segments with the same dialogue act class and the classification correctly predicts the class for this joint segment, both segments are marked as a correct classification.

Table 6.8: Two-step dialogue act classification. Context size 5. The first model is the segmentation model, and the second one is for the DA classification.

Model	**DA Acc**	**Tag Acc**
TBRNN + Albert	64.8	76.5
Albert + Albert	70.2	80.6
DistilBERT + Albert	69.9	81.3
BERT + Albert	70.4	81.7
TBRNN + BERT	65.7	77.5
Albert + BERT	69.5	81.4
DistilBERT + BERT	**71.0**	82.2
BERT + BERT	70.8	**82.3**

[4]https://github.com/huggingface/transformers

6.4.2 Dialogue Act Classification

The Dialogue Act Classification task is performed on each individual segment separately. We assign one dialogue act class to each segment plus one optional tag. The dialogue acts and tag classifiers are trained as separate models. Each individual input contains the segment for which we want to know the dialogue act plus the previous segments from the same utterance. The segments are separated using `[SEG]` token. We also experimented with different context sizes from 1 to 5 turns. To add context to the base input, we concatenated consecutive turns into one string and used `[SEP]` token as a separator.

We use the same BERT and Albert models as in the segmentation task. Both models were trained for 10 epochs with batch size 32. The results on the proposed dataset, along with the results of the same models on SwDA and MIDAS datasets, are shown in Table 6.9.

Table 6.9: Dataset act classification results on SwDA, Midas, and our Segment-Level Contextual Dialogue Act (SLCDA). The Context column shows the number of turns in the context.

Dataset	Model	Context	DA Acc	Tag Acc
SwDA	BERT	1	78.9	-
SwDA	BERT	2	81.0	-
SwDA	BERT	3	81.5	-
SwDA	BERT	5	82.0	-
SwDA	Albert	3	80.6	-
SwDA	Albert	5	81.5	-
MIDAS	BERT	1	74.0	-
MIDAS	BERT	2	79.4	-
MIDAS	Albert	1	73.1	-
MIDAS	Albert	2	78.5	-
SLCDA	BERT	1	71.2	82.3
SLCDA	BERT	2	74.6	83.5
SLCDA	BERT	3	75.1	84.2
SLCDA	BERT	5	**75.9**	**84.8**
SLCDA	Albert	3	73.7	82.2
SLCDA	Albert	5	75.2	84.1

The optimized floating point 16 version of the best-performing model has a statistically insignificant accuracy drop of 0.1 %, and the quantized version using integer 8 has a drop of 3.9 %.

Table 6.10 show the results of core segment classification. We measure the accuracy of user and bot utterances separately, as the user utterances are typically shorter (see Table 6.3). Due to the low number of average segments per user utterance, the core segment classification in these utterances is an easy task, resulting in high accuracies. Therefore, the accuracy measured on bot segments is a more significant metric.

Table 6.10: Core segment classification results.

Model	**Context**	**User Acc**	**Bot Acc**
Albert	1	98.5	84.0
Albert	2	98.5	86.8
Albert	3	98.7	87.1
Albert	5	**98.8**	**87.8**
BERT	1	98.5	82.6
BERT	2	98.6	85.4
BERT	3	**98.8**	85.9
BERT	5	**98.8**	86.6

6.5 Conclusion and Main Contributions

We proposed a pragmatic level of Natural Language Understanding for the conversational systems as an addition to the intent recognition and Out of domain detection. The proposed method leverages Dialogue Act detection focused on the individual segments of the utterance. Each segment may play a different role in the conversation, and its importance may vary. Therefore, we also introduced the concept of the “core” segments.

We reviewed the standard datasets such as SwDA and Midas, where the former contains annotated human–human conversations while the latter contains human–machine conversations with a limited context. We showed that contextual information plays a crucial role in DA detection on both of these datasets. Especially in the field of conversational AI, the utterances may be short, so the relation the the previous context may contain information that is not included in a single utterance. Additionally, we experimented with optimized versions of the models to ensure the overall system is not slowed down by the additional component.

We created our custom dataset for DA classification that contains various types of human–machine conversations with the context containing the full conversation. This dataset has the individual classes labeled per utterance segment. The segmentation is manually annotated, unlike in the previous dataset, making the segments more accurate with the connection to the dialogue acts. Finally, we annotated core segments in each utterance. As the dataset statistics show, the users tend to say shorter utterances, resulting in a low average segment count per utterance. For that reason, the bot utterances are better data for the core segment classifier.

We experimented with a two-step classification consisting of segmentation and the actual classification. Additionally, we proposed a token classification approach classifying each token into a dialogue act class, resulting in a segmented classification. The results show that the second approach performs better on our dataset. Finally, we trained a core segment classifier that primarily leveraged bot utterances.

Chapter 7

Conclusion

This thesis focused on conversational systems and their underlying Natural Language Understanding components. The main goal was to create an architecture with a strong NLU performance while keeping the overall system response time on a reasonable level. The achieve this goal, we researched several approaches to hybrid architecture that combine precisely created dialogue designs with robust end-to-end models capable of handling a wide range of user inputs.

In the first chapter, we introduced the basic units of the conversation that are essential for this topic. Following a brief summary of the historical conversational systems and the recent approaches, we reviewed the advantages of the respective approaches and the overall research trend. We introduced the categorization of the conversational system from the use case perspective, i.e., goal-oriented and open-domain. This categorization has been historically tight to the actual system architecture, where goal-oriented system creators often chose the modular approach while the end-to-end approach was usually selected for the open-domain systems as it was able to respond to various inputs in a relatively coherent way.

The second chapter focused on the architecture design of the conversational socialbot. The name *socialbot* was introduced during the first annual Alexa Prize Socialbot Grand Challenge and stands for a social chatbot as an illustration of the open-domain "social" conversation between a user and the bot. The goal of the competition was to design a system capable of having a coherent and engaging conversation with a user. Focused on that goal, we designed several systems leveraging the hybrid architecture. This thesis contributed to the overall success of all of the systems by an effective architecture design with a focus on the NLU part, its performance, and its effectiveness. The systems were awarded as one of the top three winners during each year of the competition and the fourth iteration was the overall first-place winner. The most significant NLU contributions include the decomposition of the open-domain approach into multiple levels while allowing the user to navigate smoothly through the levels representing both the individual topics and the depth of the respective topic. This eventually resulted in the design of a generic conversational platform called Flowstorm.

The proposed Flowstorm platform is focused on a modular conversation design leveraging conversational units called sub-dialogues. This decomposition level was created as a result of experiments with the socialbot architectures and was used in the final winning system. The strengths of the system ensure an effective design process while using a minimal amount of data. Additionally, the flexibility of putting the units together creates a variable user experience. This architecture is supported by a visual editing tool allowing quick prototype, testing, and deployment of the applications built on top the the proposed architecture. The sub-dialogue structure comes along with the need for hierarchical intent recognition. We proposed and experimented with three approaches where the main focus was on using a separate model for each individual sub-dialogue while being able to combine them in complex scenarios. The three methods, namely Single Model with Masking, Try-Catch Mechanism, and Routing Mechanism, approach the combination of the models differently, and their pros and cons are discussed. Moreover, the intent recognition is coupled with a context-aware OOD detection, allowing the system to recognize the limits of the underlying conversational design and smooth transition to the selected generative approach. This method proved particularly useful in connection to the recent advancement in Large Language Models.

In addition to sub-dialogue-related intent recognition, we propose a pragmatic level of NLU represented by the dialogue act detection. The pragmatic level is based on dialogue act detection, allowing recognition of various acts of the utterance of both the user and the bot. This is useful for different strategies while handling out-of-domain utterances by a different logic and for influencing the result of the generative models. We propose a joint segmentation and dialogue act detection, allowing handling longer utterances and focusing only on the relevant parts. We propose a custom dataset reflecting the needs of the task, especially the source of the annotated data and manual segment annotation.

Overall, we proposed a novel architecture design with a flexible hierarchical intent recognition, augmented by Out of domain detection and dialogue act detection. The proposed concepts have been proved in multiple iterations of Alexa Prize Socialbot Grand Challenge competitions and by a widely used application called Elysai.

Bibliography

Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In *Recommender systems handbook*, pages 217–253. Springer, 2010.

James Allen. *Natural language understanding*. Benjamin-Cummings Publishing Co., Inc., 1995.

Inc Amazon.com. Create Alexa Skills Kit: Amazon Alexa Voice Development. `https://developer.amazon.com/en-US/alexa/alexa-skills-kit`, 2015. [Accessed 11-02-2024].

Gaurav Arora, Chirag Jain, Manas Chaturvedi, and Krupal Modi. HINT3: Raising the bar for intent detection in the wild. In Anna Rogers, João Sedoc, and Anna Rumshisky, editors, *Proceedings of the First Workshop on Insights from Negative Results in NLP*, pages 100–105, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.insights-1.16. URL `https://aclanthology.org/2020.insights-1.16`.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Petr Baudiš. Yodaqa: a modular question answering system pipeline. In *POSTER 2015-19th International Student Conference on Electrical Engineering*, pages 1156–1165, 2015.

Tom Bocklisch, Joey Faulkner, Nick Pawlowski, and Alan Nichol. Rasa: Open source language understanding and dialogue management, 2017.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146, 2017.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, SIGMOD

'08, pages 1247–1250, New York, NY, USA, 2008a. ACM. ISBN 978-1-60558-102-6. doi: 10.1145/1376616.1376746. URL http://doi.acm.org/10.1145/1376616.1376746.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. AcM, 2008b.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*, 2016.

Kristy Boyer, Robert Phillips, Eun Young Ha, Michael Wallis, Mladen Vouk, and James Lester. Modeling dialogue structure with adjacency pair analysis and hidden markov models. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*, pages 49–52, 2009.

Samuel Broscheit. Investigating entity knowledge in BERT with simple neural end-to-end entity linking. In Mohit Bansal and Aline Villavicencio, editors, *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 677–685, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/K19-1063. URL https://aclanthology.org/K19-1063.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Paweł Budzianowski and Ivan Vulić. Hello, it's GPT-2 - how can I help you? towards the use of pretrained language models for task-oriented dialogue systems. In Alexandra Birch, Andrew Finch, Hiroaki Hayashi, Ioannis Konstas, Thang Luong, Graham Neubig, Yusuke Oda, and Katsuhito Sudoh, editors, *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22, Hong Kong, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5602. URL https://aclanthology.org/D19-5602.

Robin Burke. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction*, 12:331–370, 2002.

Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yurii Kuratov, Denis Kuznetsov, et al. Deeppavlov: Open-source library for dialogue systems. In *Proceedings of ACL 2018, System Demonstrations*, pages 122–127, 2018.

Heloisa Candello and Claudio Pinhanez. *Recovering from Dialogue Failures Using Multiple Agents in Wealth Management Advice*, pages 139–157. 09 2018. ISBN 978-3-319-95578-0. doi: 10.1007/978-3-319-95579-7_7.

Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder for english. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174, 2018.

Harrison Chase. Introduction to Langchain. `https://python.langchain.com/docs/get_started/introduction`, 2022. [Accessed 11-02-2024].

Zheqian Chen, Rongqin Yang, Zhou Zhao, Deng Cai, and Xiaofei He. Dialogue act recognition via crf-attentive structured network. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, SIGIR '18, page 225–234, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450356572. doi: 10.1145/3209978.3209997. URL `https://doi.org/10.1145/3209978.3209997`.

Jianpeng Cheng, Zhongyuan Wang, Ji-Rong Wen, Jun Yan, and Zheng Chen. Contextual text understanding in distributional semantic space. In *ACM International Conference on Information and Knowledge Management (CIKM)*. ACM – Association for Computing Machinery, October 2015. URL `https://www.microsoft.com/en-us/research/publication/contextual-text-understanding-in-distributional-semantic-space/`.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL `https://aclanthology.org/D14-1179`.

Pierre Colombo, Emile Chapuis, Matteo Manica, Emmanuel Vignon, Giovanna Varni, and Chloe Clavel. Guiding attention in sequence-to-sequence models for dialogue act prediction. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7594–7601, 2020.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Xin Luna Dong, Seungwhan Moon, Yifan Ethan Xu, Kshitiz Malik, and Zhou Yu. Towards next-generation intelligent assistants leveraging llm techniques. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5792–5793, 2023.

Ondřej Dušek and Filip Jurčíček. A context-aware natural language generator for dialogue systems. In Raquel Fernandez, Wolfgang Minker, Giuseppe Carenini, Ryuichiro Higashinaka, Ron Artstein, and Alesia Gainer, editors, *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 185–190, Los Angeles, September 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-3622. URL `https://aclanthology.org/W16-3622`.

Mihail Eric and Christopher Manning. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. In Mirella Lapata, Phil Blunsom, and Alexander Koller, editors, *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 468–473, Valencia, Spain, April 2017. Association for Computational Linguistics. URL `https://aclanthology.org/E17-2075`.

D. A. Ferrucci. Introduction to “this is watson”. *IBM Journal of Research and Development*, 56(3.4):1:1–1:15, 2012. doi: 10.1147/JRD.2012.2184356.

David Ferrucci and Adam Lally. Uima: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3-4):327–348, 2004.

David Ferrucci, Adam Lally, Karin Verspoor, and Eric Nyberg. Unstructured information management architecture (UIMA) version 1.0. OASIS Standard, mar 2009. URL `https://docs.oasis-open.org/uima/v1.0/uima-v1.0.html`.

David Ferrucci, Anthony Levas, Sugato Bagchi, David Gondek, and Erik T. Mueller. Watson: Beyond jeopardy! *Artificial Intelligence*, 199-200:93–105, 2013. ISSN 0004-3702. doi: https://doi.org/10.1016/j.artint.2012.06.009. URL `https://www.sciencedirect.com/science/article/pii/S0004370212000872`.

David A. Ferrucci. Ibm’s watson/deepqa. In *Proceedings of the 38th Annual International Symposium on Computer Architecture*, ISCA ’11, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450304726. doi: 10.1145/2000064.2019525. URL `https://doi.org/10.1145/2000064.2019525`.

Alexander Fritzler, Varvara Logacheva, and Maksim Kretov. Few-shot classification in named entity recognition task. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 993–1000, 2019.

Raefer Gabriel, Yang Liu, Anna Gottardi, Mihail Eric, Anju Khatri, Anjali Chadha, Qinlang Chen, Behnam Hedayatnia, et al. Further advances in open domain dialog systems in the third alexa prize socialbot grand challenge. In *Alexa Prize SocialBot Grand Challenge 3 Proceedings*, 2019. URL `https://www.amazon.science/publications/further-advances-in-open-domain-dialog-systems-in-the-third-alexa-prize-socialbot-grand-challenge`.

Philip Gage. A new algorithm for data compression. *The C Users Journal archive*, 12: 23–38, 1994. URL https://api.semanticscholar.org/CorpusID:59804030.

Varun Gangal, Abhinav Arora, Arash Einolghozati, and Sonal Gupta. Likelihood ratios and generative classifiers for unsupervised out-of-domain detection in task oriented dialog. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):7764–7771, Apr. 2020. doi: 10.1609/aaai.v34i05.6280. URL https://ojs.aaai.org/index.php/AAAI/article/view/6280.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023a.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2023b.

Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1(12), 2009.

John J Godfrey, Edward C Holliman, and Jane McDaniel. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, volume 1, pages 517–520. IEEE, 1992.

Google. Dialogflow: A comprehensive platform for developing chatbots. https://cloud.google.com/dialogflow, 2017. [Accessed 11-02-2024].

Karthik Gopalakrishnan, Behnam Hedayatnia, Qinlang Chen, Anna Gottardi, Sanjeev Kwatra, Anushree Venkatesh, Raefer Gabriel, and Dilek Hakkani-Tür. Topical-chat: Towards knowledge-grounded open-domain conversations. In *Interspeech 2019*, 2019. URL https://www.amazon.science/publications/topical-chat-towards-knowledge-grounded-open-domain-conversations.

Anna Gottardi, Osman Ipek, Giuseppe Castellucci, Shui Hu, Lavina Vaz, Yao Lu, Anju Khatri, Anjali Chadha, Desheng Zhang, Sattvik Sahai, Prerna Dwivedi, Hangjie Shi, Lucy Hu, Andy Huang, Luke Dai, Bofei Yang, Varun Somani, Pankaj Rajan, Ron Rezac, Michael Johnston, Savanna Stiff, Leslie Ball, David Carmel, Yang Liu, Dilek Hakkani-Tür, Oleg Rokhlenko, Kate Bland, Eugene Agichtein, Reza Ghanadan, and Yoelle Maarek. Alexa, let's work together: Introducing the first alexa prize taskbot challenge on conversational task assistance. In *Alexa Prize TaskBot Challenge 1 Proceedings*, 2022. URL https://www.amazon.science/publications/alexa-lets-work-together-introducing-the-first-alexa-prize-taskbot-challenge-on-conversational-task-assistance.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. Incorporating copying mechanism in sequence-to-sequence learning. In Katrin Erk and Noah A. Smith, editors, *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640, Berlin, Germany, August 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1154. URL https://aclanthology.org/P16-1154.

Ramanathan Guha, Rob McCool, and Eric Miller. Semantic search. In *Proceedings of the 12th international conference on World Wide Web*, pages 700–709, 2003.

Masato Hagiwara. *Real-World Natural Language Processing: Practical applications with deep learning*. Manning, 2021.

Dilek Hakkani-Tür, Gökhan Tür, Asli Celikyilmaz, Yun-Nung Chen, Jianfeng Gao, Li Deng, and Ye-Yi Wang. Multi-domain joint semantic frame parsing using bi-directional rnn-lstm. In *Interspeech*, pages 715–719, 2016.

Zihao He, Leili Tavabi, Kristina Lerman, and Mohammad Soleymani. Speaker turn modeling for dialogue act classification. In Marie-Francine Moens, Xuanjing Huang, Lucia Specia, and Scott Wen-tau Yih, editors, *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 2150–2157, Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.findings-emnlp.185. URL https://aclanthology.org/2021.findings-emnlp.185.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, nov 1997. ISSN 0899-7667. doi: 10.1162/neco.1997.9.8.1735. URL https://doi.org/10.1162/neco.1997.9.8.1735.

Shui Hu, Yang Liu, Anna Gottardi, Behnam Hedayatnia, Anju Khatri, Anjali Chadha, Qinlang Chen, Pankaj Rajan, Ali Binici, Varun Somani, Yao Lu, et al. Further advances in open domain dialog systems in the fourth alexa prize socialbot grand challenge. In *Alexa Prize SocialBot Grand Challenge 4 Proceedings*, 2021. URL https://www.amazon.science/publications/further-advances-in-open-domain-dialog-systems-in-the-fourth-alexa-prize-socialbot-grand-challenge.

Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional LSTM-CRF models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015.

Vít Jakimiv, Tomáš Zajíček, Jan Pichl, Petr Marek, Jakub Konrád, Jan Šedivý, Klára Lorencová, Tereza Pazderová, Markéta Vaňková, Ondřej Hrách, Martin Matulík, Tereza Matoulek, Vojtěch Badalec, Tomáš Kormaňák, and Petr Lorenc. Elysai - Ease your mind. https://www.elysai.com/, 2021. [Accessed 11-02-2024].

Dan Jurafsky and Elizabeth Shriberg. Switchboard swbd-damsl shallow-discourse-function annotation coders manual. 1997.

C. Maria Keet. *Open World Assumption*, pages 1567–1567. Springer New York, New York, NY, 2013. ISBN 978-1-4419-9863-7. doi: 10.1007/978-1-4419-9863-7_734. URL https://doi.org/10.1007/978-1-4419-9863-7_734.

J. F. Kelley. An iterative design methodology for user-friendly natural language office information applications. *ACM Trans. Inf. Syst.*, 2(1):26–41, January 1984. ISSN 1046-8188. doi: 10.1145/357417.357420. URL https://doi.org/10.1145/357417.357420.

Chandra Khatri, Behnam Hedayatnia, Anu Venkatesh, Jeff Nunn, Yi Pan, Qing Liu, Han Song, Anna Gottardi, Sanjeev Kwatra, Sanju Pancholi, et al. Advancing the state of the art in open domain dialog systems through the alexa prize. *2nd Proceedings of Alexa Prize*, 2018a.

Chandra Khatri, Anu Venkatesh, Behnam Hedayatnia, Raefer Gabriel, Ashwin Ram, and Rohit Prasad. Alexa prize—state of the art in conversational ai. *AI Magazine*, 39(3): 40–55, 2018b.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751, 2014.

Bowon Ko and Ho-Jin Choi. Paraphrase bidirectional transformer with multi-task learning. In *2020 IEEE International Conference on Big Data and Smart Computing (BigComp)*, pages 217–220, 2020. doi: 10.1109/BigComp48618.2020.00-72.

Jakub Konrád, Jan Pichl, Petr Marek, Petr Lorenc, Van Duy Ta, Ondřej Kobza, Lenka Hýlová, and Jan Šedivý. Alquist 4.0: Towards social intelligence using generative models and dialogue personalization. In *4th Proceedings of Alexa Prize (Alexa Prize 2021)*, 2021.

Gakuto Kurata, Bing Xiang, Bowen Zhou, and Mo Yu. Leveraging sentence-level information with encoder LSTM for semantic slot filling. In Jian Su, Kevin Duh, and Xavier Carreras, editors, *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2077–2083, Austin, Texas, November 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1223. URL https://aclanthology.org/D16-1223.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*, 2019.

George Larionov, Zachary Kaden, Hima Varsha Dureddy, Gabriel Bayomi T Kalejaiye, Mihir Kale, Srividya Pranavi Potharaju, Ankit Parag Shah, and Alexander I Rudnicky. Tartan: A retrieval-based socialbot powered by a dynamic finite-state machine architecture. 2018.

Stefan Larson, Anish Mahendran, Joseph J. Peper, Christopher Clarke, Andrew Lee, Parker Hill, Jonathan K. Kummerfeld, Kevin Leach, Michael A. Laurenzano, Lingjia Tang, and Jason Mars. An evaluation dataset for intent classification and out-of-scope prediction. In Kentaro Inui, Jing Jiang, Vincent Ng, and Xiaojun Wan, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1311–1316, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-1131. URL `https://aclanthology.org/D19-1131`.

Heeyoung Lee, Andreas Stolcke, and Elizabeth Shriberg. Using out-of-domain data for lexical addressee detection in human-human-computer dialog. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 221–229, Atlanta, Georgia, June 2013. Association for Computational Linguistics. URL `https://aclanthology.org/N13-1022`.

Sungjin Lee and Igor Shalyminov. Contextual out-of-domain utterance handling with counterfeit data augmentation. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7205–7209, 2019. doi: 10.1109/ICASSP.2019.8683019.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195, 2015.

Xiujun Li, Yun-Nung Chen, Lihong Li, Jianfeng Gao, and Asli Celikyilmaz. End-to-end task-completion neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 733–743, 2017.

Bing Liu and Ian Lane. Attention-based recurrent neural network models for joint intent detection and slot filling. *Interspeech 2016*, 2016.

Fei Liu and Julien Perez. Gated end-to-end memory networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 1–10, 2017.

Pasquale Lops, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. *Recommender systems handbook*, pages 73–105, 2011.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the*

49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/P11-1015`.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *ACL (System Demonstrations)*, pages 55–60, 2014.

Petr Marek. Dialog manager for conversational AI. 2018.

Petr Marek. *Dialogue Management for Conversational AI.* Phd thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, Department of Cybernetics, 2023. Available at `https://dspace.cvut.cz/handle/10467/112773`.

Petr Marek, Vishal Ishwar Naik, Anuj Goyal, and Vincent Auvray. Oodgan: Generative adversarial network for out-of-domain data generation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers*, pages 238–245, 2021.

Michael L. Mauldin. Chatterbots, tinymuds, and the turing test entering the loebner prize competition. In *Proceedings of the Twelfth AAAI National Conference on Artificial Intelligence*, AAAI'94, page 16–21. AAAI Press, 1994.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013b.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*, 2018.

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015.

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.

Attila Nagy, Bence Bial, and Judit Ács. Automatic punctuation restoration with bert models. *arXiv preprint arXiv:2101.07343*, 2021.

Pandu Nayak. Understanding searches better than ever before. `https://blog.google/products/search/search-language-understanding-bert/`, 2019. Accessed: 10. 12. 2023.

Joakim Nivre. Dependency parsing. *Language and Linguistics Compass*, 4(3):138–152, 2010.

Christopher Olah. Understanding LSTM Networks – colah's blog — colah.github.io. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`, 2015. [Accessed 24-01-2024].

OpenAI. Introducing ChatGPT — openai.com. `https://openai.com/blog/chatgpt`, 2022. [Accessed 11-01-2024].

Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project, 1998. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.31.1768`.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Claudia Pateras, Nicolas Chapados, Remi Kwan, Dominic Lavoie, and Réal Tremblay. A mixed-initiative natural dialogue system for conference room reservation. In *EUROSPEECH*, 1999.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL `https://aclanthology.org/N18-1202`.

Jan Pichl. Enhancing Question Answering with Structured Database Queries. Master's thesis, Czech Technical University in Prague, Faculty of Electrical Engineering, 2016.

Jan Pichl. Dialogue act detection using contextual knowledge. In *POSTER 2018-22nd International Student Conference on Electrical Engineering*, 2018.

Jan Pichl, Petr Marek, Jakub Konrád, Martin Matulík, Hoang Long Nguyen, and Jan Šedivỳ. Alquist: The alexa prize socialbot. In *1st Proceedings of Alexa Prize (Alexa Prize 2017)*, 2017.

Jan Pichl, Petr Marek, Jakub Konrád, Martin Matulík, and Jan Šedivỳ. Alquist 2.0: Alexa prize socialbot based on sub-dialogue models. In *2nd Proceedings of the Alexa Prize (AP 2018).*, 2018.

Jan Pichl, Petr Marek, Jakub Konrád, Petr Lorenc, Van Duy Ta, and Jan Šedivỳ. Alquist 3.0: Alexa prize bot using conversational knowledge graph. In *3rd Proceedings of Alexa Prize (Alexa Prize 2020).*, 2020.

Jan Pichl, Petr Marek, Jakub Konrád, Petr Lorenc, Ondřej Kobza, Tomáš Zajícek, and Jan Šedivỳ. Flowstorm: Open-source platform with hybrid dialogue architecture. *NAACL 2022*, page 39, 2022.

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

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.

Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, et al. Conversational ai: The science behind the alexa prize. 2017.

Ashwin Ram, Rohit Prasad, Chandra Khatri, Anu Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, et al. Conversational ai: The science behind the alexa prize. *arXiv preprint arXiv:1801.03604*, 2018.

Lance A Ramshaw and Mitchell P Marcus. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer, 1999.

Suman Ravuri and Andreas Stolcke. Recurrent neural network and lstm models for lexical utterance classification. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

Paul Resnick and Hal R Varian. Recommender systems. *Communications of the ACM*, 40 (3):56–58, 1997.

John Roach. Microsoft researchers release graph that helps machines conceptualize, Nov 2016. URL `https://blogs.microsoft.com/ai/microsoft-researchers-release-graph-that-helps-machines-conceptualize/`.

Carl R Rogers. The necessary and sufficient conditions of therapeutic personality change. *Journal of consulting psychology*, 21(2):95, 1957.

Harvey Sacks, Emanuel A Schegloff, and Gail Jefferson. A simplest systematics for the organization of turn taking for conversation. In *Studies in the organization of conversational interaction*, pages 7–55. Elsevier, 1978.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*, 2019.

Jost Schatzmann, Blaise Thomson, and Steve Young. Error simulation for training statistical dialogue systems. In *Automatic Speech Recognition & Understanding, 2007. ASRU. IEEE Workshop on*, pages 526–531. IEEE, 2007.

Mike Schuster and Kaisuke Nakajima. Japanese and korean voice search. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5149–5152. IEEE, 2012.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, 2016.

Min Joon Seo, Hannaneh Hajishirzi, and Ali Farhadi. Query-regression networks for machine comprehension. *CoRR, abs/1606.04582*, 2016.

Iulian Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. A hierarchical latent variable encoder-decoder model for generating dialogues. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31, 2017.

Tom Seymour, Dean Frantsvog, Satheesh Kumar, et al. History of search engines. *International Journal of Management & Information Systems (IJMIS)*, 15(4):47–58, 2011.

Hangjie Shi, Leslie Ball, Govind Thattai, Desheng Zhang, Lucy Hu, Qiaozi (QZ) Gao, Suhaila Shakiah, Xiaofeng Gao, Aishwarya Padmakumar, Bofei Yang, Cadence Chung, Dinakar Guthy, Gaurav Sukhatme, Karthika Arumugam, Matthew Wen, Osman Ipek, Patrick Lange, Rohan Khanna, Shreyas Pansare, Vasu Sharma, Chao Zhang, Cris Flagg, Daniel Pressel, Lavina Vaz, Luke Dai, Prasoon Goyal, Sattvik Sahai, Shaohua Liu, Yao Lu, Anna Gottardi, Shui Hu, Yang Liu, Dilek Hakkani-Tür, Kate Bland, Heather Rocker, James Jeun, Yadunandana Rao, Michael Johnston, Akshaya Iyengar, Arindam Mandal, Prem Natarajan, and Reza Ghanadan. Alexa, play with robot: Introducing the first alexa prize simbot challenge on embodied ai. In *Alexa Prize SimBot Challenge Proceedings*, 2023. URL https://www.amazon.science/alexa-prize/proceedings/alexa-play-with-robot-introducing-the-first-alexa-prize-simbot-challenge-on-embodied-ai.

Amit Singhal. Introducing the knowledge graph: things, not strings. https://blog.google/products/search/introducing-knowledge-graph-things-not/, 2012. Accessed: 10. 12. 2023.

Kaitao Song, Xu Tan, Tao Qin, Jianfeng Lu, and Tie-Yan Liu. Mpnet: Masked and permuted pre-training for language understanding. *arXiv preprint arXiv:2004.09297*, 2020.

Valentin I. Spitkovsky and Angel X. Chang. A cross-lingual dictionary for english wikipedia concepts. In *LREC*, 2012.

Dima Suleiman, Arafat Awajan, and Nailah Al-Madi. Deep learning based technique for plagiarism detection in arabic texts. 10 2017. doi: 10.1109/ICTCS.2017.42.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. Out-of-domain detection for low-resource text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3557–3563, 2019a.

Ming Tan, Yang Yu, Haoyu Wang, Dakuo Wang, Saloni Potdar, Shiyu Chang, and Mo Yu. Out-of-domain detection for low-resource text classification tasks. *arXiv preprint arXiv:1909.05357*, 2019b.

Henry S. Thompson, Anne Anderson, Ellen Gurman Bard, Gwyneth Doherty-Sneddon, Alison Newlands, and Cathy Sotillo. The HCRC map task corpus: Natural dialogue for speech recognition. In *Human Language Technology: Proceedings of a Workshop Held at Plainsboro, New Jersey, March 21-24, 1993*, 1993. URL https://aclanthology.org/H93-1005.

Ottokar Tilk and Tanel Alumäe. Bidirectional recurrent neural network with attention mechanism for punctuation restoration. In *Interspeech 2016*, 2016.

Erik F. Tjong Kim Sang and Fien De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 142–147, 2003. URL https://aclanthology.org/W03-0419.

Maxim Tkachenko, Mikhail Malyuk, Andrey Holmanyuk, and Nikolai Liubimov. Label Studio: Data labeling software, 2020-2022. URL https://github.com/heartexlabs/label-studio. Open source software available from https://github.com/heartexlabs/label-studio.

Loan Trinh Van, Thuy Dao Thi Le, Thanh Le Xuan, and Eric Castelli. Emotional speech recognition using deep neural networks. *Sensors*, 22(4):1414, 2022.

Alper Kursat Uysal and Serkan Gunal. The impact of preprocessing on text classification. *Information processing & management*, 50(1):104–112, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.

Oriol Vinyals and Quoc Le. A neural conversational model. *arXiv preprint arXiv:1506.05869*, 2015.

Voiceflow. Voiceflow: Build amazing AI chatbots. https://www.voiceflow.com/, 2019. [Accessed 11-02-2024].

Atro Voutilainen. *Part-of-speech tagging*, volume 219. The Oxford handbook of computational linguistics, 2003.

Richard Wallace. The elements of aiml style. *Alice AI Foundation*, 139, 2003.

Yao Wan, Wenqiang Yan, Jianwei Gao, Zhou Zhao, Jian Wu, and Philip S. Yu. Improved dynamic memory network for dialogue act classification with adversarial training. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 841–850, 2018. doi: 10.1109/BigData.2018.8622245.

Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. ISBN 9781713829546.

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.

Tsung-Hsien Wen, David Vandyke, Nikola Mrkšić, Milica Gašić, Lina M. Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. In Mirella Lapata, Phil Blunsom, and Alexander Koller, editors, *Proceedings of the 15th Conference of the European Chapter of* *the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449, Valencia, Spain, April 2017. Association for Computational Linguistics. URL https://aclanthology.org/E17-1042.

Jason Weston, Antoine Bordes, Sumit Chopra, Alexander M Rush, Bart van Merriënboer, Armand Joulin, and Tomas Mikolov. Towards ai-complete question answering: A set of prerequisite toy tasks. *arXiv preprint arXiv:1502.05698*, 2015.

Jason D Williams, Antoine Raux, and Matthew Henderson. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33, 2016.

Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. *arXiv preprint arXiv:1702.03274*, 2017.

Ledell Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. Starspace: Embed all the things! *arXiv preprint arXiv:1709.03856*, 2017.

Wei Wu, Can Xu, Yu Wu, and Zhoujun Li. Towards interpretable chit-chat: Open domain dialogue generation with dialogue acts. 2018.

Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Q. Zhu. Probase: A probabilistic taxonomy for text understanding. May 2012. URL https://www.microsoft.com/en-us/research/publication/probase-a-probabilistic-taxonomy-for-text-understanding/.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Jia-Yu Yao, Kun-Peng Ning, Zhen-Hui Liu, Mu-Nan Ning, and Li Yuan. Llm lies: Hallucinations are not bugs, but features as adversarial examples. *arXiv preprint arXiv:2310.01469*, 2023.

Fanghua Ye, Meng Fang, Shenghui Li, and Emine Yilmaz. Enhancing conversational search: Large language model-aided informative query rewriting. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 5985–6006, 2023.

Jui-Feng Yeh. Speech act identification using semantic dependency graphs with probabilistic context-free grammars. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 15(1):1–28, 2016.

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

Dian Yu and Zhou Yu. Midas: A dialog act annotation scheme for open domain human-machine spoken conversations. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1103–1120, 2021.

Dian Yu, Michelle Cohn, Yi Mang Yang, Chun Yen Chen, Weiming Wen, Jiaping Zhang, Mingyang Zhou, Kevin Jesse, Austin Chau, Antara Bhowmick, Shreenath Iyer, Giritheja Sreenivasulu, Sam Davidson, Ashwin Bhandare, and Zhou Yu. Gunrock: A social bot for complex and engaging long conversations. In Sebastian Padó and Ruihong Huang, editors, *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP): System Demonstrations*, pages 79–84, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-3014. URL `https://aclanthology.org/D19-3014`.

Hongxia Jin Yu Wang, Yilin Shen. A bi-model based rnn semantic frame parsing model for intent detection and slot filling, 2018. URL `https://arxiv.org/abs/1812.10235`.

Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. *Dive into Deep Learning*. Cambridge University Press, 2023. `https://D2L.ai`.

Hanlei Zhang, Hua Xu, and Ting-En Lin. Deep open intent classification with adaptive decision boundary. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14374–14382, 2021.

Yizhe Zhang, Siqi Sun, Michel Galley, Yen-Chun Chen, Chris Brockett, Xiang Gao, Jianfeng Gao, Jingjing Liu, and Bill Dolan. Dialogpt: Large-scale generative pre-training for conversational response generation. *arXiv preprint arXiv:1911.00536*, 2019.

Reviewed Publications of the Author Relevant to the Thesis

[R.1] **Pichl Jan**; *Alquist: An Open-Domain Dialogue System* In: POSTER 2017 - 21st International Student Conference on Electrical Engineering. Prague, Czech Republic, 2017.

Jan Pichl (100%): Conceptualization, System architecture, NLU, Implementation, Experiments, Writing, Project administration.

[R.2] **Pichl Jan**, Konrád Jakub, Nguyen Hoang Long, Matulík Martin, Marek Petr, Šedivý Jan; *Alquist: The alexa prize socialbot* In: 1st Proceedings of Alexa Prize (Alexa Prize 2017) Las Vegas, NV, USA, 2017.

Jan Pichl (55%): Conceptualization, System architecture, NLU, Implementation, Experiments, Writing, Project administration.

Petr Marek (10%): Conceptualization, Dialogue management, Implementation, Experiments, Writing.

Jakub Konrád (20%): Conceptualization, Knowledge management, Implementation, Experiments, Writing.

Long Hoang Nguyen (5%): Conceptualization, Implementation, Writing.

Martin Matulík (5%): Conceptualization, Implementation, Writing.

Jan Šedivý (5%): Supervision, Conceptualization, Project administration, Review of writing. The paper has been cited in:

1. Ram Ashwin, Rohit Prasad, Chandra Khatri, Anushree Venkatesh, Raefer Gabriel, Qing Liu, Jeff Nunn, Behnam Hedayatnia, Ming Cheng, Ashish Nagar, others. Conversational AI: The science behind the Alexa Prize. 2017.

   Indexed in: Google Scholar.

2. Khatri Chandra, Anu Venkatesh, Behnam Hedayatnia, Raefer Gabriel, Ashwin Ram, Rohit Prasad. Alexa prize—state of the art in conversational AI. *AI Magazine 39*. 3: 40–55, 2018.

   Indexed in: Scopus, Web of Science, Google Scholar.

3. Chen Chun-Yen, Dian Yu, Weiming Wen, Yi Mang Yang, Jiaping Zhang, Mingyang Zhou, Kevin Jesse, Austin Chau, Antara Bhowmick, Shreenath Iyer, others. Gunrock: Building a human-like social bot by leveraging large scale real user data. 2018.

   Indexed in: Google Scholar.

4. Larionov George, Zachary Kaden, Hima Varsha Dureddy, Gabriel Bayomi T Kalejaiye, Mihir Kale, Srividya Pranavi Potharaju, Ankit Parag,Shah, Alexander I Rudnicky. Tartan: A retrieval-based socialbot powered by a dynamic finite-state machine architecture. 2018.

   Indexed in: Google Scholar.

5. Ostendorf Mari. Several Steps Forward in Conversational AI. 2018.

   Indexed in: Google Scholar.

6. Malik Szymon, Adrian Lancucki, Jan Chorowski. Efficient Purely Convolutional Text Encoding. 2018.

   Indexed in: Scopus, Google Scholar.

7. Jonell Patrik, Per Fallgren, Fethiye Irmak Doğan, Jose Lopes, Ulme Wennberg, Gabriel Skantze. Crowdsourcing a self-evolving dialog graph. *Proceedings of the 1st International Conference on Conversational User Interfaces*, 2019.

   Indexed in: Scopus, Web of Science, Google Scholar.

8. Li Aaron W, Veronica Jiang, Steven Y Feng, Julia Sprague, Wei Zhou, Jesse Hoey. Follow alice into the rabbit hole: Giving dialogue agents understanding of human level attributes. 2019.

   Indexed in: Google Scholar.

9. Choi Jason Ingyu, Ali Ahmadvand, Eugene Agichtein. Offline and online satisfaction prediction in open-domain conversational systems. *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019.

   Indexed in: Scopus, Web of Science, Google Scholar.

10. Huang Minlie, Xiaoyan Zhu, and Jianfeng Gao. Challenges in building intelligent open-domain dialog systems. *ACM Transactions on Information Systems (TOIS)*, 38.3 1-32, 2020.

    Indexed in: Scopus, Web of Science, Google Scholar.

11. Liang Kaihui, Austin Chau, Yu Li, Xueyuan Lu, Dian Yu, Mingyang Zhou, Ishan Jain, Sam Davidson, Josh Arnold, Minh Nguyen, others. Gunrock 2.0: A user adaptive social conversational system. 2020.

Indexed in: Google Scholar.

12. Zolitschka Jan Felix. A novel multi-agent-based chatbot approach to orchestrate conversational assistants. *Business Information Systems: 23rd International Conference, BIS 2020,* Colorado Springs, CO, USA, June 8–10, 2020, Proceedings 23, 2020.

    Indexed in: Scopus, Google Scholar.

13. Fang Hao. Building A User-Centric and Content-Driven Socialbot. 2020.

    Indexed in: Google Scholar.

14. Song Haoyu, Yan Wang, Weinan Zhang, Zhengyu Zhao, Ting Liu, Xiaojiang Liu. Profile Consistency Identification for Open-domain Dialogue Agents. *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020.

    Indexed in: Scopus, Web of Science, Google Scholar.

15. Li Aaron W, Veronica Jiang, Steven Y Feng, Julia Sprague, Wei Zhou, Jesse Hoey. Aloha: Artificial learning of human attributes for dialogue agents. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2020.

    Indexed in: Scopus, Web of Science, Google Scholar.

16. Nehring Jan, Nils Feldhus, Harleen Kaur, Akhyar Ahmed. Combining open domain question answering with a task-oriented dialog system. *Proceedings of the 1st Workshop on Document-grounded Dialogue and Conversational Question Answering (DialDoc 2021)*, 2021.

    Indexed in: Scopus, Web of Science, Google Scholar.

17. Walker Nicholas Thomas, Stefan Ultes, Pierre Lison. GraphWOZ: Dialogue Management with Conversational Knowledge Graphs. 2022.

    Indexed in: Google Scholar.

[R.3] **Pichl Jan**; *Dialogue Act Detection Using Contextual Knowledge* In: POSTER 2018 - 22st International Student Conference on Electrical Engineering. Prague, Czech Republic, 2018.

Jan Pichl (100%): Conceptualization, Implementation, Experiments, Writing.

- The second best paper award in Informatics and Cybernetics category

[R.4] **Pichl Jan**, Konrád Jakub, Matulík Martin, Marek Petr, Lorenc Petr, Šedivý Jan; *Alquist 2.0: Alexa Prize Socialbot Based on Sub-Dialogue Models* In: 2nd Proceedings of Alexa Prize (Alexa Prize 2018) Las Vegas, NV, USA, 2018.

Jan Pichl (30%): Conceptualization, System architecture, NLU, Implementation, Experiments, Writing, Project administration.

Petr Marek (30%): Conceptualization, Dialogue management, Implementation, Experiments, Writing.

Jakub Konrád (30%): Conceptualization, Knowledge management, Implementation, Experiments, Writing.

Martin Matulík (5%): Conceptualization, Implementation, Writing.

Jan Šedivý (5%): Supervision, Conceptualization, Project administration, Review of writing. The paper has been cited in:

1. Bowden Kevin K, Jiaqi Wu, Wen Cui, Juraj Juraska, Vrindavan Harrison, Brian Schwarzmann, Nicholas Santer, Steve Whittaker, Marilyn Walker. Entertaining and opinionated but too controlling: a large-scale user study of an open domain Alexa prize system. *Proceedings of the 1st International Conference on Conversational User Interfaces,* 2019.

   Indexed in: Scopus, Web of Science, Google Scholar.

2. Gabriel Raefer, Yang Liu, Anna Gottardi, Mihail Eric, Anju Khatri, Anjali Chadha, Qinlang Chen, Behnam Hedayatnia, Pankaj Rajan, Ali Binici, others. Further advances in open domain dialog systems in the third alexa prize socialbot grand challenge. 2019.

   Indexed in: Google Scholar.

3. Szpektor Idan, Deborah Cohen, Gal Elidan, Michael Fink, Avinatan Hassidim, Orgad Keller, Sayali Kulkarni, Eran Ofek, Sagie Pudinsky, Asaf Revach, others. Dynamic composition for conversational domain exploration. *Proceedings of The Web Conference 2020,* 2020.

   Indexed in: Scopus, Web of Science, Google Scholar.

4. Paranjape Ashwin, Abigail See, Kathleen Kenealy, Haojun Li, Amelia Hardy, Peng Qi, Kaushik Ram Sadagopan, Nguyet Minh Phu, Dilara Soylu, Christopher D Manning. Neural generation meets real people: Towards emotionally engaging mixed-initiative conversations. 2020.

   Indexed in: Google Scholar.

5. Fang Hao. Building A User-Centric and Content-Driven Socialbot. 2020.

   Indexed in: Google Scholar.

6. Kuratov Yuri, Idris Yusupov, Dilyara Baymurzina, Denis Kuznetsov, Daniil Cherniavskii, Alexander Dmitrievskiy, Elena Ermakova, Fedor Ignatov, Dmitry Karpov, Daniel Kornev, others. Dream technical report for the alexa prize 2019. 2020.

   Indexed in: Google Scholar.

7. Juraska Juraj, Kevin Bowden, Lena Reed, Vrindavan Harrison, Wen Cui, Omkar Patil, Rishi Rajasekaran, Angela Ramirez, Cecilia Li, Eduardo Zamora, others.

Athena 2.0: Contextualized Dialogue Management for an Alexa Prize SocialBot. *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, 2021.

Indexed in: Scopus, Web of Science, Google Scholar.

8. Vasquez-Correa Juan Camilo, Juan Carlos Guerrero-Sierra, Jose Luis Pemberty-Tamayo, Juan Esteban Jaramillo, Andres Felipe Tejada-Castro. One system to rule them all: A universal intent recognition system for customer service chatbots. 2021.

   Indexed in: Google Scholar.

9. Walker Nicholas Thomas, Torbjørn Dahl, Pierre Lison. Dialogue management as graph transformations. *Conversational AI for Natural Human-Centric Interaction: 12th International Workshop on Spoken Dialogue System Technology, IWSDS 2021,* Singapore, 2022.

   Indexed in: Scopus, Google Scholar.

10. Vavrecka Michal, Gabriela Sejnova, Petr Schimperk. Personified Robotic Chatbot Based On Compositional Dialogues. *2022 International Conference on Interactive Media, Smart Systems and Emerging Technologies (IMET),* 2022.

    Indexed in: Scopus, Google Scholar.

[R.5] **Pichl Jan**, Konrád Jakub, Marek Petr, Lorenc Petr, Duy Ta Van, Šedivý Jan; *Alquist 3.0: Alexa Prize Bot Using Conversational Knowledge Graph* In: 3rd Proceedings of Alexa Prize (Alexa Prize 2020) Seattle, USA, 2020.

Jan Pichl (19%): Conceptualization, System architecture, NLU, Implementation, Experiments, Writing, Project administration.

Petr Marek (20%): Conceptualization, Dialogue management, Implementation, Experiments, Writing.

Jakub Konrád (20%): Conceptualization, Knowledge management, Implementation, Experiments, Writing.

Petr Lorenc (20%): Conceptualization, NLP, Implementation, Experiments, Writing.

Van Duy Ta (20%): Conceptualization, Implementation, System architecture, Writing.

Jan Šedivý (1%): Supervision, Project administration, Review of writing. The

paper has been cited in:

1. Gabriel Raefer, Yang Liu, Anna Gottardi, Mihail Eric, Anju Khatri, Anjali Chadha, Qinlang Chen, Behnam Hedayatnia, Pankaj Rajan, Ali Binici, others. Further advances in open domain dialog systems in the third alexa prize socialbot grand challenge. 2019.

Indexed in: Google Scholar.

2. Axelsson Agnes, Hendrik Buschmeier, Gabriel Skantze. Modeling feedback in interaction with conversational agents—a review. *Frontiers in Computer Science 4,* 2022.

   Indexed in: Scopus, Web of Science, Google Scholar.

3. Ahlberg Sofie, Agnes Axelsson, Pian Yu, Wenceslao Shaw Cortez, Yuan Gao, Ali Ghadirzadeh, Ginevra Castellano, Danica Kragic, Gabriel Skantze, Dimos V Dimarogonas. Co-adaptive Human–Robot Cooperation: Summary and Challenges. *Unmanned Systems* 10. 02(2022): 187–203, 2022.

   Indexed in: Scopus, Web of Science, Google Scholar.

4. Kann Katharina, Abteen Ebrahimi, Joewie Koh, Shiran Dudy, Alessandro Roncone. Open-domain dialogue generation: What we can do, cannot do, and should do next. *Proceedings of the 4th Workshop on NLP for Conversational AI,* 2022.

   Indexed in: Scopus, Web of Science, Google Scholar.

5. Yamashita Sanae, Ryuichiro Higashinaka. Data Collection for Empirically Determining the Necessary Information for Smooth Handover in Dialogue. *Proceedings of the Thirteenth Language Resources and Evaluation Conference,* 2022.

   Indexed in: Scopus, Web of Science, Google Scholar.

6. Axelsson Agnes, Gabriel Skantze. Do you follow? a fully automated system for adaptive robot presenters. *Proceedings of the 2023 ACM/IEEE International Conference on Human-Robot Interaction,* 2023.

   Indexed in: Google Scholar.

7. Harrison Vrindavan, Rishi Rajasekaran, Marilyn Walker. A Transformer-based Response Evaluator for Open-Domain Spoken Conversation. 2023.

   Indexed in: Google Scholar.

8. Reddy Revanth Gangi, Sharath Chandra, Mankeerat Singh Sidhu, H Bai, Wenta, Yao, Pranav Pillai, Karan Aggarwal, Liliang Ren, Prathamesh Sonawane, Keyu Han, others. CharmBana: Progressive Responses with Real-Time Internet Search for Knowledge-Powered Conversations. 2023.

   Indexed in: Google Scholar.

[R.6] Konrád Jakub, **Pichl Jan**, Marek Petr, Lorenc Petr, Duy Ta Van, Kobza Ondřej, Hýlová Lenka, Šedivý Jan; *Alquist 4.0: Towards Social Intelligence Using Generative Models and Dialogue Personalization* In: 4th Proceedings of Alexa Prize (Alexa Prize 2021) Seattle, USA, 2021.

Jakub Konrád (16%): Conceptualization, Knowledge management, Implementation, Experiments, Writing, Project administration.

Jan Pichl (16%): Conceptualization, System architecture, NLU, Implementation, Experiments, Writing.

Petr Marek (16%): Conceptualization, Dialogue management, Neural Response Generator, Implementation, Experiments, Writing.

Petr Lorenc (16%): Conceptualization, NLP, Experiments, Writing.

Van Duy Ta (16%): Implementation, Writing.

Ondřej Kobza (16%): Implementation, Writing.

Lenka Hýlová (2%): Neural Response Generator, Implementation, Experiments.

Jan Šedivý (2%): Supervision, Project administration, Review of writing.

The paper has been cited in:

1. Kuznetsov Denis, Dmitry Evseev, Lidia Ostyakova, Oleg Serikov, Daniel Kornev, Mikhail Burtsev. Discourse-Driven Integrated Dialogue Development Environment for Open-Domain Dialogue Systems. *Proceedings of the 2nd Workshop on Computational Approaches to Discourse,* 2021.

   Indexed in: Scopus, Google Scholar.
2. Jin Di, Sijia Liu, Yang Liu, Dilek Hakkani-Tur. Improving Bot Response Contradiction Detection via Utterance Rewriting. *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue,* 2022.

   Indexed in: Google Scholar.
3. Hedayatnia Behnam, Di Jin, Yang Liu, Dilek Hakkani-Tur. A Systematic Evaluation of Response Selection for Open Domain Dialogue. *Proceedings of the 23rd Annual Meeting of the Special Interest Group on Discourse and Dialogue,* 2022.

   Indexed in: Google Scholar.
4. Bowden Kevin K, Marilyn Walker. Let's Get Personal: Personal Questions Improve SocialBot Performance in the Alexa Prize. 2023.

   Indexed in: Google Scholar.
5. Vicente, Frederico, Rafael, Ferreira, David, Semedo, Joao, Magalhaes. The Wizard of Curiosities: Enriching Dialogues with Fun Facts. *Proceedings of the 24th Meeting of the Special Interest Group on Discourse and Dialogue,* 2023.

   Indexed in: Google Scholar.
6. Mai Long, Julie Carson-Berndsen. I already said that! Degenerating redundant questions in open-domain dialogue systems. *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics* (Volume 4: Student Research Workshop), 2023.

   Indexed in: Google Scholar.

[R.7] Marek Petr, Müller Štěpán, Konrád Jakub, Lorenc Petr, **Pichl Jan**, Šedivý Jan; *Text Summarization of Czech News Articles Using Named Entities* In: The Prague Bulletin of Mathematical Linguistics 116 (2021): 5-25. Prague, Czech Republic, 2021.

Petr Marek (55%): Conceptualization, Implementation, Experiments, Writing.

Štěpán Müller (25%): Conceptualization, Implementation, Experiments.

Jakub Konrád (5%): Consultations, Review of writing.

Petr Lorenc (5%): Consultations, Review of writing.

Jan Pichl (5%): Consultations, Review of writing.

Jan Šedivý (5%): Supervision, Review of writing. The paper has been cited in:

1. Matsushita Kyoumoto, Takuya Makino, Tomoya Iwakura. Improving Neural Language Processing with Named Entities. *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021),* 2021.

   Indexed in: Scopus, Google Scholar.
2. Rathi Kartik, Saumy Raj, Sudhir Mohan, Yash Vardan Singh. A Review of state-of-the-art Automatic Text Summarisation. *International Journal of Creative Research Thoughts (2022),* 2022.

   Indexed in: Google Scholar.
3. Rehman Tohida, Debarshi Kumar Sanyal, Prasenjit Majumder, Samiran Chattopadhyay. Named Entity Recognition Based Automatic Generation of Research Highlights. *Third Workshop on Scholarly Document Processing,* 2022.

   Indexed in: Google Scholar.
4. Lima Khadija Akter, Khan Md Hasib, Sami Azam, Asif Karim, Sidratul Montaha, Sheak Rashed Haider Noori, Mirjam Jonkman. A novel Data and Model Centric artificial intelligence based approach in developing high-performance Named Entity Recognition for Bengali Language. *Plos one 18.* 9(2023): e0287818, 2023.

   Indexed in: Google Scholar.
5. Choudhary Ambrish, Mamatha Alugubelly, Rupal Bhargava. A Comparative Study on Transformer-based News Summarization. *2023 15th International Conference on Developments in eSystems Engineering (DeSE),* 2023.

   Indexed in: Scopus, Google Scholar.
6. Vajdecka Peter, Vojtech Svatek, Martin Vita. A Novel Approach to Abstractive Summarization Based on LOF, Sentence-BERT and T5–with Fact Checking Use Case. 2023.

   Indexed in: Google Scholar.

[R.8] Lorenc Petr, Marek Petr, **Pichl Jan**, Konrád Jakub, Šedivý Jan; *Benchmark of public intent recognition services* Language Resources and Evaluation (2022): volume 56, 1023-1041.

Petr Lorenc (80%): Conceptualization, Implementation, Experiments, Writing.

Petr Marek (5%): Consulatations, Review of writing.

Jan Pichl (5%): Consulatations, Review of writing.

Jakub Korád (5%): Consulatations, Review of writing.

Jan Šedivý (5%): Supervision, Review of writing. The paper has been cited in:

1. Vasquez-Correa Juan Camilo, Juan Carlos Guerrero-Sierra, Jose Luis Pemberty-Tamayo, Juan Esteban Jaramillo, Andres Felipe Tejada-Castro. One system to rule them all: A universal intent recognition system for customer service chatbots. 2021.

   Indexed in: Google Scholar.

[R.9] Lorenc Petr, Gargiani Tommaso, **Pichl Jan**, Konrád Jakub, Marek Petr, Šedivý, Jan, Kobza Ondřej *Metric Learning and Adaptive Boundary for Out-of-Domain Detection* In: Natural Language Processing and Information Systems (2022): 127-134.

Petr Lorenc (30%): Conceptualization, Implementation, Experiments, Writing.

Tommaso Gargiani (30%): Conceptualization, Implementation, Experiments, Writing.

Jan Pichl (10%): Consultations, Review of writing.

Jakub Korád (10%): Consultations, Review of writing.

Petr Marek (10%): Consultations, Review of writing.

Ondřej Kobza (5%): Consultations, Review of writing.

Jan Šedivý (5%): Supervision, Review of writing.

[R.10] **Pichl Jan**, Marek Petr, Konrád Jakub, Lorenc Petr, Kobza Ondřej, Zajíček Tomáš, Šedivý Jan; *Flowstorm: Open-Source Platform with Hybrid Dialogue Architecture* Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations (2022): 39-45.

Jan Pichl (55%): Conceptualization, Implementation, Experiments, Writing.

Petr Marek (10%): Conceptualization, Implementation, Review of writing.

Jakub Korád (10%): Conceptualization, Implementation, Review of writing.

Petr Lorenc (10%): Conceptualization, Implementation, Review of writing.

Ondřej Kobza (5%): Review of writing.

Tomáš Zajíček (5%): Conceptualization, Supervision, Implementation.

Jan Šedivý (5%): Supervision, Review of writing.

The paper has been cited in:

1. Pande Charuta, Andreas Martin, Christoph Pimmer. Towards hybrid dialog management strategies for a health coach chatbot. *Proceedings of the AAAI 2023 Spring Symposium on Challenges Requiring the Combination of Machine Learning and Knowledge Engineering (AAAI-MAKE 2023).* Vol. 3433, 2023.

   Indexed in: Scopus, Google Scholar.

Remaining Publications of the Author Relevant to the Thesis

[O.1] Baudiš Petr, **Pichl Jan**, Vyskočil Tomáš, Šedivý Jan; *Sentence pair scoring: Towards unified framework for text comprehension.* arXiv preprint arXiv:1603.06127, 2016.

Petr Baudiš (70%): Conceptualization, Implementation, Experiments, Writing.

Jan Pichl (15%): Implementation, Experiments.

Tomáš Vyskočil (10%): Implementation, Experiments.

Jan Šedivý (5%): Supervision, Review of writing. The paper has been cited in:

1. Henderson, Matthew, et al. Training neural response selection for task-oriented dialogue systems. arXiv preprint arXiv:1906.01543, 2019.
2. Lowe, Ryan, et al. Training end-to-end dialogue systems with the ubuntu dialogue corpus. Dialogue & Discourse, 2017, 8.1: 31-65.
3. Irving, Geoffrey, et al. Deepmath-deep sequence models for premise selection. Advances in neural information processing systems, 2016, 29.
4. Shao, Yang. Hcti at SemEval-2017 task 1: Use convolutional neural network to evaluate semantic textual similarity. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017). 2017. p. 130-133.
5. Opitz, Juri; DAZA, Angel; Frank, Anette. Weisfeiler-leman in the bamboo: Novel AMR graph metrics and a benchmark for AMR graph similarity. Transactions of the Association for Computational Linguistics, 2021, 9: 1425-1441.
6. Alemi, Alex A., et al. DeepMath-deep sequence models for premise selection. arXiv preprint arXiv:1606.04442, 2016.
7. Gogár, Tomas; Hubáček, Ondrej; Šedivý, Jan. Deep neural networks for web page information extraction. In: Artificial Intelligence Applications and Innovations: 12th IFIP WG 12.5 International Conference and Workshops, AIAI

2016, Thessaloniki, Greece, September 16-18, 2016, Proceedings 12. Springer International Publishing, 2016. p. 154-163.

8. Yoon, Seunghyun; Shin, Joongbo; Jung, Kyomin. Learning to rank question-answer pairs using hierarchical recurrent encoder with latent topic clustering. arXiv preprint arXiv:1710.03430, 2017.
9. Zhang, Guanhua, et al. Selection bias explorations and debias methods for natural language sentence matching datasets. arXiv preprint arXiv:1905.06221, 2019.
10. Chaudhuri, Debanjan, et al. Improving response selection in multi-turn dialogue systems by incorporating domain knowledge. arXiv preprint arXiv:1809.03194, 2018.
11. Ma, Wentao, et al. TripleNet: Triple attention network for multi-turn response selection in retrieval-based chatbots. arXiv preprint arXiv:1909.10666, 2019.
12. Liu, Bang, et al. Matching natural language sentences with hierarchical sentence factorization. In: Proceedings of the 2018 World Wide Web Conference. 2018. p. 1237-1246.
13. Dong, Jianxiong; HUANG, Jim. Enhance word representation for out-of-vocabulary on ubuntu dialogue corpus. arXiv preprint arXiv:1802.02614, 2018.
14. Nanda, Rohan, et al. Legal Information Retrieval Using Topic Clustering and Neural Networks. In: COLIEE@ ICAIL. 2017. p. 68-78.
15. Shou, Ziyi; Lin, Fangzhen. Evaluate AMR Graph Similarity via Self-supervised Learning. In: Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). 2023. p. 16112-16123.
16. GU, Jia-Chen; Ling, Zhen-Hua; Liu, Quan. Utterance-to-utterance interactive matching network for multi-turn response selection in retrieval-based chatbots. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2019, 28: 369-379.
17. Opitz, Juri; Frank, Anette. SBERT studies meaning representations: decomposing sentence embeddings into explainable AMR meaning features. arXiv preprint arXiv:2206.07023, 2022.
18. Gu, Jia-Chen, et al. Deep contextualized utterance representations for response selection and dialogue analysis. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2021, 29: 2443-2455.
19. Peinelt, Nicole; Liakata, Maria; Nguyen, Dong. Aiming beyond the obvious: Identifying non-obvious cases in semantic similarity datasets. In: Proceedings of the 57th annual meeting of the association for computational linguistics. 2019. p. 2792-2798.
20. Morales, Alvaro, et al. Learning to answer questions from wikipedia infoboxes. In: Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. 2016. p. 1930-1935.

21. Mousset, Paul; Pitarch, Yoann; Tamine, Lynda. End-to-end neural matching for semantic location prediction of tweets. ACM Transactions on Information Systems (TOIS), 2020, 39.1: 1-35.

22. Kolawole John, Adebayo, et al. Textual inference with tree-structured LSTM. In: BNAIC 2016: Artificial Intelligence: 28th Benelux Conference on Artificial Intelligence, Amsterdam, The Netherlands, November 10-11, 2016, Revised Selected Papers 28. Springer International Publishing, 2017. p. 17-31.

23. An, Guozhen; Shafiee, Mehrnoosh; Shamsi, Davood. Improving retrieval modeling using cross convolution networks and multi frequency word embedding. arXiv preprint arXiv:1802.05373, 2018.

24. Adouane, Wafia; Bernardy, Jean-Philippe; Dobnik, Simon. Neural models for detecting binary semantic textual similarity for algerian and msa. In: Proceedings of the Fourth Arabic Natural Language Processing Workshop. 2019. p. 78-87.

25. Ahmadi, Sina. Monolingual alignment of word senses and definitions in lexicographical resources. arXiv preprint arXiv:2209.02465, 2022.

26. Adebayo, Kolawole John, et al. An approach to information retrieval and question answering in the legal domain. 2016.

27. BOUSSAHA, Basma El Amel, et al. Multi-level context response matching in retrieval-based dialog systems. In: Proceedings of the 7th edition of the Dialog System Technology Challenges Workshop at AAAI. 2019.

28. Battula, Venkataramana; Vuyyala, Sathish; Ambati, Saritha. Retrieval of Answers from FAQ's using Word Embedding Techniques. In: 2023 10th International Conference on Computing for Sustainable Global Development (INDIACom). IEEE, 2023. p. 1321-1324.

29. Chaudhuri, Debanjan. Enriching Text-Based Human-Machine Interactions with Additional World Knowledge. 2022. PhD Thesis. Universitäts-und Landesbibliothek Bonn.

30. Opitz, Juri. Metrics of Graph-Based Meaning Representations with Applications from Parsing Evaluation to Explainable NLG Evaluation and Semantic Search. 2024. PhD Thesis.

31. John, Adebayo Kolawole; Di Caro, Luigi; Boella, Guido. Siamese network with soft attention for semantic text understanding. In: Proceedings of the 13th international conference on semantic systems. 2017. p. 160-167.

32. Boussaha, Basma El Amel, et al. End-to-end response selection based on multi-level context response matching. Computer Speech & Language, 2020, 63: 101080.

33. Boussaha, Basma El Amel, et al. Towards simple but efficient next utterance ranking. Computación y Sistemas, 2019, 23.3: 1055-1064.

34. Zhu, Jinghua; Yuan, Dandan; Xi, Heran. Context-Aware Network for Multi-Turn Response Selection in Retrieval-Based Chatbots. In: Proceedings of the 2021 International Conference on Human-Machine Interaction. 2021. p. 23-28.
35. Morales, Alvaro, et al. Learning to answer questions from semi-structured knowledge sources. 2016. PhD Thesis. Massachusetts Institute of Technology.
36. Sileo, Damien. Semantic and Discursive Representation for Natural Language Understanding. 2019. PhD Thesis. Université Paul Sabatier-Toulouse III.
37. Adebayo, Kolawole, et al. Textual Inference with Deep Learning Technique. In: Proceedings of the 28th Annual Benelux Conference on Artificial Intelligence (BNAIC2016). 2016.
38. John, A., et al. An approach to information retrieval and question answering in the legal domain. In: Proceedings of 10th International Workshop on Jurisinformatics. 2018. p. 1-14.

[O.2] **Pichl Jan**; *Open-Domain Dialogue Systems.* Doctoral Study Report, 2018.

Jan Pichl (15%): Conceptualization, Experiments, Writing.

[O.3] Ta Van Duy, **Pichl Jan**, Marek Petr, Konrád Jakub, Lorenc Petr, Šedivý Jan; *Leveraging Pre-trained Multilingual Models for Context-Aware Generation of Inflectional Word-Forms.* 2022.

Van Duy Ta (40%): Conceptualization, Implementation, Experiments, Writing.

Jan Pichl (40%): Conceptualization, Experiments, Writing.

Petr Marek (5%): Conceptualization, Experiments, Writing.

Jakub Konrád (5%): Consultations, Review of writing.

Petr Lorenc (5%): Consultations, Review of writing.

Jan Šedivý (5%): Supervision, Review of writing.

[O.4] **Pichl Jan**, Marek Petr, Konrád Jakub, Ondřej Kobza, Šedivý Jan; *Segment-Level Contextual Conversational Dialogue Act Dataset.* 2023.

Jan Pichl (80%): Conceptualization, Experiments, Writing.

Petr Marek (5%): Conceptualization, Experiments, Writing.

Jakub Konrád (5%): Consultations, Review of writing.

Ondřej Kobza (5%): Consultations, Review of writing.

Jan Šedivý (5%): Supervision, Review of writing.

Awards Relevant to the Thesis

[A.1] **Pichl Jan**, Konrád Jakub, Nguyen Hoang Long, Matulík Martin, Marek Petr, Šedivý Jan. Alexa Prize Socialbot Grand Challenge 1, **Second** Place Winners
`https://www.amazon.science/alexa-prize/socialbot-grand-challenge/2017`
Seattle, Washington, USA, 2017.

[A.2] **Pichl Jan**. Forbes 30 under 30
`https://30pod30-2018.forbes.cz/30.php?d=jan-pichl`
Pregue, Czech Republic, 2018.

[A.3] **Pichl Jan**, Konrád Jakub, Matulík Martin, Marek Petr, Lorenc Petr, Šedivý Jan. Alexa Prize Socialbot Grand Challenge 2, **Second** Place Winners
`https://www.amazon.science/alexa-prize/socialbot-grand-challenge/2018`
Seattle, Washington, USA, 2018.

[A.4] **Pichl Jan**, Konrád Jakub, Matulík Martin, Marek Petr, Lorenc Petr, Šedivý Jan. Project of the Year, AI Awards
`https://www.aiawards.cz/`
Prague, Czech Republic, 2018.

[A.5] **Pichl Jan**, Konrád Jakub, Marek Petr, Lorenc Petr, Duy Ta Van, Šedivý Jan. Alexa Prize Socialbot Grand Challenge 3, **Third** Place Winners
`https://www.amazon.science/alexa-prize/socialbot-grand-challenge/2019`
Seattle, Washington, USA, 2020.

[A.6] Konrád Jakub, **Pichl Jan**, Marek Petr, Lorenc Petr, Duy Ta Van, Kobza Ondřej Šedivý Jan. Alexa Prize Grand Challenge 4, **First** Place Winners
`https://www.amazon.science/alexa-prize/socialbot-grand-challenge/2020`
Seattle, Washington, USA, 2021.

Other Conferences, Seminars, and Research Popularization Relevant to the Thesis

Conferences

[P.1] **Pichl Jan**. Machine Learning Prague (ML Prague)
https://2018.mlprague.com/
Prague, Czech Republic, 2018.

[P.2] **Pichl Jan**. The Next Big Thing, Forbes
https://forbes.cz/akce/forbes-next-big-thing2018/
Prague, Czech Republic, 2018.

[P.3] **Pichl Jan**. Future Port Prague
https://2019.futureportprague.com/involvement/festival-speaker-2018/page/2/
Prague, Czech Republic, 2018.

[P.4] **Pichl Jan**. Forbes 30 under 30 summit
https://forbes.cz/akce/forbes-summit-30-pod-30-cz-sk-2019/
Bratislava, Slovakia, 2019.

[P.5] **Pichl Jan**, Marek Petr. Amper Brno, Czech Republic, 2019.

[P.6] **Pichl Jan**. Kognice a umělý život
https://kuz2022.ciirc.cvut.cz/
Třešť, Czech Republic, 2022.

Seminars

[P.7] **Pichl Jan**, Konrád Jakub, Nguyen Hoang Long, Matulík Martin, Marek Petr, Šedivý Jan. Machine Learning Meetups (MLMU).
https://www.meetup.com/Prague-Machine-Learning/events/243981269/
Prague, Czech Republic, 2017.

[P.8] **Pichl Jan**, Šedivý Jan, Marek Petr, Konrád Jakub, Lorenc Petr, Matulík Martin. ČVUT CIIRC meetup Prague, Czech Republic, 2018.

[P.9] **Pichl Jan**. Czechbots
https://www.youtube.com/watch?v=FihTE4AcQPU&ab_channel=Czechbots
Prague, Czech Republic, 2019.

[P.10] **Pichl Jan**, Marek Petr. Linguistic mondays, UFAL MFF Charles University.
https://ufal.mff.cuni.cz/events/alquist-how-develop-engaging-conversation
Prague, Czech Republic, 2019.

[P.11] **Pichl Jan**. Robots Prague: Conversational UI & AI Assistants https://www.meetup.com/CzechRobots/events/257861963/ Prague, Czech Republic, 2019.

[P.12] **Pichl Jan**. Summer School on Voice Interaction and Voice Assistants in Health Care. Technische Universität Dresden, Dresden, Germany, 2019.

[P.13] **Pichl Jan**, Vodrážka Tomáš. Machine Learning Meetups (MLMU)
https://www.meetup.com/Prague-Machine-Learning/events/282546297/
Prague, Czech Republic, 2021.

[P.14] **Pichl Jan**. *ChatGPT and the Future of Conversational Technologies: Advancements, Implications, and Challenges*, IDA Seminar
https://cs.fel.cvut.cz/en/news/detail/1715
Prague, Czech Republic, 2023.

Research Popularization

[P.15] **Pichl Jan**, Šedivý Jan. Studio 6, Czech Television
https://www.ceskatelevize.cz/porady/1096902795-studio-6/217411010101205/
Prague, Czech Republic, 2017.

[P.16] **Pichl Jan**, Konrád Jakub, Matulík Martin, Marek Petr, Lorenc Petr. Voice assistants for Kids, Planeta Yo, CT :D, Czech Television.
https://www.ceskatelevize.cz/porady/10315711050-planeta-yo/219553117450001/
Prague, Czech Republic, 2019.

[P.17] **Pichl Jan**. Inspiration Forum, MFDF 22
https://www.ji-hlava.cz/inspiracni-forum/inspiracni-forum-2018 Jihlava, Czech Republic, 2018.

[P.18] **Pichl Jan**, Chatbot as a Friend, Studenta Magazine
https://www.studenta.cz/life/chatbot-muze-byt-nas-novy-kamarad-s-honzou-pichlem-o-vyvoji/r~002393a06d8311e89f2fac1f6b220ee8/
2018.

[P.19] **Pichl Jan**, Konrád Jakub, Chat with AI, Science Festival Göteborg
https://stockholm.czechcentres.cz/sv/program/ceske-dny-v-goteborgu
Gothenburg, Sweden, 2019.

[P.20] **Pichl Jan**, Conversational AI, Maker Fair
https://stockholm.czechcentres.cz/sv/program/ceske-dny-v-goteborgu
Brno, Czech Republic, 2019.

[P.21] **Pichl Jan**, Konrád Jakub. Studio 6, Czech Television
https://www.ceskatelevize.cz/porady/1096902795-studio-6/221411010100728/
Prague, Czech Republic, 2021.

[P.22] **Pichl Jan**. Pražská technika
https://media.cvut.cz/cs/publikace/20211208-prazska-technika-6-2021#page/24
Prague, Czech Republic, 2021.

Podcasts

[P.23] **Pichl Jan**. Forbes 30 under 30 podcast Prague, Czech Republic
https://forbes.cz/jan-pichl-30-pod-30-podcast/
2018.

[P.24] **Pichl Jan**, Šedivý Jan. Alexa Prize Socialbot Grand Challenge 3 Summary, Czech Radio
https://radiozurnal.rozhlas.cz/jeste-nedavno-byli-outsidery-dnes-patri-vyvojari-z-cvut-mezi-svetovou-spicku-v-7695126
Prague, Czech Republic, 2018.

[P.25] **Pichl Jan**. Alexa Prize Socialbot Grand Challenge 3 Summary, Czech Radio
https://plus.rozhlas.cz/cesky-program-alquist-pro-chytre-reproduktory-se-uz-potreti-ucastni-celosvetove-8098946
Prague, Czech Republic, 2019.

[P.26] **Pichl Jan**, Marek Petr. Podcast Future R, Czech Radio
https://radiozurnal.rozhlas.cz/kamarad-ktery-dokaze-dobre-poradit-popisuji-

vyvojari-svoji-predstavu-budoucnosti-8271828
Prague, Czech Republic, 2020.

[P.27] **Pichl Jan**. Digibiz Podcast about Voice Assistants Prague, Czech Republic, 2020.