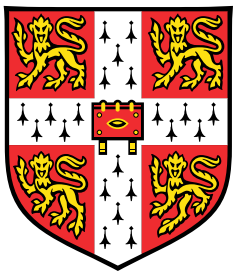
Natural Language Understanding and Generation for Task-Oriented Dialogue



Bo-Hsiang Tseng

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
Doctor of Philosophy

Clare Hall

September 2021

I would like to dedicate this thesis to my loving parents and wife ...

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains roughly 48,799 words including appendices, bibliography, footnotes and equations and has 57 figures and tables. Some of the material included in this thesis has been published in the Special Interest Group on Discourse and Dialogue (Tseng et al., 2019a), the Empirical Methods on Natural Language Processing (Tseng et al., 2019b), the Association for Computational Linguistics (Tseng et al., 2020, 2021b), and the North American Chapter of the Association for Computational Linguistics (Tseng et al., 2021a).

Bo-Hsiang Tseng
September 2021

Abstract

Natural Language Understanding and Generation for Task-Oriented Dialogue

Bo-Hsiang Tseng

The success of deep learning methods has stimulated the rapid development of many NLP research areas. Still, task-oriented dialogue modelling remains challenging due to both the inherent complexity of human language and task difficulty. Moreover, building such systems usually relies on large amounts of data with fine-grained annotations, and in many situations, it is difficult to obtain such data. It is thus important for dialogue systems to learn efficiently in low-resource scenarios so that the models can still effectively fulfill their tasks. This thesis aims to provide novel methods to tackle these difficulties in dialogue modelling.

To communicate, most commonly, a dialogue system converts a semantic representation (e.g., a dialogue act) into natural language in a process known as Natural Language Generation (NLG). A tree-based NLG model is proposed and shown to be more easily adapted to unseen domains in comparison to other models. This desirable property arises due to the fact that modelling semantic structure facilitates knowledge sharing between source and target domains. We also show that the NLG task can be jointly learned with its dual task, the natural language understanding (NLU), which maps natural language utterances to semantic counterparts. Our approach consists of a stochastic generative model with a shared latent variable for two tasks, which can be trained with significantly less data than individual components.

The focus then shifts to a more general setup of dialogue generation. In end-to-end dialogue modelling, systems consume user utterances and learn to directly generate responses, where intermediate dialogue acts are usually used as auxiliary learning signals for model optimisation. We show that semi-supervised methods that were proposed for computer vision tasks can be beneficial to dialogue modelling. We also address the problem of developing dialogue systems when little training data is available. To this end, we propose a learning framework where user and dialogue models are jointly optimised. We show that the data generated by their interaction can be used to further optimise the two models and leads to improved model performance. Yet again, this approach reduces the amount of data for end-to-end dialogue modelling on low-resource domains.

Lastly, an understanding model is proposed to address the prevalent phenomena of coreference and ellipsis in dialogues. This model first performs coreference resolution and then rewrites the input user utterance into a complete sentence that resolves coreferent entities

and omitted information. As a side contribution, the acquired data for model training is released to the research community.

Acknowledgements

First and foremost, I would like to acknowledge my supervisor, Bill Byrne. I realise that words cannot express my sincere gratitude to you, but I truly appreciate your dedicated supervision for the past three years. Under your guidance, I enjoyed freedom to explore topics that interest me; your open-mindedness towards and support for my research ideas have been extremely valuable. I will always think fondly of our insightful and constructive discussions, and I am also grateful for your patience and advice on my questions beyond research. Were it not for your support and guidance, I would not have been able to finish my PhD study. Thank you, Bill. From the bottom of my heart, it has been my pleasure to be your student.

I am thankful to Milica Gašić for accepting my candidacy to the Dialogue Systems Group and the supervision on my first year PhD. I also would like to acknowledge Steve Young for founding the group and his feedback on some of my work. Thank you both for believing in my potential and offering me the opportunity to start my research journey at Cambridge.

I was privileged to work with many talented people in the Dialogue Systems Group. Paweł, you are like a big bother to me. I will always treasure the time we worked and lived together. Yen-Chen, I love those great research ideas and daily life stories you shared with me. Thank you both for being such a good housemate and research collaborator. Florian, thank you for your great effort put in our work. We really should drink more coffee together at Hot Numbers. Iñigo, I enjoy the collaboration and jokes we said during work. I also would like to thank Stefan and Lina for offering help whenever I had difficulties at the lab. In addition, I also thank Rachel for your amazing administrative support.

Special thanks to Yinpei, who spent hours and hours on brainstorming and discussion with me. After several revisions and improvements, we finally managed to publish our fine work. Another special thanks goes to Alex, my research buddy and writing mentor. I admire your enthusiasm of sharing and ambition of doing great work. I truly wish we could meet earlier during our PhDs.

During my two internships at Apple, I have worked with many brilliant people. I would like to thank David, Jianpeng and Yimai for wonderful discussions and brainstorming sessions. I also want to thank Joel, Dhivya, Jiarui and Shruti for your kind and selfless help

to our work. I especially thank Jason, Lin and Hong for offering me the opportunity to learn from and grow with the team.

I sincerely thank my two viva examiners, Prof. Andreas Vlachos and Prof. Ondřej Dušek. They gave very constructive comments and feedback that are tremendously helpful to improve the thesis. Thank you so much.

Outside of work, I met wonderful people during my life at Cambridge. I thank all of my friends at the Taiwanese Society, Clare Hall and St Catharine's College for the joy you brought to my life. I also express my sincere gratitude towards the Cambridge Trust and the Ministry of Education in Taiwan for funding me throughout my PhD study.

Finally, I want to say thank you to my family for your endless love and encouragement. For my parents, Ying-Chen and Kuang-Hua. Were it not for your upbringing and support, I would not have been able to reach to this point and go this far. For my wife, Chenyi, meeting you has been without doubt the best thing that could have ever happened to me along this journey.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Overview

John McCarthy, one of the founders of the discipline of artificial intelligence, stated in the Dartmouth proposal (McCarthy et al., 1955):

> *"Every aspect of learning or any other feature of intelligence can be so precisely described that a machine can be made to simulate it."*

Indeed, having a machine equipped with human-level intelligence and knowledge that can discourse about arbitrary topics has been a fascinating, long-term quest. In 1950, Alan Turing had already discussed a profound question: "Can machines think?" and proposed the Imitation Game (Turing and Haugeland, 1950), known as the Turing test nowadays, according to which a computer would deserve to be called intelligent if it could deceive a human into believing that it was human.

An early and influential dialogue system was ELIZA (Weizenbaum, 1966), which was designed to simulate a therapist and could chat with patients using natural language. Although the system was simple, built using heuristics such as pattern matching and word substitutions/re-ordering, it was strikingly capable of delivering engaging conversations with users and made them believe they were chatting with a real person. The very first system that passed the Turning test was PARRY (Colby et al., 1971), created for simulating a patient with schizophrenia (Colby et al., 1972). Although these systems seemed anthropomorphic and convincing, they were not really intelligent and not equipped with the ability to learn and evolve.

With the development of dialogue systems, there are currently two mainstream research disciplines in the field: *Chat-oriented* and *Task-oriented* dialogue systems. The former category, also known as chit-chat bots, aims at conversing with users in an open-domain

setup, with no well-defined tasks or domains in its scope (Li et al., 2016; Ritter et al., 2011; Serban et al., 2016; Sordoni et al., 2015). In contrast, *task-oriented* dialogue systems are programmed to facilitate solving various tasks (Levin et al., 2000; Young et al., 2010, 2013), such as flight reservations (Seneff and Polifroni, 2000) and tourist guidance (Misu and Kawahara, 2007). These systems are designed based on pre-defined problems and complete the dialogues through interacting with the users who have a set of constraints related to the task in their minds (e.g., in the tourist guidance domain, a user is looking for an attraction place in *college type* with *no entrance fee*).

Nowadays, the need for task-oriented dialogue agents is largely growing in industry. More and more people rely on personal assistants, such as Apple's Siri, Amazon's Alexa and Google Assistant, to solve daily-life problems (e.g., asking about weather, alarm setup). Another trend is the application of conversational AI to customer services that are usually centralised around problem solving or task completion within limited domains (e.g., restaurant reservation, payment/transaction in banking domain). Compared to human resources, the promise of intelligent agents is that of lower cost and scalability.

Although task-oriented dialogue systems have a long history of development and have been attracting much research attention in recent years, building such systems is still challenging. There are two main research questions explored in this thesis. First, the complex nature of human language (e.g., language variations, phenomena of coreference and ellipsis) poses problems for the machine to understand and correctly decode the user intent. Hypothetically, even with the perfect understanding results, the complex dialogue flow makes it difficult to generate coherent and engaging responses. For example, the user might want to complete multiple tasks (such as modifying banking details and conducting a transaction) within one dialogue session.

Second, the prevalent, data-driven deep learning methods require great amounts of data to appropriately learn a task, but collecting dialogue data at large scale is problematic due to the need to provide specialised annotations for the data. Specifically, modern task-oriented dialogue agents operate on semantic representations (i.e., *dialogue acts* (Traum, 1999)) and the corresponding labels could be noisy and error-prone if not annotated by dialogue experts (Eric et al., 2020; Zhou and Small, 2019). A lack of data not only makes dialogue modelling more challenging, but also limits a system's scalability across various domains. It thus becomes important to design algorithms that can learn how to efficiently complete tasks from a limited amount of data.

The aim of this thesis is to investigate and tackle the aforementioned issues. Concretely, we explore the data sparsity problems in natural language understanding and generation in task-oriented dialogue and propose novel data-efficient algorithms to address these issues. To

verify our claims and gain an insight into how close are dialogue models to being deployed in the real world, we test the proposed models on the latest benchmark corpora in low-resource and domain adaptation settings.

1.2 Contributions and Outline

The dissertation is split into three main parts. After an overview of spoken dialogue systems (Chapter 2), studies on the task of Natural Language Generation (NLG) are presented in Chapters 3 and 4, in which system responses are generated given dialogue acts as input. Chapters 5 and 6 focus on a more general dialogue generation setting, where the system consumes a user utterance and dialogue history as input and generates system responses. In this *end-to-end* framework, the ability of the system to understand the user intent greatly affects the generation quality. Lastly, we focus on the prevalent phenomena of coreference and ellipsis in dialogue understanding (Chapter 7). Below we briefly summarise the contributions presented in each chapter.

- **Chapter 2 - Overview of Spoken Dialogue Systems**
  This chapter presents fundamental concepts of the spoken dialogue system and the most important system architectures. We discuss the development of each component in a *pipeline based dialogue system* and provide an overview of key research directions in each of these areas. A relatively new research line on end-to-end modelling is reviewed as well.

- **Chapter 3 - Domain Adaptation through Knowledge Sharing for Dialogue Generation**
  Developing dialogue systems across various domains is important but difficult due to insufficient data in new domains. In this chapter, we present a tree-based NLG model for knowledge sharing in domain adaptation. A tree-based encoder captures the structure of input dialogue acts, which facilitates information sharing between source and target domains, leading to better generalisation in NLG. This research work was presented in the publication (Tseng et al., 2019a):

  – **Tseng, B. H.**, Budzianowski, P., Wu, Y. C., and Gasic, M, "Tree-Structured Semantic Encoder with Knowledge Sharing for Domain Adaptation in Natural Language Generation". *In Proc of SIGDIAL 2019*.

- **Chapter 4 - Joint Modelling of Natural Language Understanding and Generation**
  Natural language understanding and generation are two important and correlated tasks

in dialogue modelling. The former translates natural language into dialogue acts, while the latter performs the reverse operation. However, they are usually studied separately. In this chapter, we propose a single generative model that performs joint understanding and generation by leveraging the inverse property between the two tasks. More importantly, our model can be trained in a semi-supervised manner by making the use of unlabelled data, resulting in a significant reduction in annotated data requirements for training. This study was presented in the publication (Tseng et al., 2020):

  - **Tseng, B. H.**, Cheng, J., Fang, Y., & Vandyke, D, "A Generative Model for Joint Natural Language Understanding and Generation". *In proc of ACL 2020*.

- **Chapter 5 - Semi-supervised Bootstrapping of Dialogue State Trackers**
  The annotations required for training dialogue state trackers are complicated and difficult to collect at a large scale. This chapter investigates two well-known semi-supervised learning methods that were initially proposed for the task of image recognition, and applies them on dialogue state tracking in an end-to-end framework, where all the dialogue components are jointly optimised. The results suggest that our model could use only 50% labelled data to reach the comparable performance with models trained on the complete dataset. This study was presented in the publication (Tseng et al., 2019b):

  - **Tseng, B. H.**, Rei, M., Budzianowski, P., Turner, R., Byrne, B., & Korhonen, A. "Semi-Supervised Bootstrapping of Dialogue State Trackers for Task-Oriented Modelling". *In proc of EMNLP 2019*.

- **Chapter 6 - Transferable Dialogue Systems and User Simulators**
  This chapter tackles the problem of data sparsity in training end-to-end dialogue systems and their adaptation to new dialogue scenarios (such as unseen domains or new dialogue flows). We propose a learning framework that jointly optimises a dialogue agent and a user simulator. When adapting dialogue systems to new domains, the corresponding user models can be transferred together in this framework, and the additional data can be generated by allowing the user and system agents to interact freely. Both systems are then optimised on the simulated data using reinforcement learning. This research work was presented in the publication (Tseng et al., 2021b):

  - **Tseng, B. H.**, Dai, Y., Kreyssig, F., & Byrne, B. "Transferable Dialogue Systems and User Simulators". *In proc of ACL 2021*.

- **Chapter 7 - Combined Resolution of Ellipses and Anaphora in Dialogues**
  In this chapter, we focus on the phenomena of coreference and ellipsis in dialogues. A transformer based model is proposed to solve the problem. The model is optimised against two tasks: it first performs coreference resolution on the dialogue context, and then rewrites the user utterance into a complete sentence that provides more comprehensive information for dialogue understanding. To train and evaluate the proposed model, as a side contribution, we have annotated a dialogue dataset with rewritten utterances and released the acquired data for the research community. This research work was presented in the publication (Tseng et al., 2021a):

  – **Tseng, B. H.**, Bhargava, S., Lu, J., Moniz, J. R. A., Piraviperumal, D., Li, L., & Yu, H. "CREAD: Combined Resolution of Ellipses and Anaphora in Dialogues". *In Proc of NAACL 2021*.

- **Chapter 8 - Conclusions**
  The contributions of the proposed methods are summarised and future research directions are discussed in this final chapter.

Chapter 2

Overview of Spoken Dialogue Systems

This chapter introduces fundamental concepts of spoken dialogue systems for task-oriented dialogues. Spoken dialogues considered throughout this dissertation are a series of exchanges between a user and a dialogue agent. At the beginning of a dialogue, the user has a *user goal* in mind, containing constraints pertaining to an entity they wish to search or find more information about (e.g., a suitable restaurant and its address). As the dialogue proceeds, the dialogue agent collects the constraints provided by the user and executes the corresponding actions to complete the task successfully. The user has always at least one entity to search for in a dialogue. Below we discuss the key concepts of task-oriented dialogue systems and their architecture, and we encourage the reader to refer to more thorough reviews provided in the chapter 24 of Jurafsky and Martin (2019), Gao et al. (2018) and McTear (2020).

Frames The frame is a key component of modern dialogue systems (Bobrow et al., 1977). A frame is a knowledge representation containing a set of `intents` along with a collection of `slots`. Intents define the user intention that the dialogue agent is dealing with (e.g., finding a restaurant or booking a movie ticket). A set of slots defines the attributes of the entity that the user is looking for, and each slot can take a set of values. In the restaurant domain, for example, the slot *food type* can take either *British* or *American*. There are two slot categories: informable and requestable. The former are the attributes specified by the user (e.g., the slot *food type* in the utterance *"I want British food."*), while the latter are the information the user desires to know (e.g., the slot *address* found in response to the query *"What is the address of the restaurant?"*). A frame is usually devised by dialogue developers, and is created for a particular domain or service. A combination of frames constitutes the `ontology`. A frame example in the schema-guided dialogue (SGD) dataset (Rastogi et al., 2020) is shown in Figure 2.1.

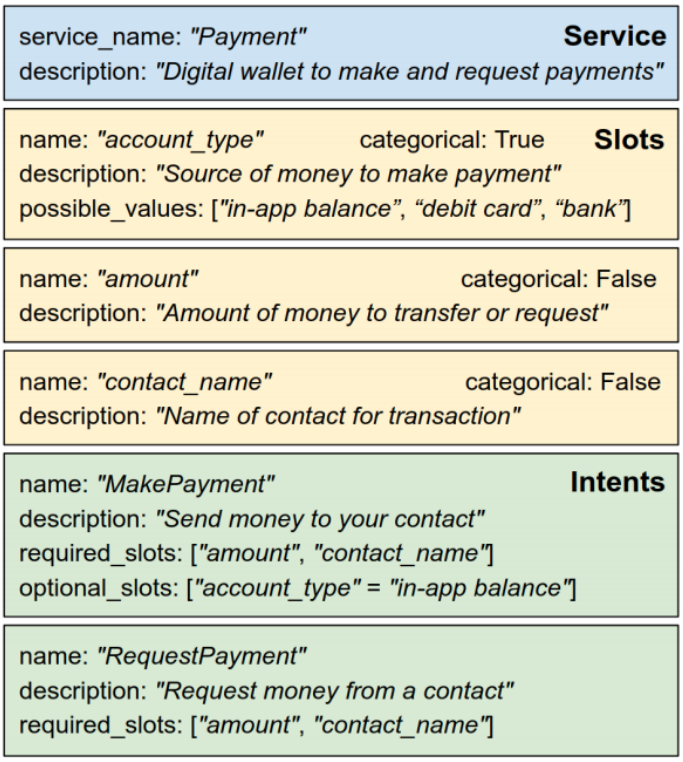


Fig. 2.1 Example dialogue frame for the digital wallet service in the SGD dataset (Rastogi et al., 2020).

Dialogue Act	**Description**
INFORM	Inform the user of the value for a slot.
REQUEST	Request the value of a slot from the user.
CONFIRM	Confirm the value of a slot before making a transactional service call.
OFFER	Offer a certain value for a slot to the user.
NOTIFY_SUCCESS	Inform the user that their request was successful.
NOTIFY_FAILURE	Inform the user that their request failed.
INFORM_COUNT	Inform the number of items found that satisfy the user's request.
OFFER_INTENT	Offer a new intent to the user.
REQ_MORE	Asking the user if they need anything else.
GOODBYE	End the dialogue.

Table 2.1 Dialogue act and the corresponding description in the SGD dataset.

Dialogue Acts Another important concept in task-oriented dialogues modelling is the `Dialogue Act` (Core and Allen, 1997; Traum, 1999). The dialogue act represents the meaning of an utterance and capture the speaker's intent in a conversation (Austin, 1975; Young, 2007). The identification of dialogue acts helps analysing the discourse structure of dialogue (Stolcke et al., 2000). In literature, different schemes of dialogue acts have been proposed and studied (Bunt, 2009; Bunt et al., 2012; Jurafsky et al., 1997). Throughout this thesis, a set of dialogue acts defines the possible communicative actions that the user and the system can execute (e.g., *inform*, *request* and *confirm*). A dialogue act can take slots and values as arguments and together they form a semantic representation of natural language. The designs of dialogue acts depend on the target task, and can vary in different dialogue systems (Paul et al., 2019). As an example, the dialogue acts defined in the SGD dataset is shown in Table 2.1. Note that the user intents are modelled separately from dialogue acts. For example, in the SGD corpus, the intent is treated as a slot type and a set of corresponding values describes the possible user intents in the corpus.

System Architecture The most notable design of spoken dialogue systems is the pipelined architecture (Pieraccini and Huerta, 2005; Young et al., 2010) comprising six components: automatic speech recognition (ASR), natural language understanding (NLU), dialogue state tracking (DST), dialogue manager (DM), natural language generation (NLG), and text-to-speech (TTS) synthesis, along with a predefined ontology and knowledge base (database). The overview of the architecture is presented in Figure 2.2. Given a user utterance audio input, the ASR module transcribes the audio signal into text. The NLU module then detects the domain, intent and slot values in the user utterance, and DST incorporates the user's

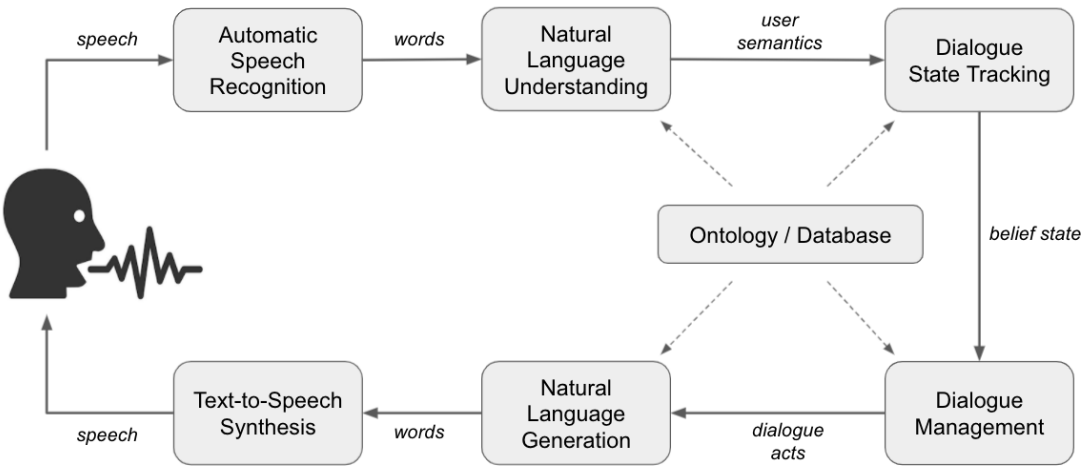


Fig. 2.2 Spoken dialogue system overview.

constraints up to the current turn. The DM determines the system actions with corresponding arguments (i.e., slot-value pairs). Finally, the NLG module generates the the system response in text format, which is then converted into speech by the TTS model. Note that there are also text-based dialogue systems, which communicate with the user via text. These systems are not equipped with ASR and TTS models.

The rest of the chapter is structured as follows. First, each component in the system architecture is discussed through Section 2.1 to Section 2.6. In addition, joint learning of system components, which has shown advantages over the conventional pipelined system described above, is discussed in Section 2.7.

2.1 Automatic Speech Recognition

The automatic speech recognition (ASR) module, along with the text-to-speech (TTS) module (Section 2.6), form the speech interface for the user to communicate with the dialogue agent. The ASR module transcribes the input audio signal into its textual form and the TTS operates in reverse.

Early ASR systems relied on extracting features of speech characteristics (such as fast Fourier Transform (FFT) (Cooley and Tukey, 1965)) and made use of dynamic programming to match the stored templates against the incoming speech (Itakura, 1975). Later on, statistical models such as the hidden Markov model (HMM) were applied to ASR (Baker, 1975; Jelinek et al., 1975) and gradually became prevailing in the field (Bahl et al., 1986; Levinson, 1986). With the rise of neural networks, hybrid systems of HMMs and neural networks were proposed (Morgan and Bourlard, 1995; Robinson and Fallside, 1991; Waibel et al., 1989). The more recent encoder-decoder architecture (Chan et al., 2016; Chorowski et al., 2014) and the CTC-based models (Graves et al., 2006) have led to a great improvement in the ASR accuracy and robustness.

As the first step in the pipelined architecture, the quality of transcription makes a marked difference to the performance of the dialogue system. Although the state-of-the-art techniques achieve high performance, many scenarios (e.g., noisy environments, long distances between speakers and microphones) remain challenging. To reduce error propagation to downstream modules, instead of using the top-1 result, a good practice is to use either the *n-best list* of the sentence hypotheses where each candidate has a likelihood score (Williams et al., 2014), or a *word lattice* which subsumes all possible paths with transition probabilities (Purver et al., 2011). We note that in this thesis, an accurate ASR transcription is assumed, so issues related to ASR or TTS are not considered. This is because recent challenging dialogue corpora are

mainly text-based, and text-based dialogue systems trained on these datasets can be further combined with the off-the-shelf ASR and TTS modules to provide a speech interface.

2.2 Natural Language Understanding

The natural language understanding (NLU) model extracts semantics from the user utterance transcription. The semantic extraction usually involves three tasks: domain detection, intent prediction and slot filling. For example, the user utterance *"Find a train from Cambridge to London Liverpool Street this Saturday"* has an intent `find_train` in `train` domain, with the slot-value pairs {`departure=Cambridge;destination=London Liverpool Street;Day=Saturday`}. The decoded NLU result is then consumed by the downstream modules in dialogue systems.

Domain and intent prediction Domain and intent prediction can both be framed as a classification problem (Schapire and Singer, 2000; Yaman et al., 2008), where the ontology defines the possible output classes. Deep learning based methods have achieved good performance in classification tasks (Deng et al., 2012; Tur et al., 2012). Models such as recurrent neural networks (RNNs) (Ravuri and Stolcke, 2015, 2016) and convolutional neural networks (CNNs) (Hashemi et al., 2016; Lee and Dernoncourt, 2016) have been explored to obtain textual representations that consider the sentence context and produce good performance in the two tasks.

Slot filling Slot filling requires predictions of slot-value pairs at the word level and is usually modelled as a sequence tagging problem (Mesnil et al., 2013; Wang et al., 2005) using the Inside-Outside-Beginning (IOB) annotation style (Ramshaw and Marcus, 1999). The ATIS (Airline Travel Information System) (Price, 1990) is the widely-used benchmark for evaluating NLU systems. Early approaches to the problem include rule-based semantic grammars (Ward, 1994; Zue et al., 2000), but the need to handcraft rules by dialogue experts made the system difficult to extend to new domains. Conventional NLP methods such as context-free-grammars (CFG) (Charniak, 1997; Kwiatkowski et al., 2011; Zettlemoyer and Collins, 2012) and dependency trees (Chen and Manning, 2014; Nivre et al., 2007) were explored to deal with more rich and diverse natural language. With the development of statistical modelling, discriminative models such as conditional random fields (CRF) (Jeong and Lee, 2008; Lafferty, 2001) and support vector machines (SVM) (Kate and Mooney, 2006; Mairesse et al., 2009) were applied to the task and were later superseded by the more recent deep learning methods. Among these, RNNs (Hakkani-Tür et al., 2016; Mesnil et al.,

2014; Yao et al., 2013) perfectly fit the nature of the task to predict the semantic label for each word. They also provide the flexibility to consider the dialogue context to augment the understanding ability and reduce the semantic ambiguity (Chen et al., 2016; Hori et al., 2015). Joint learning of intent detection and slot filling using RNNs has shown benefits for the model performance (Hakkani-Tür et al., 2016; Liu and Lane, 2016). More recently, the powerful pre-trained transformer based models such as BERT (Devlin et al., 2019) achieved state-of-the-art performance and become a paradigm nowadays (Chen et al., 2019; Wu et al., 2020; Zhu et al., 2020).

To engage in an appropriate and coherent dialogue, it is imperative for the dialogue agent to not only understand the meaning of the current turn, but also to maintain the historical information in the dialogue context. Turn-level NLU predictions will be tracked across turns to collect context-aware information at dialogue level by the dialogue state tracking (DST) module, which is discussed in the next section.

2.3 Dialogue State Tracking

As aforementioned, the DST module collects information across dialogue turns, which allows dialogue systems to have a comprehensive understanding of the conversation so far. Slot-value pairs detected up to the current turn together with the detection uncertainty forms the *dialogue state* (or *belief state*). For each informable slot, a distribution over possible values is predicted, and the aggregation of these distributions form the belief state. As seen in the example in Table 2.2, the user changes the date constraint to *Sunday* due to the unavailability on *Saturday*. The system should be aware of the goal change while tracking the confirmed attributes in previous turns.

Dialogue excerpt	**DST annotation**
user: *Find a train from Cambridge to London Liverpool Street this Saturday.* system: *I am sorry. There are no trains available on Sunday meeting your criteria.*	Departure=Cambridge Destination=London Liverpool Street Day=Saturday
user: *How about Sunday?* system: *There are several options. What is your departure time?*	Departure=Cambridge Destination=London Liverpool Street Day=Sunday

Table 2.2 Dialogue excerpt with the corresponding DST annotation.

Early approaches to DST relied on heuristics designed by dialogue developers (Larsson and Traum, 2000; Wang and Lemon, 2013). The focus then shifted to data-driven methods for a better generalisation ability in unseen scenarios. This included studies using deep belief networks (Thomson and Young, 2010; Williams and Young, 2007), and log-linear models (Lee and Eskenazi, 2013; Ren et al., 2013). More recently, the application of deep learning

models (Henderson et al., 2013) and the Dialogue State Tracking Challengings (DSTC2 & 3) (Henderson et al., 2014a,b) stimulated the development of DST modelling. Neural networks such as RNNs are used to take as input the hypotheses from the ASR and NLU modules to predict a belief state. Research has found that joint learning of the NLU and DST produces a more robust performance (Mrkšić et al., 2015; Sun et al., 2014; Zilka and Jurcicek, 2015). Recent studies on DST follow this research trend, and can be categorised based on the decoding mechanism employed as follows:

- **Classification** This model type learns classifiers to predict a distribution over values for each slot. A representative example is the neural belief tracker model (Mrkšić et al., 2017), which reduced the dependency on feature engineering used in prior work (Henderson et al., 2014c,d) and even achieved better performance. However, as the dialogue complexity (e.g., number of slots) increases, these models inevitably become larger and inefficient. Knowledge sharing between slots was then proposed to solve the problem (Ren et al., 2018; Zhong et al., 2018). One disadvantage of classification models is that they can only predict the slot value based on a pre-defined ontology, and is not able to handle slots with arbitrary value sets such as phone numbers. Generation and span prediction models, introduced below, are not limited by these constraints.

- **Generation** Wu et al. (2019a) proposed to use an encoder-decoder architecture with a copy mechanism (Gu et al., 2016) to generate the value for each slot. Powerful generative transformer based models (Vaswani et al., 2017) were adopted for DST as well. Hosseini-Asl et al. (2020) used the GPT-2 (Radford et al., 2019) to generate a linearised belief state (i.e., slots and values are treated as tokens and a belief state is represented as a sequence of tokens). Lin et al. (2020) proposed to model the belief state differences between turns using the T5 model (Raffel et al., 2020) or BART (Lewis et al., 2020).

- **Span Prediction** Non-categorical slots, such as postcodes, can have loosely defined value sets and the textual form of their values can vary significantly. Predictions of these slots are therefore not suitable to be modelled as a classification problem. Zhang et al. (2020) proposed to learn the word span predictions of values presented in the input utterance. Gao et al. (2019) formulated the task as reading comprehension. A query is prepared for each slot, e.g., *"what is the value of postcode?"*, and the model then finds an answer span in the dialogue context.

The performance of DST dramatically affects the overall performance of a dialogue system (Li et al., 2017a,b) as error recovery is difficult and often not implemented in systems.

It is thus essential to build a robust DST model. A well-known issue in the model training is the need for detailed annotation at each dialogue turn. We address this issue by adopting popular semi-supervised learning techniques to DST, and show the significant reduction on data usage while maintaining the model performance. More details are discussed in Chapter 5.

Another important perspective in DST modelling is coreference (Ng and Cardie, 2002; Soon et al., 2001) and ellipsis (McShane et al., 2005), which is prevalent in human language. Coreference occurs in a conversation where two or more expressions refer to the same entity. For example, in Table 2.3 the song in user turn 1 and the pronoun *"it"* in user turn 2 refers to the same entity. Ellipsis happens when the speaker omits some information to avoid repeats, e.g., the complete utterance of user turn 3 should be *"Who is the singer of the song?"*. Without resolving these references correctly, dialogue systems are not capable of understanding the user intent and might lead to sub-optimal performance.

Recently, there have been more studies dedicated to this line of research. Quan et al. (2019) augmented the CamRest676 dataset (Wen et al., 2017b) with the annotations of complete user utterances that resolve coreference and ellipsis. The coreference phenomenon is more evident in multi-domain dialogues in that entities are commonly shared between domains within a dialogue, e.g., a hotel name and a taxi departure place. Han et al. (2020) reported that 20% of the dialogues in the MultiWOZ dataset (Budzianowski et al., 2018) contain coreference, and they showed that DST performance can be boosted by coreference modelling. This shows the importance of the coreference resolution to DST. We address this issue by proposing a novel transformer-based model that jointly optimises coreference resolution and query rewriting. More details are provided in Chapter 7.

2.4 Dialogue Management and User Simulators

Following the DST module, the dialogue manager is used to determine the system action. The quality of the dialogue policy is key to user experience and successful task completion.

Dialogue excerpt
user-turn-1: *Give me the information of the song Bohemian Rhapsody.*
system-turn-1: *What kind of information you are looking for?*
user-turn-2: *When does it release?*
system-turn-2: *It was released at 31 October 1975.*
user-turn-3: *Who is the singer?*
system-turn-3: *The singer is Freddie Mercury.*

Table 2.3 Example dialogue with coreference and ellipsis in user utterances.

Early approaches relied on handcrafted engineering of dialogue flow by experts. Finite state automata were used to represent the dialogue flow (Abowd et al., 1995; Larsson and Traum, 2000; McTear, 1998). However, expert human knowledge was required to model new conversational flows, so these models were not easily scalable to new/multiple domains. Moreover, errors caused by upstream modules (e.g., ASR errors and belief state uncertainty) cannot be easily accounted for in this approach. Researchers subsequently resorted to statistical methods (Lemon and Pietquin, 2007; Levin et al., 2000; Singh et al., 1999; Young, 2002) to solve this problem. In this approach, the dialogue policy learning can be cast as a classification problem, where the model consumes the dialogue state as input and selects one dialogue act from a pre-defined action space. Given a large collection of pairs of dialogue states and actions, a machine, theoretically, should be able to learn the dialogue planning well. However, training corpora only contain a limited number of dialogue situations in practice. The resulting data sparsity issue affects policy exploration, leading to sub-optimal policy learning.

Reinforcement learning (RL) (Sutton et al., 2000) is a common paradigm in dialogue policy learning as it formulates the task as a long-term sequential decision making process. In an RL context, the model learns to predict a series of actions in order to maximise the reward obtained at the end of the dialogue. The reward is usually designed as the `dialogue success`, i.e., a measurement of task completion of a dialogue. A dialogue is successful when the dialogue system provides an entity that meets the user's constraints and answers the corresponding questions correctly. Other properties such as user satisfaction can also be used as a reward signal (Schmitt and Ultes, 2015). Learning dialogue policy through RL enables the model to explore various dialogue states and actions that might not be covered in a static corpus once collected. If the true dialogue state is known, dialogue policy optimisation could be cast as a Markov decision process (MDP) (Levin et al., 2000; Singh et al., 2002; Walker, 2000). In practice, the uncertainty in the belief state requires one to resort to techniques such as partially observable Markov decision process (POMDP) (Lemon et al., 2006; Williams and Young, 2007).

Using RL in dialogue policy learning, ideally, requires a user as an environment to interact with the system. However, it is time-consuming and costly to communicate with real users. The user simulator (Schatzmann et al., 2006) is thus introduced to substitute for real humans during system development. The quality of the simulated user plays a vital role in the system training. Unrealistic, unnatural human behaviours due to poor user simulators would train policies that are not capable of interacting with real users once deployed. Therefore, studying user simulation techniques carries equal weight to studying dialogue system policy

optimisation. We show how to improve the performance of a user simulator through joint learning with a dialogue agent in Chapter 6.

2.5 Natural Language Generation

To generate the system response, the natural language generation (NLG) module converts the dialogue act, which is produced by the dialogue management, into natural language. To illustrate, the dialogue act: `inform(food=British, choices=5, area=centre)` is translated into natural language *"There are 5 restaurants serving British food in the centre."*. The quality of the resulting utterances significantly influences users' experience.

The template-based method (Geldof and Van de Velde, 1997) was the first to handle the NLG problem. A set of heuristics was designed by experts to map dialogue actions to utterances. Although the produced responses are grammatically correct and perfectly convey the system's intention, the requirement of human effort in creating templates limits its scope. Therefore, research increasingly focused on statistical approaches that allow learning from data.

Early approaches to trainable generators divide the generation into two stages: 1) sentence planning and 2) surface realization (Stent et al., 2004; Walker et al., 2002). First, each slot-value pair in the dialogue act corresponds to an utterance. Pre-defined operations are then applied to incrementally combine those utterances into the final sentence, represented as a sentence plan tree. Multiple trees are generated and re-ranked using the RankBoost algorithm (Freund et al., 2003; Schapire, 1999). The top ranked sentence plan output is then input to the surface realiser which produces a surface linguistic utterance (Lavoie and Rainbow, 1997). Although this method is able to produce various sentences by considering linguistic structure, it still requires a significant degree of handcrafting same as the template method.

Neural network-based approaches to NLG are now popular, with outstanding performance and no heuristics. In Wen et al. (2015a), several candidate utterances were generated using RNNs and then re-ranked using CNNs. Wen et al. (2015b) improved the LSTM by introducing an additional gate to ensure that the output sentence conveys the desired semantics. Afterwards, Dušek and Jurčíček (2016) proposed an encoder-decoder model (Sutskever et al., 2014) to generate the output sentence[1] based on a linearised dialogue act. Re-ranking models are also adopted in these neural models to re-rank the output candidates by validating their semantic accuracy. However, a high quality of NLG model requires a large volume of dialogue act and sentence pairs, which is as labor-intensive as annotating dialogue

[1]Note that the delexicalised utterances (i.e., the slot values in an utterance are replaced with slot placeholders) are used in these prior work to reduce the complexity of natural language and training difficulty of RNNs.

belief states. More recently, transformers such as GPT-2 are used in few-shot NLG learning (Chen et al., 2020; Peng et al., 2020b) and demonstrate that the linguistic knowledge learned from the large amount of unannotated data is very useful. We discuss how to address the data sparsity issue through domain adaptation and unsupervised learning in Chapters 3 and 4. It is worth noting that this type of NLG modelling is now widely used beyond dialogue, with input as any structured data such as tables (Gardent et al., 2017; Lebret et al., 2016; Puduppully et al., 2019).

2.6 Text to Speech Synthesis

The TTS module, the last component in the dialogue system, converts the generated natural language in textual form into speech. Early approaches to TTS concatenated the pre-recorded audio fragments to form the complete utterance (Hunt and Black, 1996; Moulines and Charpentier, 1990). Although this method is able to create realistic speech to some extent, it requires a large collection of pre-recorded speech fragments. In addition, the discontinuity between fragments makes the generated speech sound unnatural.

Researchers then turned their attention to statistical modelling, where HMM based models demonstrated promising results (Taylor, 2009; Zen et al., 2009). Although not producing more natural sound, this type of model provides the flexibility to control the voice attributes (e.g., word emphasis, context sensitivity) in dialogue systems (Tsiakoulis et al., 2014; Yu et al., 2010, 2011). More recently, neural network based models (Ling et al., 2015; Oord et al., 2018; Zen et al., 2016) and transformers (Li et al., 2019a) have been applied to TTS and demonstrate impressive results. Note that as with ASR, we do not consider the TTS module in this thesis.

2.7 Joint Optimisation of Dialogue Components

The components discussed thus far are typically learned separately and their combination forms the pipelined dialogue system. Even though such systems are popular and commonly used in industry (Williams, 2009), they have several major limitations. First, the workflow interdependency between components restricts the overall system scalability (Bordes et al., 2016). When a new domain or service is to be incorporated, all of the system components have to be retrained or even redesigned (e.g., larger vocabulary size, more acts/slots) to perform well on the new target domain. This considerably increases system development effort. Second, it is not guaranteed that the optimality of each individual component results in the optimality of the entire system. For example, Takanobu et al. (2020b) found that among

various pipelined dialogue systems, the one including the DST module with the highest accuracy achieved the worst dialogue success rate.

The aforementioned problems drive the research interest in joint optimisation of system components. In this learning framework, the system, treated as a large neural network, learns to directly generate the response given the current user input and the dialogue context.[2] Wen et al. (2016, 2017b) proposed trainable network-based dialogue systems based on CNNs and LSTMs. Bordes et al. (2016) adapted the memory network (Sukhbaatar et al., 2015; Weston et al., 2014), a common architecture in question-answering task, to task-oriented dialogues. Lei et al. (2018) presented the Sequicity model, where the entire system is based on a two-stage sequence-to-sequence structure with copying, largely simplifying the complexity in end-to-end models and becoming the main paradigm later. Learning dialogue structure through latent variables was also explored (Wen et al., 2017a; Zhao and Eskenazi, 2018; Zhao et al., 2019). One of the advantages of joint learning models is that, during training, the error signals received at the end of system will be back-propagated through the entire network. This is especially meritorious in domain adaptation and transfer learning. More recently, transformer based models and pretrained language models have been utilised (Budzianowski and Vulić, 2019; Hosseini-Asl et al., 2020; Peng et al., 2020a). During training, the input sequence passed to the model is merely the concatenation of the dialogue context, belief state, dialogue acts and system response, and the model is optimised towards predictions of the last three. The trend of using powerful generative transformers makes the system design easier and less dependent on expert knowledge, but also less interpretable at the same time. In Chapter 6, we discuss our end-to-end modelling approach to both dialogue system and user simulator and show how joint optimisation of the two agents benefits domain adaptation.

2.8 Evaluation

To evaluate whether a dialogue system is capable of completing the given task in an efficient way, two metrics are commonly used in literature: task-success and dialogue efficiency. Task-success measures if the dialogue agent successfully accomplishes the task for end users, which confirms whether it provides the correct entity that meets the users' constraints (e.g., restaurant's price range, food type) and whether all the user's requests are answered (e.g., restaurant's address, phone number) (Schatzmann et al., 2007). Dialogue efficiency is computed with respect to the dialogue length. The system that uses fewer turns to finish the task is considered to be more efficient (Walker et al., 1997). There is also a line of research

[2] In literature, this joint learning framework of all dialogue components is often called end-to-end dialogue system.

focusing on end-user satisfaction and it is quantified through a satisfaction rating given by either the interactive users or the dialogue experts (Schmitt and Ultes, 2015; Ultes et al., 2013). These metrics are usually used to evaluate the policy manager in the pipelined system (Section 2.4) or the joint learning framework (Section 2.7).

In an ideal evaluation setup, the dialogue system to be evaluated should be deployed for interaction with end users, and the resulting conversations are then collected for analysis. Yet, as mentioned in Section 2.4, it is impractical to conduct such user trials due to the cost and inefficiency. A user simulator is used as proxy instead (Lee et al., 2019; Ultes et al., 2017; Zhu et al., 2020). Although this method can provide as many as interactions desired for evaluation, the performance obtained from these simulations is largely affected by the quality of the user simulator, which is not guaranteed to emulate the wide variety of behaviours exhibited by real users (Coca et al., 2021). Another way of obtaining interactions is to leverage the test corpus: given a dialogue context and a most recent user utterance, the dialogue agent generates the next response.[3] This method overestimates the model performance, especially because the test set can be linguistically similar to and feature the same set of user behaviours as the training set. Therefore, unless a powerful user simulator capable of modelling a wider range of human behaviours is available, this evaluation method fails to take into account a variety of user behaviours unseen in the available corpora. As a consequence, dialogue systems are seldom robust when interacting with real users (Takanobu et al., 2020b).

There are other metrics used for evaluation of sub-tasks in the pipelined system. For instance, intent accuracy and joint goal accuracy (Henderson et al., 2014a) are employed to judge how well the understanding modules (i.e., NLU and DST) perform, whereas the semantic accuracy and the widely used NLG metrics (Wen et al., 2015a) such as BLEU (Papineni et al., 2002) or ROUGE (Lin, 2004) are adopted to examine how well the language generation model works. Through Chapter 3 to Chapter 7, the experiment section in each chapter describes which evaluation methods are utilised to evaluate our proposed models.

2.9 Summary

In this chapter, we have reviewed the spoken dialogue architecture together with individual components. Despite promising results brought about by advances in neural architectures, dialogue modelling remains challenging due to the fact that state-of-the-art methods require large amounts of data to perform well, and collecting such training data with fine-grained annotations is not a trivial task. For example, NLG modules require turn-level dialogue act

[3]The popular MultiWOZ benchmark (Budzianowski et al., 2018) relies on this approach as no user simulator was released with this corpus initially.

annotation (Section 2.5). This requirement has limited the scalability of systems to new domains. We address this issue from two perspectives: domain adaptation and unsupervised learning. In Chapter 3, we propose a tree-based NLG model that captures the hierarchy of dialogue acts to facilitate knowledge sharing in domain adaptation. In Chapter 4, we present a generative modelling approach that allows learning from un-annotated data through performing both natural language understanding and generation via a shared latent variable.

Dialogue state annotations are also difficult to collect because labelling dialogue requires reasoning over dialogue context (Section 2.3). We alleviate this reliance by exploring popular semi-supervised learning techniques and their applicability to end-to-end dialogue modelling (Chapter 5). Furthermore, dialogue policy learning on new domains requires the corresponding user simulator to be adaptive (Section 2.4). We tackle this problem by joint learning with a transferable user simulator which is able to adapt to new domains together with a dialogue system (Chapter 6). The two learned models can generate additional dialogues through interaction and can be further optimised using RL to conquer the data sparsity problem. Lastly, coreference and ellipsis in user utterances render dialogue understanding difficult (Section 2.3). This problem is addressed by a novel model that jointly learns coreference resolution and query rewriting (Chapter 7).

Chapter 3

Domain Adaptation through Knowledge Sharing for Dialogue Generation

Building a high quality natural language generation (NLG) system requires a significant number of annotated pairs of dialogue acts and corresponding natural language utterances. When a new domain is to be incorporated, it is unlikely and undesirable to spend vast resources on collecting sufficient data to obtain satisfactory performance. In practice, extending dialogue systems to a new domain often starts with limited data. Domain adaptation is therefore crucial for the system to deal with evolving dialogue situations. In this chapter, we propose an NLG model that encodes the input dialogue act using a tree-structure encoder to share knowledge between the source and target domains. This work was published in Tseng et al. (2019a):

- **Tseng, B. H.**, Budzianowski, P., Wu, Y. C., and Gasic, M, "Tree-Structured Semantic Encoder with Knowledge Sharing for Domain Adaptation in Natural Language Generation". *In Proc of SIGDIAL 2019.*

3.1 Motivation

Compared to single-domain dialogues (such as the CamRest676 dataset (Wen et al., 2017b) for restaurant domain), multi-domain dialogues cover more dialogue topics and situations, leading to a considerable increase in the ontology complexity. For example, there are up to 24 slots and 4510 values in the MultiWOZ dataset, as in the ontology shown in Figure 3.1 (Budzianowski et al., 2018). By contrast, the CamRest676 dataset (Wen et al., 2017b) has only 4 slots and 99 values. In addition, the input dialogue act to the NLG model could be

more complicated in terms of its semantic representation. For example, a single-domain instance in the restaurant domain (Novikova et al., 2017b) could be:

- Dialogue Act: `domain=restaurant, inform(name=Clowns, rating=average, type=coffee shop, food=English)`
- Natural Language: "*Clowns is a three-star coffee shop that provides breakfast.*"

By contrast, a multi-domain instance could be:

- Dialogue Act:
  - `domain=train`
    - `inform(train-id=TR0904, leave=14:48)`
  - `domain=restaurant`
    - `inform(price=expensive, food=Chinese, area=centre)`
    - `recommend(name=Ugly Duckling)`
    - `offer-book()`
- Natural Language: *"The train TR0904 leaves at 14:48. We have several expensive Chinese restaurants in the centre. I could recommend Ugly Duckling. Would you like a reservation?"*

By comparison, the multi-domain example not only covers information across two domains, but also contains several dialogue acts concurrently in a single turn. High complexity in both the ontology and dialogue act of multi-domain dialogues makes NLG modelling challenging.

The aforementioned issue could be alleviated to some extent if a sufficient amount of data is accessible across domains. However, in practice it is not feasible to re-collect data for each new domain. A good strategy is to extend an existing pre-trained system to a new dialogue situation with limited resources via domain adaptation. In this low-resource regime, it is thus important to leverage the source domain knowledge for learning a better generalisation to the target domain.

A dialogue act together with slot-value pairs in fact has a hierarchical structure, where each domain has several dialogue acts and each dialogue act has its associated slot-value pairs. Modelling this hierarchical structure could facilitate information sharing between domains. To illustrate this, Figure 3.2 presents dialogue act examples in two domains. Semantic features that consider dialogue act hierarchy would be similar in the two cases, with only domain information being different at top. When adapting from a rich domain (e.g., restaurant) to a low-resource domain (e.g., attraction), analogous dialogue acts with structure

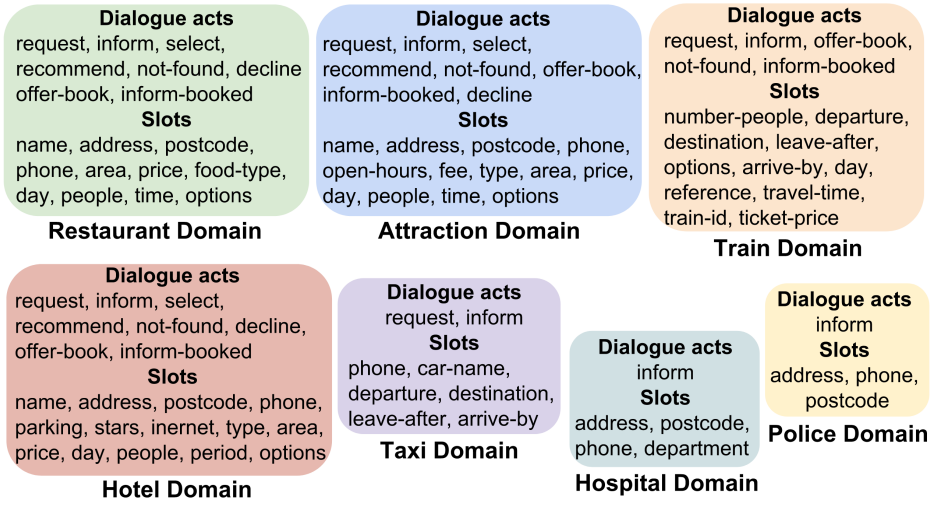


Fig. 3.1 The ontology of the MultiWOZ datatset.

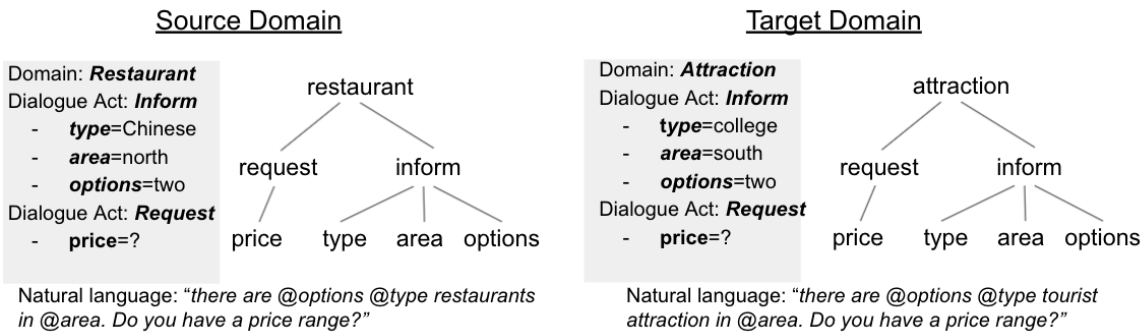


Fig. 3.2 Examples of dialogue acts in two domains together with diagrams of their hierarchy and the corresponding natural language.

knowledge obtained from the source domain facilitate learning of mapping between dialogue act and natural language in the target domain. Earlier, previous NLG models encoded a dialogue act either using a flat representation such as a binary vector (Wen et al., 2015a,b), or using a sequential model such as an LSTM (Dušek and Jurčíček, 2016; Tran and Nguyen, 2017). These encoding methods do not capture the structure of dialogue acts.

This chapter presents an NLG model where a tree-based encoder is used to model the dialogue act structure. An LSTM decoder generates the output sentence based on this encoded tree representation of dialogue acts. A tree-based attention mechanism is proposed to facilitate information flow between the tree encoder and the LSTM decoder. On the MultiWOZ dataset (Budzianowski et al., 2018), results show that the proposed model outperforms previous systems on both automatic metrics and human evaluation, suggesting that modelling the semantic structure in dialogue acts benefits domain adaptation in NLG.

3.2 Modelling

As shown in Figure 3.3, the proposed NLG model has two components: a tree-structured semantic encoder and an LSTM based decoder.[1] The tree-structured encoder extracts a semantic embedding of a dialogue act, and the decoder is then conditioned on the obtained embedding to generate natural language[2]. Inspired by the attention mechanism used in sequence-to-sequence models (Bahdanau et al., 2015), a tree-based attention mechanism is proposed between the tree encoder and the LSTM decoder to increase the generation accuracy of slot tokens. Model details are illustrated below.

3.2.1 Tree-Structured Semantic Encoder

Tree Hierarchy in Dialogue Act

First, we convert the input dialogue act into a tree representation and extract its semantic embedding. An example of a dialogue act and its tree representation is presented in Figure 3.3 (left). The middle three layers of the tree (besides the tree root and leaves) corresponds to the hierarchy of the dialogue act: `domain` -> `act` -> `slot`. Links between layers model the relationships between different information types. For example, the slots `area` and `options` are informed, so both of their nodes are linked to the `inform` act node. By contrast, the slot `price` is requested, so its node is linked to the `request` act node. This is the same for the relationship between the domain and act layers. Furthermore, slot types (such as `informable` and `requestable`, mentioned in Chapter 2) are modelled through the bottom layer. Finally, the tree root represents the aggregated information of a dialogue act. With this tree representation, the structure of a dialogue act that contains several actions concurrently across multiple domains can be captured. Next, we show how to compute vector representations in a bottom-up manner using a tree-based LSTM.

Tree Encoding

We make a use of the tree LSTM (Tai et al., 2015), which is proposed to model syntactic properties of natural language. Given a tree that corresponds to a dialogue act, we first compute word embeddings at each node where node information (i.e., `domain`, `action`, `slot` and slot properties) are viewed as word tokens. Information is then propagated from

[1] As the encoder adopted in our approach follows the equations of the tree-structured LSTM (Tai et al., 2015), we use the term 'tree-based' LSTM to describe our model.

[2] Following prior studies (Wen et al., 2015b), delexicalised tokens are used in system responses. In this work, a value for a slot is replaced by a special token in the format of `@domain-act-slot`. For instance, the informed restaurant *"Golden House"* is replaced by the token `@restaurant-inform-name`.

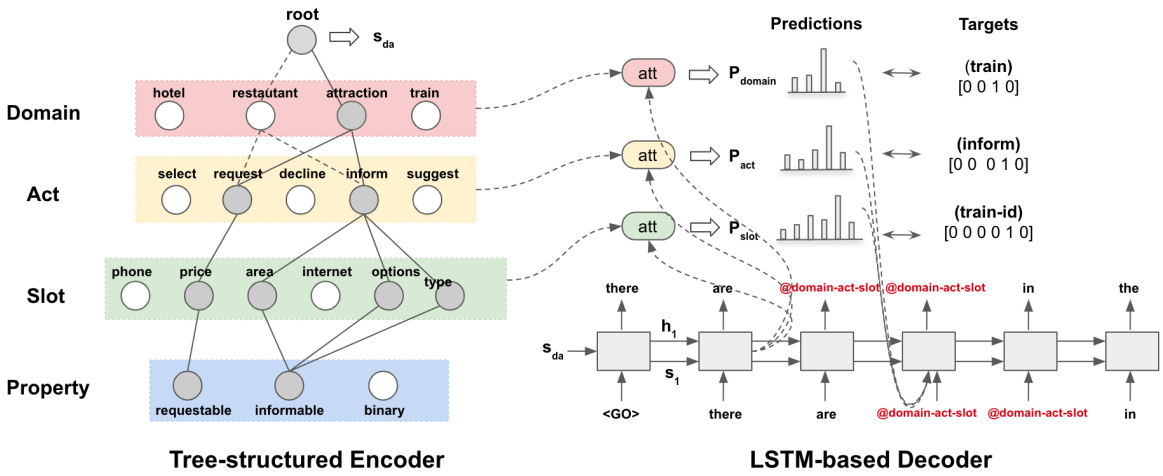


Fig. 3.3 Overview of the proposed NLG model. The tree-structured encoder encodes the input dialogue act. The obtained dialogue act embedding s_{da} is then used to condition the LSTM decoder. Active nodes in grey color specify the information presented in the dialogue act. The tree-based attention between two components occurs when the delexicalised token `@domain-act-slot` is generated. The example sentence here is *"there are @attraction-inform-options @attraction-inform-type in the @attraction-inform-area, do you have a price range in mind?"*

bottom to top through iterative transitions. A non-leaf node j has two inputs: its token embedding e_j and its children states h_k, c_k, and the hidden state of node h_j is computed as:

$$
\begin{aligned}
\tilde{h}_j &= \sum_{k \in C(j)} h_k, \\
\tilde{c}_j &= \sum_{k \in C(j)} c_k, \\
i_j &= \sigma(W_E^{(i)} e_j + U_E^{(i)} \tilde{h}_j + b_E^{(i)}), \\
f_j &= \sigma(W_E^{(f)} e_j + U_E^{(f)} \tilde{h}_j + b_E^{(f)}), \\
o_j &= \sigma(W_E^{(o)} e_j + U_E^{(o)} \tilde{h}_j + b_E^{(o)}), \\
g_j &= \tanh(W_E^{(g)} e_j + U_E^{(g)} \tilde{h}_j + b_E^{(g)}), \\
c_j &= i_j \circ g_j + f_j \circ \tilde{c}_j, \\
h_j &= o_j \circ \tanh(c_j),
\end{aligned}
\tag{3.1}
$$

where $C(j)$ is the set of children nodes and k is the children index; $\tilde{h}_j$ and $\tilde{c}_j$ are the sum of children's hidden states and memory cells respectively. The dialogue act embedding s_{da} is obtained at the tree root and and will be used to condition the decoder for generation.

3.2.2 Decoder

The decoder for sentence generation is an LSTM based model (Hochreiter and Schmidhuber, 1997). As shown in Wen et al. (2015b), the introduction of a control cell over the dialogue act feature benefits the generation performance. Unlike the 1-hot dialogue act feature used in their work, the feature s_{da} obtained by our tree encoder is a continuous vector. In order to handle the more abstract and complex vector representation during generation, we introduce two additional gates, reading gate r and writing gate w, into the LSTM.

At each time step t of generation, as in a vanilla LSTM, the memory cell c_t is computed:

$$\begin{aligned}
i_t &= \sigma(W_D^{(i)} x_t + U_D^{(i)} h_{t-1} + b_D^{(i)}),\\
f_t &= \sigma(W_D^{(f)} x_t + U_D^{(f)} h_{t-1} + b_D^{(f)}),\\
o_t &= \sigma(W_D^{(o)} x_t + U_D^{(o)} h_{t-1} + b_D^{(o)}),\\
g_t &= \tanh(W_D^{(g)} x_t + U_D^{(g)} h_{t-1} + b_D^{(g)}),\\
c_t &= i_t \circ g_t + f_t \circ c_{t-1}.
\end{aligned} \tag{3.2}$$

where x_t, h_{t-1} are the input word embedding and the previous hidden state. The two additional gates are then used to update the dialogue act feature s_t at each time step. The reading gate r_t determines the information carried over from the previous step, while the writing gate g_t controls how much the new information to be added:

$$\begin{aligned}
r_t &= \sigma(W_D^{(r)} x_t + U_D^{(r)} h_{t-1} + V_D^{(r)} s_{t-1} + b_D^{(r)}),\\
w_t &= \sigma(W_D^{(w)} x_t + U_D^{(w)} h_{t-1} + V_D^{(w)} s_{t-1} + b_D^{(w)}),\\
d_t &= \tanh(W_D^{(d)} x_t + U_D^{(d)} h_{t-1} + V_D^{(d)} s_{t-1} + b_D^{(d)}),\\
s_t &= w_t \circ d_t + r_t \circ s_{t-1}.
\end{aligned} \tag{3.3}$$

The hidden state h_t is then computed as the weighted sum of the memory cell c_t and the dialogue act feature s_t:

$$h_t = o_t \circ \tanh(c_t) + (1 - o_t) \circ \tanh(s_t). \tag{3.4}$$

The probability of the output token is then obtained by applying a softmax layer on the hidden state.

3.2.3 Tree-based Attention Mechanism

The attention mechanism (Bahdanau et al., 2015) in a sequence-to-sequence model (Sutskever et al., 2014) allows the decoder to fully access the encoder states at each generation step, resulting in a significant improvement in generation quality. However, this attention mechanism could not be directly applied to a non-sequential structure (e.g., the tree encoder in our model). Here we propose a tree-based attention mechanism to increase the information sharing between the two components in the proposed model.

During generation, when a delexicalised token (i.e., `@domain-act-slot`) is generated at a time step t, the decoder attends over states in the tree encoder to determine the actual values of domain, act and slot. Specifically, we use the dialogue act feature s_t as a query to attend over states at different tree layers so as to obtain domain, act and slot distributions:

$$\begin{aligned} p(d_t|x_{<t},s_t) &= \frac{\exp(\text{score}(s_t,h_d))}{\sum_{d'\in D}\exp(\text{score}(s_t,h_{d'}))}, \\ p(a_t|x_{<t},s_t) &= \frac{\exp(\text{score}(s_t,h_a))}{\sum_{a'\in A}\exp(\text{score}(s_t,h_{a'}))}, \\ p(s_t|x_{<t},s_t) &= \frac{\exp(\text{score}(s_t,h_s))}{\sum_{s'\in S}\exp(\text{score}(s_t,h_{s'}))}, \end{aligned} \tag{3.5}$$

where h_d, h_a and h_s are the hidden states at layers of domain, act and slot in the tree encoder. D, A and S are numbers of domains, acts and slots in the input dialogue act. The score function used to calculate the similarity between two vectors is their dot-product:

$$\text{score}(f,h) = f^T h. \tag{3.6}$$

Distributions $p(d_t)$, $p(a_t)$ and $p(s_t)$ are then used to predict the domain, act and slot at time step t by taking the `argmax` to realise the delexicalised token `@domain-act-slot`.

A good NLG model should precisely generate information presented in the dialogue act. In order to not over generate or miss any slot tokens, the three predicted distributions, $p(d_t)$, $p(a_t)$ and $p(s_t)$, are used as additional signals to guide further generation. The input vector at next time step $t+1$ is a concatenation of the word vector x_{t+1} and three predicted distributions.[3] In such a way, the model has a better sense of what slot tokens have been generated.

[3] At a time step t' where the output is not a delexicalised token, the tree-based attention is not triggered and the input to the next time step $t'+1$ the input word vector padded with zeros.

3.2.4 Optimisation

The model is optimised against both the language model loss and the tree-based attention loss. The former is the cross-entropy between predicted word distributions and 1-hot vectors of ground-truth. The latter is the cross-entropy between the predicted domain/act/slot distributions and the corresponding 1-hot vectors of ground-truth. The overall training loss is the sum of two losses with equal weights.

3.3 Experiments

3.3.1 Dataset

Experiments were conducted on the MultiWOZ dataset (Budzianowski et al., 2018), which contains 10k dialogues spanning over 7 domains. Unlike prior single-domain NLG datasets (such as the CamRest676 dataset (Wen et al., 2017b) or the E2E dataset (Novikova et al., 2017b)) where only one act (e.g., inform) presents in the whole dialogue act representation, multi-domain dialogues with concurrent acts per single turn make the MultiWOZ more challenging for NLG modelling. Data statistics of the five major domains are shown in Table 3.1.

Domain	Dialogue turns	Distinct dialogue act	Number of acts	Number of slots
Restaurant	8.5k	346	8	11
Hotel	6.6k	378	8	14
Attraction	6.4k	314	8	13
Train	11k	338	5	11
Taxi	3.4k	47	2	6

Table 3.1 Statistics for each domain in the MultiWOZ dataset.

3.3.2 Experiment Design

NLG systems were implemented using the Pytorch library (Paszke et al., 2017). The number of hidden units in LSTMs was 100 with one hidden layer. The dropout rate was 0.25 and the Adam optimizer (Kingma and Ba, 2014) was used. In domain adaptation experiments, the learning rate was 0.0025 for training from scratch on source domains, and 0.001 for adapting to target domains. Beam search was used during decoding with a beam size 10. Our complete model was trained against both the LM loss and tree-based attention loss, denoted as *Tree+Att*. A model variant without attention was also investigated, denoted as *Tree*.

3.3.3 Evaluation

For automatic metrics, both BLEU scores (Papineni et al., 2002) and the semantic error rate (SER) used in Wen et al. (2015b) are reported. The SER is measured based on the delexicalised slot tokens generated in output sentences compared to the input dialogue act, and is defined as $(p+q)/N$, where p,q are numbers of missing and redundant slots in the generated sentence, and N is the number of slots in the dialogue act. A slot token (`@domain-act-slot`) is correct only if its domain, act and slot are all matched to the ground-truth. Results in all experiments are averaged over 5 random seeds and 10 generated sentences per example obtained by beam search.

3.3.4 Baselines

Three baselines were compared in our experiments:

1. the semantically-conditioned LSTM (SCLSTM) model (Wen et al., 2015b) that has an additional control cell to update the binary representation of a dialogue act.

2. the TGen model (Dušek and Jurčíček, 2016) that is a seq2seq model with attention where the dialogue act is represented in a linear sequence.

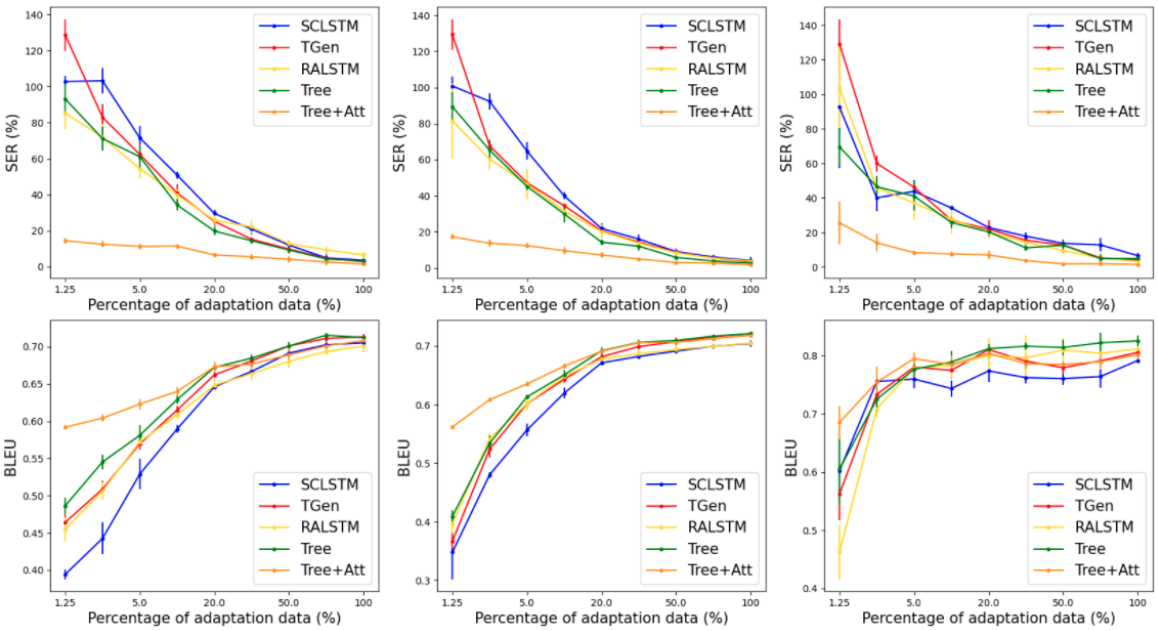
3. the refinement adjustment LSTM model (RALSTM) (Tran and Nguyen, 2017) that is an improved seq2seq model with a refinement gate and an adjustment gate in the decoder.

All the baselines and our models were optimised against the validation set during training.

3.3.5 Metric-based Evaluation

Domain adaptation experiments were conducted in three setups: (a) from restaurant to hotel domain; (b) from restaurant to attraction domain and (c) from train to taxi domain. NLG models were first trained using the full data of source domains and then fine-tuned on target domains using various data amounts. Figure 3.4 shows results of the SER (the first row) and BLEU scores (the second row).

In general, with sufficient adaptation data (i.e., more than 75% target domain data being used), all the systems perform similarly with low SER results, and the model performance increases with the data size. In a low-resource regime (from 1.25% to 20%), the proposed model without attention (Tree) performs on par with the RALSTM model, and performs slightly better than the other two baselines. The complete model with attention (Tree+Att),



(a) Restaurant to Hotel domain (b) Restaurant to Attraction domain (c) Train to Taxi domain

Fig. 3.4 Domain adaptation results in three setups: (a) from restaurant to hotel domain; (b) from restaurant to attraction domain and (c) from train to taxi domain. SER results and BLEU scores are reported in two rows respectively, with their standard deviations over 5 random seeds. The size of adaptation data varies from 1.25% to 100% of original training corpus, with 2.5%, 5%, 10%, 20%, 30%, 50%, 75% increments considered.

however, significantly outperforms all the others. Especially with only 1.25% adaptation data, the SER obtained by the Tree+Att model is below 25%, while that by the others is in the range of between 75% and 125%. We observe that baseline models tend to predict slots with either an incorrect act or incorrect domain when adapting using limited data. By contrast, the tree-based attention allows our model to concentrate on information at different granularity in the tree encoder, leading to a large improvement in SER. An illustrative example is provided in Section 3.3.8 where attention distributions are visualised. A similar trend in model performance can be observed in BLEU scores.

3.3.6 Human Evaluation

Since automatic metrics (such as BLEU) may not consistently agree with human perception (Novikova et al., 2017a; Reiter, 2018; Stent et al., 2005), it is important to perform human evaluation for NLG. The Amazon Mechanical Turk service was used here. To focus on the models' adaptation ability in low-resource situations, we showed the hired workers model outputs obtained from experiments using 1.25% to 10% adaptation data. Five models (three

baselines and our two model variants) were included in the evaluation. For each model, the two sentences with the highest probabilities out of the 10 generated sentences by beam search were selected. The workers were asked to score each sentence from 1 (bad) to 5 (good) in terms of its *informativeness* and *naturalness*. The informativeness is defined as the degree to which the generated sentence contains all the information specified in the given dialogue act, and the naturalness is defined as whether the generated sentence sounds like human language.

Ipeirotis et al. (2010) pointed out that malicious workers might take advantage of the difficulty of verifying results and therefore submit answers with low quality. In order to filter out submissions with bad quality, we also asked workers to score the ground truth sentence and an artificial sentence that was irrelevant to the given dialogue act. If the worker assigned a ground truth sentence a low score (< 3) or an artificial sentence a high score (> 3) in informativeness, the submission was discarded.

Results are reported in Table 3.2. For informativeness, our models (both Tree+Att and Tree) outperform all three baseline models. These results are consistent with SER results in the automatic evaluation, which indicates that the proposed model is more capable of conveying correct semantics of dialogue acts, even in a low-resource regime. For naturalness, the Tree+Att performs the best in two setups, while the SCLSTM performs better when adapting from train to the taxi domain. The reason might be that the SCLSTM is good at generating utterances with simple patterns and the taxi domain is relatively easy due to its low complexity in dialogue acts (only 2 actions and 6 slots defined in the ontology). When adapting to more complex domains such as hotel or attraction, our model produces high quality utterances in both informativeness and naturalness. Table 3.3 presents two examples of a dialogue act - natural language pair and sentences generated by each model.

Table 3.2 Human evaluation in three adaptation settings: Restaurant (Rest.) to Hotel domain; Restaurant to Attraction (Attr.) domain and Train to Taxi domain. Informativeness (Info.) and Naturalness (Nat.) are reported in a scale of 1 (worse) to 5 (best).

Model	**Rest. to Hotel**		**Rest. to Attr.**		**Train to Taxi**	
	Info.	**Nat.**	**Info.**	**Nat.**	**Info.**	**Nat.**
SCLSTM	2.96	3.85	2.81	3.69	3.05	**4.26**
TGen	2.87	3.33	3.00	3.23	3.42	3.90
RALSTM	2.79	3.48	2.91	3.40	3.48	3.15
Tree	3.08	3.54	3.38	3.41	3.81	3.81
Tree+Att	**4.04**	**4.10**	**4.30**	**3.92**	**4.29**	3.78

Table 3.3 Two examples of dialogue act - natural language pair and generated sentences by different models trained using 1.25% adaptation data. Delexicalised slot tokens are filled by corresponding values and highlighted in bold. Only the Tree+Att model produces semantically correct sentences.

Domain	**Attraction**	**Hotel**
Dialogue Act	Dialogue Act: `Inform` Slot-Value pairs: [Area: **west**] [Options: **five**] [Type: **colleges**] Dialogue Act: `Request` Slot-Value pairs: [Price=?]	Dialogue Act: `Inform` Slot-Value pairs: [Options=**two**] Dialogue Act: `Select` Slot-Value pairs: [type1=**guesthouse**] [type2=**hotel**]
Ground Truth	*there are **five colleges** in the **west**.* *do you mind paying an entrance fee ?*	*i have **two**, would you* *prefer a **guesthouse** or **hotel** ?*
SCLSTM	*what type of place are you looking for ?* (miss 3 slots & request wrong)	*what area would you* *like to stay in ?* (miss 3 slots)
TGen	*there are located in the . do you have a price* *range in mind ?* (miss 3 slots)	*i have found options. would you* *prefer or ?* (miss 3 slots)
RALSTM	*we have **five colleges** in the **west** area . do you* *have an attraction type in mind ?* (request wrong)	*i have **two** options.* *do you have a preference ?* (miss 2 slots)
Tree	*there are **five colleges** in the **west** . do you have an* *area of town you would prefer ?* (request wrong)	*i have found **two** options for you.* *do you have a preference ?* (miss 2 slots)
Tree+Att	*there are **five colleges** in the **west** .* *do you have a price range in mind ?* (correct)	*i have **two** options for you. would* *you prefer **guesthouse** or **hotel** ?* (correct)

3.3.7 Error Analysis

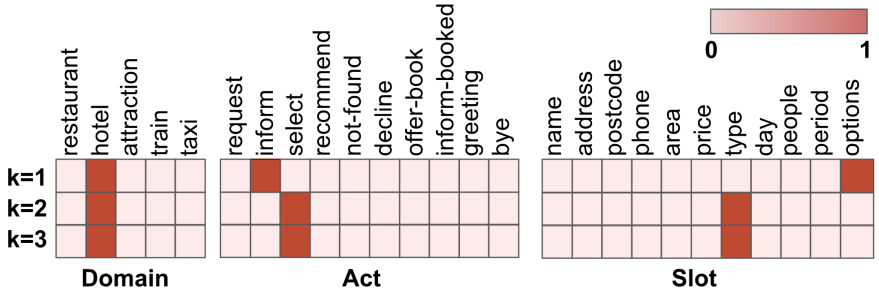
To further investigate errors made by NLG systems and reasons behind the superiority of our model over baselines, the following error analysis is performed. The whole test set is divided into two subsets - *seen* and *unseen*. An example is counted as *seen* if its dialogue act appears during training. Otherwise, it is marked as *unseen*. As shown in Table 3.4, for each domain adaptation experiment, numbers of test examples and incorrect generation outputs (with at least a missing or redundant slot) by three models are reported. With more adaptation data, more dialogue acts are seen during training. Our model produces more correct generations on both seen and unseen subsets, and the better performance on the *unseen* subset indicates a stronger generalisation ability in our model. For example, with 1.25% adaptation data, the Tree+Att model produces only 134 incorrect generations out of 902 unseen examples, while the SCLSTM makes 729 erroneous predictions.

3.3.8 Visualisation of Tree-based Attention

Here we visualise the tree-based attention in the Tree+Att model using a testing example. As shown in Figure 3.5, the generated sentence contains three slot tokens and attention distributions over domains, acts and slots at the corresponding time steps are plotted. It is observed that the model is confident of generating slot tokens with the correct combination of

Table 3.4 Error analysis on two test subsets under various adaptation data size. Numbers of testing examples and incorrect generation (at least a missing or redundant slot) are reported.

Percentage	**1.25%**		**5%**		**10%**		**50%**	
Testing	**seen**	**unseen**	**seen**	**unseen**	**seen**	**unseen**	**seen**	**unseen**
examples	439	902	858	483	1069	272	1330	11
SCLSTM	248	729	307	412	302	190	111	5
TGen	309	741	176	353	178	168	102	6
Tree+Att	10	134	31	103	60	55	76	3



Generated utterance: *i have @hotel-inform-options (two) options for you. would you prefer @hotel-select-type (guesthouse) or @hotel-select-type (hotel) ?*

Fig. 3.5 Visualisation of tree-based attention distributions over domains, acts and slots at the three time steps of generating delexicalised tokens `@domain-act-slot`. The color shades signify attention weights.

domain, act and slot through learning tree-based attention distributions, even though the data used here for training is only 1.25% of all adaptation data (the hotel domain in this example). We acknowledge that there are different views on whether attention distributions could be a reliable interpretation towards models' decision making process in NLP community (Jain and Wallace, 2019; Wiegreffe and Pinter, 2019). Hence, the visualisation here merely serves as a reference.

3.4 Conclusion

The data sparsity issue in domain adaptation makes NLG modelling challenging. To address this problem, this chapter presents an NLG model that captures the hierarchy of dialogue acts using a tree encoder to facilitate knowledge sharing between domains. The proposed tree-based attention allows the decoder to focus on information at different levels (such as domains, acts and slots) during generation. On the MultiWOZ dataset, results show that our model outperforms strong baselines in both automatic and human evaluation, especially in a

low-resource regime, which indicates that the proposed tree-based encoder and attention are effective in NLG adaptation.

Chapter 4

Joint Modelling of Natural Language Understanding and Generation

Natural language understanding (NLU) and natural language generation (NLG) are two fundamental and related tasks in pipelined dialogue systems (Chapter 2). They have complementary objectives: NLU tackles the transformation from natural language to formal representations (i.e., dialogue acts), whereas NLG does the reverse. A key to success in either task is parallel training data which is expensive to obtain at a large scale. In this chapter, we propose a generative model which couples both NLU and NLG through two latent variables. This approach allows us to explore the spaces of both natural language and formal representations, and it facilitates information sharing through the latent space to eventually benefit the two tasks. Our model can make use of unlabelled data and can be trained using semi-supervised learning, which significantly reduces the high demand for annotated paired data for building NLU/NLG models. This study was published in Tseng et al. (2020):

- **Tseng, B. H.**, Cheng, J., Fang, Y., & Vandyke, D, "A Generative Model for Joint Natural Language Understanding and Generation". *In proc of ACL 2020*.

4.1 Motivation

In research literature, NLU and NLG are well-studied as separate problems. State-of-the-art NLU systems tackle the task as a classification problem (Zhang and Wang, 2016) or as a structured prediction problem (Damonte et al., 2019), depending on the formal representation used which can be parallel slot-value pairs (Henderson et al., 2014d), a first-order logical form (Zettlemoyer and Collins, 2012) or structured queries (Pasupat et al., 2019; Yu et al., 2018).

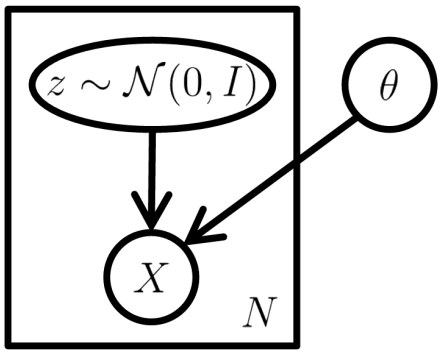


Fig. 4.1 The VAE model, represented as a graphical model. One could generate X by sampling z from a latent space Z (typically it is a standard normal distribution $\mathcal{N}(0, I)$).

On the other hand, approaches to NLG vary from pipelined methods subsuming content planning and surface realisation (Stent et al., 2004) to more recent end-to-end modelling approaches (Dušek et al., 2020; Wen et al., 2015b).

However, the duality between NLU and NLG has been less explored. In fact, both tasks can be treated as a translation problem: NLU converts natural language to formal language while NLG does the reverse. Both tasks require a substantial amount of paired utterances and semantic representations to succeed, and such data is costly to collect due to the complexity of annotation involved. In contrast, unlabelled data could be easily obtained. For instance, large amounts of unlabeled text are available online; reasonable combinations of slot-value pairs in semantic representations can be simulated using heuristics. Despite the fact that these unannotated data can be easily collected, it is less clear how they can be leveraged as the two languages stand in different spaces.

In this chapter, we propose a generative model for **J**oint natural language **U**nderstanding and **G**eneration (**JUG**), which couples NLU and NLG with latent variables representing the shared intent between natural language and formal representations. We aim to learn the association between two discrete spaces through continuous latent variables to facilitate information sharing between the two tasks. Moreover, JUG can be trained in a semi-supervised fashion, which enables us to explore each space of natural language and formal representations when unlabelled data is accessible. We examine our model on two dialogue datasets with different formal representations: the E2E dataset (Novikova et al., 2017b), where the semantics are represented as a collection of slot-value pairs, and the weather dataset (Balakrishnan et al., 2019) where the formal representations are tree structured. Experimental results show that our model improves over standalone NLU/NLG models and existing methods on both tasks. The performance can be further boosted by utilising unlabelled data.

4.2 Background Knowledge: Variational Autoencoder

Before introducing the proposed model, we first briefly review the background knowledge of variational autoencoders (VAEs) (Kingma and Welling, 2013), which serves the foundation of our model. We recommend the readers to refer Doersch (2016) and Blei et al. (2017) for a thorough explanation of VAEs.

In VAEs, our aim is to learn a deterministic function $p_\theta(x|z)$, parameterised by θ, that allows to generate the data x by sampling z from a latent space Z, as shown in Figure 4.1. Intuitively, to make the output very much like samples from the data space X, one can maximise $p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz$ using some optimisation techniques such as gradient descent. This is, however, intractable due to the integration over z, and would require extremely high number of sampled z for a good estimation of $p_\theta(x)$. Instead, the VAE introduces the posterior distribution $q_\phi(z|x)$ that infers z given x with the aim to better estimate those z that could help to reconstruct x.

Here we derive the objective for learning VAEs. By starting with the Kullback-Leibler divergence (KL divergence, or D) between $q_\phi(z|x)$ and $p_\theta(z|x)$ (the true posterior that is unknown to us):

$$D[q_\phi(z|x)||p_\theta(z|x)] = \mathbb{E}_{z\sim q_\phi(z|x)}[\log q_\phi(z|x) - \log(p_\theta(z|x))] \tag{4.1}$$

Expanding $p_\theta(z|x)$ by applying Bayes rule:

$$D[q_\phi(z|x)||p_\theta(z|x)] = \mathbb{E}_{z\sim q(z|x)}[\log q_\phi(z|x) - \log(p_\theta(x|z)) - \log p_\theta(z) + \log(p_\theta(x))] \tag{4.2}$$

Rearranging both sides, we can obtain the evidence lower bound (ELBO) of VAEs:

$$\begin{aligned}\log p_\theta(x) &= D[q_\phi(z|x)||p_\theta(z|x)] + \mathbb{E}_{z\sim q_\phi(z|x)}[\log(p_\theta(x|z))] - D[q_\phi(z|x)||p_\theta(z)] \\ &\geq \mathbb{E}_{z\sim q_\phi(z|x)}[\log(p_\theta(x|z))] - D[q_\phi(z|x)||p_\theta(z)] = \mathscr{L}(\theta,\phi)\end{aligned} \tag{4.3}$$

Note that the inequality is because of the non-negative term $D[q_\phi(z|x)||p_\theta(z|x)]$. During training, the first term in the ELBO $\mathscr{L}(\theta,\phi)$ requires the model to reconstruct the input data x given z, sampled from the posterior $q_\phi(z|x)$. The second term can be seen as the regularisation that pulls the posterior $q_\phi(z|x)$ close to the prior $p_\theta(z)$.

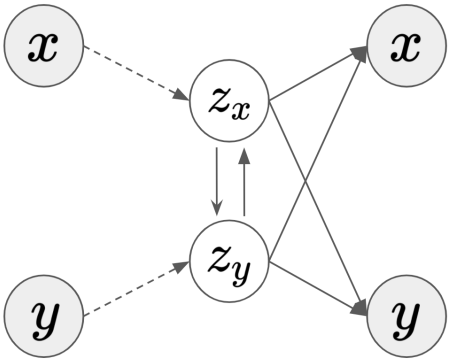


Fig. 4.2 The framework of our generative model. The variables x and y denote utterances and formal representations respectively. The two latent variables, z_x and z_y, are inferred from x and y respectively, and both can be used to generate a pair of x and y.

4.3 The Proposed Model: JUG

As described in Section 4.2, the vanilla VAE focuses on one data space X (e.g., images) and the optimisation concerns the reconstruction/generation of X alone. Instead, in the task of natural language understanding and generation, we aim to learn the mapping between the spaces of natural language and formal representation. This motivates us to come up with a generative model that better captures the relationship between the two data sources. In our modelling, we assume there exists a latent variable z underlying a pair of an utterance x and a formal representation y, and we would like to infer this z from either x or y. From the two inference paths we could obtain z_x and z_y, as depicted in Figure 4.2. This model allows us to perform both NLU (path $x \rightarrow z_x \rightarrow y$) and NLG ($y \rightarrow z_y \rightarrow x$). Details of the proposed model are described in the following.

4.3.1 Natural Language Generation

As aforementioned, the task of NLG requires us to infer z_y from y, and then generate x using both z_y and y. By choosing the posterior distribution $q(z_y|y)$ to be Gaussian[1], the task of inferring z_y can be cast as computing the mean μ and the standard deviation σ of the Gaussian distribution using an NLG encoder. To do this, we use a bi-directional LSTM (Hochreiter and Schmidhuber, 1997) to encode a formal representation y, which is linearised and represented as a sequence of symbols. After the encoding, a list of hidden vectors $\mathbf{H}$ is obtained, with each vector representing the concatenation of forward and backward LSTM states. These hidden vectors are then average-pooled and passed through two feed-forward

[1] A multivariate gaussian distribution, $\mathcal{N}(\nu, \Sigma)$, with Σ being constrained to be a diagonal matrix.

neural networks to compute the mean $\boldsymbol{\mu}_{y,z}$ and the diagonal covariance matrix $\boldsymbol{\Sigma}_{y,z}$ of the posterior $q(z_y|y)$.

$$\begin{aligned}
\mathbf{H} &= \text{Bi-LSTM}(\mathbf{y}) \\
\bar{\mathbf{h}} &= \text{Pooling}(\mathbf{H}) \\
\boldsymbol{\mu}_{y,z} &= \mathbf{W}_{\mu}\bar{\mathbf{h}} + \mathbf{b}_{\mu} \\
\boldsymbol{\Sigma}_{y,z} &= \mathbf{W}_{\sigma}\bar{\mathbf{h}} + \mathbf{b}_{\sigma}
\end{aligned} \tag{4.4}$$

where $\mathbf{W}$ and $\mathbf{b}$ represent neural network weights and biases. Then the latent vector $\mathbf{z_y}$ can be sampled from the approximated posterior using the re-parameterisation trick (Kingma and Welling, 2013):

$$\begin{aligned}
\boldsymbol{\varepsilon} &\sim \mathcal{N}(0, \mathbf{I}) \\
\mathbf{z_y} &= \boldsymbol{\mu}_{y,z} + \boldsymbol{\Sigma}_{y,z}^{1/2} * \boldsymbol{\varepsilon}
\end{aligned} \tag{4.5}$$

The final step is to generate natural language x based on the latent variable $\mathbf{z_y}$ and the formal representation y. We use an LSTM to generate x conditioned on both $\mathbf{z_y}$ and y with the attention mechanism (Bahdanau et al., 2015). At each time step, the decoder computes:

$$\begin{aligned}
\mathbf{g}_i^x &= \text{LSTM}(\mathbf{g}_{i-1}^x, \mathbf{x}_{i-1}) \\
\mathbf{c}_i &= \text{attention}(\mathbf{g}_i^x, \mathbf{H}) \\
p(x_i) &= \text{softmax}(\mathbf{W_v}[\mathbf{c}_i \oplus \mathbf{g}_i^x \oplus \mathbf{z_y}] + \mathbf{b}_v)
\end{aligned} \tag{4.6}$$

where $\oplus$ denotes concatenation of vectors. $\mathbf{x}_{i-1}$ is the word vector of input token; $\mathbf{g}_i^x$ is the decoder hidden state and is initialised with z_y. $p(x_i)$ is the distribution of output word at time step i.

4.3.2 Natural Language Understanding

NLU performs the reverse procedure of NLG. First, an NLU encoder infers a latent variable $\mathbf{z_x}$ from an utterance x. Again, the encoder is implemented using a bi-directional LSTM that converts the utterance into a list of hidden states. These hidden states are average-pooled and passed through feed-forward neural networks to compute the mean $\boldsymbol{\mu}_{x,z}$ and standard deviation $\boldsymbol{\sigma}_{x,z}$ of the posterior $q(z_x|x)$. This procedure follows Equation 4.4 in NLG.

However, note that a subtle difference between natural language and formal language is that the former is ambiguous while the latter is precisely defined. This makes NLU a many-to-one mapping problem but NLG is one-to-many. To better reflect the fact that the

outputs of NLU require less variance, during NLU decoding we choose the latent vector $\mathbf{z_x}$ to be the mean vector $\boldsymbol{\mu}_{x,z}$, instead of sampling it from $q(z_x|x)$ as in Equation 4.5.[2]

The formal representation y is predicted from both z_x and x using an NLU decoder. Since the space of y depends on the formal language construct, we consider two scenarios common in dialogue systems. In the first scenario, y is represented as a set of slot-value pairs, e.g., {*food type=British*, *area=north*} in the restaurant search domain (Mrkšić et al., 2017). The decoder here consists of several classifiers, one for each slot, to predict the corresponding values. Each classifier is modelled by a 1-layer feed-forward neural network that takes z_x as input:

$$p(y_s) = \text{softmax}(\mathbf{W_s}\mathbf{z_x} + \mathbf{b_s}) \tag{4.7}$$

where $p(y_s)$ is the predicted distribution over values of slot s.

In the second scenario, y is a tree-structured formal representation (Banarescu et al., 2013). We then generate y as a linearised token sequence using an LSTM decoder relying on both z_x and x via the attention mechanism (Bahdanau et al., 2015). The decoding procedure follows exactly Equation 4.6.

4.3.3 Model Summary

Thus far we have presented how the JUG model couples NLU and NLG through latent variables z_x and z_y. One characteristic that differentiates our model from standard variational models is that the latent variable can be inferred from one of the input sources (natural language x or formal language y), and therefore can be used to generate data in both output spaces. In the next section, we show the model optimisation and how the semi-supervised learning is achieved by utilising unlabelled x and y.

4.4 Optimisation

We now describe how JUG can be optimised with a pair of x and y (Section 4.4.1) and also unpaired x or y (Section 4.4.2). We specifically discuss the prior choice of JUG objectives in Section 4.4.3. A combined objective can be thus derived for semi-supervised learning: a practical scenario when we have a small set of labelled data but abundant unlabelled data (Section 4.4.4).

[2]Note that it is still necessary to compute the standard deviation $\boldsymbol{\sigma}_{x,z}$ in NLU, since the term is needed for model optimisation. More details are provided in Section 4.4.

4.4.1 Optimising $p(x,y)$

Given a pair of an utterance x and a formal representation y, our objective is to maximise the log-likelihood of the joint probability $p(x,y)$:

$$\log p(x,y) = \log \int_z p(x,y,z) \tag{4.8}$$

The optimisation is not directly tractable since it requires us to marginalise out the latent variable z. However, it can be solved by following the standard practice of neural variational inference (Kingma and Welling, 2013). An objective based on the variational lower bound can be derived as: (z_x is specified)[3]

$$\mathscr{L}_{x,y} = \mathbb{E}_{q(z_x|x)} \log p(y|z_x,x) + \mathbb{E}_{q(z_x|x)} \log p(x|z_x,y) - \mathrm{KL}[q(z_x|x)||p(z)] \tag{4.9}$$

where the first term on the right side includes the NLU model; the second term includes the reconstruction of x; and the last term is the Kullback−Leibler divergence between the approximate posterior $q(z_x|x)$ and the prior $p(z)$. The prior choice is discussed in Section 4.4.3.

The symmetry between x and y in our model offers an alternative way of inferring z by introducing the approximate posterior $q(z_y|y)$. Analogously we can derive a variational optimisation objective: (same here, z_y is specified)

$$\mathscr{L}_{y,x} = \mathbb{E}_{q(z_y|y)} \log p(x|z_y,y) + \mathbb{E}_{q(z_y|y)} \log p(y|z_y,x) - \mathrm{KL}[q(z_y|y)||p(z)] \tag{4.10}$$

where the first term includes the NLG model; the second term includes the reconstruction of y; and the last term is the KL divergence between the approximate posterior $q(z_y|y)$ with the prior $p(z)$.

It can be observed that our model has two posterior inference paths from either x or y, and also two generation paths. All paths are optimised during training.

4.4.2 Optimising $p(x)$ or $p(y)$

Additionally, when we have access to unlabelled utterances x (or formal representations y), the goal is to maximise the marginal likelihood $p(x)$ (or $p(y)$):

$$\log p(x) = \log \int_y \int_z p(x,y,z) \tag{4.11}$$

[3]Detailed derivations of objectives are provided in the Appendix A.

Note that both z and y are unobserved in this case.

We can develop an objective based on the variational lower bound for the marginal:[3]

$$\mathscr{L}_x = \mathbb{E}_{q(y|z_x,x)}\mathbb{E}_{q(z_x|x)} \log p(x|z_y, y) - \mathrm{KL}[q(z_x|x)||p(z)] \tag{4.12}$$

where the first term is the auto-encoder reconstruction of x with a cascaded NLU-NLG path. The second term is the KL divergence which regularizes the approximated posterior distribution.

When computing the reconstruction term of x, it requires us to first run through the NLU model to obtain the prediction of y, from which we run through the NLG model to reconstruct x. The full information flow is ($x \rightarrow z_x \rightarrow y \rightarrow z_y \rightarrow x$) and it requires us to sample both z and y in reconstructing x. Since y is a discrete sequence, we use REINFORCE (Williams, 1992) to pass the gradient from NLG to NLU in the cascaded NLU-NLG path. Connections can be drawn with recent work which use back-translation to augment training data for machine translation (He et al., 2016; Sennrich et al., 2016). Unlike back-translation, the presence of latent variable in our model requires us to sample z along the NLU-NLG path. The introduced stochasticity allows the model to explore a larger area of the data space.

The above describes the objective when we have unlabelled x. We can derive a similar objective for leveraging unlabelled y:

$$\mathscr{L}_y = \mathbb{E}_{q(x|z_y,y)}\mathbb{E}_{q(z_y|y)} \log p(y|z_x, x) - \mathrm{KL}[q(z_y|y)||p(z)] \tag{4.13}$$

where the first term is the auto-encoder reconstruction of y with a cascaded NLG-NLU path. The full information flow in this case is ($y \rightarrow z_y \rightarrow x \rightarrow z_x \rightarrow y$).

4.4.3 Choice of Prior

The objectives described in Section 4.4.1 and Section 4.4.2 require us to match an approximated posterior (either $q(z_x|x)$ or $q(z_y|y)$) to a prior $p(z)$. A common choice of $p(z)$ in the research literature is the Normal distribution (Kingma and Welling, 2013). However, it should be noted that even if we attempt to match both $q(z_x|x)$ and $q(z_y|y)$ to the same prior, it does not guarantee that the two inferred posteriors are close to each other for paired x and y, which is a desired property of the shared latent space.

To better address this concern, we propose a novel prior choice: when the latent variable is inferred from x using the posterior $q(z_x|x)$, we choose the parameterised distribution $q(z_y|y)$ as our prior belief of $p(z)$. Similarly, when z is sampled from $q(z_y|y)$, we have the freedom of defining $p(z)$ to be $q(z_x|x)$.

Finally, note that it is straightforward to compute both $q(z_x|x)$ and $q(z_y|y)$ when we have parallel x and y. However, when we only have the access to unlabelled data, as described in Section 4.4.2, we can only use the pseudo x-y pairs that are generated by either the NLU or NLG models.

4.4.4 Training Summary

In general, the JUG subsumes the following three training scenarios which we will experiment with. When we have fully labelled x and y, the JUG jointly optimises NLU and NLG in a supervised fashion with the objective:

$$\mathscr{L}_{basic} = \sum_{(x,y)\sim(X,Y)} (\mathscr{L}_{x,y} + \mathscr{L}_{y,x}) \tag{4.14}$$

where (X,Y) denotes the set of labelled examples.

Additionally, in the fully supervised setting, the JUG can be trained to optimise not only NLU and NLG models, but also auto-encoding paths described in Section 4.4.2 (e.g., the path of $x \rightarrow z \rightarrow y \rightarrow z \rightarrow x$). This corresponds to the following objective:

$$\mathscr{L}_{marginal} = \mathscr{L}_{basic} + \sum_{(x,y)\sim(X,Y)} (\mathscr{L}_x + \mathscr{L}_y) \tag{4.15}$$

Furthermore, when we have additional unlabelled x or y, we can optimise the model in a semi-supervised manner as follows:

$$\mathscr{L}_{semi} = \mathscr{L}_{basic} + \sum_{x\sim X} \mathscr{L}_x + \sum_{y\sim Y} \mathscr{L}_y \tag{4.16}$$

where X denotes the set of utterances and Y denotes the set of formal representations.

Dataset	Train	Valid	Test
E2E	42061	4672	4693
Weather	25390	3078	3121

Table 4.1 Number of examples in the two datasets

Natural Language
"sousa offers british food in the low price range. it is family friendly with a 3 out of 5 star rating. you can find it near the sunshine vegetarian cafe."
Semantic Representation
restaurant_name=sousa, food=english, price_range=cheap, customer_rating=average, family_friendly=yes, near=sunshine vegetarian cafe

Table 4.2 An example of paired data in the E2E dataset (Novikova et al., 2017b).

Natural Language (original)
"[__DG_YES__ *Yes*] , [__DG_INFORM__ [__ARG_DATE_TIME__ [__ARG_COLLOQUIAL__ *today's*]] *forecast is* [__ARG_CLOUD_COVERAGE__ *mostly cloudy*] *with* [__ARG_CONDITION__ *light rain showers*]] ."
Natural Language (processed by removing tree annotations)
"Yes, today's forecast is mostly cloudy with light rain showers."
Semantic Representation
[__DG_YES__ [__ARG_TASK__ get_weather_attribute]] [__DG_INFORM__ [__ARG_TASK__ get_forecast] [__ARG_CONDITION__ light rain showers] [__ARG_CLOUD_COVERAGE__ mostly cloudy] [__ARG_DATE_TIME__ [__ARG_COLLOQUIAL__ today's]]]

Table 4.3 An example of the weather dataset (Balakrishnan et al., 2019). Original natural language with tree annotations provided in the dataset (first row) is used for benchmark comparison to existing models (Section 4.5.4). The processed utterance (second row) is used in our semi-supervised experiment (Section 4.5.5).

4.5 Experiments

4.5.1 Dataset

We experimented with two dialogue datasets that contain different formats of formal representations to test the generality of our model. The first is the E2E dataset (Novikova et al., 2017b), which contains utterances annotated with a set of slot-value pairs as their semantic representations. The second is the weather dataset (Balakrishnan et al., 2019), where both utterances and semantics are represented as tree structures. Statistics of the two datasets can be found in Table 4.1, and examples are provided in Tables 4.2 and 4.3.

4.5.2 Model Hyper-parameters

Among all systems in our experiments, the number of units in the LSTM encoder and decoder are set to 150 and 300 respectively. The dimension of the latent space is 150. The Adam

optimiser (Kingma and Ba, 2014) was used, with a learning rate of 1e-3. Batch size is set to {32, 64} for training on two datasets respectively.

4.5.3 Model Training

As described in Section 4.4.4, the proposed model can be optimised in different ways. Variants of our model are explained below:

- **JUG$_{\textbf{basic}}$** jointly optimises the NLU and NLG models using labelled data only, per Equation 4.14.
- **JUG$_{\textbf{marginal}}$** jointly optimises the NLU and NLG models, and auto-encoders of x or y using labelled data only, per Equation 4.15.
- **JUG$_{\textbf{semi}}$** jointly optimises NLU and NLG models using labelled data and auto-encoders using unlabelled data, per Equation 4.16.

For each model variant, during training the best model checkpoint is selected by the average of NLU and NLG results on the validation set.

4.5.4 Benchmark Comparison

We first conduct a benchmark comparison to existing NLU/NLG models by following the data splits of the two datasets. For a fair comparison, we only compare the model variant $\texttt{JUG}_{\texttt{basic}}$ that is trained using the whole training set, without any additional unlabelled data. Baseline systems together with results are shown in Table 4.4. On the E2E dataset, we report F1 scores of slot-values pairs for NLU evaluation, and BLEU scores (Papineni et al., 2002) for NLG evaluation.

As seen from Table 4.4, our model outperforms the previous state-of-the-art NLU and NLG systems on the E2E dataset, and also for NLG on the weather dataset. These results prove the effectiveness of joint training of NLU and NLG through the introduced latent variables. The impact of the latent variables is discussed in Section 4.5.6.

4.5.5 Semi-supervised Learning

In order to test the ability of our model to leverage unlabelled data, we set up a semi-supervised learning experiment, with different proportion of labelled (5%, 10%, 25%, 50%, 100%) and unlabelled data (with the rest being unlabelled).[4] Note that the E2E dataset

[4]Note that two semi-supervised learning models have no results on 100% labelled case due to no unlabelled data available.

E2E NLU	**F1**
Dual supervised learning (Su et al., 2019b)	0.7232
$\text{JUG}_{\text{basic}}$	**0.7337**
E2E NLG	**BLEU**
TGEN (Dušek and Jurčíček, 2016)	0.6593
SLUG (Juraska et al., 2018)	0.6619
Dual supervised learning (Su et al., 2019b)	0.5716
$\text{JUG}_{\text{basic}}$	**0.6855**
Weather NLG	**BLEU**
S2S-CONSTR (Balakrishnan et al., 2019)	0.7660
$\text{JUG}_{\text{basic}}$	**0.7768**

Table 4.4 Comparison to previous systems on the two datasets. Note that there is no previous system trained for NLU on the weather dataset.

Model / Data	5%	10%	25%	50%	100%
Decoupled	52.77 (0.874)	62.32 (0.902)	69.37 (0.924)	73.68 (0.935)	76.12 (0.942)
Augmentation*	54.71 (0.878)	62.54 (0.902)	68.91 (0.922)	73.84 (0.935)	-
$\text{JUG}_{\text{basic}}$	60.30 (0.902)	67.08 (0.918)	72.49 (0.932)	74.74 (0.937)	78.05 (0.945)
$\text{JUG}_{\text{marginal}}$	62.96 (0.907)	68.43 (0.920)	73.35 (0.933)	**75.74 (0.939)**	**78.93 (0.948)**
$\text{JUG}^{*}_{\text{semi}}$	**68.09 (0.921)**	**70.33 (0.925)**	**73.79 (0.935)**	75.46 (0.939)	-

Table 4.5 NLU results on the E2E dataset. Joint accuracy (%) and F1 score (in bracket) are both reported with varying percentage of labelled training data. Models using unlabelled data are marked with *, and have no results on 100% labelled case due to no unlabelled data available.

is designed with a great amount of unseen slot-values in the test set, and is aimed to test NLG models' generalisation ability to unseen situations (Dušek et al., 2018, 2020). However, this setup would make it too difficult for the classification-based NLU models to perform reasonably and therefore makes the comparison of NLU performance somewhat meaningless. We removed this distribution bias by randomly re-splitting the E2E dataset. On the contrary, utterances in the weather dataset contain additional tree structured annotations,

Model / Data	5%	10%	25%	50%	100%
Decoupled	0.693 (83.47)	0.723 (87.33)	0.784 (92.52)	0.793 (94.91)	0.813 (96.98)
Augmentation*	0.747 (84.79)	0.770 (90.13)	0.806 (94.06)	0.815 (96.04)	-
$\text{JUG}_{\text{basic}}$	0.685 (84.20)	0.734 (88.68)	0.769 (93.83)	0.771 (95.06)	0.810 (95.07)
$\text{JUG}_{\text{marginal}}$	0.724 (85.57)	0.775 (93.59)	0.803 (94.99)	0.817 (98.67)	**0.830 (99.11)**
$\text{JUG}^{*}_{\text{semi}}$	**0.814 (90.47)**	**0.792 (94.76)**	**0.819 (95.59)**	**0.827 (98.42)**	-

Table 4.6 NLG results on the E2E dataset. BLEU and semantic accuracy (%) (in bracket) are both reported.

Model / Data	5%	10%	25%	50%	100%
Decoupled	73.46	80.85	86.00	88.45	90.68
Augmentation*	74.77	79.84	86.24	88.69	-
$\text{JUG}_{\text{basic}}$	73.62	80.13	86.15	87.94	90.55
$\text{JUG}_{\text{marginal}}$	74.61	81.14	86.83	89.06	**91.28**
$\text{JUG}^{*}_{\text{semi}}$	**79.19**	**83.22**	**87.46**	**89.17**	-

Table 4.7 NLU results in exact match accuracy (%) on the weather dataset, with varying percentage of labelled training data. Models using unlabelled data are marked with *, and have no results on 100% labelled case due to no unlabelled data available.

Model / Data	5%	10%	25%	50%	100%
Decoupled	0.632	0.667	0.703	0.719	0.725
Augmentation*	0.635	0.677	0.703	0.727	-
$\text{JUG}_{\text{basic}}$	0.634	0.673	0.701	0.720	**0.726**
$\text{JUG}_{\text{marginal}}$	0.627	0.671	0.711	0.721	0.722
$\text{JUG}^{*}_{\text{semi}}$	**0.670**	**0.701**	**0.725**	**0.733**	-

Table 4.8 NLG results in BLEU on the weather dataset.

which considerably simplifies the NLU task. We removed these annotations to make NLU more realistic, as shown in the second row of Table 4.3.

In this experiment, two baselines are considered as follows:

- **Decoupled**: The NLU and NLG models are trained separately by supervised learning. The two models have the same encoder-decoder structure as the JUG. The main difference to our model is that there is no latent variables between the two individual NLU and NLG models.

- **Augmentation**: The pre-trained `Decoupled` models are used to generate pseudo labels from unlabelled data (Lee, 2013) in a setup similar to back-translation (Sennrich et al., 2016). The pseudo data and labelled data are then used together to fine-tune the pre-trained models.

Results on the E2E dataset are presented in Table 4.5 and 4.6.[5] We computed both F1 score and joint accuracy (Mrkšić et al., 2017) of slot-values as a more solid NLU measurement. Joint accuracy is defined as the proportion of test examples whose slot-value pairs are all correctly predicted. For NLG, both BLEU and semantic accuracy were computed. Semantic accuracy measures the proportion of correctly generated slot values in the produced utterances. From results, we can observe that the `Decoupled` model can be improved with

[5]Note that the results of semi-supervised learning presented in this section are not comparable to the benchmark result (Table 4.4) due to the applied data processing as mentioned abovve.

Ground Truth
x: *"for those prepared to pay over £30 , giraffe is a restaurant located near the six bells ."*
y: {name=giraffe, eat_type=restaurant, price_range=more than £30, near=the six bells}
Predictions by the Decoupled model
x: *"near the six bells , there is a restaurant called giraffe that is children friendly ."* (miss price_range)
y: {name=travellers rest beefeater, price_range=more than £30, near=the six bells} (wrong name, miss eat_type)
Predictions by the JUG$_{\texttt{semi}}$ model
x: *"giraffe is a restaurant near the six bells with a price range of more than £30 ."* (semantically correct)
y: {name=giraffe, eat_type=restaurant, price_range=more than £30, near=the six bells} (exact match)

Table 4.9 An example of the E2E dataset and generation by the baseline model `Decoupled` and the proposed model JUG$_{\texttt{semi}}$. x and y denote natural language and the corresponding semantic representation. Judgement of each prediction is appended at the end and highlighted in color.

the aid of pseudo data (`Augmentation`), which forms a stronger baseline. Even that, our models outperform baselines among various setups of different proportions of labelled data. When using only labelled data (100%), our model JUG$_{\texttt{marginal}}$ can surpass the `Decoupled` in all measurements. The performance gain mainly comes from the fact that our model uses auto-encoding objectives to help the learning of the two latent spaces. Compared to the `Augmentation` model, JUG$_{\texttt{marginal}}$ also has a 'built-in mechanism' to bootstrap pseudo data on the fly of training (see Section 4.4.4). When adding unlabelled data, our model JUG$_{\texttt{semi}}$ obtains further performance boosts and outperforms the baselines by a considerable margin.

With varying proportions of unlabelled data in the training set, we see that unlabelled data is helpful in almost all cases to the proposed model. Moreover, performance gain is pronounced with the smaller amounts of labelled data. This indicates that JUG is especially efficient in low-resource setups where only a limited amount of annotated data is accessible.

Results on the weather dataset are presented in Table 4.7 and 4.8. In this dataset, NLU is more like a semantic parsing task (Berant et al., 2013) so we used exact match accuracy as its measurement. Results reveal a very similar trend observed in the E2E dataset. An E2E example together with model generation outputs are provided in Table 4.9. Note that we did not conduct human evaluation as the NLU performance can be fully reflected by automatic evaluation (i.e., joint accuracy).

4.5.6 Analysis

In this section we further analyse the impact of the introduced latent variables and also the benefit of utilising unlabelled data.

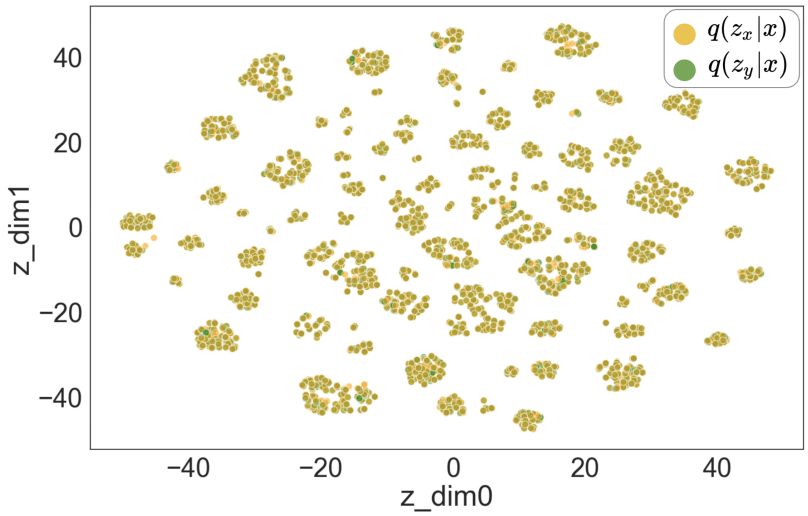


Fig. 4.3 Visualisation of latent variables. Given a pair of x and y, z_x and z_y can be sampled from the posterior $q(z_x|x)$ and $q(z_y|y)$ respectively, denoted by yellow and green dots respectively.

Visualisation of Latent Variables

As mentioned in Section 4.3.1, latent variables can be sampled from either the posterior approximation $q(z_x|x)$ or $q(z_y|y)$. We aim to inspect the latent space to find out how well the model learns intent sharing. We sampled z from the E2E dataset and plotted the samples on a 2-dimentional space using the t-SNE projection (Maaten and Hinton, 2008), as shown in Figure 4.3.

Two interesting properties can be observed. First, for each data point (x, y), two latent variables, z_x and z_y, sampled from $q(z_x|x)$ and $q(z_y|y)$ respectively are close to each other. This suggests that the meanings of x and y are tied in the latent space. Second, there exists distinct clusters in the space of z. By further inspecting the actual examples within each cluster, we found that a cluster represents a type of meaning composition. For instance, the cluster centered at (-20, -40) contains {`name`, `foodtype`, `price`, `rating`, `area`, `near`}, while the cluster centered at (45, 10) contains {`name`, `eattype`, `foodtype`, `price`}. This indicates that the shared latent space serves as conclusive global feature representations for NLU and NLG.

Impact of Latent Variables

One novelty of our model is the introduction of the latent variable z for natural language x and formal representations y. A common problem in neural variational models is that when they are coupled a powerful autogressive decoder, this decoder tends to learn to ignore z and solely rely on itself to generate the data (Bowman et al., 2016; Chen et al., 2017; Goyal et al.,

Model	NLU	NLG
$\text{JUG}_{\text{basic}}$	90.55	0.726
$\text{JUG}_{\text{basic}}$ (using random z)	38.13	0.482

Table 4.10 A comparative study to evaluate the contribution of the learned latent variable z in NLU/NLG decoding. Models are trained on the whole weather dataset.

Method	NLU		NLG	
	Mi	Wr	Mi	Wr
Decoupled	714	2382	5714	2317
$\text{JUG}_{\text{basic}}$	**594**	**1884**	**4871**	**2102**

Table 4.11 Error analysis on the E2E dataset. Numbers of missing (Mi) and wrong (Wr) predictions of slot-value pairs are reported. Lower number indicates the better results. Both models are trained using 5% training data.

2017). In order to examine to what extent our model actually relies on the introduced latent variables in both NLU and NLG, we seek for an empirical answer by comparing the $\texttt{JUG}_{\texttt{basic}}$ with a model variant which uses a random value of z sampled from a normal distribution $N(\mathbf{0},\mathbf{1})$ during testing. As seen from Table 4.10, we observe a large performance drop if z is assigned with random values. This suggests that JUG indeed relies greatly on the latent variables to produce natural language x or formal language y.

Error Analysis

We further analyse sources of error in the two tasks. Errors in semantics mainly come from predicting either missing or wrong slot-value pairs. These two error types are investigated in model predictions by the `Decoupled` and $\texttt{JUG}_{\texttt{basic}}$, reported in Table 4.11. Both models are trained using only 5% training data on the E2E dataset. By comparison, in general our model produces fewer mistakes in conveying the correct meaning in both NLG and NLG. We believe this is because the clustering property learned in the latent space provides better feature representations at a global scale, eventually benefiting the two tasks.

Impact of Unlabelled Data Source

In Section 4.5.5, we found that the performance of our model can be further enhanced by leveraging unlabelled data. In this semi-supervised learning setup, since unlabelled utterances and semantic representations are used together, it is unclear if both contribute to the performance gain. To answer this question, we start with the $\texttt{JUG}_{\texttt{basic}}$, and experiment with adding unlabelled data from 1) only unlabelled utterances x; 2) only semantic representations

	E2E		Weather	
Method	NLU	NLG	NLU	NLG
JUG_{basic}	60.30	0.685	73.62	0.634
+unlabelled x	62.89	0.765	74.97	0.654
+unlabelled y	59.55	**0.815**	76.98	0.621
+unlabelled x and y	**68.09**	0.814	**79.19**	**0.670**

Table 4.12 Comparison of sources of unlabelled data for semi-supervised learning in three setups: (a) only utterances x; (b) only semantic representations y and (c) both x and y. Only 5% of annotated data is used here, with the rest of 95% data being as unlabelled.

Model	BLEU	Semantic Accuracy (%)
JUG_{basic} (z is sampled)	0.771	95.05
JUG_{basic} (z is not sampled)	**0.783**	**96.55**

Table 4.13 Comparison of NLG performance when z is sampled or not during decoding. Models are trained on the E2E dataset using 50% of training data.

y; 3) both x and y. As shown in Table 4.12, when adding any uni-sourced unlabelled data (x or y), most of the time the performance can be improved to a certain extent. However, the maximum improvement is reached when both data sources are utilised. This strengthens the argument that our model can leverage bi-sourced unlabelled data effectively via latent space sharing, which leads to improvements in NLU and NLG at the same time.

Ablation Study of Sampling z in NLG

Due to the one-to-many property in NLG, we sampled z_y using Equation 4.5 to handle the language variety in the utterances. Here we are interested in whether the sampling of z_y would affect the NLG performance. As shown in Table 4.13, surprisingly, when z_y is not sampled (i.e., z_y is set to $\mu_{y,z}$, the mean of the posterior $q(z_y|y)$), the NLG performance is better than that of sampling z_y. This implies that although sampling z_y introduces language variety, it comes with the risk of precision loss of its semantics.

4.6 Related Work

Literature of modelling NLU or NLG alone is discussed in Section 2.2 and Section 2.5 respectively. By having the two models trained separately, pseudo data of paired text and semantics can be generated and added to the original training corpus as augmented data to boost the model performance (Kedzie and Mckeown, 2019; Nie et al., 2019). There has also been recent work which models the two tasks jointly. Both Ye et al. (2019) and Cao

et al. (2019) explored the duality of semantic parsing and NLG. The former optimised two seq2seq models using dual information maximisation, while the latter introduced a dual learning framework for semantic parsing. Su et al. (2019b) proposed a learning framework for dual supervised learning Xia et al. (2017), where both NLU and NLG models are optimised towards a joint objective. Their method brings benefits with annotated data in supervised learning, but does not allow semi-supervised learning with unlabelled data. The back-translation concept (Sennrich et al., 2016) has been also explored in the joint learning of NLU and NLG. In Qader et al. (2019), two seq2seq models are learned for each task and the connection of the two models can be used to handle the unlabelled data. In contrast to these prior work, the proposed generative model couples NLU and NLG with latent variables. We focus on exploring a coupled representation space between natural language and corresponding semantic annotations.

4.7 Conclusion

In this chapter, we presented a generative model which couples natural language and formal representations via the two close latent spaces. Since the two data sources are coupled, we gain the luxury of exploiting each unpaired data source and transferring the acquired knowledge to the latent spaces, which eventually benefits both NLU and NLG modelling. Extensive experiments and analyses demonstrate that our model is highly effective in a low-resource regime, which reduces the high demand on annotated pairs of natural language and semantic representations.

Chapter 5

Semi-supervised Bootstrapping of Dialogue State Trackers

Dialogue systems benefit greatly from optimizing on detailed annotations, such as dialogue state representations and dialogue act labels. However, collecting these annotations is expensive and time-consuming. In this chapter, we investigate semi-supervised learning methods and their applications to end-to-end dialogue modelling (Section 2.7). We found that these methods enable dialogue systems to effectively leverage un-annotated data and to reach equivalent system performance as if labelled data was available. This work was published in Tseng et al. (2019b):

- **Tseng, B. H.**, Rei, M., Budzianowski, P., Turner, R., Byrne, B., & Korhonen, A. "Semi-Supervised Bootstrapping of Dialogue State Trackers for Task-Oriented Modelling". *In proc of EMNLP 2019*.

5.1 Motivation

Training modern dialogue systems requires significant amounts of annotated data to obtain satisfactory performance. However, collecting labels (e.g., dialogue states and acts) for each dialogue turn is not a trivial task because it often requires domain and expert knowledge to obtain high quality data (Asri et al., 2017). Due to this restriction, existing datasets for task-oriented dialogue are several orders of magnitude smaller relative to general open-domain dialogue corpora (Henderson et al., 2019; Lowe et al., 2015).

Arguably, one of the most challenging parts of dialogue modelling is maintaining an interpretable internal memory over crucial domain concepts, i.e., tracking dialogue state (Williams et al., 2016). Although there is increasing research effort to learn dialogue state

tracking (DST) jointly with the text generation component (Eric et al., 2017; Wu et al., 2019b), the most effective models use DST as an intermediate signal in joint modelling of dialogue components[1] (Hosseini-Asl et al., 2020; Lei et al., 2018; Wen et al., 2017b). The difficulty of state tracking has made this task a driving force behind most of the Dialog System Technology Challenges in recent years (Henderson et al., 2014b; Kim et al., 2017).

In this chapter, we aim to reduce the high demand for fine-grained dialogue annotations by leveraging semi-supervised training. Two approaches, initially proposed for the task of image classification (LeCun et al., 1998; Russakovsky et al., 2015) in the field of computer vision, were investigated and evaluated for providing an improved training signal to the DST component in an end-to-end dialogue system. One approach is *pseudo-labelling* (Lee, 2013), where the automatically predicted DST output on unlabelled utterances is treated as additional annotation if the model confidence is sufficiently high. Another approach is the Π-*model* (Laine and Aila, 2017), which regularises the network outputs to be as close as possible given the same input during training. Differences caused in model outputs can come from stochasticity in model (e.g., dropout) or randomness in input data (e.g., Gaussian noise).

The main contribution of this chapter is two-fold: (1) an end-to-end dialogue system with joint modelling of DST, policy and NLG is proposed; (2) the two aforementioned semi-supervised learning methods are investigated and adopted to the scenario of dialouge modelling, with the aim of reducing DST annotations. Our analysis on the MultiWOZ corpus (Budzianowski et al., 2018) finds that these methods can effectively reduce the needed amounts of DST annotations by 30% to 50% while maintaining equivalent system performance.

5.2 End-to-end Neural Dialogue Model

We now present the end-to-end neural dialogue model, which is composed of three main components: dialogue state tracker, policy network and natural language generation (NLG) module. These components are jointly trained as a connected network and will be introduced in detail below. The overall architecture can be seen in Figure 5.1, with the DST model emphasised as the focus here is the reduction of DST annotations.

[1] As mentioned in Section 2.7, the approach of joint optimisation of dialogue components, including DST, policy and NLG, is often named "end-to-end". We therefore use this term in this chapter.

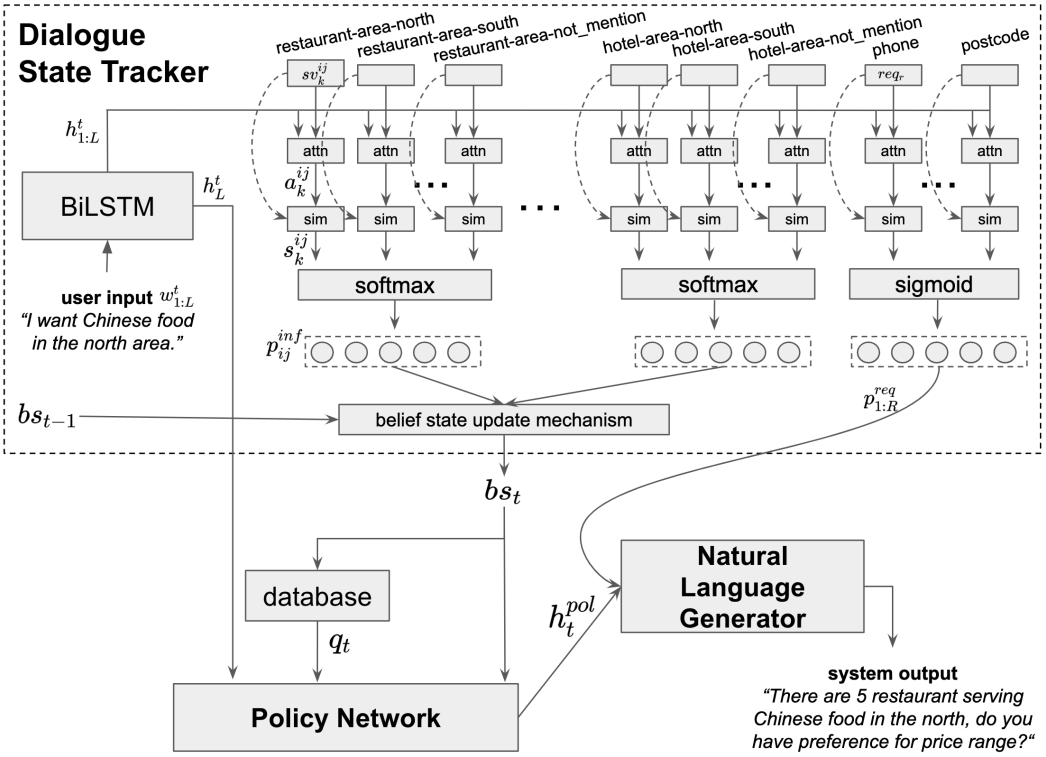


Fig. 5.1 Overview of our end-to-end neural dialogue model, comprised of three main components: dialogue state tracker, policy network and natural language generator. In the dialogue state tracker, *attn* and *sim* denote the attention operation and the similarity computation of two vectors respectively.

5.2.1 Dialogue State Tracker

As introduced in Section 2.3, dialogue systems should have a comprehensive understanding of both the current user utterance and the dialogue context. The DST module is typically used to maintain a belief state by tracking information across turns. The presented DST module consumes the user utterance and predicts distributions over values for each domain-slot pair. The resulting distributions and the previous belief state are used to obtain the belief state for the current turn.

Let i, j and k denote the index of domain, slot and value. As depicted at the top of Figure 5.1, the user utterance, with L tokens $\{w_1{:}w_L\}$, at dialogue turn t is first encoded by a BiLSTM (Hochreiter and Schmidhuber, 1997) to obtain the hidden states $h^t_{1:L}$. Each slot-value pair sv^{ij}_k is encoded using a feedforward neural network that takes the input the concatenation of domain i, slot j and value k embeddings. The encoded vector of each slot-value pair is used as query to attend over the user utterance by the attention mechanism

(*attn*) (Luong et al., 2015) to obtain the context vector a_k^{ij}:

$$e_l = \text{sim}(h_l, sv_k^{ij}) \tag{5.1}$$

$$a_k^{ij} = \sum_{l=1}^{L} e_l h_l, \tag{5.2}$$

where l is the token index of the user utterance and `sim` denotes a score function that calculates the similarity of two vectors. Here we use the dot product as a score function. The likelihood of a slot-value pair s_k^{ij} being presented in the input utterance is then computed as the similarity score between a_k^{ij} and sv_k^{ij}. The mentioned slot-value pair should have a higher similarity score to its context vector than those which are not mentioned. Finally, for each informable slot s_{ij}^{inf}, a softmax layer is applied to normalise the output vector into a probability distribution p_{ij}^{inf}. The value of a slot can be determined by taking *argmax* on the obtained distribution. The same decoding mechanism is applied for each requestable slot to predict whether a slot has been requested in the current user turn. A sigmoid layer is used instead as it is a binary classification problem. The prediction of requestable slots will be used as input to the natural language generator.

Thus far, the model predicts slot-value pairs solely based on the input user utterance. To consider context information, the belief state bs_t for the current turn t is obtained using a defined heuristics, similar to the practice in Mrkšić et al. (2017). For each informable slot, if it is predicted as not mentioned in the user utterance (i.e., the predicted value is either `not mentioned` or `don't care`), the value distribution obtained at previous turns is carried over to this turn. Otherwise, the value distribution obtained at current turn is used.

It is worth noting that although our DST model is a classification based method, while this type of modelling approach has some disadvantages (e.g., it cannot handle arbitrary value sets), as introduced in Section 2.3, it is naturally more suitable for us to apply semi-supervised learning techniques proposed for image classification tasks, compared to a generation based DST model.

5.2.2 Policy and Natural Language Generation

We use a policy network to fuse signals from the belief state $\mathbf{bs}_t$, the encoding of the user utterance $\mathbf{h}_L^t$ and the database result $\mathbf{q}_t$. The database result is constructed based on the number of matched entities against the belief state.[2] We use a feed-forward layer as the

[2]Following Wen et al. (2017b), by querying the database using $\mathbf{bs}_t$ as query, $\mathbf{q}_t$ is a 1-hot representation with each element indicating the number of matched entities in the database. We use 5 bins to indicate the matched number from 0 to 3 and more than 3.

policy network:

$$\mathbf{z}_t = \tanh(W^z[\mathbf{bs}_t \oplus \mathbf{h}_L^t \oplus \mathbf{q}_t]). \tag{5.3}$$

where $\oplus$ denotes the concatenation of vectors.

The NLG model, implemented using an LSTM, then consumes both $\mathbf{z}_t$ (from the policy network) and predictions of requestable slots (from the DST model) as a hidden state initialisation to generate a system response.

5.2.3 Supervised Learning

The model is jointly optimized against two learning signals: DST labels and system utterances. The DST loss consists of cross-entropy over the multi-class classification of informable slots and binary cross-entropy for requestable slots:

$$L_{dst} = -\sum_i \sum_j t_{ij}^{inf} \log p_{ij}^{inf} - \sum_r t_r^{req} \log p_r^{req}, \tag{5.4}$$

where i, j and r are the index of domain, informable slot and requestable slot respectively; t^{inf} and t^{req} denote the corresponding target distributions. The generation error is the cross-entropy between distributions of the predicted word and the ground truth:

$$L_{gen} = -\sum_{l'} t_{l'} \log p_{l'}, \tag{5.5}$$

where l' is the word index in the generated sentence and $t_{l'}$ is the ground-truth 1-hot distribution. To jointly train the DST module, the policy network and the generator as a connected network, the final objective used is $L = L_{dst} + L_{gen}$.

5.2.4 Semi-supervised Learning

The DST loss requires each dialogue turn to be manually annotated with the correct slot-value pairs, which makes it difficult to collect such data at large scale. Here we experiment with two semi-supervised training methods that take advantage of unlabelled examples instead, allowing us to reduce the amount of required annotation.

The first approach is the *pseudo-labelling* (Lee, 2013). The idea is that if the prediction probability of an unlabelled data point for a particular class is larger than a given threshold ν, the predicted (*pseudo*) label is used as if it were a true label. Lee (2013) showed that this simple method is effective on leveraging unlabelled image data and further improves image classification accuracy on the MNIST dataset (LeCun et al., 1998). In our experiments, the

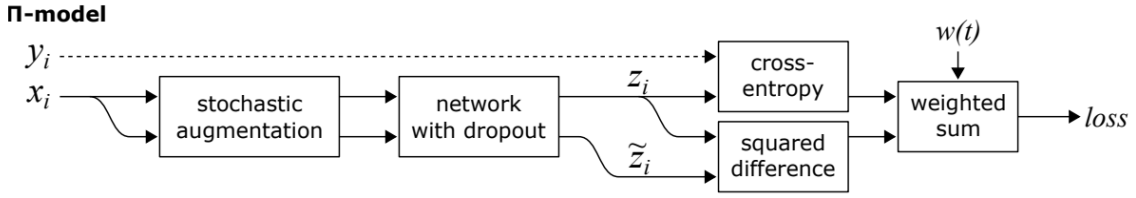


Fig. 5.2 Structure of the Π model, extracted from Laine and Aila (2017).

input unlabelled user utterances were first evaluated using a pre-trained DST model. If a slot-value pair was predicted with a likelihood higher than the threshold ν, it was included in the training set for the next round model training. The parameter ν is optimized against the validation set during training.

The second approach is the Π-*model* (Laine and Aila, 2017). As shown in Figure 5.2, a data point x_i is evaluated twice using a neural network to obtain output vectors z_i and $\widetilde{z}_i$ respectively. These two model outputs are not guaranteed to be the same because of dropout regularisation or data noise. This difference between outputs given the same data point can be thought of as a prediction error, and therefore minimising it is a reasonable objective. Mean square error between z_i and $\widetilde{z}_i$ is used together with the cross-entropy to train the network in a semi-supervised learning manner. Laine and Aila (2017) showed that the Π-*model* significantly reduced the amount of labelled images needed, and obtained comparable performance to 100% supervised training across several benchmarks such as the CIFAR-10 (Krizhevsky et al., 2010) and the SVHN (Netzer et al., 2011) datasets. Connections can be drawn with adversarial training (Goodfellow et al., 2014; Miyato et al., 2015; Szegedy et al., 2013), which requires the model to be robust to both original and adversarial image inputs. Miyato et al. (2016) further extended this technique to the task of text classification and applied adversarial perturbations on continuous word embeddings to RNNs. One key difference between adversarial training and the Π-*model* is that the former seeks the worse perturbations to the model training (i.e., a perturbation that causes the largest increase in training loss), while the latter adds a random noise to the inputs.

In our implementation of Π-*model*, we perturbed the word embeddings to the BiLSTM used in our DST model (see Figure 5.1) with a Gaussian noise:

$$\begin{aligned} \varepsilon &\sim \mathcal{N}(\mathbf{0}, \sigma) \\ E'(x_t) &= E(x_t) + \varepsilon \end{aligned} \tag{5.6}$$

where $E(.)$ is the word embedding function and x_t is the one-hot vector of the input word at the step t. We then require the model to be robust to this perturbation by having an additional

regularisation term defined as follows:

$$L_{\Pi} = \sum_{ij} ||\widetilde{p}_{ij}^{inf} - p_{ij}^{inf}||^2, \tag{5.7}$$

where p_{ij}^{inf} is the predictive distribution over slot values for slot j in domain i, while $\widetilde{p}_{ij}^{inf}$ is its counterpart but having a perturbation to the input per Equation 5.6. Note that this introduced loss does not require training labels, which allows us to perform semi-supervised learning. Finally, the DST model is trained using the loss $L = L_{dst} + \alpha L_{\Pi}$ where α is a hyper-parameter tuned on the validation set.

5.3 Experiments

We investigated the effects of semi-supervised training on optimizing the end-to-end neural dialogue system. In particular, we evaluated how much annotation of the dialogue state at the intermediate level could be reduced while preserving comparable overall system performance. Experiments were conducted on the MultiWOZ dataset (Budzianowski et al., 2018), which consists of 10k multi-domain task-oriented dialogues. The size of the dataset allows us to control the amount of available fully labelled data points.

As mentioned in Section 2.8, there are two metrics of importance when evaluating task-oriented dialogue systems. The first is the DST joint goal accuracy, defined as an average joint accuracy over all slots per turn (Williams et al., 2016). The second is the *success rate* that informs how many times systems have presented the entity satisfying the user's goal and provided them with all the additional requested information (Wen et al., 2017b). The models were optimized using the validation set and the results were averaged over 10 different random seeds.

5.3.1 Results

We examine the performance of our model trained only using labelled data (baseline) and the two model variants using semi-supervised techniques, as the amount of labelled data varies. The results of the DST joint accuracy is presented in Figure 5.3 (left). The *pseudo-labelling* model performs better than the baseline when more than 50% of the dataset is labelled. At the scarce data levels (10% and 30%) , the *pseudo-labelling* model is not producing pseudo training points that help improve DST predictions. In contrast, the Π-*model* takes advantages of the additional regularization loss and effectively leverages unlabelled data to enhance the performance over the baseline. The improvements are consistent in joint accuracy with a 5%

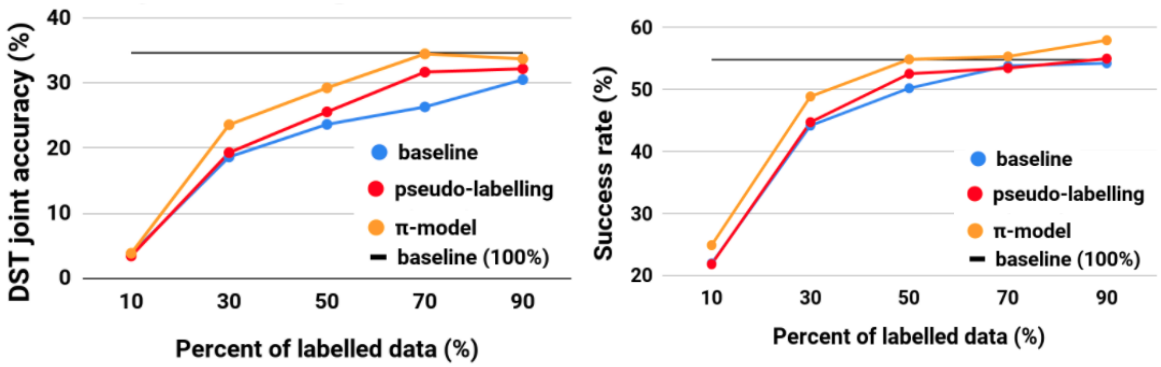


Fig. 5.3 Results of DST joint accuracy (left) and success rate (right) for the three considered models, as the amount of labelled data varies. The horizontal line denotes the baseline model trained on 100% labelled data.

margin when labelled data is more than 30%. The Π-*model* even reaches the performance of the fully trained baseline model with only 70% labelled data.

Figure 5.3 (right) shows the results in success rate. The *pseudo-labelling* method is not able to improve performance over the baseline regardless of the amount of labelled data. However, the Π-*model* is capable of improving the success rate consistently and manages to reach the performance of the fully trained model with only 50% of the intermediate DST signal. Note that a better DST joint accuracy does not necessarily translate to a better success rate as the final metric is also influenced by the quality of the generator.

5.3.2 Analysis

Results over slot-value pairs

DST joint accuracy considers all slot-value pairs in an utterance and cannot give us further insight regarding the source of the improvements. We are particularly interested in whether the semi-supervised models can leverage unlabelled data to improve the prediction of rarely seen slot-value pairs. In this analysis, we classify all slot-value pairs in the test set in terms of their number of training examples in the setup of 50% labelled data. Table 5.1 presents the accuracy of slot-value pairs of each group. We can see that the Π-*model* shows a consistent improvement over the baseline model across all groups, and provides a better generalisation to low frequent slot-value pairs (1-10 times seen during training) with a 5% gap in accuracy. The improvement on few-shot examples contributes to the improvement of joint accuracy reported in Figure 5.3.

No. of examples	0	1-5	6-10	10-15	16-20
Baseline	6.17	15.93	25.71	35.07	28.88
Pseudo-labelling	6.50	16.27	26.96	33.46	28.55
Π-model	**6.60**	**21.93**	**31.29**	**36.22**	**30.72**

Table 5.1 The accuracy (%) of different groups of slot-value pairs in terms of their number of training examples.

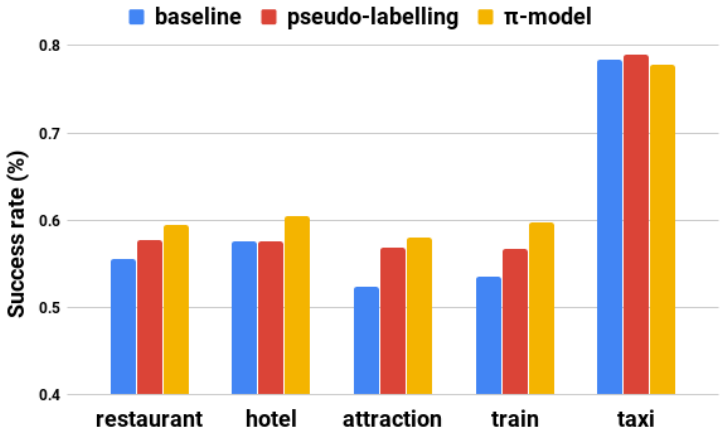


Fig. 5.4 Success rates on each domain in the case of 50% labelled data.

Results over domains

We also investigate if the improvements in the success rate are consistent among all domains. Figure 5.4 shows the success rate on individual domains in the case of 50% labelled data. In general, both semi-supervised models improve performance over the baseline in all domains except for the taxi domain. We hypothesize this comes from the fact that the taxi domain is a relatively easy domain with only 4 possible slots. Among the four complicated and difficult domains, the Π-*model* performs the best with a large performance gap compared to the baseline system, which, once again, verifies the efficacy of the Π-*model* in dialogue modelling.

5.4 Conclusions

In this chapter, we presented an end-to-end dialogue system and explored two semi-supervised learning techniques, the *pseudo-labelling* and the Π-*model*, in dialogue modelling. The aim is to reduce the complex DST annotations through leveraging unlabelled data. The results suggest that we do not need to annotate the dialogue states for the complete dataset and are

able to reach the comparable success rate using only 50% labelled data, which is a significant number when training on the MultiWOZ dataset.

Chapter 6

Transferable Dialogue Systems and User Simulators

As noted in the previous chapter, one of the challenges in dialogue modelling is the lack of training data. In this chapter, we explore the possibility of creating dialogue data through the interaction between a dialogue system and a user simulator. The goal here is to develop a modelling framework that can incorporate new dialogue scenarios through interaction between the two agents. In this framework, we first pre-train the two agents on a collection of source domain dialogues, which equips the agents to converse with each other via natural language. With further fine-tuning on a small amount of target domain data, the agents continue to interact with the aim of improving their behaviors using reinforcement learning with structured reward functions. In our experiments, two practical transfer learning problems are investigated: domain adaptation and single-to-multiple domain transfer. We demonstrate that the proposed framework is highly effective in bootstrapping the performance of the two agents in transfer learning. We also show that our method leads to improvements when training on the whole dataset compared with prior work. This work was published in Tseng et al. (2021b)

- **Tseng, B. H.**, Dai, Y., Kreyssig, F., & Byrne, B. "Transferable Dialogue Systems and User Simulators". *In proc of ACL 2021*.

6.1 Motivation

One of the challenges in task-oriented dialogue modelling is to obtain adequate and relevant training data. A practical approach in moving to a new domain is via transfer learning, where pre-training on a general domain with rich data is first performed and then fine-tuning the

model on the target domain. End-to-end dialogue systems (Dhingra et al., 2017; Li et al., 2017a; Wen et al., 2017b), as introduced in Section 2.7, are particularly suitable for transfer learning, in that such models are optimised as a single system. By comparison, pipelined dialogue systems with multiple individual components (Young et al., 2013) require fine-tuning of each component. These separate steps can be done independently, but it becomes difficult to ensure optimality of the overall system (Takanobu et al., 2020b).

A similar problem arises in data-driven user simulators, which are commonly used in interaction with the dialogue system. Though many user simulators have been proposed and widely studied, they usually operate at the level of semantic representation (El Asri et al., 2016; Kreyssig et al., 2018). These models can capture user intent, but are otherwise somewhat artificial as user simulators in that they do not consume or produce natural language. As discussed above for dialogue systems, the end-to-end architecture for the user simulator also offers simplicity in transfer learning across domains.

There are also potential advantages in continued joint training of a dialogue system and a user simulator. If a user model is less than perfectly optimised after supervised learning over a fixed training corpus, further learning through interaction between the two agents offers the dialogue system the opportunity to refine its behavior. Prior work has shown benefits from this approach to dialogue policy learning, with a higher success rate at dialogue level (Liu and Lane, 2017; Papangelis et al., 2019; Takanobu et al., 2020a), but there has not been previous work that addresses multi-domain end-to-end dialogue modelling for both agents. Takanobu et al. (2020a) address refinement of the dialogue policy alone at the semantic level, but do not touch end-to-end system architectures. Liu and Lane (2017); Papangelis et al. (2019) address single-domain dialogues (Henderson et al., 2014a), instead of the more realistic and complex multi-domain dialogues.

In this chapter, we propose a novel learning framework for developing dialogue systems that performs **J**oint **O**ptimisation with a **U**ser **S**imula**T**or (**JOUST**). Through pre-training on a complex multi-domain dataset, the two agents are able to interact using natural language, and further create more diverse and rich dialogues. Using reinforcement learning (RL) to optimise both agents enables them to depart from known strategies learned from a fixed limited corpus, and to explore new, potentially better policies. Importantly, the end-to-end design in the framework makes it easier for transfer learning of two agents from one domain to another. We also investigate and compare two reward designs within this framework: 1) the common choice of task success at dialogue level; 2) a fine-grained reward that operates at turn level. Results on the MultiWOZ dataset (Budzianowski et al., 2018) show that our method is effective in boosting the performance of dialogue systems in complicated multi-domain

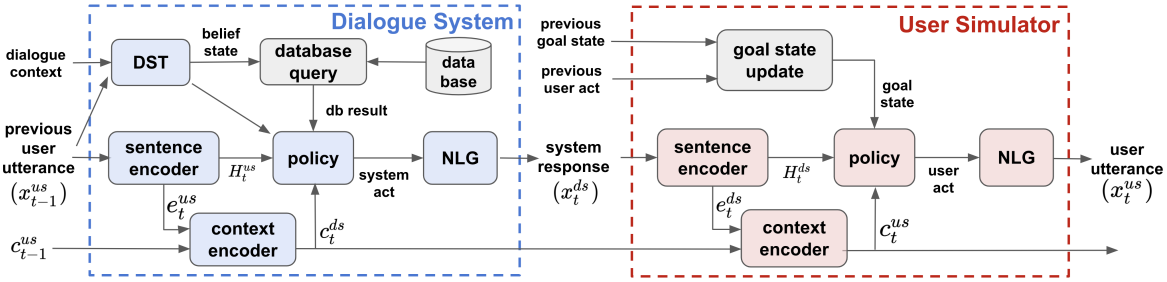


Fig. 6.1 Overall architecture of the proposed framework, where the dialogue system (*ds*) and user simulator (*us*) converse with each other at the t^{th} dialogue turn. The context encoder is shared between the two agents.

conversation. To further test our method in more realistic scenarios, we design specific experiments on two low-resource setups that address different aspects of data sparsity.

6.2 Model Architecture and Pre-training

In our joint learning framework, we first pre-train both the dialogue system and user simulator using supervised learning so that two models are able to interact via natural language. This section presents the architectures of two agents, as shown in Figure 6.1, and the objectives used for supervised learning.

6.2.1 Dialogue System

Dialogue state tracking (DST)

As discussed in Chapter 2, the first task of a dialogue system is to process the dialogue history in order to maintain the belief state which records essential information of the dialogue. A DST model is utilized to predict the set of slot-value pairs which constitute the constraints of the entity for which the user is looking for, e.g., {`hotel_area=north`, `hotel_name=gonville_hotel`}.

Different from the classification based design of the DST model presented in Chapter 5, the DST model used here is a generation based model which adopts an encoder-decoder architecture with the attention mechanism (Bahdanau et al., 2015). The set of slot-value pairs in a belief state is formulated as a slot sequence together with a value sequence. For the t^{th} dialogue turn, the DST model first encodes the dialogue context and the most recent user utterance x^{us}_{t-1} using a bi-directional LSTM (Hochreiter and Schmidhuber, 1997) to obtain hidden states $H^{enc}_t = \{h^{enc}_1, ..., h^{enc}_j, ...\}$. At the i^{th} decoding step, the previous decoder hidden

state h_{i-1}^{dst} is used to attend over H_t^{enc} to obtain the attention vector a_i^{dst}. The decoder takes as input a_i^{dst}, h_{i-1}^{dst} and $x_{s,i}$, the embedding of the slot token predicted at the $i-1^{\text{th}}$ step, to produce the current hidden state h_i^{dst}. The vector h_i^{dst} is then passed through affine transforms followed by the softmax function to predict the distributions over slots and values for step i.

$$\begin{aligned} a_i^{dst} &= \text{attention}(H_t^{enc}, h_{i-1}^{dst}) \\ h_i^{dst} &= \text{LSTM}^{dst}(a_i^{dst}, h_{i-1}^{dst}, x_{s,i}) \\ p_{s,i}, p_{v,i} &= \text{softmax}(f_{\text{affine}}(h_i^{dst})) \end{aligned} \tag{6.1}$$

The final belief state is the aggregation of predicted slot-value pairs of all decoding steps.

Database Query

Based on the current belief state, the system searches the database and retrieves the matched entities at every turn. We use a one-hot vector of size 3 to characterise the result of database query: 1) no matched entities exist; 2) only one matched entity exists and 3) multiple matched entities exist.

Context Encoding

To better capture the dialogue flow, we use a hierarchical LSTM (Serban et al., 2016) to encode the dialogue context from turn to turn throughout the dialogue. At each turn t, the most recent user utterance x_{t-1}^{us} is encoded by an LSTM-based sentence encoder[1] to obtain a sentence embedding e_t^{us} and hidden states H_t^{us}. Another LSTM is used as the context encoder, which encodes e_t^{us} as well as the output of the context encoder on the user side c_{t-1}^{us} from the previous turn (see Figure 6.1). The context encoder produces the next dialogue context state c_t^{ds} for the downstream dialogue manager.

Policy

The dialogue manager determines the system dialogue act based on the current state of the dialogue. Here the system dialogue act is treated as a sequence of act tokens in order to handle the concurrent actions in a single turn. For instance, when the system informs the user of several hotels in the north and also requests the desired star rating, the system dialogue act is presented as `{act_inform hotel_choice hotel_area act_request hotel_star}`. The problem can be therefore formulated as a sequence generation task using an LSTM

[1] Note that this sentence encoder is separate from the encoder inside the DST model as the former takes the input only one user utterance, while the latter consumes the whole dialogue context.

based decoder. At the i^{th} decoding step, the input to the policy decoder includes: 1) $x_{a,i}$, the embedding of the act token predicted at the previous step; 2) h_{i-1}^{pol}, the previous hidden state; 3) a_i^{pol}, the attention vector obtained by attending over the hidden states of the user utterance H_t^{us}; 4) v_i^{db}, the database retrieval vector; 5) v_i^{bs}, the summarized belief state, which is a binary vector where each entry corresponds to a domain-slot pair. The distribution over act tokens is then generated using the following equations:

$$\begin{aligned} a_i^{pol} &= \text{attention}(H_t^{us}, h_{i-1}^{pol}) \\ h_i^{pol} &= \text{LSTM}^{pol}(x_{a,i}, h_{i-1}^{pol}, a_i^{pol}, v_i^{db}, v_i^{bs}) \\ p_{a,i} &= \text{softmax}(f_{\text{affine}}(h_i^{pol})) \end{aligned} \tag{6.2}$$

For better modeling of the dialogue flow, the context information is injected by initialising the hidden state with the context state c_t^{ds} obtained by the hierarchical LSTM.

Natural language generation (NLG)

The final task is to generate the system response based on the predicted dialogue act, and we use another LSTM to perform that. During decoding, the attention mechanism is performed between the policy and the NLG models. That is to say, at the i^{th} decoding step, the hidden state of the NLG LSTM, (h_i^{nlg}) attends over the hidden states of the policy LSTM (H^{pol}). The resulting attention vector (a_i^{nlg}) and the embedding of the previous output word (x_i^{nlg}) are the inputs to the NLG model and the model outputs the distribution of *delexicalized* tokens ($p_{w,i}$) .[2]

$$\begin{aligned} a_i^{nlg} &= \text{attention}(H^{pol}, h_{i-1}^{nlg}) \\ h_i^{nlg} &= \text{LSTM}^{nlg}(x_{w,i}, a_i^{nlg}) \\ p_{w,i} &= \text{softmax}(f_{\text{affine}}(h_i^{nlg})) \end{aligned} \tag{6.3}$$

These *delexicalized* tokens are then replaced by the retrieval results to form the final utterance.

6.2.2 User Simulator

As in the dialogue system, the proposed user simulator is comprised of a dialogue manager, an NLG model and a dialogue context encoder. However, in place of a DST to maintain a belief state, the user simulator possesses an internal goal state to track progress towards satisfying the user goal.

[2] As described in Section 2.5, the slot values are replaced with slot placeholders in delexicalised utterances.

Goal State

To perform a coherent interaction with the dialogue agent, the user simulator should be aware of the requested information and unfulfilled goal constraints based on the dialogue context. In our design, a goal state is modelled to reflect the user status up to the current turn. The goal state is implemented using a binary vector, where each entry of the vector corresponds to a domain-slot pair in the ontology. At the start of a dialogue, goal state entries are turned on for the slots that are specified in the user goal. At each dialogue turn, the goal state is updated deterministically based on the previous user dialogue act: if a slot appears in the previous dialogue act, either as a constraint or as a request, the corresponding entry is turned off. In such a way, this goal state serves as a signal of goal satisfaction to the downstream components to produce appropriate user acts and utterances along the entire dialogue session.

Context Encoding, Policy and Natural Language Generation

These steps follow their implementations in the dialogue system. For context encoding in the user model, a sentence encoder first encodes the system response using an LSTM to obtain hidden states H_t^{ds} and a sentence embedding e_t^{ds}. The context encoder then takes e_t^{ds} and the system context state c_t^{ds} as inputs to produce the user context state c_t^{us}, which is passed to the dialogue agent at the next turn.

Also as in the dialogue system, the policy and the NLG models of the user simulator are implemented using LSTMs. The input to the policy are the goal state, hidden states of the sentence encoder H_t^{ds} and the context state c_t^{us}, and the output is the user dialogue act, represented as a sequence of act tokens as in the dialogue agent. The NLG model takes the hidden states of the policy decoder as input to generate the user utterance, which is then *lexicalised* by replacing *delexicalised* placeholder tokens using the user goal.

6.2.3 Supervised Learning

For each dialogue turn, the ground truth dialogue acts and the output word sequences are used as supervision for training both two agents. The objectives of the policy and the NLG models are the cross-entropy between the probability of predicted sequence p and the ground-truth vectors y:

$$
\begin{aligned}
L_{pol}^{*} &= \sum_{i=1}^{|A|} -y_{a,i}^{*} \log p_{a,i}^{*} \\
L_{nlg}^{*} &= \sum_{i=1}^{|W|} -y_{w,i}^{*} \log p_{w,i}^{*}
\end{aligned}
\tag{6.4}
$$

where the superscript $*$ can be either ds or us, referring either to the dialogue system or the user simulator: e.g., $p^{ds}_{a,i}$ is the probability of the act token at the i^{th} decoding step on the system side. The ground-truth vector y contains both word sequences and act sequences with W and A as their lengths.

The DST annotations are also used as supervision for training the dialogue agent. The loss of the DST model is defined as the sum of the cross-entropy for sequences of both slot and value:

$$L^{ds}_{dst} = \sum_{i=1}^{|SV|} -y^{ds}_{s,i} \log p^{ds}_{s,i} - y^{ds}_{v,i} \log p^{ds}_{v,i} \tag{6.5}$$

where $|SV|$ is the number of slot-value pairs in a turn; i is the decoding step index. $p^{ds}_{s,i}$ and $p^{ds}_{v,i}$ are the probabilities of slot and value predictions at the i^{th} step.

The overall objectives are as:

$$\begin{aligned} L^{ds}(\theta^{ds}) &= L^{ds}_{dst} + L^{ds}_{pol} + L^{ds}_{nlg} \\ L^{us}(\theta^{us}) &= L^{us}_{pol} + L^{us}_{nlg} \end{aligned} \tag{6.6}$$

where θ^{ds} and θ^{us} are parameters of the dialogue agent and the user model respectively. The two agents are optimised jointly to minimize the sum of the losses ($L^{ds}+L^{us}$). The success rate of the dialogues generated from thee two agents' interaction is used as the stopping criterion during supervised training.

6.3 Reinforcement Learning

After pre-training from the corpus using supervised learning, both agents are fine-tuned using reinforcement learning (RL) based on the dialogues generated during their interactions. Two reward designs are presented after which the optimisation strategy is given.

6.3.1 Dialogue-Level Reward

Following a common practice (Casanueva et al., 2018; El Asri et al., 2014; Su et al., 2017; Zhao et al., 2019), the success of the simulated dialogue is used as the reward, which can only be observed at the end of the dialogue session. A small negative value as penalty is given at each turn to discourage lengthy dialogues. During training, the reward is shared between the two agents.

6.3.2 Turn-Level Reward

While the dialogue-level reward is straightforward, it only considers the final task success rate of the dialogues and neglects the quality of the individual turns. For complex multi-domain dialogues, this sparse reward will make it difficult for the system to learn the relationship between actions and rewards, resulting in a credit assignment problem (Fu and Anderson, 2008; Sutton, 1984). We thus propose a turn-level reward function that encapsulates the desired behavioural features of the fundamental dialogue task. The rewards are designed separately for the user simulator and the dialogue system according to their characteristics.

Dialogue System Reward

A good dialogue system should learn to refine the search by requesting needs from the user and providing the correct entities, with their attributes, that the user wishes to know. Therefore, at each turn, a positive reward is assigned to the dialogue system if: 1) it requests slots that had not been requested before; 2) it provides an correct entity; or 3) it answers precisely all additional requests by the user. Otherwise, a negative reward is given.

User Simulator Reward

A good user model should not repeatedly provide the same information or request attributes that have already been informed by the dialogue agent. Therefore, a positive reward is assigned to the user simulator if: 1) it provides slots specified in the user goal without repetition; 2) it requests attributes about an entity without repetition, or 3) it replies correctly to a request by the dialogue agent. Otherwise a negative reward is given. Note that all the rewards aforementioned are real values that are tuned using grid search during development. Details are provided in Section 6.4.2.

6.3.3 Optimization

We apply the *Policy Gradient Theorem* (Sutton et al., 2000) to the space of user/system dialogue acts. In the t^{th} dialogue turn, the reward r_t^{ds} or r_t^{us} is assigned to the two agents at the last step of their generated act sequence. The return for the action at the i^{th} step is

$$R_i^* = \gamma^{|A^*|-i} r_t^* \tag{6.7}$$

where $*$ denotes *ds* or *us*, and $|A^*|$ is the length of the act sequence of each agent. $\gamma \in [0,1]$ is usually set as a discounting factor. Note that we do not use reward discounting in our implementation as we find this works well in experiments. Each action is assigned with the

same return in an action sequence $R_i^* = r_t^*$. The policy gradient of each turn can then be written as:

$$\nabla_{\theta^*} J^*(\theta^*) = \sum_{i}^{|A^*|} R_i^* \nabla_{\theta^*} \log p_{a,i}^* \tag{6.8}$$

where $p_{a,i}^*$ is the probability of the act token at the i^{th} step in the predicted dialogue act sequence. The two agents are updated using Equation 6.8 at each turn within the entire simulated dialogue.

6.4 Experiments

6.4.1 Dataset

The MultiWOZ dataset (Budzianowski et al., 2018) is used for our experiments. Detailed description and the statistics of the dataset can be found in Section 3.3.1. We use the additional user dialogue act labels annotation provided by Lee et al. (2019).

6.4.2 Training Details

Both the dialogue system and the user simulator are trained in an end-to-end fashion using the Adam optimizer (Kingma and Ba, 2014). The sizes of the embedding and of the hidden layers are set to 300. During supervised training the batch size is 100 and the learning rate is 0.001, while during RL they are 10 and 0.0001 respectively for stability. The turn-level rewards in Section 6.3 are tuned in the range of [-5, 5] based on the development set using grid search. As for dialogue-level rewards, a positive reward 1.0 is given if a dialogue is successful; a negative reward -0.1 is assigned to each turn to discourage lengthy dialogues. The user goals employed for interaction during RL are taken from the training data without synthesizing new goals.

6.4.3 Evaluation Metrics

The proposed model is evaluated in terms of the inform rate (Info), the success rate (Succ), and BLEU, as introduced in Section 2.8.[3] Following Mehri et al. (2019), the combined performance (Comb) is reported, calculated as $0.5 * (\text{Info} + \text{Succ}) + \text{BLEU}$.

[3]Note that the numbers reported on MultiWOZ could be not comparable due to different pre-processing or evaluation scripts (Nekvinda and Dušek, 2021). For a fair comparison to previously proposed models, we use the standard evaluation script provided by the MultiWOZ organizers (https://github.com/budzianowski/multiwoz) and the official data split for train/dev/test sets with the same pre-processing.

Model	Info	Succ
(a) Supervised Learning	69.77	58.02
(b) RL-DS w/ dial-R	81.38	70.67
(c) RL-Joint w/ dial-R	82.83	71.57
(d) RL-DS w/ turn-R	85.62	70.34
(e) RL-Joint w/ turn-R	**86.49**	**73.04**

Table 6.1 Quality for dialogues generated by two agents in JOUST using the test corpus user goals. *DS* denotes only the dialogue system is optimised during reinforcement learning (RL), while *Joint* optimises the user simulator together. *dial-R* and *turn-R* refer two reward types respectively introduced in Section 6.3. BLEU is not reported since no reference sentences are available for these interactions.

6.4.4 Interaction Quality

First, it is examined whether the proposed learning framework improves the discourse between the dialogue system and user simulator. Several variants of our model are investigated: 1) two agents were pre-trained using supervised learning, serving as a baseline; 2) RL was used to fine-tune only the dialogue system (RL-DS) or both agents (RL-Joint). In each RL case, we can either use rewards at the dialogue level (dial-R, Section 6.3.1) or rewards at the turn-level (turn-R, Section 6.3.2). The two trained agents interacted based on 1k user goals from the test corpus, with the generated dialogues being evaluated using the metrics above.

From Table 6.1, we can see that the application of RL in our framework improves the success rate by more than 10% (b-e vs. a). This indicates that the dialogue agent learns to be better at completing the task successfully through interaction with the trained user agent and through the designed rewards. Moreover, the joint optimisation of both agents provides dialogues with a higher success rate than optimising the dialogue system only (c & e vs. b & d). It shows that the behaviour of the user model is realistic and diverse enough to interact with the system and can be improved together during RL optimisation. Finally, by comparing the two reward designs, the fine-grained rewards at the turn level seem to be more effective towards guiding two agents' interaction (b & c vs. d & e), which is reasonable since they reflect more than a simple success rate in terms of the nature of the tasks.

Table 6.2 provides a dialogue pair example that is generated by the two agents' interaction[4] using supervised learning (model (a) in Table 6.1) and using RL (model (e) in Table 6.1) respectively. As seen from the top dialogue, since the SL user model gives a wrong answer to the system's request, the system keeps asking the unsolved question, leading to an unsuccess-

[4]Generated utterances are lexicalised with corresponding slot values (belief state for the system; user goal for the user). If the system generates attributes that are not grounded in the belief state, the slot placeholders are kept in the output sentence, e.g., the placeholder "@train-departure" in the top dialogue of Table 6.1.

ful dialogue with a loop (highlighted in color). Note that there are interaction examples where the SL models fail to escape the repetition loop; it is somewhat unclear how the SL system did not repeat to ask the question on the turn 19-sys. In contrast, the interaction between RL models (the bottom dialogue) is much smoother and efficient since both agents answer each other correctly. This indicates that the designed rewards are useful for improving the agents' behaviors. The corresponding error analysis of both agents is provided later in Section 6.4.7.

User Goal
Train: destination: cambridge; day: wednesday; arriveBy: 18:15; departure: kings lynn; request=[trainID, price, leaveAt]
Attraction: name: club salsa; request=[address]
Dialogue generated by two agents trained using SL
(Dialogue has a loop and fails as the provided train is incorrect due to the wrong departure)
00-usr: i am looking for a train that leaves on wednesday and arrives at cambridge .
01-sys: there are few trains that meet your criteria . what time would you like to leave ?
02-usr: i would like to arrive by 18:15 .
03-sys: there are few trains leaving @train-departure to cambridge on wednesday . what time would you like to leave ?
04-usr: i would like to arrive by 18:15 .
05-sys: there are few trains leaving @train-departure to cambridge . is there a particular time you would like to leave ?
06-usr: i would like to arrive by 18:15 .
07-sys: there are few trains leaving @train-departure to cambridge . is there a particular time you would like to leave ?
08-usr: i would like to arrive by 18:15 .
09-sys: there are few trains leaving @train-departure to cambridge . is there a particular time you would like to leave ?
10-usr: i would like to arrive by 18:15 .
11-sys: there are few trains leaving @train-departure to cambridge . what time would you like to leave ?
12-usr: i would like to arrive by 18:15 .
13-sys: there are few trains leaving @train-departure to cambridge . what time would you like to leave ?
14-usr: i would like to arrive by 18:15 .
15-sys: there are few trains leaving @train-departure to cambridge . what time would you like to leave ?
16-usr: i would like to arrive by 18:15 .
17-sys: there are few trains leaving @train-departure to cambridge . what time would you like to leave ?
18-usr: i would like to arrive by 18:15 .
19-sys: TR0269 arrives at 18:15 . would you like me to book it for you ?
20-usr: yes , please . i need the price and travel time .
21-sys: the train id is TR0269 , the price is 10 and the travel time is 50 .
22-usr: i am also looking for a particular attraction called club salsa .
23-sys: club salsa is a nightclub in the centre . the address is 1 station road . entrance is free . is there anything else i can help you with ?
24-usr: that is all i need . thank you for your help .
25-sys: thank you for using our system !
Dialogue generated by two agents trained using RL
(Dialogue is smooth and successful)
00-usr: i am looking for a train from kings lynn on wednesday .
01-sys: what is your destination ?
02-usr: i need to arrive in cambridge by 18:15 .
03-sys: i have train TR0415 that leaves at 17:11 and arrives at 17:58 . would you like me to book it for you ?
04-usr: yes , please book 2 tickets .
05-sys: booking was successful , the total fee is 9.80 pounds payable at the station . reference number is fztwszhh . is there anything else i can help you with ?
06-usr: yes , i am also looking for a particular attraction called club salsa .
07-sys: club salsa is located at 1 station road , postcode cb12jb . is there anything else i can help you with ?
08-usr: no , that s all i need today . thank you .
09-sys: thank you for using our service . have a great day !

Table 6.2 Example of dialogue pairs: two dialogues generated by the SL system and the RL system respectively. The undesirable dialogue loop with repeated information is highlighted.

Model	Info	Succ	BLEU	Comb
HRED-TS (Peng et al., 2019)	70.0	58.0	17.5	81.5
DAMD (Zhang et al., 2019)	76.3	60.4	16.6	85.0
SimpleTOD*(Hosseini-Asl et al., 2020)	84.4	70.1	15.0	92.3
SOLOIST* (Peng et al., 2020a)	**85.5**	72.9	16.5	95.7
MinTL-BART* (Lin et al., 2020)	84.9	**74.9**	**17.9**	**97.8**
JOUST Supervised Learning	77.4	66.7	17.4	89.5
JOUST RL-Joint w/ dial-R	80.6	69.4	17.5	92.5
JOUST RL-Joint w/ turn-R	**83.2**	**73.5**	**17.6**	**96.0**

Table 6.3 Empirical comparison with state-of-the-art dialogue systems using the predicted belief state. * indicates leveraging of pre-trained transfomer-based models.

Model	Info	Succ	BLEU	Comb
SimpleTOD*(Hosseini-Asl et al., 2020)	88.9	67.1	16.9	94.9
MoGNet (Pei et al., 2020)	85.3	73.3	20.1	99.4
ARDM* (Wu et al., 2019c)	87.4	72.8	20.6	100.7
DAMD (Zhang et al., 2019)	89.2	77.9	18.6	102.2
SOLOIST* (Peng et al., 2020a)	89.6	**79.3**	18.3	102.5
PARG (Gao et al., 2020)	91.1	78.9	18.8	103.8
MarCo* (Wang et al., 2020)	**92.3**	78.6	**20.0**	**105.5**
JOUST Supervised Learning	88.5	79.4	18.3	102.3
JOUST RL-Joint w/ dial-R	93.9	85.7	16.9	106.7
JOUST RL-Joint w/ turn-R	**94.7**	**86.7**	**18.7**	**109.4**

Table 6.4 Empirical comparison with state-of-the-art dialogue systems using the oracle belief state. * indicates leveraging of pre-trained transfomer-based models.

6.4.5 Benchmark Results

We conduct experiments on the official test set for comparison to existing end-to-end dialogue systems. The trained systems are used to interact with the fixed test corpus following the same setup of Budzianowski et al. (2018). Results are reported using a predicted belief state (Table 6.3) and using an oracle belief state (Table 6.4). In general, we can observe similar performance trends as in Section 6.4.4. Joint learning of two agents using RL with the fine-grained rewards reaches the best combined score and success rate. This implies that the exploration of more dialogue states and actions in the simulated interactions reinforces the behaviors that lead to improved performances, and that these generalise well to unfamiliar states encountered in the test corpus.

Our best RL model produces competitive results in Table 6.3 when using the predicted belief state, and can further outperform the previous work in Table 6.4 when using the oracle belief state. Note that we do not leverage the powerful pre-trained transformer-based models as in SOLOIST (Peng et al., 2020a) or MinTL-BART model (Lin et al., 2020). We found

Model	Info.	Succ.	BLEU	Comb.
Predicted Belief State				
Supervised Learning	70.37	55.43	17.29	80.19
RL-Joint w/ turn-R	**74.83**	**60.60**	**17.41**	**85.12**
Oracle Belief State				
Supervised Learning	89.67	74.5	16.96	99.04
RL-Joint w/ turn-R	**94.27**	**81.47**	**17.20**	**105.06**

Table 6.5 Results of JOUST using 50% training data in supervised learning.

that with RL optimisation, our LSTM-based models can still perform competitively. In terms of the model structure, the most similar work would be the DAMD model (Zhang et al., 2019). The performance gain found in comparing *JOUST Supervised Learning* to DAMD is partially due to the better performance of our DST model.[5]

We also conduct experiments using only 50% of the training data for supervised learning to verify the efficacy of the proposed method under different amounts of data. As shown in Table 6.5, it is observed that our method also improves the model upon supervised learning when trained with less data and the improvements are consistent with the complete data scenario.

6.4.6 Transfer Learning

In this section, we demonstrate the transfer learning capability of the proposed framework under two low-resource setups: Domain Adaptation and Single-to-Multiple Domain Transfer. Two fine-tuning methods are adopted: straightforward fine-tuning without any constraints (Naive) and elastic weight consolidation (EWC) (Kirkpatrick et al., 2017). We show that the proposed RL can be further applied to both methods and produces significantly improved results. Here we experiment with the best RL variants using turn-level rewards (same as model (e) in Table 6.1). The evaluation script used in transfer learning is the same as the one used in benchmark results (Section 6.4.5).

Domain Adaptation

In these experiments, each of five domains is selected as the target domain. Taking the *hotel* domain for example, 300 dialogues[6] involving the hotel domain are sampled from the training corpus as adaptation data. The rest of the dialogues, without the hotel domain

[5] In correspondence, the DAMD authors report a DST model with a joint accuracy of 35%, while ours is 45%.

[6] For each domain, 300 dialogues accounts for 10% of all target-domain data.

Model	Restaurant	Hotel	Attraction	Train	Taxi	Avg.
Predicted Belief State						
Source	21.1	28.6	25.2	59.6	48.7	36.6
Naive	46.7	56.2	66.1	68.5	66.3	60.8
EWC	56.7	58.2	71.6	69.3	78.7	66.9
Naive+RL	57.0	66.8	72.5	**72.3**	75.4	68.8
EWC+RL	**64.6**	**67.8**	**75.8**	71.6	**87.6**	**73.5**
Oracle Belief State						
Source	33.2	40.1	34.3	70.7	55.4	46.7
Naive	85.6	84.2	77.9	96.7	93.4	87.5
EWC	84.1	85.1	89.8	101.7	97.5	91.6
Naive+RL	**97.6**	99.2	88.5	104.0	103.4	98.5
EWC+RL	97.5	**100.7**	**96.0**	**104.9**	**106.3**	**101.1**

Table 6.6 Combined scores in domain adaptation. 300 dialogues are used for each target domain adaptation.

involved, form the source data. Both the dialogue system and the user simulator are first trained on the source data (Source), and then are fine-tuned on the limited data of the target domain (Naive or EWC). Afterwards, the pair of agents is trained in interaction using the proposed RL training regime (+RL).

Results in terms of the combined score are given in Table 6.6. As expected, models pre-trained on source domains obtain low results on target domains without fine-tuning. Fine-tuning using the Naive or EWC method considerably bootstraps the systems, where the regularization of the EWC method benefits more for the low-resource training. By applying our proposed RL framework to the two sets of fine-tuned models, the performance can be further improved by 7-10% on average, with both predicted and oracle belief states. This indicates that through the interaction with the user model, the dialogue system is not constrained by having seen only a very limited amount of target domain data, and that it can learn effectively from the simulated dialogues using the simple reward structure. Interestingly, although the EWC models are shown to be more robust to the original task/domain in continual learning due to their superiority of combating catastrophic forgetting, we find that they also provide better performances on the adaptation domain. On average, the final performance obtained by the EWC+RL model doubles that of *Source* model, which demonstrates the efficacy of the proposed method in domain adaptation.

Single-to-Multiple Domain Transfer

Another transfer learning scenario is investigated where only limited multi-domain data is accessible but sufficient numbers of single-domain dialogues are available. This setup

Model	H+T	R+T	A+T	A+H+X	H+R+X	A+R+X	Avg.
Predicted Belief State							
Source	46.0	55.4	34.3	22.0	26.6	19.9	34.0
Naive	57.2	69.2	65.0	40.3	36.0	42.8	51.7
EWC	57.4	72.1	66.1	43.7	39.0	45.0	53.9
Naive+RL	63.2	74.4	**68.4**	**47.4**	42.7	**48.7**	57.5
EWC+RL	**64.7**	**77.6**	67.6	46.6	**43.2**	48.5	**58.0**
Oracle Belief State							
Source	82.3	93.3	76.2	36.8	55.4	42.4	64.4
Naive	88.8	98.4	85.9	72.2	79.8	76.7	83.6
EWC	95.5	96.9	89.6	70.0	81.5	79.6	85.5
Naive+RL	99.7	**104.3**	92.0	80.6	**97.2**	**89.3**	93.9
EWC+RL	**100.2**	103.0	**93.9**	**82.6**	95.0	89.2	**94.0**

Table 6.7 Combined scores in single-to-multiple domain transfer where 100 dialogues on each target scenario are used for adaptation. R, H, A, T, X represent Restaurant, Hotel, Attraction, Train, Taxi domain.

is based on a practical fact that single-domain dialogues are often easier to collect than multi-domain ones. All single-domain dialogues in the training set form the source data. For each target multi-domain combination, 100 dialogues[7] are sampled as adaptation data. As before, both agents are first pre-trained on the source data and then fine-tuned on the adaptation data. Afterwards, the two agents improve themselves through interaction. The models are tested using the multi-domain dialogues of the test corpus.

Results in terms of of the combined score are reported in Table 6.7. Although the *Source* models capture individual domains, they cannot manage the complex flow of multi-domain dialogues and hence produce poor combined scores, with worst results on combinations of three domains. Fine-tuning improves performance significantly, as the systems learn to transition between domains in the multi-domain dialogue. Finally, applying our RL optimization further increases performance by 6-9% on average. This indicates that the dialogue agents can learn more complicated policies through exploring more dialogue states and actions while interacting with the user simulator. We analyse the sources of improvements in the following section.

[7]There are 6 types of domain combinations in MultiWOZ, as shown in Table 6.7. For each multi-domain combination, 100 dialogues accounts for 11% of its multi-domain data.

Model	Dialogue System		User Simulator	
	Miss Ent.	Wrong Ans.	Rep. Att.	Miss Ans.
Naive	17.59	36.99	10.12	47.27
Naive+RL	**2.73**	**9.54**	**1.47**	**32.60**

Table 6.8 Error analysis (%) of the both agents averaged over 5 adaptation domains. Lower is better.

6.4.7 Analysis

Error Analysis

We first investigate the two agents' behaviors to gain insights into the model improvement in transfer learning. Two models, one fine-tuned (Naive) and one further trained using RL (Naive+RL), are examined here. For the dialogue system, rates of missing entities (Miss Ent.) and of wrong answers (Wrong Ans.) are reported. For the user simulator, rates of repetitions of attributes (Rep. Att.) and of missing answers (Miss Ans.) are reported. The results, shown in Table 6.8, are averaged over the five adaptation domains. We can see that with RL optimisation errors made by the two agents are reduced significantly, which, again, shows the efficacy of the designed rewards at turn level and verifies improvements observed in Section 6.4.6. Notably, the user model learns not to repeat the information already provided and attempts to answer more of the questions from the dialogue agent. These are the behaviors the reward structure of Section 6.3.2 is intended to encourage, and they lead to more successful interactions in policy learning.

Exploration of Dialogue States and Actions

We now investigate whether our framework encourages exploration through increased interaction in transfer learning. We report the number of unique belief states in the training corpus and in the dialogues generated during RL interaction that the dialogue system experienced, as well as the number of unique action sequences per state that the agent predicted.

As shown in Table 6.9, the dialogue model encounters more states in interaction with the user model and also adopts more unique actions in reinforcement learning relative to what it sees in supervised learning. In this way the system considers additional strategies during the dialogue simulation, with the opportunity to reach better performance even with only limited supervised data.

	Domain adaptation		Single-to-Multiple	
	states	actions	states	actions
Corpus	614	3.34	223	3.61
Interact.	**1425**	**6.22**	**399**	**15.33**

Table 6.9 Number of unique dialogue states and average dialogue actions per state in the training corpus and in the RL interactions in two transfer learning setups.

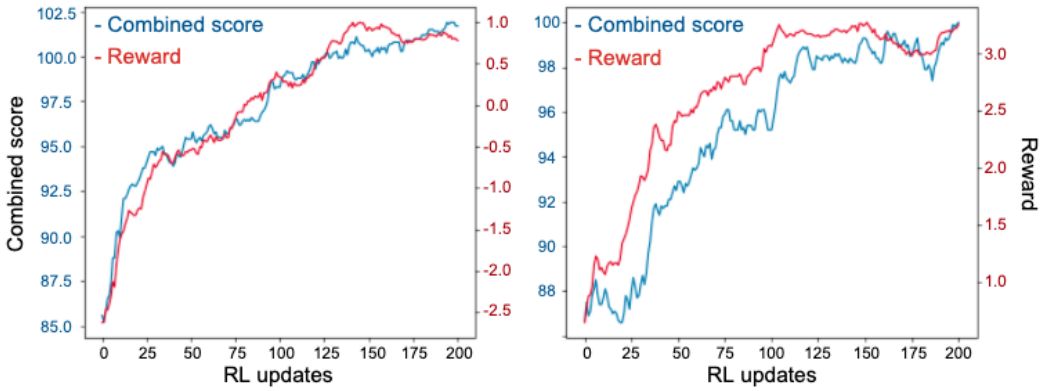


Fig. 6.2 Learning curves observed on the development set during RL optimization. Two domain adaptation cases are presented, with restaurant (left) and hotel (right) as target domain respectively.

RL Learning Curve

Here we show that the designed reward structure is indeed a useful objective for training. Figure 6.2 shows learning curves of the model performance and the received (turn-level) rewards during RL training. The two examples are from the domain adaptation experiments in Section 6.4.6, where restaurant (left) and hotel (right) are the target domains. We can see that both the reward value and model performance are consistently improved during RL, and their high correlation verifies the efficacy of the proposed reward design for training task-oriented dialogue systems.

6.4.8 Human Evaluation

A human assessment of dialogue quality is performed to confirm the improvements in metric based evaluation. 400 dialogues, generated by the two trained agents, are evaluated by 14 human assessors who are graduate students in different scientific fields. Each assessor is shown a comparison of two dialogues where one dialogue is generated by the models using supervised learning and another is generated by the models with RL optimization. Note that here we are evaluating the performance gain in interactions between two agents

Win Ratio (%)	SL	RL
DS Success	26.0	**74.0**
US Human-like	29.5	**70.5**
Dialogue Flow	21.0	**79.0**

Table 6.10 Human assessment of the interaction quality generated by supervised learning models (SL) and reinforcement learning models (RL). Numbers reported are human's preference (win ratio) of the two models' outputs with respect to the three criteria.

(Section 6.4.4), instead of the gain in benchmark results by interacting with the static corpus (Section 6.4.5). The assessor offers judgement regarding:

- Which dialogue system completes the task more successfully (DS Success)?
- Which user simulator behaves more like a real human user (US Human-like)?
- Which dialogue is more natural, fluent and efficient (Dialogue Flow)?

The results with relative win ratio are shown in Table 6.10. With the proposed RL optimisation, the dialogue agent is more successful in dialogue completion. More importantly, joint optimisation with the user simulator is found to produce more human-like behavior. The improvement under the two agents leads to a more natural and efficient dialogue flow, which is consistent with the metric based evaluation, as observed in Table 6.1.

6.5 Conclusion

In this chapter, we tackled the issue of the lack of training data in dialogue modelling. A novel joint learning framework of training both the dialogue agent and the user simulator for complex multi-domain dialogues via RL is presented. Under low-resource scenarios, the two agents can generate additional dialogue data through interacting with each other, and their behaviors can be significantly improved using RL through the two agents' interaction. Two types of reward are investigated and the turn-level reward performs better due to its fine-grained structure. Experiments shows that our framework produces comparable to state-of-the-art dialogue systems on the MultiWOZ dataset. In the two practical transfer learning setups, our method can further improve the well-performing EWC models and boosts the final performance substantially.

Chapter 7

Combined Resolution of Ellipsis and Anaphora in Dialogues

As discussed in Section 2.3, anaphora and ellipsis are two common phenomena in dialogues. Without resolving referring expressions and omitted information, dialogue systems may fail to generate consistent and coherent responses. Traditionally, anaphora is resolved by coreference resolution and ellipsis by query rewrite. In this chapter, we propose a novel multi-task learning framework for modelling coreference resolution and query rewriting in complex, multi-turn dialogue understanding. Given an ongoing dialogue between a user and a dialogue assistant, for the user query, our model first predicts coreference links between the query and the dialogue context, and then generates a self-contained rewritten user query. To train and evaluate our model, we have annotated a dialogue based coreference resolution dataset, MuDoCo (Martin et al., 2020), with rewritten queries. This work was published at (Tseng et al., 2021a):

- **Tseng, B. H.**, Bhargava, S., Lu, J., Moniz, J. R. A., Piraviperumal, D., Li, L., & Yu, H. "CREAD: Combined Resolution of Ellipses and Anaphora in Dialogues". *In Proc of NAACL 2021*.

7.1 Motivation

Despite rapid progress in the development of dialogue systems, several difficulties remain in the understanding of complex, multi-turn conversations. Two major problems are anaphora resolution (Clark and Manning, 2016a,b) and ellipsis (Kumar and Joshi, 2016) in follow-up turns. Take the dialogue in Figure 7.1 as an example: ellipsis happens in the `user turn 2` where the user is asking for the capital of *"Costa Rica"* without explicitly mentioning the

(Turn 1)
User: *Who is the president of Costa Rica?*
System: *Carlos Alvarado Quesada is the president of Costa Rica.*

(Turn 2)
User: *And what is the capital?*
[Rewrite: *And what is the capital of Costa Rica?]*
System: *San Jose is the capital of Costa Rica.*

(Turn 3)
User: *What is the population of the capital?*
[Rewrite: *What is the population of San Jose?]*
System (i): *The population of San Jose is 1.03 million.* (California)
System (ii): *The population of San Jose is 342 thousands.* (Costa Rica)

Fig. 7.1 A dialogue example where coreference and ellipsis happen in user queries, along with the corresponding query rewrite annotations. References to the same entity are highlighted in the same color and can be resolved by coreference resolution. The two system responses in Turn 3 indicate two possible interpretations of the city *San Jose* by the dialogue system.

country again; coreference happens in the `user turn 3` where *"the capital"* refers to *"San Jose"*. Without resolving the anaphoric reference and the ellipsis, dialogue systems may fail to generate coherent responses.

Query rewrite (Quan et al., 2019; Rastogi et al., 2019; Su et al., 2019a) is an approach that converts a context-dependent user query into a self-contained utterance so that it can be understood and executed independent of previous dialogue context. This technique can solve many cases where coreference or ellipsis happens. For instance, "*the capital*" in the `user turn 3` is changed to *"San Jose"* in the rewrite. Furthermore, the ellipsis of the country name *"Costa Rica"* in the `user turn 2` can be revealed through rewriting. This technique provides a more complete user utterance, which alleviates the difficulty of multi-turn dialogue understanding for task-oriented dialogues (Yang et al., 2019).

Although query rewrite implicitly resolves coreference resolution, there is information still missing from the rewritten utterances. First, rewriting does not provide a distinct coreference link between mentions[1] across dialogue turns as in the classic coreference resolution task (Lee et al., 2017; Ng, 2010). This is particularly disadvantageous when there is entity ambiguity in the rewritten sentence. For example, in Figure 7.1, since *"San Jose"* in the `Rewrite turn 3` can be either San Jose in Costa Rica or San Jose in California, it is possible that the system ends up with an incorrect response by generating System Response

[1] In the typical coreference resolution task, mentions of named entities/events in text could be pronouns, noun phrase (NP) or verb phrases (VP) (Pradhan et al., 2012).

(i) instead of System Response (ii) due to the wrong interpretation of San Jose. Second, mention detection, an essential step in coreference resolution (Peng et al., 2015) where mentions of entities are identified, is not involved in query rewrite. By knowing which span in an utterance is a mention, downstream systems such as intent detection can perform better (Bikel et al., 2009).

To resolve the above issues, we propose a novel multi-task learning framework that incorporates the benefits of reference resolution into the query rewrite task. To the best of our knowledge, there does not exist, at the time of writing, an English conversation dataset that couples annotations of both query rewrite and coreference resolution (as links or clusters). This motivates us to collect annotations of query rewrite on a recent dialogue dataset - MuDoCo (Martin et al., 2020), which has been labelled with coreference links between the user query and dialogue context. Compared to existing query rewrite datasets (Anantha et al., 2020; Quan et al., 2019), rewriting in MuDoCo is much more challenging since it involves reasoning over multiple turns and spans multiple domains. We design a multi-task learning model based on the GPT-2 (Radford et al., 2019) architecture, which learns both query rewrite and coreference resolution simultaneously. Given an ongoing dialogue, our model first predicts the coreference links, if any, between the latest user query and the dialogue context. It then generates the rewritten query by drawing upon the coreference results. Our experiments show that query rewrite performance can be substantially boosted with the aid of coreference resolution modelling. In addition, our model outperforms strong baselines for the two individual tasks, indicating the efficacy of the proposed multi-task learning framework.

7.2 Dataset and Task

The MuDoCo dataset contains 7.5k task-oriented multi-turn dialogues across 6 domains (calling, messaging, music, news, reminders and weather). A dialogue has an average of 2.6 turns and a maximum of 5 (where a turn includes a user query and a system response). Figure 7.2 shows an example. For each partial dialogue, the coreference links, if present, are annotated between the latest user query and its dialogue context. For example, when we consider the partial dialogue up to the `user turn 2` in Figure 7.2, there is a coreference link between the anaphor "*this*" in the `user turn 2` and the antecedent "*song*" in the `user turn 1`. When an anaphor has multiple antecedents in the context, e.g., both "*song*" (in the `user turn 1`) and "*Yellow Submarine*" (in the `system turn 1`) are antecedents to the anaphor "*this*", only one of them is annotated in the coreference link.

On top of the existing coreference labels, we annotate the rewrite for each utterance. The goal is to rewrite the query into a self-contained sentence that is independent of the dialogue

(Turn 1)
User: *Play the Beatles song about submarines.*
System: *Yellow Submarine by the Beatles playing on YouTube Music.*

(Turn 2)
User: *What album is this from?*
[Rewrite: *What album is Yellow Submarine by the Beatles from?*]
System: *Yellow Submarine is from the album Yellow Submarine.*

(Turn 3)
User: *Can I hear the next song on the album?*
[Rewrite: *Can I hear the next song on the Yellow Submarine album?*]
System: *Sure, Only A Northern Song on the album now playing.*

Fig. 7.2 An example from the MuDoCo dataset in the music domain with our query rewrite annotation. Word spans in the same color belong to the same mention cluster.

context. 30 annotators are recruited for the data collection.[2] Each of them is shown a partial dialogue and is asked (1) to decide if the query needs to be rewritten due to coreference or ellipsis; and (2) to provide the rewritten query, when rewriting is required. We notice that there can be various ways of rewriting an utterance. For example, some annotators might include every detail of the rewritten entity, while others might choose a precise term; some might paraphrase the rewritten utterance, while others keep the same expression. To ensure data consistency and high annotation quality, we designed a comprehensive guideline for the annotators to follow and undertook a two-stage collection process. First, we organized two training sessions with annotators. In each session, 50 representative examples were selected and assigned to each annotator. We inspected these training results individually and provided feedback to the annotators. Second, 5% of the annotation results were manually evaluated for quality assurance. Only when the examined annotations are consistent and (almost) correct, we finish the data collection. Otherwise, the process starts again. The detailed annotations guideline can be found in the Appendix B.

The proposed multi-task learning task requires the machine to predict both coreference links and the rewritten query for the latest user query given an ongoing dialogue. The outputs of the two individual tasks complement each other and provide more comprehensive information for dialogue understanding. For instance, the "*Yellow Submarine*" in Figure 7.2 can be either a song name or an album name. Explicit coreference resolution helps to disambiguate between various possibilities by linking entities to previously resolved ones.

[2]The annotators belong to the annotation team in the Apple Inc.

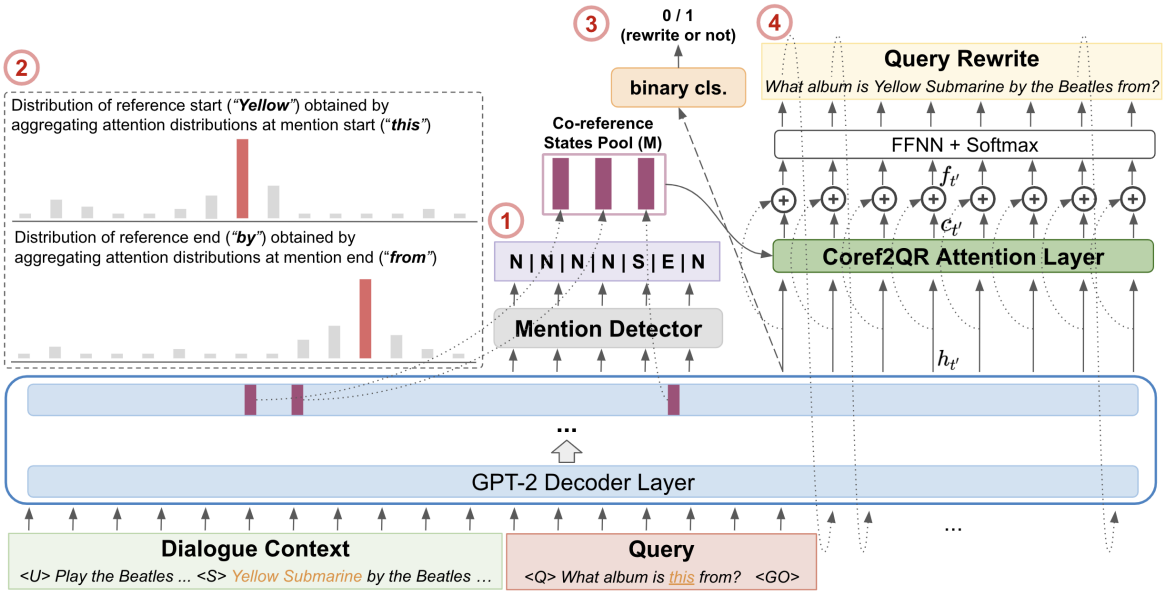


Fig. 7.3 The proposed model for multi-task learning of coreference resolution and query rewrite, designed based on the GPT-2 model. Given a dialogue context and a user query, the model first detects the mentions in the query (Step 1); resolves the corresponding reference spans (Step 2); predicts whether the query needs a rewrite or not (Step 3); and, if the model decides to rewrite, generates the rewritten query (Step 4). In this example dialogue, there is a coreference link existing between the mention "*this*" and its referent "*Yellow Submarine*".

More importantly, the supervision of coreference resolution can be beneficial to rewriting the anaphora to its antecedent.

7.3 Modelling

Our proposed model for learning coreference resolution and query rewrite is designed based on the GPT-2 model (Radford et al., 2019), with its architecture presented in Figure 7.3. The input to the model is the concatenation of the dialogue context and the latest user query, where special tokens are used to separate utterances and indicate speaker information. Passing through the GPT-2 decoder layers, the hidden state $h_t^l \in \mathbb{R}^d$ and attention score $a_t^{l,j} \in \mathbb{R}^T$ at each position of the input sequence are calculated, where l, j and t denote the index of the decoder layer, that of the attention head, and the input token position respectively; d and T denote the embedding size and the length of the input sequence respectively. Inspired by the end-to-end coreference resolution model of Lee et al. (2017), our model first predicts mentions in the user query and grounds them to their corresponding referent in the dialogue context using attention heads. The model then generates the rewritten query conditioned

on the resolved coreference links. The prediction process has four main steps, described in detail below.

7.3.1 Step 1: Mention Detection

First, the model detects any possible referring expressions in the user query. Here we use the term `mention` to include all expressions that require reference resolution (e.g., pronouns or partial entity names). We formulate mention detection as a sequence labeling problem: each token in a query is labelled as one of three classes $\{S, E, N\}$, referring to the *Start* and *End* of a mention span and *None* respectively.[3] This sequence tagger in the mention detector, parameterized by a feed-forward network, takes the hidden states of the query from the last decoder layer as input, and predicts a sequence of class labels. Then the mention spans in the query can be determined by a pair of mention tags (i.e., the start S and end E tokens). For instance, in Figure 7.3 the output for the word token "*this*" is the class S and that of "*from*" is the class E, while the rest in the query are predicted as the class N. We use m_S and m_E to respectively denote the start and end position index of a predicted mention m. We found that there is almost no invalid span predictions (i.e., m_E is predicted before m_S). This is mainly because the mentions in our task have clear patterns (usually pronouns or domain key terms, e.g., "song" in Music domain and "call" in Calling domain), which makes the model easier to capture them.

Note that our tagging mechanism shares the same concept to the Inside-Outside-Beginning (IOB) tagging (Ramshaw and Marcus, 1999), a commonly used technique in many NLP applications such as name entity recognition (Lample et al., 2016; Nadeau and Sekine, 2007). Subtle differences between ours and IOB tagging are: (1) for internal tokens (i.e., tokens between the first and end tokens of a mention), we assign the class *None*, while the IOB tagging uses the *Inside* class, which is the counterpart of our *Start* class; (2) for mentions containing a single token, we still assign an *End* class, while the IOB tagging has not.

7.3.2 Step 2: Reference Resolution

For each detected mention m, the model resolves it to the antecedent (or referent) in the dialogue context by predicting the span boundaries: the position index of the referent start r_S and end r_E. Essentially, the distributions of the boundaries (r_S and r_E) are learned by supervising multiple attention heads associated with the target mention m. In other words, the attention distribution a_{m_S} (the attention score of each position associated with the mention

[3] If a mention has more than two tokens, internal tokens between the *Start* and *End* classes are labelled as *None*.

start m_S) is supervised to focus on the referent start r_S. Similarly, attention scores a_{m_E} associated with the mention end m_E are used to learn the boundary of referent end r_E. Concretely:

$$q_{r_S} = \frac{1}{L'J'} \sum_{l}^{L'} \sum_{j}^{J'} a_{m_S}^{l,j},$$
$$q_{r_E} = \frac{1}{L'J'} \sum_{l}^{L'} \sum_{j}^{J'} a_{m_E}^{l,j}, \tag{7.1}$$

where q_{r_S} and q_{r_E} are the probability distributions of the referent start r_S and end r_E respectively; L' and J' are the specified number of the involved decoder layers and attention heads, e.g., $a_{m_S}^{l,j}$ is the j^{th} attention head from the l^{th} decoder layer associated to the mention start m_S. We then take the argmax of these two boundary distributions to resolve the referent r.[4] Our design of reference resolution effectively leverages the powerful attention mechanism in GPT-2 without adding any extra components for reference resolution.

7.3.3 Step 3: Binary Rewriting Classification

After completing the coreference resolution, the model starts to rewrite the user query. Unlike existing query rewrite systems that directly generate the rewrite given the input, our model, equipped with a binary classifier, predicts whether the incoming query requires to be rewritten before the generation process. As shown in Figure 7.3, the classifier, parameterized by a two-layer feed-forward network followed by a softmax layer, takes as input the hidden state of the first decoding step (the <*GO*> token) and predicts a vector with two entries representing the rewrite and no-rewrite classes. Only when the binary prediction is true, i.e., the classifier predicts the class indicating that a rewrite is required, does the model enter *Step 4* to generate the rewritten query; otherwise, the input query will be directly copied as the output. We show that a well-learned binary classifier with 93% accuracy functions as a filter that helps the model not only minimize the risk of incorrectly rewriting already self-contained queries, but also ease the difficulty in the rest of the generation process to solely focus on how to rewrite incomplete queries.

7.3.4 Step 4: Query Rewrite Generation

In this final step, the model completes the generation based on its binary decision of whether or not to rewrite. Unlike the standard language modelling setup in GPT-2, where the output

[4] It is likely that r_E is predicted before r_S. To handle these invalid span predictions, we first decide r_S by taking the argmax of the distribution q_{r_S}, and then look for a valid r_E with a largest probability in the distribution q_{r_E}.

sequence is generated simply using the last hidden states, we design the Coref2QR attention layer that allows results of coreference resolution to effectively benefit the query rewrite generation.

First, all relevant hidden states of mentions and referents predicted in *Steps 1 & 2* are assembled to form a memory pool M. Note that it is possible for an example to have more than one coreference link. As shown in Figure 7.3, at each time step t' during the rewrite generation, the Coref2QR attention layer, operating as the standard multi-head attention mechanism, takes the hidden state $h_{t'} \in \mathbb{R}^d$ as query to attend over the coreference related states $M \in \mathbb{R}^{d \times n}$ where n is the number of related coreference states:

$$\begin{aligned} \alpha_{t'} &= \text{softmax}(W^Q h_{t'} (W^K M)^T) \in \mathbb{R}^n \\ c_{t'} &= \alpha_{t'} (W^V M)^T \in \mathbb{R}^d \end{aligned} \tag{7.2}$$

where $W^Q \in \mathbb{R}^{d \times d}$, $W^K \in \mathbb{R}^{d \times d}$ and $W^V \in \mathbb{R}^{d \times d}$ are trainable matrices. d is the embedding size as mentioned in Section 7.3.

The resulting attention head $c_{t'}$ is summed with $h_{t'}$ to obtain the feature $f_{t'}$, which is then used to generate the output sequence. This design improves information flow between the two tasks, enabling the model to directly utilize information regarding previously resolved coreferents during the rewrite generation.

The Coref2QR attention can be applied to any arbitrary decoder layer to facilitate a deeper interaction between rewrite and coreference resolution in the model. Formally, at each decoder layer l, the memory pool M^l stores coreference related states produced at layer l. At the generation step t', the Coref2QR layer takes the hidden state $h^l_{t'}$ as query to attend over M^l to obtain $c^l_{t'}$, following Equation 7.2. The final feature $f_{t'}$, used in generation, is then obtained by aggregating information across decoder layers: $f_{t'} = h^L_{t'} + \frac{1}{L}\sum_l c^l_{t'}$. For simplicity, in Figure 7.3 we only illustrate the Coref2QR attention involved with the last decoder layer. Our results and analysis show that this Coref2QR attention design benefits the quality of query rewrite, especially in rewriting an anaphora into its antecedent.

7.3.5 Optimization

For each training example, an input sequence with length T is formed by the concatenation of the dialogue context, the user query and the target query rewrite. Four objectives, corresponding to each step discussed above, are used for training. For mention detection, the objective is the cross-entropy (CE) between the predicted distribution of mention p^m and its

ground-truth vector y^m over the user query:

$$L^M = \sum_{t=q_S}^{q_E} CE(\mathbf{y}_t^m, \mathbf{p}_t^m), \tag{7.3}$$

where q_S and q_E denote the start and end position index of the user query respectively.

For each coreference link n, the loss is calculated using cross-entropy between the predicted distributions of the antecedent boundaries q_n^* and the corresponding ground-truth y_n^*, where the superscript $*$ indicates both the referent start r_S and end r_E. The final loss for reference resolution is the sum of losses over the existing coreference links:

$$L^R = \sum_{n=1}^{N} (CE(\mathbf{y}_n^{r_S}, \mathbf{q}_n^{r_S}) + CE(\mathbf{y}_n^{r_E}, \mathbf{q}_n^{r_E})) \tag{7.4}$$

where N is the number of coreference links in an example. $\mathbf{q}_n^{r_S}$ and $\mathbf{q}_n^{r_E}$ represent the predicted distributions of the reference start r_S and end r_E respectively for the coreference link n. L^R would be 0 if the example does not contain any coreference link.

For query rewrite, the binary classification loss is the binary cross entropy between the prediction vector p^B and the binary rewriting label y^B:

$$L^B = CE(\mathbf{y}^B, \mathbf{p}^B) \tag{7.5}$$

For generation, as in the standard language modelling task, the loss is calculated using the cross-entropy between the predicted distribution p^Q and its ground-truth vector y^Q over the rewritten utterance:

$$L^Q = \sum_{t'=q_E+1}^{T} CE(\mathbf{y}_{t'}^Q, \mathbf{p}_{t'}^Q), \tag{7.6}$$

where t' is the time step in the word sequence of query rewrite, starting after the end of user query until the end of input sequence. Note that L^Q is 0 for examples that do not need rewrite. The final loss is the sum of all these losses:

$$L = L^M + L^R + L^B + L^Q. \tag{7.7}$$

7.4 Experiments

7.4.1 Dataset

As discussed in Section 7.2, experiments are conducted on the MuDoCo dataset and we follow the provided data split[5]. Data from 6 domains are aggregated to form train/development/test sets with 16k, 1.9k and 1.9k examples respectively. Each example contains a dialogue context, a current user query and corresponding annotations of coreference resolution and query rewrite. Statistics for each domain are provided in Table 7.1. Out of all examples that are not the first turn, 64.2% contain coreference links and 43.7% require query rewriting. This makes the task more challenging, as the model also needs to learn when not to rewrite a query and when to predict no coreference links.

7.4.2 Setup

The GPT-2 decoder layers and the output word classification layer in our model are initialized with the pre-trained weights from the GPT-2 small model (Radford et al., 2019). We fine-tune the model using the Adam optimizer (Kingma and Ba, 2014) with a learning rate 5e-05 and batch size 15. The criterion for early stopping is the averaged performance of coreference resolution and query rewrite on the development set. Results are obtained with the average of 5 different random seeds.

[5]https://github.com/facebookresearch/mudoco

Domain	Total	Coref.	Rewrite
Calling	10.7k	4.0k (60.5%)	2.2k (33.7%)
Messaging	3.9k	1.7k (69.0%)	1.0k (41.2%)
Music	2.8k	1.4k (77.7%)	1.4k (76.7%)
News	387	156 (66.4%)	189 (76.6%)
Reminders	1.7k	632 (56.4%)	357 (31.9%)
Weather	254	38 (28.6%)	102 (76.6%)
All	19.8k	8.0k (64.2%)	5.3k (43.7%)

Table 7.1 Total number of examples across six domains in the MuDoCo dataset, with number of examples requiring coreference resolution (Coref.) and those that need query rewrite (Rewrite). Percentages are calculated across all follow-up turns (i.e., excluding the first turn).

7.4.3 Query Rewrite

Evaluation Metrics

The standard BLEU-4 (Papineni et al., 2002) between the generated and the target rewritten sentences are reported. In addition, to highlight the quality of the rewritten parts in generated sentences, following the post-processing in Quan et al. (2019), we measure the F1 score calculated by comparing machine-generated words with ground-truth words for only the ellipsis/coreference parts in the rewritten utterances. In other words, non-rewritten words presented in both the ground-truth and generation are filtered out in calculating F1 score. We also report the percentage of all referents in ground-truth coreference links that were successfully generated in the query rewrite, denoted as reference match (RM). The RM ratio explicitly reflects the quality of coreference resolution in the generated rewritten query.

Baselines

The standard seq-to-seq LSTM based model with attention (seq2seq) and its pointer network (PN; Vinyals et al. (2015)) and pointer-generator network (PG; See et al. (2017)) variants are implemented as baselines. The concatenation of the dialogue context and the query are fed as input, and the output is the target rewrite. The size of the hidden states is 300 and word vectors are initialized with GloVe embeddings (Pennington et al., 2014).

Results

Table 7.2 shows the query rewrite results. The low F1 score and high BLEU score is due to filtering out the non-rewritten repeated tokens in post-processing when calculating F1. This allows us to better evaluate the quality of rewritten parts and to better differentiate between good and bad generation in our task. We find that our multi-task learning model substantially outperforms all LSTM-based seq-to-seq models on all metrics. Although the pointer-generator in LSTMs can effectively copy words from the input to its generation, the powerful transformer architecture with pre-trained weights allows better learning of rewriting patterns.

To fairly investigate the impact of coreference modelling on the generation of query rewrite, we train a variant of our model using only the query rewrite objectives (Equations 7.5 and 7.6), denoted as QR-only model. That is to say, this QR-only model is a GPT-2 model fine-tuned on the task of query rewrite, and the comparison to it gives us the idea how much exactly the introduced design of coreference modelling could benefit. We can see that without coreference resolution, the F1 score drops from 60.2 to 57.9 and the reference match drops

Model	Prec.	Rec.	F1	BLEU	RM
seq2seq model	38.3	29.6	33.4	81.0	54.7
+ pn (Vinyals et al., 2015)	42.4	34.1	37.6	86.0	61.2
+ pg (See et al., 2017)	41.4	39.5	40.4	86.4	63.2
Our QR-only model	58.9	57.1	57.9	89.8	78.7
Our multi-task model	**61.0**	**59.5**	**60.2**	**90.2**	**82.0**

Table 7.2 Query rewrite results in F1, BLEU and reference match rate (RM). QR-only model is our model variant trained using only objectives of query rewrite.

from 82.0 to 78.7. This illustrates the efficacy of leveraging coreference resolution for the task of query rewrite, resulting in an improved ability of rewriting anaphoric expressions in our multi-task learning model. A detailed case study with model predictions is presented in Section 7.4.7.

7.4.4 Coreference Resolution

Evaluation Metrics

The MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), and $CEAF_{\phi_4}$ (Luo, 2005) metrics that are widely-used in coreference resolution task are reported. Here we briefly explain these metrics, and recommend readers to refer to Màrquez et al. (2013) for more details.

Let K and R be the key entity set (annotations) and response entity set (predictions) respectively. **MUC** is a link-based metric and it basically compares the entities defined by the links in the key and response entity sets. The MUC recall and precision are defined as:

$$\begin{aligned} \text{MUC Recall} &= \frac{\sum_{k_i \in K}(|k_i| - p(k_i)))}{\sum_{k_i \in K}(|k_i| - 1)} \\ \text{MUC Precision} &= \frac{\sum_{r_i \in R}(|r_i| - p(r_i)))}{\sum_{r_i \in R}(|r_i| - 1)} \end{aligned} \tag{7.8}$$

where $|k_i|$ is the number of link of entity k_i, and $p(k_i)$ ($p(r_i)$) is the partition that is created by intersecting k_i (r_i) with the corresponding response (key) entities.

B^3 is a mention-based metric and it computes precision and recall for each mention. It essentially reflects, for each mention, how close the key and response entities are. For a

mention m_i, its precision and recall are calculated as:

$$\begin{aligned} \text{B}^3 \text{ Recall}(m_i) &= \frac{|R_{m_i} \cap K_{m_i}|}{|K_{m_i}|} \\ \text{B}^3 \text{ Precision}(m_i) &= \frac{|R_{m_i} \cap K_{m_i}|}{|R_{m_i}|} \end{aligned} \tag{7.9}$$

where R_{m_i} (K_{m_i}) is the response (key) entity of the mention m_i. The overall recall and precision are averaged over all mentions.

The CEAF assumes that there should be a one-to-one mapping (g^*) between the key and response entities. The best alignment is found using the Kuhn-Munkres algorithm, with the defined similarity measure (ϕ) of the two entities. The CEAF recall and precision are defined as:

$$\begin{aligned} \text{CEAF Recall} &= \frac{\sum_{k_i \in K^*} \phi(k_i, g^*(k_i))}{\sum_{k_i \in K} \phi(k_i, k_i)} \\ \text{CEAF Precision} &= \frac{\sum_{k_i \in K^*} \phi(k_i, g^*(k_i))}{\sum_{r_i \in R} \phi(r_i, r_i)} \end{aligned} \tag{7.10}$$

where K^* is the set of key entities that is included in the optimal mapping.

Note that these three metrics are calculated based on coreference clusters and we only have ground-truth annotations for coreference links between mentions and referents. To align the coreference links and clusters, during evaluation we post-process both the ground-truth and model predictions: all the word spans that are identical to the annotated referent in the dialogue context are combined into a cluster so that a link between a mention and a referent can be transformed into a cluster for the standard coreference resolution evaluation.

Baselines

To the best of our knowledge, there is no suitable coreference resolution model that is proposed for dialogues[6]. We therefore experiment with the state-of-the-art models for document-based coreference resolution, including the end-to-end model (Lee et al., 2017, 2018) using BERT (Joshi et al., 2019) or SpanBERT (Joshi et al., 2020). Note that these models can only serve as a reference since they are not specifically designed for dialogue-based coreference resolution. Coreference clusters required for these models' training are as in the post-processing in evaluation as mentioned above.

[6]The baseline presented in Martin et al. (2020) is not compared for two reasons: 1) their setups in training/evaluation is different than ours in many ways, e.g., they only consider complete dialogues; 2) their source code is not released for reproduction.

	MUC			B^3			$CEAF_{\phi_4}$			
	P	R	F1	P	R	F1	P	R	F1	Avg. F1
c2f-coref + BERT (Joshi et al., 2019)	72.2	66.7	69.3	74.5	67.9	71.0	77.7	72.6	75.1	71.8
c2f-coref + SpanBERT (Joshi et al., 2020)	71.7	**71.4**	71.5	73.5	**72.5**	73.0	77.8	74.9	76.3	73.6
Our coref-only model	**78.8**	69.4	**73.8**	**79.6**	71.3	**75.2**	80.7	75.1	77.8	**75.6**
Our multi-task model	78.3	69.4	73.6	79.5	71.2	75.1	**81.1**	**75.1**	**78.0**	**75.6**

Table 7.3 Coreference resolution results. The coref-only model is our model variant trained only using the objectives of coreference resolution.

Results

As seen in Table 7.3, SpanBERT obtains better results than BERT, which is consistent with the findings of Joshi et al. (2020). This is mainly because SpanBERT is better at capturing span information, which facilitates tasks such as coreference resolution where reasoning about relationships between word spans is required. In comparison, our multi-task learning model achieves competitive and even slightly better results. This indicates that the design of our model leveraging attention heads inside GPT-2 is effective at predicting coreference links in dialogues. To test if the supervision of query rewrite affects the optimization of coreference resolution in multi-task learning, we train a model variant using only the objectives for coreference resolution (Equations 7.3 and 7.4), denoted as `coref-only` model. It is observed that the results of the `coref-only` model are very close to that of the proposed model, showing that the addition of coreference resolution in multi-task learning is beneficial to query rewrite without sacrificing the performance of the coreference task itself.

7.4.5 Ablation Study

Here we investigate how the different components in our model contribute to the performance of query rewrite. We remove one component at a time and examine the results of query rewrite. As shown in Table 7.4, without the `coref2qr` attention layer, the performance degrades with a drop of 2.9% F1 and 1.4% RM rate, suggesting the `coref2qr` design is beneficial towards query rewriting. By further removing the supervision of coreference modelling from our multi-task learning model, the model is solely optimized towards the

	Prec.	Rec.	F1	BLEU	RM
complete model	**61.0**	**59.5**	**60.2**	**90.2**	**82.0**
- coref2qr attention	55.5	59.3	57.3	89.3	80.6
- coref. modelling	58.9	57.1	57.9	89.8	78.7
- binary head	54.6	54.4	54.3	88.9	78.9

Table 7.4 Ablation study of our multi-task learning model on query rewrite performance.

Model	Calling		Messaging		Music		All	
	coref.	elp.	coref.	elp.	coref.	elp.	coref.	elp.
seq2seq+pg	56.0	36.2	63.6	36.2	45.5	38.2	52.0	34.5
QR-only	75.4	51.4	77.6	**66.5**	59.8	45.5	69.0	49.3
Multi-task	**78.3**	**52.0**	**81.3**	64.3	**63.1**	**51.1**	**72.1**	**50.9**

Table 7.5 Query rewrite performance, F1 scores, on three major domains and the whole test set (All) with respect to two types of rewriting: coreference (coref.) and ellipsis (elp.).

objectives of query rewrite, and as observed, the quality of rewrites is affected with drops of 2.3% F1 and 3.3% RM, compared to the complete model. These results, again, verify the significance of coreference modelling for the task of query rewrite, especially for rewriting anaphora into its antecedent, reflected in results of RM. Finally, the binary (classification) head plays an essential role in our model, with an accuracy of 93.9%. Without this design, the performance would drop 5.9% F1 (60.2 -> 54.3). This shows that with the binary classification, the model is able to concentrate on how to rewrite the input query if it decides to do so, instead of being confused about whether to rewrite or not during generation process.

7.4.6 Detailed Analysis

In this section we analyse the query rewrite performance on the two types of rewriting: coreference (coref.) and ellipsis (elp.). The F1 scores on three major domains and the whole test set (All) are reported in Table 7.5. The seq2seq+pg model is the baseline seq2seq model with pointer-generator; QR-only model is our model variant but trained without coreference modelling. The overall trend shows that 1) when the dialogue contains coreference, the multi-task learning model is more capable of rewriting the query by leveraging its own coreference prediction; 2) when coreference is not present but the query still needs rewriting on account of information omission, the multi-task model can still perform competitively with the QR-only model.

7.4.7 Case Study

We show several examples of query rewrites generated by different models to provide more insights into the task and into the benefits of multi-task learning. The coreference links predicted by the model are appended after its generated rewrite. Two examples that require coreference resolution in query rewriting are shown in Table 7.6. In the left dialogue, *"the song"* in the user query refers back to *"Talking to the Moon"* mentioned in the first user turn. Both seq2seq+pg and QR-only models fail to generate the correct reference in the rewrite,

Dialogue Context	usr: When was Talking to the Moon by Bruno Mars released? sys: On April 12, 2011. usr: Who produced the song? sys: The Smeezingtons and Bhasker.	usr: I want to send a message. sys: Who do you want to send it to? usr: To Ariana. sys: Ariana Smith or Ariana Taylor?
User Query	Could you play the song for me?	The second one.
Rewrite Label	Could you play **Talking to the Moon** for me?	**Ariana Taylor**.
seq2seq+pg	Could you play **the moon** for me? ✗	**Ariana Smith**. ✗
QR-only	Could you play the song for me? ✗	**Ariana Smith**. ✗
Multi-task	Could you play **Talking to the Moon** for me? (song -> Talking to the Moon) ✓	**Ariana Taylor**. (one -> Ariana Taylor) ✓

Table 7.6 Two coreference examples from the test set with rewrites generated by three models. The rewritten parts in generations are highlighted in bold. The coreference links predicted by the multi-task learning model are appended, presented in the format of (mention -> antecedent).

Dialogue Context	usr: What's the temperature like in Richmond today? sys: The temperature is going to be a warm 85%, but there is a chance of rain.
User Query	What are the chances of rain today?
Rewrite Label	What are the chances of rain today **in Richmond**?
seq2seq+pg	What are the chances of rain today? ✗
QR-only	What are the chances of rain today **in Richmond**? ✓
Multi-task	What are the chances of rain today **in Richmond**? ✓

Table 7.7 An example with ellipsis from the test set, with rewrites generated by three different models. The rewritten parts are highlighted in bold.

probably because of the high complexity of a long dialogue. In contrast, the multi-task learning model not only correctly predicts the coreference link pointing from the mention to its referent in the first turn, but also generates a rewrite perfectly consistent with its coreference prediction. A similar trend can be observed in the right example. Again, the first two models cannot identify which *"Ariana"* to generate, while the multi-task learning model is able to rewrite with the correct one with the aid of coreference modelling.

Table 7.7 shows an ellipsis example. The implicit location in the user query can be recovered through rewriting by both GPT-2 based models, while the LSTM-based model tends to keep the query. This indicates that 1) even with copy mechanism in the pointer-generator, the seq2seq model is not capable enough of handling the difficult information omission in rewriting; 2) the multi-task learning model still performs well on ellipsis, while substantially benefiting in examples involving coreference.

7.5 Conclusion

In this chapter, we have proposed a novel multi-task learning framework for coreference resolution and query rewrite in task-oriented dialogues. Modelling coreference resolution not only complements the missing information in query rewrite, but is also beneficial to rewriting anaphoric expressions. Our multi-task learning model is able to first predict coreference links between the user query and dialogue context, and then to generate the rewritten query based on the resolved coreference results. We show that with the aid of coreference modelling, the performance of query rewrite can be substantially boosted. Furthermore, our model produces competitive results in coreference resolution, compared to state-of-the-art BERT-based systems. We hope that the presented multi-task learning task with the release of our query rewrite annotations on the MuDoCo dataset provides a promising research direction in multi-turn dialogue understanding.

Chapter 8

Conclusions

8.1 Conclusions

The creation of human-like digital assistants has been a long-standing goal of dialogue systems research. To reach this goal, complex systems capable of handling multi-domain or unseen domain dialogues have to be developed. However, collecting training data with fine-grained annotations to develop these systems is non-trivial. Therefore, extending existing systems to new domains or languages has been problematic. An alternative is to train these models with a limited amount of data (i.e., in a low-resource setting), while ensuring that the system still learns effectively. This thesis explores these practical issues in dialogue modelling, especially for natural language understanding and generation, and presents algorithms to combat the data sparsity issue from various perspectives.

We start with the NLG modelling in pipeline dialogue systems and propose a tree-based NLG system for domain adaptation. Compared to the previous systems which encode input dialogue acts as either one-hot vectors or linear sequences, the proposed model is capable of capturing the hierarchical structure of dialogue acts and that facilitates knowledge sharing between the source and target domains, leading to a significant performance improvement in low-resource transfer learning setups. We then explore the duality between the NLG and NLU components and propose a single generative model to accomplish both tasks. The proposed JUG model learns a shared latent variable between the spaces of natural language and formal language (i.e., dialogue acts) and enables predictions of both data types. Furthermore, semi-supervised learning can be achieved in the JUG by leveraging dialogues whose semantic annotations are not aligned to each turns, resulting in large reduction of labelled data required for both NLU and NLG training.

We then study a more a more general setup of dialogue generation where user utterances and dialogue context are considered as inputs. We investigate two semi-supervised training

strategies and their applications to DST training in end-to-end dialogue modelling, showing that they allow significant reduction in DST annotation quantity while preserving system performance. Another outstanding problem in end-to-end training is the insufficient data on new dialogue situations. We propose a transferable learning framework that jointly optimises a user simulator together with a dialogue system. The end-to-end design of both models enables the self-play between the two agents on target domains; the models are then optimised through RL during interaction. The proposed framework is shown to boost the dialogue system performance in two practical transfer learning setups.

Lastly, we explore the coreference and ellipsis in conversations that pose problems for system's understanding. The presented transformer based model tackles the problem by first performing coreference resolution and then rewriting the user query into a complete utterance. Experiments show that not only does our model outperforms strong baselines on the two individual tasks, but also that the coreference modelling considerably benefits the quality of user query rewriting. As a side contribution, the dataset acquired for the study, the first to couple annotations of both coreference resolution and query rewriting, is released to the research community.

8.2 Limitations and Future Work

There are some limitations to the methods proposed in this thesis. First, when modelling NLU and NLG jointly in JUG (Chapter 4), the difference in nature of the two data resources is not considered. Specifically, natural language is richer and more informal, requiring NLU module to handle ambiguous or erroneous user inputs. By comparison, formal representations utilised by NLG are more precisely-defined and structured. Refinement of the model by emphasising this difference between the two tasks is worth investigating.

Another limitation exists in the presented coreference resolution model (Chapter 7). The model is designed to predict the boundaries of a referent, which makes it only able to handle cases involving continuous spans of words but not discontinuous mentions.[1] In addition, the influence of the query rewrite process on coreference resolution is limited due to the nature of the information flow in our model design. Improvement of these limitations could be one future research direction.

The architectures of end-to-end models presented in Chapters 5 and 6 rely on LSTMs (Hochreiter and Schmidhuber, 1997). Despite the fact that LSTMs can model sequences

[1]To elaborate, an instance from the coreference dataset CRAFT (Cohen et al., 2017), *"The protein belongs to a family of evolutionarily conserved proteins of a bipartite structure with a variable N-terminal and a conserved C-terminal domain."* has a discontinuous mention *"a variable N-terminal .. domain"*.

with arbitrary lengths, it is shown that they are good at making use of nearby context but less sensitive to the distant history (Khandelwal et al., 2018). Transformers (Devlin et al., 2019; Vaswani et al., 2017), by comparison, have been shown to adequately model long-range dependencies in sequences (Dai et al., 2019; Li et al., 2019b). Given their remarkable dialogue pattern learning ability (Hosseini-Asl et al., 2020; Liu et al., 2021; Yang et al., 2021), it is worth investigating our proposed training strategies combined with powerful transformers to see if the model performance could be further enhanced.

Dataset complexity is also of great importance. Despite being one of the most commonly used benchmarks in dialogue modelling due to its large size, the MultiWOZ dataset (Budzianowski et al., 2018) used in this thesis has some limitations which hinder the development of large-scale, realistic digital assistants. First, it only covers five (major) domains, which limits the possibility of testing a system's ability of transfer learning to a variety of dialogue situations. Second, the dialogue flow has a clear pattern, which makes policy learning not that difficult. For instance, the tasks in multi-domain dialogues are sequentially completed with distinct domain boundaries, and the user never goes back to the previous domain to add or modify constraints. Even within a single-domain session, the interaction always starts with narrowing the search space, followed by answering of requests and booking completion. High dialogue completion rate (93.5% success rate in policy optimisation (Lubis et al., 2020) in the benchmark[2] suggests the need for more complex and realistic corpora to bridge the gap between academic research and practical applications.

To answer the aforementioned issue, in our view, one possible choice for more challenging dialogue modelling could be the SGD dataset (Rastogi et al., 2020). Although the policies in this corpus are simulated, with utterances being paraphrased by humans, the interaction between the two agents is actually more complex than MultiWOZ due to the following facts: (a) more domains (20) are supported (e.g., banking, weather and media), resulting in more domain combinations within dialogues; (b) more diversified user policies with 11 user acts in total (e.g., *negate*, *affirm* and *request_alternative*), leading to more realistic user behaviours and interactions, compared to MultiWOZ. However, to the best of our knowledge, dialogue generation or policy learning on the SGD corpus are not explored, partially because the corpus is initially proposed as a dialogue state tracking benchmark[3] and no standard evaluation script is provided for its dialogue completion and generation. Learning generation models, especially in an end-to-end framework, on the SGD corpus would be a worthwhile research direction.

[2]https://github.com/budzianowski/multiwoz
[3]https://dstc8.dstc.community/tracks

User simulators play a vital role in dialogue policy learning. As verified by our transferable framework (Chapter 6), dialogues simulated by well-learned user models and systems can be used to enhance model performance by continuing model optimisation using reinforcement learning. Unlike dialogue systems, user modelling is less studied. Here we provide several directions to improve user simulators. First, pre-training on large heterogeneous dialogue corpora benefits the system performances on target domains (Peng et al., 2020a; Wu et al., 2020). The same practice could be applied to user modelling to learn more complex user behaviours across corpora. Second, transformer based user models are expected to learn better from data, compared to the proposed LSTM counterparts. A recent user model proposed by Lin et al. (2021) is based on transformers, but it only learns interactions at the semantic level, disregarding the language understanding and generation components. Finally, to inject more human-like characteristics for more realistic user modelling, learning from open-domain chit-chat corpora (such as the Persona-Chat dataset (Mazare et al., 2018)) could be an interesting research direction.

As a final remark, the methodologies proposed and the corpora conducted in this thesis are limited to English alone. Multi- or cross-lingual dialogue system is an emerging and less studied research topic, primarily due to the absence of multilingual corpora and evaluation benchmarks. As pointed out by Razumovskaia et al. (2021), the development of a multilingual dialogue system requires annotations for each sub-tasks (e.g., NLU, DST, NLG) in the pipelined dialogue system, for each language of interest, which largely increases the difficulty of annotation process. How to build a cross-lingual dialogue system that is capable of transferring knowledge from resource-rich languages to resource-lean languages is the key to the success (Lauscher et al., 2020; Wu and Dredze, 2020) in this line of research.

All in all, this thesis presents novel approaches to improving natural language understanding and generation capabilities of dialogue systems. Specifically, we have proposed algorithms which can be used for dialogue systems development when the amount of training data is limited. We hope that our findings and the critical analysis of our methods in the previous chapters will support and encourage further research in this area.

References

Abowd, G. D., Wang, H.-M., and Monk, A. F. (1995). A formal technique for automated dialogue development. In *Proceedings of the 1st conference on Designing interactive systems: processes, practices, methods, & techniques*, pages 219–226.

Anantha, R., Vakulenko, S., Tu, Z., Longpre, S., Pulman, S., and Chappidi, S. (2020). Open-domain question answering goes conversational via question rewriting. *arXiv preprint arXiv:2010.04898*.

Asri, L. E., Schulz, H., Sharma, S., Zumer, J., Harris, J., Fine, E., Mehrotra, R., and Suleman, K. (2017). Frames: A corpus for adding memory to goal-oriented dialogue systems. *Proceedings of SigDial*, pages 207–219.

Austin, J. L. (1975). *How to do things with words*. Oxford university press.

Bagga, A. and Baldwin, B. (1998). Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566. Citeseer.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Bahl, L., Brown, P., De Souza, P., and Mercer, R. (1986). Maximum mutual information estimation of hidden markov model parameters for speech recognition. In *ICASSP'86. IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 11, pages 49–52. IEEE.

Baker, J. (1975). The dragon system–an overview. *IEEE Transactions on Acoustics, speech, and signal Processing*, 23(1):24–29.

Balakrishnan, A., Rao, J., Upasani, K., White, M., and Subba, R. (2019). Constrained decoding for neural nlg from compositional representations in task-oriented dialogue. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 831–844.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract meaning representation for sembanking. In *Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse*, pages 178–186, Sofia, Bulgaria. Association for Computational Linguistics.

Berant, J., Chou, A., Frostig, R., and Liang, P. (2013). Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Bikel, D. M., Castelli, V., Florian, R., and Han, D.-j. (2009). Entity linking and slot filling through statistical processing and inference rules. In *TAC*.

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. (2017). Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877.

Bobrow, D. G., Kaplan, R. M., Kay, M., Norman, D. A., Thompson, H., and Winograd, T. (1977). Gus, a frame-driven dialog system. *Artificial intelligence*, 8(2):155–173.

Bordes, A., Boureau, Y.-L., and Weston, J. (2016). Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683*.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A., Jozefowicz, R., and Bengio, S. (2016). Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.

Budzianowski, P. and Vulić, I. (2019). Hello, it's GPT-2-How can I help you? Towards the use of pretrained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22.

Budzianowski, P., Wen, T.-H., Tseng, B.-H., Casanueva, I., Ultes, S., Ramadan, O., and Gasic, M. (2018). Multiwoz-a large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 5016–5026.

Bunt, H. (2009). The dit++ taxonomy for functional dialogue markup. In *AAMAS 2009 Workshop, Towards a Standard Markup Language for Embodied Dialogue Acts*, pages 13–24.

Bunt, H., Alexandersson, J., Choe, J.-W., Fang, A. C., Hasida, K., Petukhova, V., Popescu-Belis, A., and Traum, D. (2012). Iso 24617-2: A semantically-based standard for dialogue annotation. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 430–437.

Cao, R., Zhu, S., Liu, C., Li, J., and Yu, K. (2019). Semantic parsing with dual learning. *ACL*.

Casanueva, I., Budzianowski, P., Ultes, S., Kreyssig, F., Tseng, B.-H., Wu, Y.-C., and Gašić, M. (2018). Feudal dialogue management with jointly learned feature extractors. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*.

Chan, W., Jaitly, N., Le, Q., and Vinyals, O. (2016). Listen, attend and spell: A neural network for large vocabulary conversational speech recognition. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4960–4964. IEEE.

Charniak, E. (1997). Statistical parsing with a context-free grammar and word statistics. *AAAI/IAAI*, 2005(598-603):18.

Chen, D. and Manning, C. D. (2014). A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 740–750.

Chen, Q., Zhuo, Z., and Wang, W. (2019). Bert for joint intent classification and slot filling. *arXiv preprint arXiv:1902.10909*.

Chen, X., Kingma, D. P., Salimans, T., Duan, Y., Dhariwal, P., Schulman, J., Sutskever, I., and Abbeel, P. (2017). Variational lossy autoencoder. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Chen, Y.-N., Hakkani-Tür, D., Tür, G., Gao, J., and Deng, L. (2016). End-to-end memory networks with knowledge carryover for multi-turn spoken language understanding. In *Interspeech*, pages 3245–3249.

Chen, Z., Eavani, H., Chen, W., Liu, Y., and Wang, W. Y. (2020). Few-shot nlg with pre-trained language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 183–190.

Chorowski, J., Bahdanau, D., Cho, K., and Bengio, Y. (2014). End-to-end continuous speech recognition using attention-based recurrent nn: First results. *arXiv preprint arXiv:1412.1602*.

Clark, K. and Manning, C. D. (2016a). Deep reinforcement learning for mention-ranking coreference models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2256–2262.

Clark, K. and Manning, C. D. (2016b). Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653.

Coca, A., Tseng, B.-H., and Byrne, B. (2021). Gcdf1: A goal-and context-driven f-score for evaluating user models. In *The First Workshop on Evaluations and Assessments of Neural Conversation Systems*, pages 7–14.

Cohen, K. B., Lanfranchi, A., Choi, M. J.-y., Bada, M., Baumgartner, W. A., Panteleyeva, N., Verspoor, K., Palmer, M., and Hunter, L. E. (2017). Coreference annotation and resolution in the colorado richly annotated full text (craft) corpus of biomedical journal articles. *BMC bioinformatics*, 18(1):1–14.

Colby, K. M., Hilf, F. D., Weber, S., and Kraemer, H. C. (1972). Turing-like indistinguishability tests for the validation of a computer simulation of paranoid processes. *Artificial Intelligence*, 3:199–221.

Colby, K. M., Weber, S., and Hilf, F. D. (1971). Artificial paranoia. *Artificial Intelligence*, 2(1):1–25.

Cooley, J. W. and Tukey, J. W. (1965). An algorithm for the machine calculation of complex fourier series. *Mathematics of computation*, 19(90):297–301.

Core, M. G. and Allen, J. (1997). Coding dialogs with the damsl annotation scheme. In *AAAI fall symposium on communicative action in humans and machines*, volume 56, pages 28–35. Boston, MA.

Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q., and Salakhutdinov, R. (2019). Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988.

Damonte, M., Goel, R., and Chung, T. (2019). Practical semantic parsing for spoken language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 16–23.

Deng, L., Tur, G., He, X., and Hakkani-Tur, D. (2012). Use of kernel deep convex networks and end-to-end learning for spoken language understanding. In *2012 IEEE Spoken Language Technology Workshop (SLT)*, pages 210–215. IEEE.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2019). BERT: pre-training of deep bidirectional transformers for language understanding. In Burstein, J., Doran, C., and Solorio, T., editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Dhingra, B., Li, L., Li, X., Gao, J., Chen, Y.-N., Ahmad, F., and Deng, L. (2017). Towards end-to-end reinforcement learning of dialogue agents for information access. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 484–495.

Doersch, C. (2016). Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*.

Dušek, O. and Jurčíček, F. (2016). Sequence-to-sequence generation for spoken dialogue via deep syntax trees and strings. In *The 54th Annual Meeting of the Association for Computational Linguistics*, page 45.

Dušek, O., Novikova, J., and Rieser, V. (2018). Findings of the E2E NLG challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 322–328.

Dušek, O., Novikova, J., and Rieser, V. (2020). Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG challenge. *Computer Speech & Language*, 59:123–156.

El Asri, L., He, J., and Suleman, K. (2016). A sequence-to-sequence model for user simulation in spoken dialogue systems. *Interspeech 2016*, pages 1151–1155.

El Asri, L., Laroche, R., and Pietquin, O. (2014). Task completion transfer learning for reward inference. In *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*.

Eric, M., Goel, R., Paul, S., Sethi, A., Agarwal, S., Gao, S., Kumar, A., Goyal, A., Ku, P., and Hakkani-Tur, D. (2020). Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 422–428.

Eric, M., Krishnan, L., Charette, F., and Manning, C. D. (2017). Key-value retrieval networks for task-oriented dialogue. In *Proceedings of SigDial*, pages 37–49.

Freund, Y., Iyer, R., Schapire, R. E., and Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *Journal of machine learning research*, 4(Nov):933–969.

Fu, W.-T. and Anderson, J. R. (2008). Solving the credit assignment problem: explicit and implicit learning of action sequences with probabilistic outcomes. *Psychological research*, 72(3):321–330.

Gao, J., Galley, M., and Li, L. (2018). Neural approaches to conversational ai. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*, pages 1371–1374.

Gao, S., Sethi, A., Agarwal, S., Chung, T., Hakkani-Tur, D., and AI, A. A. (2019). Dialog state tracking: A neural reading comprehension approach. In *20th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 264.

Gao, S., Zhang, Y., Ou, Z., and Yu, Z. (2020). Paraphrase augmented task-oriented dialog generation. *ACL*.

Gardent, C., Shimorina, A., Narayan, S., and Perez-Beltrachini, L. (2017). The webnlg challenge: Generating text from rdf data. In *Proceedings of the 10th International Conference on Natural Language Generation*, pages 124–133.

Geldof, S. and Van de Velde, W. (1997). An architecture for template based (hyper) text generation. In *Proceedings of the 6th European Workshop on Natural Language Generation*, pages 28–37.

Goodfellow, I. J., Shlens, J., and Szegedy, C. (2014). Explaining and harnessing adversarial examples. *ICLR*.

Goyal, A. G. A. P., Sordoni, A., Côté, M.-A., Ke, N. R., and Bengio, Y. (2017). Z-forcing: Training stochastic recurrent networks. In *Advances in neural information processing systems*, pages 6713–6723.

Graves, A., Fernández, S., Gomez, F., and Schmidhuber, J. (2006). Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Gu, J., Lu, Z., Li, H., and Li, V. O. (2016). Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1640.

Hakkani-Tür, D., Tür, G., Celikyilmaz, A., Chen, Y.-N., Gao, J., Deng, L., and Wang, Y.-Y. (2016). Multi-domain joint semantic frame parsing using bi-directional RNN-LSTM. In *Interspeech*, pages 715–719.

Han, T., Liu, X., Takanobu, R., Lian, Y., Huang, C., Peng, W., and Huang, M. (2020). Multiwoz 2.3: A multi-domain task-oriented dataset enhanced with annotation corrections and co-reference annotation. *arXiv preprint arXiv:2010.05594*.

Hashemi, H. B., Asiaee, A., and Kraft, R. (2016). Query intent detection using convolutional neural networks. In *International Conference on Web Search and Data Mining, Workshop on Query Understanding*.

He, D., Xia, Y., Qin, T., Wang, L., Yu, N., Liu, T.-Y., and Ma, W.-Y. (2016). Dual learning for machine translation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, pages 820–828. Curran Associates Inc.

Henderson, M., Budzianowski, P., Casanueva, I., Coope, S., Gerz, D., Kumar, G., Mrkšić, N., Spithourakis, G., Su, P.-H., Vulić, I., and Wen, T.-H. (2019). A repository of conversational datasets. In *Proceedings of the Workshop on NLP for Conversational AI*.

Henderson, M., Thomson, B., and Williams, J. D. (2014a). The second dialog state tracking challenge. In *Proceedings of the 15th annual meeting of the special interest group on discourse and dialogue (SIGDIAL)*, pages 263–272.

Henderson, M., Thomson, B., and Williams, J. D. (2014b). The third dialog state tracking challenge. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 324–329. IEEE.

Henderson, M., Thomson, B., and Young, S. (2013). Deep neural network approach for the dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*, pages 467–471.

Henderson, M., Thomson, B., and Young, S. (2014c). Robust dialog state tracking using delexicalised recurrent neural networks and unsupervised adaptation. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 360–365. IEEE.

Henderson, M., Thomson, B., and Young, S. (2014d). Word-based dialog state tracking with recurrent neural networks. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 292–299.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hori, C., Hori, T., Watanabe, S., and Hershey, J. R. (2015). Context sensitive spoken language understanding using role dependent lstm layers. In *Proceedings of the NIPS 2015 Workshop on Machine Learning for Spoken Language Understanding and Interaction, Montreal, QC, Canada*, volume 11.

Hosseini-Asl, E., McCann, B., Wu, C.-S., Yavuz, S., and Socher, R. (2020). A simple language model for task-oriented dialogue. *arXiv preprint arXiv:2005.00796*.

Hunt, A. J. and Black, A. W. (1996). Unit selection in a concatenative speech synthesis system using a large speech database. In *1996 IEEE International Conference on Acoustics, Speech, and Signal Processing Conference Proceedings*, volume 1, pages 373–376. IEEE.

Ipeirotis, P. G., Provost, F., and Wang, J. (2010). Quality management on amazon mechanical turk. In *Proceedings of the ACM SIGKDD workshop on human computation*, pages 64–67. ACM.

Itakura, F. (1975). Minimum prediction residual principle applied to speech recognition. *IEEE Transactions on acoustics, speech, and signal processing*, 23(1):67–72.

Jain, S. and Wallace, B. C. (2019). Attention is not explanation. In *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 3543–3556.

Jelinek, F., Bahl, L., and Mercer, R. (1975). Design of a linguistic statistical decoder for the recognition of continuous speech. *IEEE Transactions on Information Theory*, 21(3):250–256.

Jeong, M. and Lee, G. G. (2008). Triangular-chain conditional random fields. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(7):1287–1302.

Joshi, M., Chen, D., Liu, Y., Weld, D. S., Zettlemoyer, L., and Levy, O. (2020). Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Joshi, M., Levy, O., Zettlemoyer, L., and Weld, D. S. (2019). Bert for coreference resolution: Baselines and analysis. 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 5807–5812.

Jurafsky, D. and Martin, J. H. (2019). Speech and language processing (3rd draft ed.).

Jurafsky, D., Shriberg, E., and Biasca, D. (1997). Switchboard swbd-damsl labeling project coder's manual. *Draft 13. Technical Report 97-02*.

Juraska, J., Karagiannis, P., Bowden, K., and Walker, M. (2018). A deep ensemble model with slot alignment for sequence-to-sequence natural language generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 152–162.

Kate, R. and Mooney, R. (2006). Using string-kernels for learning semantic parsers. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 913–920.

Kedzie, C. and Mckeown, K. (2019). A good sample is hard to find: Noise injection sampling and self-training for neural language generation models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 584–593.

Khandelwal, U., He, H., Qi, P., and Jurafsky, D. (2018). Sharp nearby, fuzzy far away: How neural language models use context. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 284–294.

Kim, S., D'Haro, L. F., Banchs, R. E., Williams, J. D., and Henderson, M. (2017). The fourth dialog state tracking challenge. In *Dialogues with Social Robots*, pages 435–449. Springer.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kingma, D. P. and Welling, M. (2013). Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Kirkpatrick, J., Pascanu, R., Rabinowitz, N., Veness, J., Desjardins, G., Rusu, A. A., Milan, K., Quan, J., Ramalho, T., Grabska-Barwinska, A., et al. (2017). Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526.

Kreyssig, F., Casanueva, I., Budzianowski, P., and Gašić, M. (2018). Neural user simulation for corpus-based policy optimisation of spoken dialogue systems. In *Proc. SIGDIAL*, Melbourne.

Krizhevsky, A., Nair, V., and Hinton, G. (2010). Cifar-10 (canadian institute for advanced research). *URL http://www. cs. toronto. edu/kriz/cifar. html*, 5:4.

Kumar, V. and Joshi, S. (2016). Non-sentential question resolution using sequence to sequence learning. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2022–2031, Osaka, Japan. The COLING 2016 Organizing Committee.

Kwiatkowski, T., Zettlemoyer, L., Goldwater, S., and Steedman, M. (2011). Lexical generalization in ccg grammar induction for semantic parsing. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1512–1523.

Lafferty, J. (2001). Conditional random fields: probabilistic models for segmenting and labeling sequence data. In *Proc. of the 18th Intl. Conf. on Machine Learning (ICML-2001)*.

Laine, S. and Aila, T. (2017). Temporal ensembling for semi-supervised learning. *Fifth International Conference on Learning Representations*.

Lample, G., Ballesteros, M., Subramanian, S., Kawakami, K., and Dyer, C. (2016). Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Larsson, S. and Traum, D. R. (2000). Information state and dialogue management in the trindi dialogue move engine toolkit. *Natural language engineering*, 6(3 & 4):323–340.

Lauscher, A., Ravishankar, V., Vulić, I., and Glavaš, G. (2020). From zero to hero: On the limitations of zero-shot language transfer with multilingual transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499.

Lavoie, B. and Rainbow, O. (1997). A fast and portable realizer for text generation systems. In *Fifth Conference on Applied Natural Language Processing*, pages 265–268.

Lebret, R., Grangier, D., and Auli, M. (2016). Neural text generation from structured data with application to the biography domain. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1203–1213.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, D.-H. (2013). Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on Challenges in Representation Learning, ICML*, volume 3, page 2.

Lee, J. Y. and Dernoncourt, F. (2016). Sequential short-text classification with recurrent and convolutional neural networks. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 515–520.

Lee, K., He, L., Lewis, M., and Zettlemoyer, L. (2017). End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 188–197.

Lee, K., He, L., and Zettlemoyer, L. (2018). Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 687–692.

Lee, S. and Eskenazi, M. (2013). Recipe for building robust spoken dialog state trackers: Dialog state tracking challenge system description. In *Proceedings of the SIGDIAL 2013 Conference*, pages 414–422.

Lee, S., Zhu, Q., Takanobu, R., Li, X., Zhang, Y., Zhang, Z., Li, J., Peng, B., Li, X., Huang, M., et al. (2019). Convlab: Multi-domain end-to-end dialog system platform. *ACL*.

Lei, W., Jin, X., Kan, M.-Y., Ren, Z., He, X., and Yin, D. (2018). Sequicity: Simplifying task-oriented dialogue systems with single sequence-to-sequence architectures. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1437–1447.

Lemon, O., Georgila, K., Henderson, J., and Stuttle, M. (2006). An isu dialogue system exhibiting reinforcement learning of dialogue policies: generic slot-filling in the talk in-car system. In *Demonstrations*.

Lemon, O. and Pietquin, O. (2007). Machine learning for spoken dialogue systems. In *Eighth Annual Conference of the International Speech Communication Association*.

Levin, E., Pieraccini, R., and Eckert, W. (2000). A stochastic model of human-machine interaction for learning dialog strategies. *IEEE Transactions on speech and audio processing*, 8(1):11–23.

Levinson, S. E. (1986). Continuously variable duration hidden markov models for automatic speech recognition. *Computer Speech & Language*, 1(1):29–45.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2020). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880.

Li, J., Monroe, W., Ritter, A., Jurafsky, D., Galley, M., and Gao, J. (2016). Deep reinforcement learning for dialogue generation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1192–1202.

Li, N., Liu, S., Liu, Y., Zhao, S., and Liu, M. (2019a). Neural speech synthesis with transformer network. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6706–6713.

Li, S., Jin, X., Xuan, Y., Zhou, X., Chen, W., Wang, Y.-X., and Yan, X. (2019b). Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. *Advances in Neural Information Processing Systems*, 32:5243–5253.

Li, X., Chen, Y.-N., Li, L., Gao, J., and Celikyilmaz, A. (2017a). 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.

Li, X., Chen, Y.-N., Li, L., Gao, J., and Celikyilmaz, A. (2017b). Investigation of language understanding impact for reinforcement learning based dialogue systems. *arXiv preprint arXiv:1703.07055*.

Lin, C.-Y. (2004). Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

Lin, H.-c., Lubis, N., Hu, S., van Niekerk, C., Geishauser, C., Heck, M., Feng, S., and Gašić, M. (2021). Domain-independent user simulation with transformers for task-oriented dialogue systems. *arXiv preprint arXiv:2106.08838*.

Lin, Z., Madotto, A., Winata, G. I., and Fung, P. (2020). Mintl: Minimalist transfer learning for task-oriented dialogue systems. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3391–3405.

Ling, Z.-H., Kang, S.-Y., Zen, H., Senior, A., Schuster, M., Qian, X.-J., Meng, H. M., and Deng, L. (2015). Deep learning for acoustic modeling in parametric speech generation: A systematic review of existing techniques and future trends. *IEEE Signal Processing Magazine*, 32(3):35–52.

Liu, B. and Lane, I. (2016). Attention-based recurrent neural network models for joint intent detection and slot filling. *Interspeech 2016*, pages 685–689.

Liu, B. and Lane, I. (2017). Iterative policy learning in end-to-end trainable task-oriented neural dialog models. *2017 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, pages 482–489.

Liu, Q., Yu, L., Rimell, L., and Blunsom, P. (2021). Pretraining the noisy channel model for task-oriented dialogue. *arXiv preprint arXiv:2103.10518*.

Lowe, R., Pow, N., Serban, I. V., and Pineau, J. (2015). The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 285.

Lubis, N., Geishauser, C., Heck, M., Lin, H.-c., Moresi, M., van Niekerk, C., and Gasic, M. (2020). Lava: Latent action spaces via variational auto-encoding for dialogue policy optimization. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 465–479.

Luo, X. (2005). On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32.

Luong, T., Pham, H., and Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421.

Maaten, L. v. d. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605.

Mairesse, F., Gasic, M., Jurcicek, F., Keizer, S., Thomson, B., Yu, K., and Young, S. (2009). Spoken language understanding from unaligned data using discriminative classification models. In *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4749–4752. IEEE.

Màrquez, L., Recasens, M., and Sapena, E. (2013). Coreference resolution: an empirical study based on semeval-2010 shared task 1. *Language resources and evaluation*, 47(3):661–694.

Martin, S., Poddar, S., and Upasani, K. (2020). MuDoCo: Corpus for multidomain coreference resolution and referring expression generation. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 104–111, Marseille, France. European Language Resources Association.

Mazare, P.-E., Humeau, S., Raison, M., and Bordes, A. (2018). Training millions of personalized dialogue agents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2775–2779.

McCarthy, J., Minsky, M., and Rochester, N. (1955). A proposal for the dartmouth summer research project on artificial intelligence. *Dartmouth Summer Research Project on Artificial Intelligence*.

McShane, M. J. et al. (2005). *A theory of ellipsis*. Oxford University Press on Demand.

McTear, M. (2020). Conversational ai: dialogue systems, conversational agents, and chatbots. *Synthesis Lectures on Human Language Technologies*, 13(3):1–251.

McTear, M. F. (1998). Modelling spoken dialogues with state transition diagrams: experiences with the cslu toolkit. In *Fifth international conference on spoken language processing*.

Mehri, S., Srinivasan, T., and Eskenazi, M. (2019). Structured fusion networks for dialog. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 165–177.

Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tur, D., He, X., Heck, L., Tur, G., Yu, D., et al. (2014). Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(3):530–539.

Mesnil, G., He, X., Deng, L., and Bengio, Y. (2013). Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *Interspeech*, pages 3771–3775.

Misu, T. and Kawahara, T. (2007). Speech-based interactive information guidance system using question-answering technique. In *2007 IEEE International Conference on Acoustics, Speech and Signal Processing-ICASSP'07*, volume 4, pages IV–145. IEEE.

Miyato, T., Dai, A. M., and Goodfellow, I. (2016). Adversarial training methods for semi-supervised text classification. *International Conference on Learning Representations*.

Miyato, T., Maeda, S.-i., Koyama, M., Nakae, K., and Ishii, S. (2015). Distributional smoothing with virtual adversarial training. *ICLR*.

Morgan, N. and Bourlard, H. A. (1995). Neural networks for statistical recognition of continuous speech. *Proceedings of the IEEE*, 83(5):742–772.

Moulines, E. and Charpentier, F. (1990). Pitch-synchronous waveform processing techniques for text-to-speech synthesis using diphones. *Speech communication*, 9(5-6):453–467.

Mrkšić, N., Séaghdha, D. Ó., Thomson, B., Gasic, M., Su, P.-H., Vandyke, D., Wen, T.-H., and Young, S. (2015). Multi-domain dialog state tracking using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 794–799.

Mrkšić, N., Séaghdha, D. Ó., Wen, T.-H., Thomson, B., and Young, S. (2017). Neural belief tracker: Data-driven dialogue state tracking. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1777–1788.

Nadeau, D. and Sekine, S. (2007). A survey of named entity recognition and classification. *Lingvisticae Investigationes*, 30(1):3–26.

Nekvinda, T. and Dušek, O. (2021). Shades of BLEU, flavours of success: The case of MultiWOZ. In *Proceedings of the 1st Workshop on Natural Language Generation, Evaluation, and Metrics (GEM 2021)*, pages 34–46, Online. Association for Computational Linguistics.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. (2011). Reading digits in natural images with unsupervised feature learning. *NIPS Workshop*.

Ng, V. (2010). Supervised noun phrase coreference research: The first fifteen years. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 1396–1411.

Ng, V. and Cardie, C. (2002). Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pages 104–111.

Nie, F., Yao, J.-G., Wang, J., Pan, R., and Lin, C.-Y. (2019). A simple recipe towards reducing hallucination in neural surface realisation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2673–2679.

Nivre, J., Hall, J., Nilsson, J., Chanev, A., Eryigit, G., Kübler, S., Marinov, S., and Marsi, E. (2007). Maltparser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering*, 13(2):95.

Novikova, J., Dušek, O., Curry, A. C., and Rieser, V. (2017a). Why we need new evaluation metrics for nlg. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2241–2252.

Novikova, J., Dušek, O., and Rieser, V. (2017b). The E2E dataset: New challenges for end-to-end generation. In *Proceedings of the 18th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, Saarbrücken, Germany. arXiv:1706.09254.

Oord, A., Li, Y., Babuschkin, I., Simonyan, K., Vinyals, O., Kavukcuoglu, K., Driessche, G., Lockhart, E., Cobo, L., Stimberg, F., et al. (2018). Parallel Wavenet: Fast high-fidelity speech synthesis. In *International conference on machine learning*, pages 3918–3926. PMLR.

Papangelis, A., Wang, Y.-C., Molino, P., and Tur, G. (2019). Collaborative multi-agent dialogue model training via reinforcement learning. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 92–102.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Pasupat, P., Gupta, S., Mandyam, K., Shah, R., Lewis, M., and Zettlemoyer, L. (2019). Span-based hierarchical semantic parsing for task-oriented dialog. 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 1520–1526, Hong Kong, China. Association for Computational Linguistics.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in Pytorch. In *NIPS-W*.

Paul, S., Goel, R., and Hakkani-Tür, D. (2019). Towards universal dialogue act tagging for task-oriented dialogues. *Proc. Interspeech 2019*, pages 1453–1457.

Pei, J., Ren, P., Monz, C., and de Rijke, M. (2020). Retrospective and prospective mixture-of-generators for task-oriented dialogue response generation. *ECAI*.

Peng, B., Li, C., Li, J., Shayandeh, S., Liden, L., and Gao, J. (2020a). SOLOIST: few-shot task-oriented dialog with A single pre-trained auto-regressive model. *CoRR*, abs/2005.05298.

Peng, B., Zhu, C., Li, C., Li, X., Li, J., Zeng, M., and Gao, J. (2020b). Few-shot natural language generation for task-oriented dialog. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings*, pages 172–182.

Peng, H., Chang, K.-W., and Roth, D. (2015). A joint framework for coreference resolution and mention head detection. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 12–21, Beijing, China. Association for Computational Linguistics.

Peng, S., Huang, X., Lin, Z., Ji, F., Chen, H., and Zhang, Y. (2019). Teacher-student framework enhanced multi-domain dialogue generation. *arXiv preprint arXiv:1908.07137*.

Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Pieraccini, R. and Huerta, J. (2005). Where do we go from here? research and commercial spoken dialog systems. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pages 1–10.

Pradhan, S., Moschitti, A., Xue, N., Uryupina, O., and Zhang, Y. (2012). Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40.

Price, P. (1990). Evaluation of spoken language systems: The ATIS domain. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*.

Puduppully, R., Dong, L., and Lapata, M. (2019). Data-to-text generation with content selection and planning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 6908–6915.

Purver, M., Eshghi, A., and Hough, J. (2011). Incremental semantic construction in a dialogue system. In *Proceedings of the Ninth International Conference on Computational Semantics (IWCS 2011)*.

Qader, R., Portet, F., and Labbé, C. (2019). Semi-supervised neural text generation by joint learning of natural language generation and natural language understanding models. In *Proceedings of the 12th International Conference on Natural Language Generation*, pages 552–562.

Quan, J., Xiong, D., Webber, B., and Hu, C. (2019). Gecor: An end-to-end generative ellipsis and co-reference resolution model for task-oriented dialogue. 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 4547–4557.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *arXiv preprint*.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67.

Ramshaw, L. A. and Marcus, M. P. (1999). Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer.

Rastogi, A., Zang, X., Sunkara, S., Gupta, R., and Khaitan, P. (2020). Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8689–8696.

Rastogi, P., Gupta, A., Chen, T., and Lambert, M. (2019). Scaling multi-domain dialogue state tracking via query reformulation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Industry Papers)*, pages 97–105.

Ravuri, S. and Stolcke, A. (2015). Recurrent neural network and lstm models for lexical utterance classification. In *Sixteenth Annual Conference of the International Speech Communication Association*.

Ravuri, S. and Stolcke, A. (2016). A comparative study of recurrent neural network models for lexical domain classification. In *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6075–6079. IEEE.

Razumovskaia, E., Glavaš, G., Majewska, O., Ponti, E. M., Korhonen, A., and Vulić, I. (2021). Crossing the conversational chasm: A primer on natural language processing for multilingual task-oriented dialogue systems. *arXiv preprint arXiv:2104.08570*.

Reiter, E. (2018). A structured review of the validity of bleu. *Computational Linguistics*, 44(3):393–401.

Ren, H., Xu, W., Zhang, Y., and Yan, Y. (2013). Dialog state tracking using conditional random fields. In *Proceedings of the SIGDIAL 2013 Conference*, pages 457–461.

Ren, L., Xie, K., Chen, L., and Yu, K. (2018). Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2780–2786.

Ritter, A., Cherry, C., and Dolan, W. B. (2011). Data-driven response generation in social media. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 583–593.

Robinson, T. and Fallside, F. (1991). A recurrent error propagation network speech recognition system. *Computer Speech & Language*, 5(3):259–274.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3):211–252.

Schapire, R. E. (1999). A brief introduction to boosting. In *Ijcai*, volume 99, pages 1401–1406.

Schapire, R. E. and Singer, Y. (2000). Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2):135–168.

Schatzmann, J., Thomson, B., Weilhammer, K., Ye, H., and Young, S. (2007). Agenda-based user simulation for bootstrapping a pomdp dialogue system. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Companion Volume, Short Papers*, pages 149–152.

Schatzmann, J., Weilhammer, K., Stuttle, M., and Young, S. (2006). A survey of statistical user simulation techniques for reinforcement-learning of dialogue management strategies. *The Knowledge Engineering Review*, 21(2):97–126.

Schmitt, A. and Ultes, S. (2015). Interaction quality: assessing the quality of ongoing spoken dialog interaction by experts—and how it relates to user satisfaction. *Speech Communication*, 74:12–36.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083.

Seneff, S. and Polifroni, J. (2000). Dialogue management in the mercury flight reservation system. In *ANLP-NAACL 2000 Workshop: Conversational Systems*.

Sennrich, R., Haddow, B., and Birch, A. (2016). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96.

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A., and Pineau, J. (2016). Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 3776–3783.

Singh, S., Kearns, M., Litman, D., and Walker, M. (1999). Reinforcement learning for spoken dialogue systems. *Advances in neural information processing systems*, 12:956–962.

Singh, S., Litman, D., Kearns, M., and Walker, M. (2002). Optimizing dialogue management with reinforcement learning: Experiments with the njfun system. *Journal of Artificial Intelligence Research*, 16:105–133.

Soon, W. M., Ng, H. T., and Lim, D. C. Y. (2001). A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544.

Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.-Y., Gao, J., and Dolan, W. B. (2015). A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 196–205.

Stent, A., Marge, M., and Singhai, M. (2005). Evaluating evaluation methods for generation in the presence of variation. In *International Conference on Intelligent Text Processing and Computational Linguistics*, pages 341–351. Springer.

Stent, A., Prasad, R., and Walker, M. (2004). Trainable sentence planning for complex information presentation in spoken dialog systems. In *Proceedings of the 42nd annual meeting on association for computational linguistics*, page 79. Association for Computational Linguistics.

Stolcke, A., Ries, K., Coccaro, N., Shriberg, E., Bates, R., Jurafsky, D., Taylor, P., Martin, R., Ess-Dykema, C. V., and Meteer, M. (2000). Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373.

Su, H., Shen, X., Zhang, R., Sun, F., Hu, P., Niu, C., and Zhou, J. (2019a). Improving multi-turn dialogue modelling with utterance rewriter. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 22–31.

Su, P.-H., Budzianowski, P., Ultes, S., Gasic, M., and Young, S. (2017). Sample-efficient actor-critic reinforcement learning with supervised data for dialogue management. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*, pages 147–157.

Su, S.-Y., Huang, C.-W., and Chen, Y.-N. (2019b). Dual supervised learning for natural language understanding and generation. *ACL*.

Sukhbaatar, S., Weston, J., Fergus, R., et al. (2015). End-to-end memory networks. *Advances in Neural Information Processing Systems*, 28:2440–2448.

Sun, K., Chen, L., Zhu, S., and Yu, K. (2014). The sjtu system for dialog state tracking challenge 2. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 318–326.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112.

Sutton, R. S. (1984). *Temporal credit assignment in reinforcement learning*. PhD thesis, University of Massachusetts Amherst.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I., and Fergus, R. (2013). Intriguing properties of neural networks. *ICLR*.

Tai, K. S., Socher, R., and Manning, C. D. (2015). Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, volume 1, pages 1556–1566.

Takanobu, R., Liang, R., and Huang, M. (2020a). Multi-agent task-oriented dialog policy learning with role-aware reward decomposition. In *ACL*.

Takanobu, R., Zhu, Q., Li, J., Peng, B., Gao, J., and Huang, M. (2020b). Is your goal-oriented dialog model performing really well? empirical analysis of system-wise evaluation. In *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 297–310.

Taylor, P. (2009). *Text-to-speech synthesis*. Cambridge university press.

Thomson, B. and Young, S. (2010). Bayesian update of dialogue state: A POMDP framework for spoken dialogue systems. *Computer Speech & Language*, 24(4):562–588.

Tran, V.-K. and Nguyen, L.-M. (2017). Natural language generation for spoken dialogue system using rnn encoder-decoder networks. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 442–451.

Traum, D. R. (1999). Speech acts for dialogue agents. In *Foundations of rational agency*, pages 169–201. Springer.

Tseng, B.-H., Bhargava, S., Lu, J., Moniz, J. R. A., Piraviperumal, D., Li, L., and Yu, H. (2021a). Cread: Combined resolution of ellipses and anaphora in dialogues. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3390–3406.

Tseng, B.-H., Budzianowski, P., Wu, Y.-c., and Gasic, M. (2019a). Tree-structured semantic encoder with knowledge sharing for domain adaptation in natural language generation. In *Proceedings of the 20th Annual SIGdial Meeting on Discourse and Dialogue*, pages 155–164.

Tseng, B.-H., Cheng, J., Fang, Y., and Vandyke, D. (2020). A generative model for joint natural language understanding and generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1795–1807.

Tseng, B.-H., Dai, Y., Kreyssig, F., and Byrne, B. (2021b). Transferable dialogue systems and user simulators. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 152–166, Online. Association for Computational Linguistics.

Tseng, B.-H., Rei, M., Budzianowski, P., Turner, R., Byrne, B., and Korhonen, A. (2019b). Semi-supervised bootstrapping of dialogue state trackers for task-oriented modelling. 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 1273–1278.

Tsiakoulis, P., Breslin, C., Gašić, M., Henderson, M., Kim, D., Szummer, M., Thomson, B., and Young, S. (2014). Dialogue context sensitive HMM-based speech synthesis. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2554–2558. IEEE.

Tur, G., Deng, L., Hakkani-Tür, D., and He, X. (2012). Towards deeper understanding: Deep convex networks for semantic utterance classification. In *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 5045–5048. IEEE.

Turing, A. M. and Haugeland, J. (1950). *Computing machinery and intelligence*. MIT Press Cambridge, MA.

Ultes, S., Barahona, L. M. R., Su, P.-H., Vandyke, D., Kim, D., Casanueva, I., Budzianowski, P., Mrkšić, N., Wen, T.-H., Gasic, M., et al. (2017). Pydial: A multi-domain statistical dialogue system toolkit. In *Proceedings of ACL 2017, System Demonstrations*, pages 73–78.

Ultes, S., Schmitt, A., and Minker, W. (2013). On quality ratings for spoken dialogue systems–experts vs. users. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 569–578.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Vilain, M., Burger, J. D., Aberdeen, J., Connolly, D., and Hirschman, L. (1995). A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015). Pointer networks. In *Advances in neural information processing systems*, pages 2692–2700.

Waibel, A., Hanazawa, T., Hinton, G., Shikano, K., and Lang, K. J. (1989). Phoneme recognition using time-delay neural networks. *IEEE transactions on acoustics, speech, and signal processing*, 37(3):328–339.

Walker, M., Litman, D., Kamm, C. A., and Abella, A. (1997). Paradise: A framework for evaluating spoken dialogue agents. In *35th Annual Meeting of the Association for Computational Linguistics and 8th Conference of the European Chapter of the Association for Computational Linguistics*, pages 271–280.

Walker, M. A. (2000). An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research*, 12:387–416.

Walker, M. A., Rambow, O. C., and Rogati, M. (2002). Training a sentence planner for spoken dialogue using boosting. *Computer Speech & Language*, 16(3):409–433.

Wang, K., Tian, J., Wang, R., Quan, X., and Yu, J. (2020). Multi-domain dialogue acts and response co-generation. *arXiv preprint arXiv:2004.12363*.

Wang, Y.-Y., Deng, L., and Acero, A. (2005). Spoken language understanding. *IEEE Signal Processing Magazine*, 22(5):16–31.

Wang, Z. and Lemon, O. (2013). A simple and generic belief tracking mechanism for the dialog state tracking challenge: On the believability of observed information. In *Proceedings of the SIGDIAL 2013 Conference*, pages 423–432.

Ward, W. (1994). Extracting information in spontaneous speech. In *Third International Conference on Spoken Language Processing*.

Weizenbaum, J. (1966). Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45.

Wen, T.-H., Gašic, M., Kim, D., Mrkšic, N., Su, P.-H., Vandyke, D., and Young, S. (2015a). Stochastic language generation in dialogue using recurrent neural networks with convolutional sentence reranking. In *16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, page 275.

Wen, T.-H., Gasic, M., Mrkšić, N., Barahona, L. M. R., Su, P.-H., Ultes, S., Vandyke, D., and Young, S. (2016). Conditional generation and snapshot learning in neural dialogue systems. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2153–2162.

Wen, T.-H., Gašić, M., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. (2015b). Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics.

Wen, T.-H., Miao, Y., Blunsom, P., and Young, S. (2017a). Latent intention dialogue models. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3732–3741.

Wen, T.-H., Vandyke, D., Mrkšić, N., Gasic, M., Barahona, L. M. R., Su, P.-H., Ultes, S., and Young, S. (2017b). A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 438–449.

Weston, J., Chopra, S., and Bordes, A. (2014). Memory networks. *arXiv preprint arXiv:1410.3916*.

Wiegreffe, S. and Pinter, Y. (2019). Attention is not not explanation. 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 11–20.

Williams, J., Raux, A., and Henderson, M. (2016). The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Williams, J. D. (2009). Spoken dialogue systems: Challenges, and opportunities for research. In *ASRU*, page 25.

Williams, J. D., Henderson, M., Raux, A., Thomson, B., Black, A., and Ramachandran, D. (2014). The dialog state tracking challenge series. *AI Magazine*, 35(4):121–124.

Williams, J. D. and Young, S. (2007). Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language*, 21(2):393–422.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256.

Wu, C.-S., Hoi, S. C., Socher, R., and Xiong, C. (2020). Tod-bert: Pre-trained natural language understanding for task-oriented dialogue. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 917–929.

Wu, C.-S., Madotto, A., Hosseini-Asl, E., Xiong, C., Socher, R., and Fung, P. (2019a). Transferable multi-domain state generator for task-oriented dialogue systems. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 808–819.

Wu, C.-S., Socher, R., and Xiong, C. (2019b). Global-to-local memory pointer networks for task-oriented dialogue. *ICLR*.

Wu, Q., Zhang, Y., Li, Y., and Yu, Z. (2019c). Alternating recurrent dialog model with large-scale pre-trained language models. *arXiv preprint arXiv:1910.03756*.

Wu, S. and Dredze, M. (2020). Are all languages created equal in multilingual bert? In *Proceedings of the 5th Workshop on Representation Learning for NLP*, pages 120–130.

Xia, Y., Qin, T., Chen, W., Bian, J., Yu, N., and Liu, T.-Y. (2017). Dual supervised learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3789–3798. JMLR. org.

Yaman, S., Deng, L., Yu, D., Wang, Y.-Y., and Acero, A. (2008). An integrative and discriminative technique for spoken utterance classification. *IEEE Transactions on Audio, Speech, and Language Processing*, 16(6):1207–1214.

Yang, W., Qiao, R., Qin, H., Sun, A., Tan, L., Xiong, K., and Li, M. (2019). End-to-end neural context reconstruction in Chinese dialogue. In *Proceedings of the First Workshop on NLP for Conversational AI*, pages 68–76, Florence, Italy. Association for Computational Linguistics.

Yang, Y., Li, Y., and Quan, X. (2021). Ubar: Towards fully end-to-end task-oriented dialog system with gpt-2. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14230–14238.

Yao, K., Zweig, G., Hwang, M., Shi, Y., and Yu, D. (2013). Recurrent neural networks for language understanding. In *INTERSPEECH, Lyon, France*, pages 1–5. ISCA.

Ye, H., Li, W., and Wang, L. (2019). Jointly learning semantic parser and natural language generator via dual information maximization. *arXiv preprint arXiv:1906.00575*.

Young, S. (2002). Talking to machines (statistically speaking). In *Seventh International Conference on Spoken Language Processing*.

Young, S. (2007). Cued standard dialogue acts. *Report, Cambridge University Engineering Department, 14th October*, 2007.

Young, S., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., and Yu, K. (2010). The hidden information state model: A practical framework for pomdp-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.

Young, S., Gašić, M., Thomson, B., and Williams, J. D. (2013). Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Yu, K., Mairesse, F., and Young, S. (2010). Word-level emphasis modelling in HMM-based speech synthesis. In *2010 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 4238–4241. IEEE.

Yu, K., Zen, H., Mairesse, F., and Young, S. (2011). Context adaptive training with factorized decision trees for HMM-based statistical parametric speech synthesis. *Speech communication*, 53(6):914–923.

Yu, T., Li, Z., Zhang, Z., Zhang, R., and Radev, D. (2018). Typesql: Knowledge-based type-aware neural text-to-sql generation. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2*, pages 588–594.

Zen, H., Agiomyrgiannakis, Y., Egberts, N., Henderson, F., and Szczepaniak, P. (2016). Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices. In Morgan, N., editor, *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016*, pages 2273–2277. ISCA.

Zen, H., Tokuda, K., and Black, A. W. (2009). Statistical parametric speech synthesis. *speech communication*, 51(11):1039–1064.

Zettlemoyer, L. S. and Collins, M. (2012). Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *arXiv preprint arXiv:1207.1420*.

Zhang, J., Hashimoto, K., Wu, C., Wang, Y., Yu, P. S., Socher, R., and Xiong, C. (2020). Find or classify? dual strategy for slot-value predictions on multi-domain dialog state tracking. In Gurevych, I., Apidianaki, M., and Faruqui, M., editors, *Proceedings of the Ninth Joint Conference on Lexical and Computational Semantics, *SEM@COLING 2020, Barcelona, Spain (Online), December 12-13, 2020*, pages 154–167. Association for Computational Linguistics.

Zhang, X. and Wang, H. (2016). A joint model of intent determination and slot filling for spoken language understanding. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, pages 2993–2999. AAAI Press.

Zhang, Y., Ou, Z., and Yu, Z. (2019). Task-oriented dialog systems that consider multiple appropriate responses under the same context. *AAAI*.

Zhao, T. and Eskenazi, M. (2018). Zero-shot dialog generation with cross-domain latent actions. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue*, pages 1–10.

Zhao, T., Xie, K., and Eskenazi, M. (2019). Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. In *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 1208–1218.

Zhong, V., Xiong, C., and Socher, R. (2018). Global-locally self-attentive encoder for dialogue state tracking. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1458–1467.

Zhou, L. and Small, K. (2019). Multi-domain dialogue state tracking as dynamic knowledge graph enhanced question answering. *arXiv preprint arXiv:1911.06192*.

Zhu, Q., Zhang, Z., Fang, Y., Li, X., Takanobu, R., Li, J., Peng, B., Gao, J., Zhu, X., and Huang, M. (2020). Convlab-2: An open-source toolkit for building, evaluating, and diagnosing dialogue systems. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 142–149.

Zilka, L. and Jurcicek, F. (2015). Incremental LSTM-based dialog state tracker. In *2015 Ieee Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 757–762. IEEE.

Zue, V., Seneff, S., Glass, J. R., Polifroni, J., Pao, C., Hazen, T. J., and Hetherington, L. (2000). Juplter: a telephone-based conversational interface for weather information. *IEEE Transactions on speech and audio processing*, 8(1):85–96.

Appendix A

Derivation of Lower Bounds

Here we provide the derivations of lower bounds described in Section 4.4, starting with $\log p(x,y)$:

$$\begin{aligned}
\log p(x,y) &= \log \int_z p(x,y,z) \\
&= \log \int_z \frac{p(x,y,z)q(z|x)}{q(z|x)} \\
&= \log \int_z \frac{p(x|z,y)p(y|z,x)p(z)q(z|x)}{q(z|x)} \\
&= \log \mathbb{E}_{q(z|x)} \frac{p(x|z,y)p(y|z,x)p(z)}{q(z|x)} \\
&\geq \mathbb{E}_{q(z|x)} \log \frac{p(x|z,y)p(y|z,x)p(z)}{q(z|x)} \\
&= \mathbb{E}_{q(z|x)}[\log p(x|z,y) + \log p(y|z,x)] - \mathrm{KL}[q(z|x)||p(z)]
\end{aligned} \tag{A.1}$$

where $q(z|x)$ is an approximated posterior. This derivation gives us Equation 4.9. Similarly we can derive an alternative lower bound of Equation 4.10 by introducing $q(z|y)$ instead of $q(z|x)$.

For marginal log-likelihood $\log p(x)$, discussed in Section 4.4.2, its lower bound is derived as follows:

$$
\begin{aligned}
\log p(x) &= \log \int_y \int_z p(x,y,z) \\
&= \log \int_y \int_z \frac{p(x|z,y)p(y)p(z)q(z|x)q(y|z,x)}{q(z|x)q(y|z,x)} \\
&= \log \mathbb{E}_{q(y|z,x)} \mathbb{E}_{q(z|x)} \frac{p(x|z,y)p(y)p(z)}{q(z|x)q(y|z,x)} \\
&\geq \mathbb{E}_{q(y|z,x)} \mathbb{E}_{q(z|x)} \log \frac{p(x|z,y)p(y)p(z)}{q(z|x)q(y|z,x)} \\
&= \mathbb{E}_{q(y|z,x)} \mathbb{E}_{q(z|x)} \log p(x|z,y) - \mathrm{KL}[q(z|x)||p(z)] - \mathrm{KL}[q(y|x,z)||p(y)]
\end{aligned}
\tag{A.2}
$$

Note that the resulting lower bound consists of three terms: a reconstruction of x, a KL divergence which regularises the space of z, and also a KL divergence which regularises the space of y. We have dropped the last term in the optimisation objective (Equation 4.12), since we do not impose any prior assumption on the output space of the NLU model. Analogously we can derive the lower bound for $\log p(y)$, leading us to Equation 4.13.

Appendix B

Query Rewrite Annotation Guideline

As mentioned in Chapter 7, the annotation guideline for query rewriting on the MuDoCo dataset (Martin et al., 2020) is provided below.

Overview

In this project, you will be given a conversation between a user and a virtual assistant. Certain parts of the utterances (both user and assistant) might require previous context to be fully understood. The goal of the project is to make minimal changes to the current turn so that it can be understood independently, without needing access to the prior context. This may mean one (or more) of several things.

- Replacing a **pronoun** with its full referent
- Spelling out the content of an **elided phrase**
- Adding elements mentioned in the prior context that are now a part of the understood **Common Ground** of the conversation.

Here is a simple example to get us started. This is a multi-turn conversation that contains a referring expression "that week". If we only had access to the **current turn**, we would not be able to resolve the meaning of "that week"; also, we would incorrectly resolve the intended location of the question to the speaker's location instead of to Sao Paulo, as the previous context makes clear is actually intended.

> User: Will next week be a good time to vacation in Sao Paulo?
> Assistant: There will be thunderstorms and a lot of rain.
> User: What will the temperature be like that week? (current turn)

Our goal in this project is to rewrite **those parts of the current turn that require context**. Here, we can replace "that week" with "next week", and insert "in Sao Paulo" to allow the correct location to be inferred from the request, as below:

> User: Will next week be a good time to vacation in Sao Paulo?
> Assistant: There will be thunderstorms and a lot of rain.
> User: What will the temperature be like next week in Sao Paulo? (current turn)

Case by Case Guidelines

We introduce our reference / ellipsis resolution guideline in the following categories:

1. Do rewrite ellipsis as well as references
2. Do not paraphrase or summarize but do use the phrasing that appeared in the context
3. A special case exception: calls, messages, reminders
4. Multiple references
5. Data errors

To formally define what an appropriate resolution of a reference / ellipsis is, we introduce the concept of **minimal changes**, which has the following properties we will illustrate with examples.

1. You may add information that was explicitly uttered in one of the previous turns of the conversation.
2. You should **not** add more information than necessary to make the utterance understandable without additional context, even if such information is available.
3. Special case exceptions references to calls, messages, and reminders don't need additional resolution beyond reference to "the call", "the message", or "the reminder".

Do rewrite ellipsis as well as references

Example 1:

> User: Can we expect rain anytime soon?
> Assistant: Rain is possible early next week.
> User: How much chance of rain will there be? (current turn)

As we can see, the intended time of the current utterance is understood to be "early next week", since that is the time that is added to the common ground in a previous turn in the conversation as relevant to this weather inquiry. However, if we don't have this information, we would naturally interpret the time of the current turn as "now'". Consequently, we make a minimal change to the current turn by inserting this time information.

Desired: How much chance of rain will there be next week?

Example 2:

User: It's freezing! Is it going to snow?
Assistant: There's a 65 percent chance of snow tonight.
User: What about this weekend? (current turn)

In this case, the user elided the entire clause "is it going to snow", which we should recovers.

Desired: Is it going to snow this weekend?

Example 3:
We may also see cases of noun phrases (NP) with some form of ellipsis. These are not implicit arguments but rather explicitly mentioned, but have some partially missing information. In such a case we want to fill those out according to our rules to ensure that they are recoverable based on the current turn alone.

User: Play Somewhere Over the Rainbow.
Assistant: Ok, playing Somewhere Over the Rainbow by Judy Garland.
User: Can I hear the version by Ariana Grande? (current turn)

The relevant NP here is "the version", which we want to spell out explicitly as "the version of Somewhere Over the Rainbow".

Desired: Can I hear the version of Somewhere Over the Rainbow by Ariana Grande?

Do not paraphrase or summarize but do use the phrasing that appeared in the context

Do not rephrase or summarise. Instead, stick to the phrasing that appears in the context as much as possible. We give several examples of this below.

Example 1:

User: What is the latest news on the Brewers?
Assistant: There are two recent stories. One is about their lose last night, and ..
User: What happened to him?
Assistant: He stepped on his Jesus Aguilar foot.
User: Then what happened?
Assistant: The Brewers bench cleared because they were mad at him.
User: Can you send more about this to my screen? (current turn)

In this case, the desired behavior is to simply resolve "this" into "this story", rather than providing a paraphrase that collects information from all of the previous turns to create a phrase that wasn't actually uttered in the conversation at all. There is a single story ongoing in this context, so there's not need for a fuller description of this entity.

Desired: Can you send more about **this story** to my screen?
Undesired: Can you send more about The Brewers bench cleared because they ...

Example 2:

User: Can you play Moonlight Sonata by Beethoven?
Assistant: Yes, you have the live recording Moonlight Sonata.
User: I don't want that version. Can you search the best version?
Assistant: Yes, the best version is from 1914 and can stream from Youtube.
User: Ok, stream and save to my playlist under Relaxing. (current turn)

There is an implied argument here (roughly, stream it and save it ...), which we can recover from the context. Here, we choose the simplest form we can use to complete the utterance, and do not add information across turns.

Desired: Ok, stream the Moonlight Sonata by Beethoven and save it to my playlist under Relaxing.
Undesired: Ok, stream the 1914 version of Moonlight Sonata by Beethoven from Youtube ...

A special case exception: calls, messages, reminders

For these entities, you don't need to resolve nominal ellipsis or write out its content in full, just call these entities "the call", "the message", "the reminder". This is an exception to the

rules above – there is no need to provide additional information even if it present in the exchange.

Example 1:

User: Who is calling me right now?
Assistant: Marty and Brad are calling you via conference call.
User: Answer the call but change it to voice only. (current turn)

Reference is already made to "the call", and there is only one call referenced in this exchange, so nothing needs to be done here.

Desired: Answer the call but change it to voice only.
Undesired: Answer the conference call from Marty and Brad but ...

Example 2:

User: I need to send a message.
Assistant: Ok, to who?
User: To my yoga class.
Assistant: Ok, what do you want to say?
User: Let them know that the class is on tomorrow.
Assistant: Ok, I have sent your yoga class the message. (current turn)

Likewise, there is just one message being discussed here, so there is no need for resolution or added information.

Desired: Ok, I have sent your yoga class the message.
Undesired: Ok, I have sent your yoga class the message that the class is on for tomorrow.

Example 3:

User: Is there a reminder this afternoon?
Assistant: There is a reminder for your son's school.
User: What does it say? (current turn)

Here we have a pronoun that needs to be resolved to a noun phrase. The simple noun phrase "the reminder" is all that's needed, please don't add information about the content of the reminder, since this reminder is not ambiguous with any other reminder in the context.

Desired: What does the reminder say?
Undesired: What does the reminder for my son's school say?

Multiple references

If there are multiple references towards the same entity, you only need to resolve it in the first one. If there is a reference towards an entity in the utterance itself, you don't need to resolve it at all since it's already fully interpretable from that turn alone.

Example 1:

> User: Send a message to Stephen Bradley.
> Assistant: Ok, what is the message?
> User: Ask him if he wants in on the office lottery pool. (current turn)

The current turn contains two pronouns referring to the same individuals. Once the first pronoun is resolved, it provides a reference for the second pronoun within the same sentence. Therefore, we only need to resolve that first instance (in fact, resolving the second pronoun leads to an unnatural paraphrase that should always be avoided).

> Desired: Ask Stephen Bradley if he wants in on the office lottery pool.
> Undesired: Ask Stephen Bradley if Stephen Bradley wants in on the office lottery pool.

Example 2:

> User: Set a reminder.
> Assistant: Ok, what for?
> User: Remind me to call my daughter for her wedding. (current turn)

The pronoun "her" receives a referent within the current turn (namely, "my daughter"), so there is no need to do anything here. You should skip, and not spell out the referent of the pronoun.

> Desired: Remind me to call my daughter for her wedding.
> Undesired: Remind me to call my daughter for my daughter's wedding.

Data Errors

Some times the request itself could be faulty, here's a couple of cases.

Example 1:

When there is ambiguity in the reference / nominal ellipsis, you should skip.

> User: Who called me while I was in class?
> Assistant: You have a missing call from James and also Sam.
> User: Ok, call him back and set my status to online. (current turn)

Here since there are two entities, "James" and "Sam", both common male names, it is impossible to determine its referent. You should skip in this case.

Desired: Ok, call him back and set my status to online.
Undesired: Ok, call James back and set my status to online.