LEARNING TO CONVERSE WITH LATENT ACTIONS

Tiancheng Zhao

CMU-LTI-19-005

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Maxine Eskenazi, Chair (Carnegie Mellon)
William Cohen (Carnegie Mellon)
Louis-Philippe Morency (Carnegie Mellon)
Dilek Hakkani-Tur (Amazon)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

Keywords: dialog systems, end-to-end models, deep learning, reinforcement learning, generative models, transfer learning, zero-shot learning

This work is dedicated to my beloved wife and parents who unconditionally encouraged me to pursue for my passion.

Acknowledgments

I am extremely grateful for my journey at Carnegie Mellon as a graduate student. First of all, I am deeply indebted to my Ph.D. advisor Professor Maxine Eskenazi, who introduced me to the fascinating field of dialog system research and transformed me from a fresh college graduate into a experienced computer science researcher who is ready to take on any hard challenges. In the last four years, Professor Eskenazi has provided me endless guidance, which shaped my style of research and my ways of solving problems.

I would also like to express my deepest appreciation to my thesis committee, Professor William Cohen, Professor Louis-Phillippe Morrency and Dr. Dilek Hakkani-Tur. Thank you for supporting for my thesis work and giving me priceless suggestions from both high-level ideas to low-level technical details. I greatly appreciate your valuable input.

My experience as a graduate student was wonderful. I have learned so much new knowledge about machine learning and natural language process. Just to name a few: I would like to thank Professor Abeer Alwan who has taught me speech processing, Professor Alan W Black who has taught me machine translation, Professor Ruslan Salakhutdinov who has taught me about Variational Autoencoders and Professor Roni Rosenfeld who has taught me language models and the principle of maximum entropy. I am also grateful to have the chance to get connected and work with many smart people in research projects. Special thanks to Dr. Kyusong Lee, Ran Zhao, Zhiting Hu and Professor Zhou Yu. Collaboration with you was fun and memorable.

Attending conferences was also a great part of joy as a graduate student. I am thankful to get know other colleagues who share similar passion and conduct great research in my field study, such as Dr. Gokhan Tur, Professor David Traum, Professor Milica Gasic, Dr. Bing Liu, Dr. Eddy Pei-Hao Su, Pawel Budzianowski, Eli Pincus, Dr. Stefan Ultes and many others. I also had great pleasure of discussing and collaborating with you.

Finally I would like to thank my family, especially my dear mother and father, who brought me to this beautiful world and unconditionally loved and guided me. Thank you, Yilian, my wonderful wife and best friend. Thank you for always helping me think clearly, for helping me find the answers to my questions, and for giving me the courage to try. Last but not least, I would like to thank my son who came to this world just a few weeks before the completion of this thesis. You are the best.

Abstract

The study of *actions* has been on the frontier of dialog research since day one. Rooted in speech act theory (Austin, 1962), actions represent the basic communication unit and define the types of interactions that a dialog agent is capable of. This dissertation begins with the goal of developing domain-agnostic dialog models that can learn to converse with induced action representations. Achieving this first requires the models to be expressive and general purpose so that we can create dialog agents in many different domains via the same framework. It then requires the model to produce semantic representations that encode actions in natural conversations and fulfill the requirements of real-world dialog systems. Unfortunately, current methodologies to create dialog systems are not adequate to achieve this goal. The classical frame-based dialog pipeline have assumed the actions are pre-defined by expert handcrafting, which struggle to generalize to complex domains. More recent end-to-end (E2E) dialog models based on encoder-decoder neural networks are designed to be not restricted by hand-crafted semantic representations. Unfortunately, it is far from trivial to build a full-fledged dialog system using encoder-decoder models and they suffer from a range of limitations. Moreover, current E2E models only focus on the final response word outputs and pays little attention to the action representation.

This dissertation advocates a new family of E2E dialog models based on *latent actions*. Latent actions model the hidden actions in raw conversations as latent variables and make it possible to learn explicit action representations at scale. Concretely, a general latent action framework is defined and detailed, including desired properties, optimization techniques, and we developed novel solutions to efficiently discover latent actions from large datasets and seamlessly integrate the resulting latent actions into E2E neural dialog models. Then four different types of latent action are created to address major limitations that current E2E dialog systems are facing: (1) the dull response problem where models tend to generate generic responses, (2) poor interpretability where E2E models cannot be easily interpreted (3) limited domain generalization where deep models requires a lot of in-domain training data and (4) strategy optimization where is it challenging to apply reinforcement learning for E2E models. This work shows that the above-mentioned challenges can naturally be solved by using latent actions and significant empirical performance gain can be observed. The proposed framework also offers a new perspective to create E2E dialog models that focus on action representation, which enables new research that connects to other subjects, e.g., sentence representation learning, zero-shot learning etc. This research is a first step towards bridging the classic dialog action research to neural E2E models, and lays the foundation for building dialog systems that can accomplish more complex tasks, understand and reason as human do.

Contents

List of Figures

¡

List of Tables

List of Abbreviations

AI Artificial Intelligence
ASR Automatic Speech Recognition
CVAE Conditional Variational Autoencoder
DM Dialog Management
E2E End-to-end
ELBO Evidence Lower bound
GRU Gated Recurrent Unit
KB Knowledge Base
LaRL Latent Action Reinforcement Learning
LCR Language Constrained Reward Curve
LSTM Long Short-term Memory
MDP Markov Decision Process
MI Mutual Information
MLE Maximum Likelihood Estimation
MTL Multi-task Learning
NLG Natural Language Generation
NLP Natural Language Processing
NLU Natural Language Understanding
PG Policy Gradient
POMDP Partially Observable Markov Decision Process
RL Reinforcement Learning
RNN Recurrent Neural Network
ST Skip Thought
SVI Stochastic Variational Inference
TL Transfer Learning
TTS Text-to-speech
VAE Variational Autoencoder

ZSDG Zero-shot Dialog Generation
ZSL Zero-shot Learning

Chapter 1

Introduction

1.1 Overview

Teaching machines to converse like a human for real-world applications is arguably one of the hardest challenges in Artificial Intelligence (AI). To carry out natural and meaningful conversation with a human, a dialog system needs to be competent in understanding natural language, making intelligent decisions, and generating appropriate responses. Moreover, unlike many other natural language processing (NLP) tasks, a dialog system is a sequential decision-making problem that needs to look ahead and plan about the future (Barto et al., 1989). Through decades of research and development, dialog systems have evolved tremendously and have been transformed from research projects into commercial systems that are constantly used by hundreds of millions of people every day. Apple Siri, Amazon Alexa, Google Home, and Microsoft Xiao Ice (Zhou et al., 2018) are just a few examples of planet-level dialog systems to showcase this steady growth of conversational interaction between human and machine. In the meantime, the state-of-the-art dialog systems are still very preliminary in terms of matching up to the conversational ability of human, i.e., they mostly can only talk about specific topics with limited memory and functionalities. New methodologies and frameworks are urgently needed to advance the capability of current dialog agents further.

A central topic of dialog research is the meaning representation of utterances, i.e. the basic unit of communications between two interlocutors (Searle et al., 1980; Allen et al., 2001). People have also used the term *actions* to denote the system utterances that an agent can output to human users (Williams and Young, 2003). Modeling actions in a dialog system is very challenging due to the unbounded nature of natural language. In principle, the output action space for a dialog system is infinite in order to appropriately respond to all possible conversational context. The classic dialog systems (Young, 2006) takes a divide-and-conquer approach, i.e. dividing the process of dialog into natural language understanding (NLU), dialog management (DM) and natural language generation (NLG). Then hand-crafted utterance (action) representations are used as the interface between NLU and DM, and between DM and NLG. That is the NLU recog-

nize the user input into user actions, which are processed in the DM, which outputs the next system action. Finally, the system action is transduced into natural language. For example, in CMU Let's Go Bus Information system (Raux et al., 2005), one of the early real-world dialog deployments, has utilized dialog-act based meaning representations. An example action can be: *Implicit Confirm (Departure Time=now), Inform (Bus=28X; Arrival Time=7pm)*. Dialog-act based representations contain one or more dialog acts for propositional function and a set of slot arguments to capture the propositional content. This approach has proven to be effective in simple task-oriented domains where it's feasible to enumerate most of the valid outputs from the systems. Yet this approach struggles to generalize to more complex domains or transfer to other domains because of the limitation of hand-crafted symbolic representation (Joty et al., 2011).

The recent revolution in deep learning has suggested an alternative path to model conversations, i.e., developing end-to-end models that do not use hand-tailored intermediate interfaces. This philosophy strives to create domain-agnostic models and learning algorithms that can learn useful intermediate representations by itself from data. Following this approach, deep learning models have become the state-of-the-art methods in a wide range of NLP tasks, including language modeling (Mikolov et al., 2010), syntax parsing (Socher et al., 2011), speech recognition (Hinton et al., 2012), neural machine translation (Cho et al., 2014), image captioning (Xu et al., 2015), entity recognition (Lample et al., 2016) etc. Pioneer work in end-to-end (E2E) dialog models has used encoder-decoder neural networks (Sordoni et al., 2015; Vinyals and Le, 2015) and formulates a dialog system as a response generation task: encode the dialog context via an encoder network to dense vector representations, and then decode the next system reply via a decoder network. The encoder and decoder networks are trained jointly via maximizing the word log likelihood on training dialogs without the need for hand-crafting. In this setting, the actions are implicitly modeled in the hidden representations of the decoder network.

Unfortunately, despite the undeniable advantages and promising research results, E2E models have their problems. For example, E2E dialog models are black box models that are difficult to control and challenging to interpret, and these proprieties are often required for real-world dialog applications. E2E models also suffer from the issue of dull response, i.e. the models are only able to generate generic, not engaging responses despite the fact of the large training corpus. One more problem is that E2E models are notoriously data hungry whereas most of the dialog domains are scarce of abundant data, and it is impossible to collect a large dialog dataset for every possible domain due to its universal applicability. As we can see, these challenges cannot be solved merely by collecting more data or using more computation power with larger models. Thus, before we can address these challenges with a fundamentally novel solution, E2E dialog models are still far away from becoming the standard solution to create dialog systems.

The goal of this dissertation is to develop novel methods to enable us to create dialog systems that can generalize to complex domains and accomplish more challenging goals through conversation with human users. In order to achieve this goal, we build a new family of E2E dialog models that is built upon the notion of *latent actions*. We will show that this novel framework can achieve superior performance compared to vanilla

E2E systems in the above mentioned challenges, while adequately maintaining the benefits of E2E systems compared to the classic pipeline approach. We define latent actions as the hidden discourse-level intents that the system-side speaker has used in the raw conversational data. Unlike vanilla E2E system that naively hope such abstraction over high-level decisions can be learned from next utterance prediction automatically, we develop a set of novel algorithms to encourage the models to learn helpful and meaningful latent representations about the next utterance, such as representations that are transferable across domain which in turn reduces the training size needed for domain adaption.

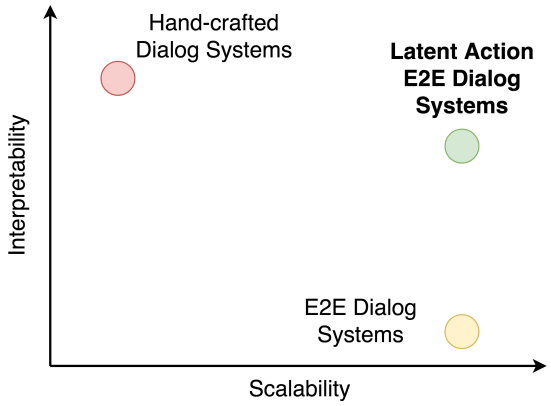

Figure 1.1: Illustration of the relationships between the proposed latent action approach and hand-crafted systems or current end-to-end systems in terms of interpretability and scalability.

There are many reasons why learning with latent actions is desired. One of them is that human intelligence is superb in operating with abstractions, or "high-level" actions, that span over multiple timestamps of primitive actions. This enables human to generalize better to unseen situations and obtain reusable knowledge to new domains (Parr and Russell, 1998). This principle also applies to natural language. Research in plan recognition (Litman and Allen, 1987) and natural language generation (Hovy, 1990; Rambow et al., 2001) suggest that human produce their utterances during a conversation in a hierarchical fashion with different level abstractions from "what to say" to "how to say it". Building systems with such hierarchy has enabled people to optimize dialog strategy in parallel with improving the natural language generation quality and provides an interpretable interface for researchers to understand the success or failure of a dialog agent.

Another reason is that the introduction of latent action creates a bridge to connect the classic symbolic dialog research with recent deep learning based NLP models. Latent variables can be used to correspond to the "dialog acts", "intentions" that were manually designed in classic research, and now can be inferred from data as part of a neural network. Also, the latent actions are modelled as probabilistic latent variables, which enable researchers to utilize techniques from Bayesian machine learning, variational inferences etc to improve the learning process of latent action E2E models. This

can not only give us a better modeling of the actual underlying dialog process, but also make neural dialog systems more explainable with human understandable interface. Figure 1.1 shows the benefits of latent actions in terms of interpretability and scalability compared to current methods.

Last but not least, as this thesis will show, having an explicit representation of system actions in E2E dialog models can lead to various empirical performance gain and bring novel features compared to standard E2E systems. Unlike standard E2E models that solely aims to optimize the entire system for the main learning objective, latent variable E2E systems creates opportunities for developers to gain insights and incorporate additional knowledge, while remaining to be end-to-end trainable. These unique features make latent action E2E dialog system powerful and practical for creating dialog systems in a variety of usage and domains.

1.2 Thesis Statement

In this dissertation, we advocate a new family of E2E dialog systems centered around latent actions, by proposing novel inference algorithms to infer latent system intentions from raw conversational data and incorporating them into the response generation process of a decoder neural network. We argue that this family of E2E models can combine the best properties of classic hand-crafted dialog systems with the ones of current encoder-decoder dialog models, but also yield entirely new proprieties that neither prior systems can achieve.

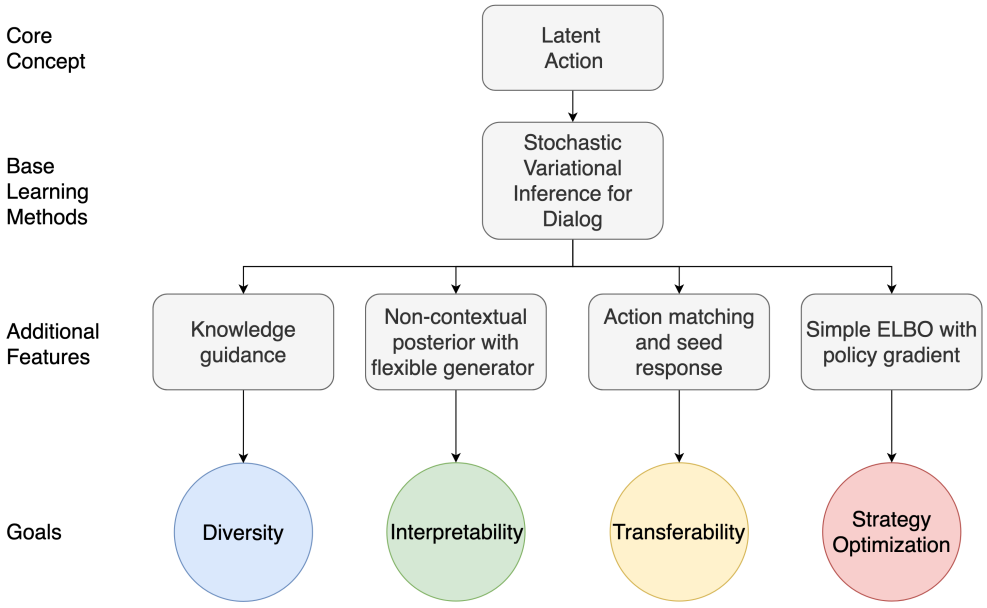

Figure 1.2: A high-level overview of the proposed latent action framework and the four pivot topics.

Specifically, we first define the framework of latent action for dialog systems. Then a

set of novel unsupervised learning algorithms based on stochastic variational inference is developed to train neural text generation systems with latent variables for a given dialog dataset. Based on these definitions, system architecture and optimization methods, we create four types of latent actions to solve the following challenges: 1) a stochastic continuous latent action that is designed to capture the distribution over discourse-level intentions and results in diverse response generation for open-domain conversation 2) a stochastic discrete latent actions that are easy for human to understand for model interpretability 3) a cross-domain latent action that is used to establish alignments between similar dialog moves from different domains and enable zero-shot domain transfer, 4) a latent action-based reinforcement learning framework that optimizes the dialog policy over the induced latent actions space. Each of the proposed latent action addresses one essential challenge of current E2E dialog modeling and they can be stacked together to provide solutions as a whole. Last but not least, the bigger picture is that this dissertation demonstrates how explicit latent variable can be incorporated into deep neural natural language generation systems. Also, it shows how such synergy between latent variable and neural networks can improve system performance, provide useful scientific insights, and open doors to new research topics that leverage ideas from other fields of study, such as zero-shot learning, discourse analysis, etc. We hope these promising directions can encourage brand-new methodologies to be developed and advance dialog and natural language processing research as a whole.

1.3 Thesis Structure

The rest of the dissertation is organized as follows:

- **Chapter 2: Foundational Work**
  This chapter gives an overview of related research areas, including both work about dialog system and related work in machine learning.

- **Chapter 3: The Latent Action Framework**
  This chapter defines the proposed the latent action approach and describes the background. We lay out the overall structure, including problem definition, evaluation metric, basic system components and advantages of latent actions.

- **Chapter 4: Learning Methods for Latent Actions**
  This chapter develops neural network architectures that create the base form of the defined latent action framework. Then we present machine learning algorithms based on stochastic variational inference to train these neural networks from raw conversational data. In particular, we show that the posterior collapse problem is a key challenge for training our proposed architectures and this chapter presents a set of novel techniques to solve the posterior collapse problem, making the model learn better latent action representations.

- **Chapter 5: Latent Actions for Discourse-level Diversity**
  This chapter demonstrates how to use latent action framework to solve the well-known dull response problem in E2E open domain chatting system. Besides using the base latent action models, we propose to use linguistic knowledge to guide the learning of latent actions and achieve significant better performance. Also, this chapter proposes a novel evaluation metric to overcome the difficulties of assessing open-domain chatting systems.

- **Chapter 6: Discrete Action Representation for Model Interpretability**
  Vanilla E2E systems are black-box systems that do not provide explainable interfaces for human to understand its internal operations. This chapter first shows that non-contextual posterior distribution is desired to learn interpretable latent variables. Then we create discrete latent actions with two distinct learning signals in order to offer a human-readable interface for reading an E2E dialog model's intention of the next response, while maintaining its ability to be trained on unlabelled dialog data

- **Chapter 7: Zero-shot Generalization with Cross-Domain Latent Action**
  This chapter presents a cross-domain latent action that enables model to transfer utterance-level knowledge from source domains to target domains, which can greatly reduce the data needed to train an E2E system for a new domain. We show that this in fact enables zero-shot transfer if two domains share similar discourse-level structure, even though the lexical distributions are completely different at utterance level.

- **Chapter 8: Dialog Strategy Optimization with Latent Action Reinforcement Learning**
  This chapter extends the latent action from supervised learning to reinforcement learning. We propose a new paradigm of training E2E models via reinforcement learning. The main novelty is to learn an induced new action space for optimizing discourse-level dialog strategy given task specific reward signals. Whereas the traditional approach tunes the dialog policy at word-level from decoder outputs.

- **Chapter 9: Conclusion and Future Work**
  This chapter concludes the main contributions and discusses a number of interesting directions that can be explored in the future.

Chapter 2

Foundational Work

This chapter presents an overview of the prior research that this work paves the foundation for this dissertation. We will first go over the background of existing models for building dialog systems and previous state-of-the-art approaches. Then we will summarize machine learning techniques that the rest of this dissertation will build on.

2.1 Dialog Systems Foundations

2.1.1 Speech Acts

Before discussing practical computational solutions for dialog systems, we will briefly discuss the linguistic foundations. The study of action has a long history. The theory of *speech acts* is first developed by Austin in the field of linguistics and language philosophy (Austin, 1962). Under this setting, utterances in conversations are actions that are used to change the mental and interactions state of the speakers. Austin distinguishes several types of actions that are performed when a speaker produces utterances. They are:

1. *Locutionary acts* the literal meaning and structure of an utterance.
2. *Illocutionary acts* the intended meaning of an utterance. Illocutionary acts are also divided into two parts: illocutionary force (the type of action, e.g. statement, promise, request) and illocutionary content, which specifies the details of an action. Many of dialog system frameworks have focused on creating action representation for capture illocutionary acts, which will be discussed in the next section.
3. *Perlocutionary acts*: the effect achieved by the actions, such as persuading, convincing etc.

Besides theoretical work in studying human-human conversations, there have been many pioneering works in developing computational models for speech acts in dialog based on classic AI expert systems to model various behavior that is needed for a dialog systems, including grounding (Clark et al., 1991; Clark, 1996; Traum, 1999), plan recognition and execution (Kautz and Allen, 1986; Litman and Allen, 1987) and mental

state modeling (Larsson and Traum, 2000) etc. These work serve the foundation of our understanding about the actions in human-human conversations.

Since then, there have been a constant effort in creating practical dialog systems that can accomplish real-world tasks, which is also the focus of this dissertation. In terms of usage, the research in dialog systems can be roughly divided into task-oriented systems and chat-oriented systems: task-oriented system are designed to achieve certain goals, e.g. flight booking, hotel recommendation etc. Chat-oriented dialog systems are designed to carry out open-domain conversations, so that are not restricted to a certain domain or a specific goal. Chat-oriented dialog systems have been mainly used for entertainment and social chat. Since the unlimited scope of chat-oriented dialog system, it is more difficult than a task-oriented dialog system in one domain. In the following sections, we will describe some of the most popular and related methods to create these two types of dialog systems.

2.1.2 Frame-based Dialog Systems

Frame-based dialog systems are one of the most successful frameworks to create task-oriented dialog systems (Glass et al., 1999; Young, 2006; Raux et al., 2005), and are sometimes also referred as slot-filling dialog systems. In this setting, a dialog state frame (aka form) is designed to contain every information needed to accomplish its goal, e.g. departure place, arrival place for a flight booking system (Glass et al., 1999). Through conversation, the dialog agent needs to acquire these missing information from users, aka. "slot filling" and provides the correct information to users once sufficient intelligence is gathered. Frame-based dialog system often consists of 3 major components as shown in Figure 2.1:

1. Natural language understanding (NLU): parses user utterances to semantic frames that represent user-side speaker's actions.
2. Dialog manager (DM): maintains the dialog state and decides the system next action
3. Natural language generation (NLG): generates a natural language sentence based on the DM's decision of next move.

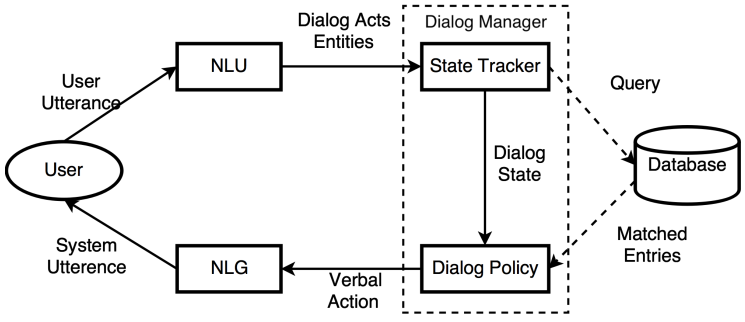

Figure 2.1: Dialog system pipeline for task-oriented dialog systems

The frame system has evolved several times from its initial design. TrindiKit (Larsson et al., 1999) defines a formalism to consume user/system input, update information dialog state and generates the next systems moves based on the information state theory for dialog management. CMU RavenClaw is another popular dialog management framework (Bohus and Rudnicky, 2003) that adds agenda-based planning into frame-based dialog systems, which enables developers to define the dialog policy via task decomposition trees. Hidden Information State (HIS) dialog management (Young et al., 2007) uses statistical models to model dialog systems as a Partially Observable Markov Decision Making Process (POMDP) and uses reinforcement learning to optimize the decision making policy. Also, HIS systems further divides the DM into two parts: dialog state tracking (DST), which accumulates information from the entire dialog history and dialog policy (DP) that selects the next system action based on the output from the DST as shown in 2.1.

One limitation of the above approaches is that component is independently optimized and may suffer from error propagation from module to modules. Therefore, end-to-end trainable dialog models based on deep learning models are created to alleviate this problem by jointly learning all components in a frame-based system (Wen et al., 2016a; Zhao and Eskenazi, 2016; Su et al., 2016; Williams and Zweig, 2016). Wen et al., (Wen et al., 2016a) first introduced a fully differentiable network architecture that can be trained on both oracle system responses and intermediate labels jointly. After supervised training, the dialog policy of this model can be fine tuned using reinforcement learning (Su et al., 2016). Zhao et al., (2016) first used deep reinforcement learning to enable a task-oriented E2E dialog system to learn to interface with external KB and learn task oriented dialog state representations. Later work has extended this idea using soft attention to reason over knowledge base (Dhingra et al., 2017). The above approaches still retain a part of the intermediate representations (e.g. dialog acts) from the classical pipeline and combine a subset of the dialog pipeline into one E2E model.

Meanwhile, despite the rapid development of learning methods in each component of frame-based dialog systems, the basic architecture has stayed the same as shown in Figure 2.1. The dialog state frame is always hand-crafted by domain experts and the action frame for both user utterances and system actions are also manually created. The improvement in machine learning models indeed continue to improve the estimation accuracy of automatically recognizing semantic frame given raw text input (Williams et al., 2013; Mesnil et al., 2015), the whole system is limited by the hand-crafted intermediate representations so it struggles to generalize to new or more complex domains. Therefore, frame-based dialog system have been only applied to task-oriented dialog systems that operate in a constrained domain, and it is not expressive enough to model conversations in domains where actions cannot be captured by hand-crafted dialog-act frames.

2.1.3 Retrieval-based Dialog Systems

An alternative approach is retrieval-based dialog systems. The basic idea is simple, i.e. giving a query dialog context, the system searches in a database of previous dialog

contexts and responses based on certain ranking function, e.g. nearest neighbour, and returns the best matched response as the answer. Matching multi-turn dialog context is a challenging task since multi-context are extremely sparse. Therefore, early work has focused on 1-turn question answering retrieval which only matches the last user question with every question in the database using semantic relatedness measures, e.g. BM-25 (Graesser et al., 2004; Jeon et al., 2005; Banchs and Li, 2012). There have also been research effort in extending nearest neighbour approach to multi-turn dialogs by developing dialog context features extracted from long-term dialog history. This approach has successfully been applied to model multi-domain task-oriented dialogs (Lee et al., 2009; Noh et al., 2012). The more recent deep learning-based E2E retrieval dialog systems solve the response matching problem by learning neural dialog context encoders, e.g. recurrent neural networks, that are trained to rank the correct response higher probability (Nio et al., 2014; Bordes and Weston, 2016; Lowe et al., 2017; Zhou et al., 2016).

From action point of view, instead of coming up with a explicit representation of the speech acts, retrieval models takes a non-parametric approach by storing all the historical responses in the database and selects one of them at the testing time. The advantages of example-based systems are (1) purely data-driven and are not limited to hand-crafted semantic frames (2) response quality improves automatically with larger databases (3) the return responses are human generated so that always grammatical and coherent. On the other hand, example-based systems are limited in the following ways: (1) they cannot generate novel responses that are not in the database, leading to poor generalization given a limited database (2) same as other non-parametric approaches, the query time linearly increases as the database becomes bigger, slowing down the response speed at testing time.

2.1.4 Generation-based Dialog Systems

Generation-based dialog systems that builds on encoder-decoder networks (Cho et al., 2014; Vinyals and Le, 2015) are perhaps the most expressive framework to create dialog systems to date. Similar to E2E retrieval models, E2E generation-based system also utilizes neural encoder networks to transforms the raw dialog context, e.g. dialog history, external knowledge etc into distributed vector representations. Then generation systems distinguish itself by using a auto-regressive decoder, e.g. recurrent neural networks that is trained to generate the response from left-to-right, word by word. Thus, a generation system in principle can learn to generate free form natural responses and generalize to novel utterances that are not included in the training data. In the mean time, it is harder to train. E2E generation systems have been successfully applied to both task and chat-oriented domains due to its flexibility.

Hierarchical encoders (Serban et al., 2015) have been proposed to exploit the hierarchical structure in dialog and has shown better results then encoding the entire discourse history word-by-word. More fine-grained encoders (Henaff et al., 2016; Xing et al., 2017) are also proposed to better extract key elements (e.g. entities) from the discourse history. Furthermore, recent research has found that encoder-decoder models

tend to generate generic and dull responses, (e.g., I don't know), rather than meaningful and specific answers (Li et al., 2015a; Serban et al., 2016c). To tackle this problem, one line of research has focused on augmenting the input of encoder-decoder models with richer context information, in order to generate more specific responses. Li et al. (Li et al., 2016a) captured speakers' characteristics by encoding background information and speaking style into the distributed embeddings, which are used to re-rank the generated response from an encoder-decoder model. Xing et al. (Xing et al., 2016) maintain topic encoding based on Latent Dirichlet Allocation (LDA) (Blei et al., 2003) of the conversation to encourage the model to output more topic coherent responses. The second category of solution is to improve the decoding algorithm to encourage more diverse responses, including decoding with beam search and its variations (Wiseman and Rush, 2016), encouraging responses that have long-term payoff (Li et al., 2016b) or adding mutual-information loss in additional to standard maximum likelihood estimation (Li et al., 2015a). The last category of solutions is to introduce a latent random variable to model a distribution of responses given a dialog context (Serban et al., 2016c; Zhao et al., 2017b; Cao and Clark, 2017).

As for generation-based task-oriented systems, the challenges mainly lies in integrating the system with external knowledge bases and improving the performance on handling entities. Decoder network with copy mechanism was applied to task-oriented setting and shows significant improvement on entity accuracy over standard decoder with attention mechanism (Eric and Manning, 2017a). Prior work applies a pre-processing technique named entity indexing that improves the entity independence of encoder-decoder models (Zhao et al., 2017a). This approach is used to transformed a real-world system, CMU Let's Go) into E2E generation model and tested with users through spoken interface. There are two types of approach to integrate generation-based systems with external knowledge bases (KBs). The first approaches treat the external knowledge base as a part of the environment and the model is learned to interact with both human (in natural language) and KBs (in API calls) (Zhao and Eskenazi, 2016; Williams et al., 2017; Lei et al., 2018). The second approaches assumes that the generation-model has full-access to the internal entries in the database and the system is trained to access the database via some forms of attention mechanism (Dhingra et al., 2017; Eric and Manning, 2017b; Madotto et al., 2018).

2.1.5 Hybrid Dialog Systems

Hybrid Domains

Although the task-oriented and chat-oriented systems have been usually explored independently, there is pioneer work in combining the two types of systems. Research has found that interleaving chat with task-oriented systems can improve the robustness of the system against misunderstanding errors, and improve user satisfaction by keeping them engaged with the systems (Zhao et al., 2017a). Yu et al. (Yu et al., 2017b) has used reinforcement learning to learn the interleaving strategy. Other work has used social chat reasoner to improve the rapport between the computer and human in order to

develop a amicable long-term relationship (Zhao et al., 2014).

Hybrid Decoders

Neither generation nor retrieval-based systems are perfect so that there have been effort in combining the best of both worlds. One popular approach is to first retrieve N responses from the database and then use these selected response as additional input to a generation-based decoder to generate a better response (Song et al., 2016; Guu et al., 2018).

2.2 Machine Learning Foundations

2.2.1 Encoder-Decoder Models

Generative modeling is an area of machine learning research which deals with models of the distribution of data $P(X)$, where X are data points that can be high-dimensional and structured. Generative models have been extensively studied in many fields, including computer vision, natural language etc., and still remains to be one of the most exciting field of research. This section will focus on backpropagation-based generative models for natural language using neural networks, which sit at the core of this dissertation. Also, besides introducing the generic generative models, we are more interested in conditional generative model $P(X|C)$ where C is an arbitrary variable in high-dimensional space that can influence the distribution of X.

The most common yet very powerful conditional generative model for natural language is the *encoder-decoder* model (Cho et al., 2014; Vinyals and Le, 2015). The standard form of encoder-decoder models the conditional distribution of target word tokens $P(X|C)$ conditioned on a given word sequence X, which is also known as the *sequence-to-sequence* model. The basic idea is to use an encoder recurrent neural network (RNN) to encode the context sentence C into a distributed representation and then use a decoder RNN to predict the words in the target sentence X. Let the $w_i^{\mathbf{x}}$ and $w_j^{\mathbf{c}}$ to denote the i^{th} and j^{th} words in the target and context sentence respectively, and RNN^e and RNN^d to denote the encoder and decoder RNNs. Then the source sentence is encoded by recursively applying:

$$h_0^e = \mathbf{0} \tag{2.1}$$

$$h_i^e = \text{RNN}^e(w_i^{\mathbf{x}}, \mathbf{h}_{i-1}^e) \tag{2.2}$$

Then the last hidden state of the encoder RNN $h_{|\mathbf{c}|}^e$ is treated as the representation of $\mathbf{c}$, which in theory is able to encode all of the information in the context sentence. Then the initial state of RNN^d is initialized to be $h_{|\mathbf{c}|}^e$, and predicts the words in the target sentence $\mathbf{x}$ sequentially via:

$$o_j = \text{softmax}(W\mathbf{h}_\mathbf{j}^\mathbf{d} + b) \tag{2.3}$$

$$h_j^d = \text{RNN}^d(w_j^{\mathbf{c}}, \mathbf{h}_{j-1}^d) \tag{2.4}$$

where o_j is the decoder RNN's output probability for every word in the vocabulary at time step j. Also, in order to make the model predict the first word in the target sentence and predict a terminal symbol indicating the end of generation, special symbols BOS and EOS are usually padded at the beginning and end of the target sentence. Moreover, it is important to note that the encoder and decoder networks are not limited to RNNs or text word sequences. Within the scope of X being word sequences, past research has investigated a variety of encoders, including convolutional neural network (CNN) to encode visual data (Vinyals and Le, 2015), a tree encoder to encode syntactic trees (Eriguchi et al., 2016) or hierarchical RNNs to encode conversation (Serban et al., 2015) etc. Although the standard encoder-decoder models are very simple, they have achieved impressive results in a wide range of natural language processing (NLP) tasks, including machine translation (Vinyals and Le, 2015), image captioning (Vinyals et al., 2015) etc.

Memory-Augmented Encoder-Decoder

Although the standard encoder-decoder models are able to learn long-term dependencies in theory, they often struggles to deal with long-term information in practice. Attention mechanism (Bahdanau et al., 2014; Luong et al., 2015) is an important extension of encoder-decoder models that enable better modeling of long term context. The general idea is instead of asking the encoder RNN to summarize a fixed-size distributed representation of the context C, but allowing it to create dynamic size distributed representation (usually a list of fixed size vectors), and then equip the decoder RNN with a reading mechanism that can retrieve a subset of the information from the dynamic source representation. Specifically, let the dynamic representation of the context sentence be $H^e = \{h^e_1, ..., h^e_{|\mathbf{c}|}\}$. Then at each decoder step, the update function now becomes:

$$o_j = \text{softmax}(W[\mathbf{h_j^d}, \mathbf{m_j^e}] + b) \tag{2.5}$$

$$m^e_j = \sum_i^{|\mathbf{c}|} \alpha^i_j \mathbf{h_i^e} \tag{2.6}$$

$$\alpha^i_j = f(\mathbf{h_i^e}, \mathbf{h_j^d}) \tag{2.7}$$

$$h^d_j = \text{RNN}^d(w^{\mathbf{x}}_j, \mathbf{h_{j-1}^d}) \tag{2.8}$$

where α is the scalar attention score computed via a matching function f which can be simple dot product, bi-linear mapping or a neural network (Luong et al., 2015).

A recent extension of attention mechanism is the copy-mechanism (Gu et al., 2016; Merity et al., 2016). Similar to the attention mechanism, the copy-mechanism also utilizes a pointer to dynamically read from a variable-length representation of the source sentence. However, rather than asking the decoder RNN to output the next word via its softmax layer, the copy-mechanism directly copies and outputs the selected word according to the attention. The main advantage of copy-mechanism is its ability to handle rare words and OOVs better in case where other encoder-decoder models fail (Merity

et al., 2016). Also past work has found that copy-mechanism results into better generalization performance when the task inherently has the copy-nature, e.g. entity references (Zhong et al., 2017; Eric and Manning, 2017a).

2.2.2 Variational Latent Variable Models

The generation of real-world data usually involves a hierarchical process. For example, given a dialog context, the speaker may first decide the high level action to respond with, e.g. ask a question or give a suggestion, and then the second stage generates the actual response in natural language which focuses on low-level factors. Such high-level decisions are often unobserved in data and are referred as *latent variables*. The objective to maximize for unconditional generation is the marginal probability of data X

$$P(X) = \int P(X|\mathbf{z};\theta)P(\mathbf{z})d\mathbf{z} \tag{2.9}$$

One of the most successful framework to model such phenomenon is the *variational autoencoder* (VAE) (Kingma and Welling, 2013; Rezende et al., 2014). The idea of VAE is to encode the input $\mathbf{x}$ into a probability distribution $\mathbf{z}$ instead of a point encoding in the autoencoder. Then VAE applies a decoder network to reconstruct the original input using samples from $\mathbf{z}$. To generate images, VAE first obtains a sample of $\mathbf{z}$ from the prior distribution, e.g. $\mathcal{N}(0, \mathbf{I})$, and then produces an image via the decoder network. To deal with the integral over the high-dimensional latent variable $\mathbf{z}$, VAE utilizes the *Stochastic Gradient Variational Bayes* (SGVB), which instead of directly optimizing the marginal log likelihood of the data, optimizes the *evidence lower bound* (ELBO), which is a lower bound of the actual data log likelihood. ELBO is usually expressed as:

$$\log P(X) \geq \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}[\log P(X|\mathbf{z})] - \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})\|P(\mathbf{z})) \tag{2.10}$$

In order to do conditional generation, VAE has been extended to the *conditional variational autoencoder* (CVAE) (Yan et al., 2015; Sohn et al., 2015). CVAE introduces a new random variable C that is given at the generation stage. The goal is then to maximize the lower bound of the conditional probability, which can be expressed as:

$$\log P(X|C) \geq \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x},\mathbf{c})}[\log P(X|\mathbf{z},C)] - \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|X,C)\|P(\mathbf{z}|C)) \tag{2.11}$$

Both VAE and CVAE were first introduced for computer vision, and were later extended to natural language. Although VAE/CVAE has achieved impressive results in image generation, adapting this to natural language generators is non-trivial. Bowman et al. (Bowman et al., 2015) have used VAE with Long-Short Term Memory (LSTM)-based recognition and decoder networks to generate sentences from a latent Gaussian variable. They showed that their model is able to generate diverse sentences with even a greedy LSTM decoder. They also reported the difficulty of training because the LSTM decoder tends to ignore the latent variable. We refer to this issue as the *posterior collapse problem*. VAEs and CVAEs play critical roles in our proposed latent action framework and are discussed in much more details starting from Chapter 3.

2.2.3 Zero-shot Learning

Zero-shot learning (ZSL) refers to an extreme situation where there is no training data available for the target domain so the label space Y is unseen in the source. Although difficult for machines, a human is indeed capable of ZSL. For example, after a person reads a detailed description about the look of a cat, this person should be able to recognize an image of cat even he/she has never seen a cat before. Therefore, the major challenge of ZSL is to construct a shared representation g of the output space Y, so that a model trained on the source $P(g|X)$ can be still used to predict meaning outputs that can be related to the new labels.

ZSL was first introduced in the computer vision community (Larochelle et al., 2008; Palatucci et al., 2009), which has focused on recognizing unseen objects from images. The major approach is to parameterize the object Y into semantic output attributes instead of directly predicting the object class index. As a result, in the test time, the model can first predict the semantic attributes of the input image. Then the final prediction can be obtained by comparing the predicted attributes with a list of candidates objects. A more recent work (Romera-Paredes and Torr, 2015) improves this idea by jointly learn a bi-linear mapping to directly fuse the information from semantic codes and input image for prediction. Besides image recognition, recent work has explored the notion of *task generalization* in robotics, so that a robot can execute a new task that is not mentioned in training (Oh et al., 2017; Duan et al., 2017). In this case, a task is described by one demonstration or a sequence of instructions and the system needs to learn to breakdown the instructions into previously learned skills. ZSL has also been applied to individual components in the dialog system pipeline. Chen et al. (Chen et al., 2016b) developed an intent classifier that can predict new intent labels that are not included in the training data. Bapna et al. (Bapna et al., 2017) extended the idea to the slot-filling module to track novel slot types. Both papers leverage a natural language description about the label (intent or slot-type) in order to learn a semantic embedding about the label space. Then, given any new labels the model can still make predictions. Moreover, there has been extensive work on learning domain-adaptable dialog policy by first training a dialog policy on K previous domains, and then testing the policy on the K+1st new domain. Gasic et al. (Gasic and Young, 2014) used the Gaussian Process with cross-domain kernel functions. The resulting policy can leverage the experience from other domains to make educated decisions in a new one. Finally, ZSL has also been applied to NLG. Wen et al. (Wen et al., 2016b) used delexicalized data to synthetically generate NLG training data for a new domain.

In summary, past ZSL research for dialog has mostly focused on adapting individual modules of a pipeline-based dialog system. We consider our proposal to be the first step in exploring the notion of adapting an entire E2E dialog system to new domains for task generalization.

2.2.4 Deep Reinforcement Learning

Reinforcement learning (RL) is used to learn dialog policies in dialog systems (Walker, 2000; Williams and Young, 2007; Gašić et al., 2010). RL models are based on the Markov Decision Process (MDP). An MDP is a tuple (S, A, P, γ, R), where S is a set of states; A is a set of actions; P defines the transition probability $P(s'|s, a)$; R defines the expected immediate reward $R(s, a)$; and $\gamma \in [0, 1)$ is the discounting factor. The goal of reinforcement learning is to find the optimal policy π^*, such that the expected cumulative return is maximized (Sutton and Barto, 1998). MDPs assume full observability of the internal states of the world, which is rarely true for real-world applications. The Partially Observable Markov Decision Process (POMDP) takes the uncertainty in the state variable into account. A POMDP is defined by a tuple $(S, A, P, \gamma, R, O, Z)$. O is a set of observations and Z defines an observation probability $P(o|s, a)$. The other variables are the same as the ones in MDPs. Solving a POMDP usually requires computing the belief state $b(s)$, which is the probability distribution of all possible states, such that $\sum_s b(s) = 1$. It has been shown that the belief state is sufficient for optimal control (Monahan, 1982), so that the objective is to find $\pi^* : b \rightarrow a$ that maximizes the expected future return.

The deep Q-Network (DQN) introduced by Mnih (Mnih et al., 2015) uses a deep neural network (DNN) to parametrize the Q-value function $Q(s, a; \theta)$ and achieves human-level performance in playing many Atari games. DQN keeps two separate models: a target network θ_i^- and a behavior network θ_i. For every K new samples, DQN uses θ_i^- to compute the target values $\mathbf{y}^{DQN}$ and updates the parameters in θ_i. Only after every C updates, the new weights of θ_i are copied over to θ_i^-. Furthermore, DQN utilizes *experience replay* to store all previous experience tuples (s, a, r, s'). Before a new model update, the algorithm samples a mini-batch of experiences of size M from the memory and computes the gradient of the following loss function:

$$\mathcal{L}(\theta_i) = E_{(s,a,r,s')}[(\mathbf{y}^{DQN} - Q(s, a; \theta_i))^2] \tag{2.12}$$

$$\mathbf{y}^{DQN} = r + \gamma \max_{a'} Q(s', a'; \theta_i^-) \tag{2.13}$$

Recently, Hasselt et al. (Van Hasselt et al., 2015) leveraged the overestimation problem of standard Q learning by introducing double DQN and Schaul et al. (Schaul et al., 2015) improves the convergence speed of DQN via *prioritized experience replay*. We found both modifications useful and included them in our studies.

An extension to DQN is a Deep Recurrent Q-Network (DRQN) which introduces a Long Short-Term Memory (LSTM) layer (Hochreiter and Schmidhuber, 1997) on top of the convolutional layer of the original DQN model (Hausknecht and Stone, 2015) which allows DRQN to solve POMDPs. The recurrent neural network can thus be viewed as an approximation of the belief state that can aggregate information from a sequence of observations. Hausknecht (Hausknecht and Stone, 2015) shows that DRQN performs significantly better than DQN when an agent only observes partial states. A similar model was proposed by Narasimhan and Kulkarni (Narasimhan et al., 2015) and learns to play Multi-User Dungeon (MUD) games (Curtis, 1992) with game states hidden in natural language paragraphs.

Chapter 3

The Latent Action Framework

This chapter describes the proposed latent action framework for dialog modeling from a general point of view. Notations are first formalized and the fundamental questions are highlighted. Then a principled framework is presented with evaluation metrics and novel machine learning methods. At last, we describe the high-level reasons why latent actions are desirable to advance the current state-of-the-art systems. The material introduced in this chapter is kept as general as possible to serve as the foundation for more specific applications presented in later chapters of this thesis.

3.1 Formulations and Notations

We first formally describe the variables involved in a dialog dataset. A dialog dataset often involves a number of complete dialogs, where each conversation is a list of turns that are interactively generated from two or more interlocutors. Without loss of generality, any dialog dataset can be always converted and represented as a list of $(\mathbf{c}, \mathbf{x})$ pairs, where $\mathbf{c}$ can be arbitrary structured data that describe a dialog context, e.g. discourse history, speaker information etc, and $\mathbf{x}$ is a system response to context $\mathbf{c}$. Further, a dialog context $\mathbf{c}$ is a list of utterances $[(u_1, m_1), ..., (u_t, m_t), ..., (u_T, m_T)]$, where each u_t is an natural language utterance expressed as a sequence of word tokens $[w_1^t, ..., w_i^t, ..., w_{|u_t|}^t]$. Also, m_t contains meta features about u_t, including speaker identity, ASR confidence etc. Meanwhile, a system response $\mathbf{x}$ is also represented by a sequence of tokens $[w_1, ...w_j, ...w_{|\mathbf{x}|}]$. At last, we use C and X to denote the random variables corresponding to the context and system response. Figure 3.1 shows an example dialog expressed in the above format.

Given the above notation, then in the supervised learning setting, the goal is often to learn model parameter θ through maximum likelihood estimation (MLE) on observed data:

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}} \, \mathbb{E}_{\mathbf{x}, \mathbf{c} \sim \hat{p}_{data}} [p_\theta(\mathbf{x}|\mathbf{c})] \tag{3.1}$$

where $\hat{p}_{data}$ is the empirical data distribution.

In the reinforcement learning setting, we assume to have access to an extra reward

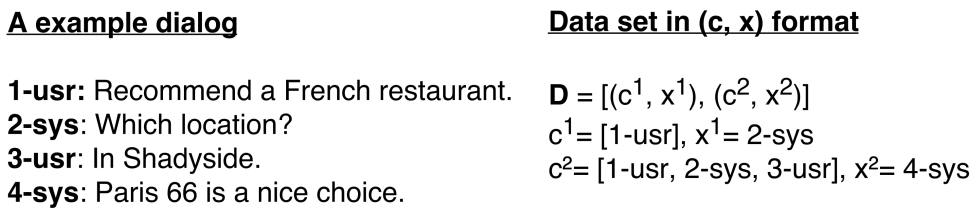

Figure 3.1: Dataset creation from an example dialog.

signal at each turn defined by a reward function $r(\mathbf{c}, \mathbf{x})$. We first define the joint probability over a dialog trajectory τ conditioned on the current dialog model θ. A trajectory corresponds to a dialog between the current model with a human users and contains a list of $\mathbf{c}, \mathbf{x}$ at each timestamp. Specifically:

$$p_\theta(\tau) = p_\theta(\mathbf{c}_1, \mathbf{x}_1, ...\mathbf{c}_T, \mathbf{x}_T) \tag{3.2}$$

$$= p(\mathbf{c}_1) \prod_{t=1}^{T} p_\theta(\mathbf{x}_t|\mathbf{c}_t) p(\mathbf{c}_{t+1}|\mathbf{c}_t, \mathbf{x}_t) \tag{3.3}$$

Then the goal is to learn model parameters θ that maximize the expected cumulative reward returns over the dialog trajectories conditioned on the current model.

$$\hat{\theta} = \underset{\theta \in \Theta}{\operatorname{argmax}}\, \mathbb{E}_{\tau \sim p_\theta(\tau)} \sum_{t=1}^{T} \gamma^t r(\mathbf{c}_t, \mathbf{x}_t) \tag{3.4}$$

where $\gamma < 1$ is a discounting factor to prevent the summation over a trajectory going to infinity.

3.2 Overview

To model the relationship between the dialog context $\mathbf{c}$ and the next response $\mathbf{x}$, a dialog system often processes information as shown in Figure 3.2. The dialog context first needs to be parsed and understood in the first layer. Then the parsed information is used as the basis for decision-making and for deciding the next move. Eventually the decided action is converted into the surface form of $\mathbf{x}$ via natural language generation. For standard E2E dialog models, the entire process is modeled by one jointly optimized neural network, e.g. encoder-decoders (Cho et al., 2014). In this case, the information is passed by as hidden states inside the E2E systems, and it is unclear which hidden states of the network correspond to the understanding results from the 1st layer, and it is neither clear about which hidden states correspond to the decision-making results.

On the contrary, the proposed latent action E2E dialog systems aim to model interlocutors' high-level actions in a conversational corpus and treat them as random variables. Since these actions are not annotated in raw conversations, they are latent and

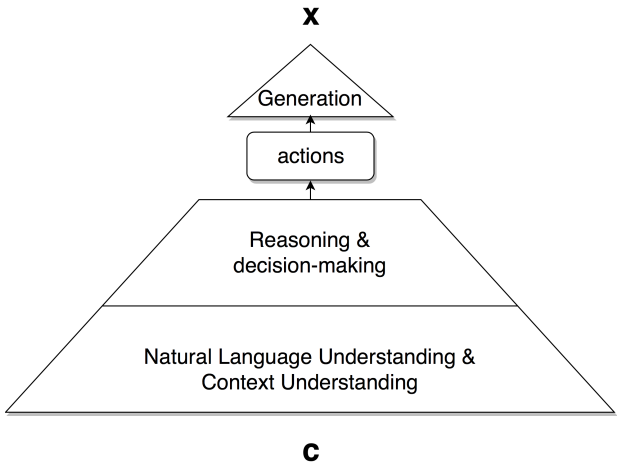

Figure 3.2: Latent actions corresponds to the interface between decision-making and generation in a dialog system.

need to be inferred from the data. These actions should represent the speaker's output ability to change the flow of the dialog and influence other interlocutors. In other words, we want to create E2E neural dialog models where the actions (as shown in Figure 3.2) is explicitly modeled as latent variables.

This goal is challenging because: even if the speaker is expressing the same action, the surface realizations are often very different from each other and may have low word overlap due to the rich variations in natural language. In MultiWoz (Budzianowski et al., 2018) corpus, a large task oriented slot-filling dataset, among the 153,262 utterances, there are 122,946 unique utterances. Therefore, most of the responses only occur once in the entire corpus but many of them are similar and share the same meaning. Thus they need to be merged judiciously in order to discover latent action groups. Another challenge is the term "action" is not a clearly defined and groups of utterances can be created at a different level of granularity. Should two utterances fall in the same group just because they have similar intentions, or these two utterances should only be combined if they carry the exact same information? These questions remain unanswered.

Moreover, as increasingly larger dialog datasets are created to fuel the training of very deep neural networks, it becomes increasingly difficult to annotate the corpus or even come up with a comprehensive annotation schema that can cover all response patterns appeared in the data. Prior research in dialog act annotation such as Switchboard Dialog Act (Godfrey et al., 1992) has created complex annotation systems that include over 200 distinct dialog acts to only model a coarse representation of the speaker intents, e.g. statement, yes/no-questions etc. Therefore the design of latent actions has to be expressive and has the potential to model a large number of fine-grained actions.

To address the above challenges, we approach this grand goal with four key research questions:

1. How to define a latent action and at what level of granularity? (this chapter)

2. How to evaluate latent actions? (this chapter)
3. How to conduct efficient inference to discover the latent assignment of each response in a large dataset? (Chapter 4)
4. Why are latent actions useful and how to evolve the basic latent actions to solve real-world challenges? (Chapter 5-8)

In short, the following sections layout a path from designing, evaluating and learning latent actions from large dialog corpora.

3.3 Latent Action Definitions

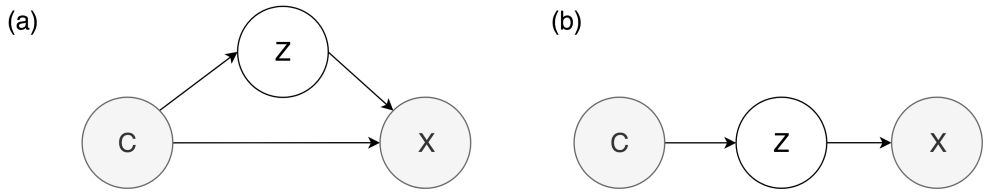

Figure 3.3: Graphic models for partial (a) and full (b) latent actions.

The definition of latent action depends on the level of granularity that it should handle. We identify two general types of latent actions and describe them via probabilistic graphic models as shown in Figure 3.3 These two types are: *full latent action* and *partial latent action*.

Full Latent Actions

First, we define full latent action Z as a random variable that captures the speech act of a dialog agent, representing the utterance-level intention and propositional content of a system. Full latent action factorizes the conditional distribution by:

$$P(X|C) = P(X|Z)P(Z|C) \tag{3.5}$$

Z is a latent variable because the ground-truth speech act is usually unobserved in the raw data. For example, in a restaurant recommendation domain,

- "Can you tell me your cuisine type preference?"
- "What type of restaurant are you looking for?"

should belong to the same latent assignment since they both express the same speech act despite the differences in word choices. Another example is

- "Noodle Head is a great Thai restaurant"
- "Thai Palace is a nice place to have Thai food"

the above two responses represent the same illocutionary force but different propositional content (Noodle Head vs. Thai Palace), and therefore should belong to different

but similar latent action assignments. Also, to infer the latent action value of an utterance should be highly contextual, i.e. depending on the dialog context that precedes this utterance. For example, the propositional content of "I need the first one" depends on the preceding question, i.e. the meaning of "first one" depends on the reference. Therefore, it is important to take account of the context when computing the posterior distribution of Z.

Partial Latent Actions

The above definition assumes Z captures both intent and content information of the speech act. However, in certain applications we only need to model certain aspect of the full speech act and can leave the rest of the information to the standard encoder-decoder neural networks. Therefore, we also define partial latent actions as opposed to the full latent actions, which factorize the conditional distribution by:

$$P(X|C) = P(X|Z, C)P(Z|C) \tag{3.6}$$

Chapter 5 will illustrate an example of auxiliary latent action used to solve the dull-response problem open domain chatbot. The graphic model for partial latent action is shown in Figure 3.3 (a). As we can see, we first sample a latent action $\mathbf{z}$ based on the context and then generate the response $\mathbf{x}$ conditioned on both $\mathbf{x}$ and $\mathbf{c}$ from conditional distribution $P(X|C, Z)$. This allows the $\mathbf{z}$ only represents some aspects of the response $\mathbf{x}$ since the generation of $\mathbf{x}$ can leverage information from both $\mathbf{c}$ and $\mathbf{z}$. For example, consider an application where latent action only needs to represent the sentiment of the response, and other detailed information, e.g. topic, intent etc should be left to the generation process $P(X|Z, C)$. So given a context:

- User said: do you like to shop at Whole Foods?

Then the following responses should belong to the same action group since they are all positive responses:

- Yes, I really like the food quality there.
- I love it, especially the furnishing style.
- It's very close to my home and I go there every week.

These three responses should be mapped to the same group if the goal is to create a latent action that represents sentiment, despite the fact that these three responses like Whole Foods for entirely different reasons.

At last, the two factorizations defined in Eq 3.5 and Eq 3.6 naturally define two sets of parameters that correspond to the encoder decoder neural networks. Specifically, $P(Z|C)$ corresponds to the encoder network, and resembles the roles of NLU, DST and DM in frame-based dialog systems. Meanwhile $P(X|Z)$ and $P(X|Z, C)$ are the decoder network and take the role of NLG in frame-based systems. The main difference between the full and partial latent action is that (1) for partial latent actions, the encoder needs to output a random variable Z as well as other deterministic hidden states that can be used by the decoder (2) for partial latent actions, the decoder can take advantage of information from the context directly.

Types of Random Variables

Now let us consider the types of distribution that latent actions should follow. In order to make the resulting latent actions capable of modeling a variety of speaker intentions, several key design choices should be made, including continuous vs. discrete variables, number of variable dimensions, distribution family and joint distribution factorization.

Both continuous and discrete random variables are considered in this thesis and we will show each of its pros and cons. For **continuous variable**, a natural choice is a Gaussian distribution. Concretely, multivariate Gaussian variable is chosen (e.g. dimension size of 300). This allows the variable to be expressive enough to represent a rich set of information about dialog responses. The joint distribution of multivariate Gaussian variable of dimension size D is:

$$p(\mathbf{x}; \boldsymbol{\mu}, \Sigma) = \frac{1}{(2\pi)^{D/2}|\Sigma|^{1/2}} \exp\Big(-\frac{1}{2}(\boldsymbol{\mu} - \mathbf{x})^T \Sigma^{-1} (\boldsymbol{\mu} - \mathbf{x}) \Big) \tag{3.7}$$

where μ is the mean vector and Σ is the covariance matrix. In addition, although learning multivariate Gaussian variable with full covariance matrix is possible, it is often preferable to constrain the covariance matrix to be diagonal and the diagonal values are denoted by σ^2

$$\text{diag}(\Sigma) = \boldsymbol{\sigma}^2 \tag{3.8}$$

This choice is backed by two reasons: (1) it encourages disentangled representations, i.e. each dimension corresponds to an orthogonal attributes of the data (2) it simplifies the optimization and computation.

In practice, μ and σ^2 are modeled by two neural networks:

$$\boldsymbol{\mu} = \text{NeuralNet}(\bullet) \tag{3.9}$$

$$\log(\boldsymbol{\sigma}^2) = \text{NeuralNet}(\bullet) \tag{3.10}$$

On the other hand, for **discrete variables**, a categorical distribution is considered. Similar to the setup in the continuous case, a discrete latent action is designed to be multi-dimensional and factorized. That is a discrete latent action is composed of M independent categorical random variable (each can take one of the K possible values) and the joint probability mass function is defined by:

$$p(\mathbf{z}_1 = \mathbf{x}_1, ...\mathbf{z}_M = \mathbf{x}_M) = \prod_i^M p(\mathbf{z}_i = \mathbf{x}_i) \tag{3.11}$$

$$= \prod_i^M \prod_j^K p_{ij}^{[\mathbf{x}_i = j]} \tag{3.12}$$

where $[\mathbf{x}_i = j]$ evaluates to be 1 if $\mathbf{x}_i$ equals j and otherwise 0. Note that the discrete latent action can only represent a finite set of values and the continuous random variable can represent an infinite number of values. However, our setup enables a discrete latent action to represent exponentially many, i.e. K^M, distinct configurations. Therefore,

given a reasonably big M and K (e.g. $K = 20$, $M = 10$), it can capture a large enough number of unique actions (e.g. 20^{10}). In terms of implementation, each categorical random variables can be achieved by passing logitis through a normalizing Softmax layer as follows:

$$p(\mathbf{z}_i|\mathbf{c}) = \text{Softmax}(\text{NeuralNet}(\bullet)) \quad i \in [1, M] \tag{3.13}$$

3.4 Objectives and Evaluation

What makes a latent action useful? We list several desired properties for latent actions and specify objectives for optimization.

3.4.1 Likelihood

Maximum likelihood estimation (MLE) is a classic learning objective for training latent variable models. That is we want to maximize the conditional distribution:

$$p(\mathbf{x}|\mathbf{c}) = \int_{\mathbf{z}} p(\mathbf{x}|\mathbf{z}, \mathbf{c}) p(\mathbf{z}|\mathbf{c}) d\mathbf{z} \tag{3.14}$$

There has been a long history in optimizing the above objective via expectation maximization, variational inference etc, which will be addressed in the next section. Regardless of the learning method, this expected conditional probability over the test dataset is a primary evaluation metric to assess if a learned latent action is useful and encodes salient information. In the context of dialog response generation, $\mathbf{x}$ is a sequence of word tokens, so that the above equation can be further written as:

$$p(\mathbf{x}|\mathbf{c}) = \int_{\mathbf{z}} \prod_{i}^{|\mathbf{x}|} p(w_i|w_{<i}, \mathbf{z}, \mathbf{c}) p(\mathbf{z}|\mathbf{c}) d\mathbf{z} \tag{3.15}$$

3.4.2 Mutual Information

Another useful metric to evaluate a latent action is by testing its mutual information with respect to the response $I(X, Z)$. After all, a latent action is designed to capture salient information in the response and should have high mutual information with it. The mutual information between X and Z can be further written as:

$$I(X, Z) = H(Z) - H(Z|X) \tag{3.16}$$

The above decomposition suggests that high mutual information leads to high $H(Z)$ and $H(Z|X)$. The second term is self-evident since we want the entropy of Z given X to be low. In the discrete random variable case, the first term $H(Z)$ implies that the marginal distribution of Z without conditions should be as uniform as possible. High $H(Z)$ leads to interesting property because it encourages all the possible latent assignment being used, at least in some situation. This is desirable for two reasons:

1. Reduce the risk of *exposure bias* (Ross et al., 2011): using all latent actions makes sure at testing time, the encoder network will not predict a latent $\mathbf{z}$ that is never observed during training, which leads to undefined behavior in the decoder.
2. Create a safe exploration space: as we will show in Chapter 8, having all possible latent actions tried create a safe exploration space for later fine-tuning based on other objectives. Similar to the exposure bias problem in the first point. We will utilize this property in Chapter 8 for better exploration in reinforcement learning.

Note that mutual information is much easier to calculate for discrete latent action, and is often intractable to compute for continuous random variables.

3.4.3 Distributional Semantics

The distributional hypothesis (Harris, 1954) in natural language semantics suggests that the meaning of words or sentences can be inferred from the context in which they are used (Mikolov et al., 2013; Kiros et al., 2015). Learning latent actions can be also considered as creating a semantic representation for dialog responses in the form of latent variables. Moreover, as we show before, the meaning of responses are also highly contextual, i.e. depends on the dialog context where it is used, and various lexically different responses can share the same meaning. These properties make the distributional hypothesis highly relevant to create better latent actions.

Concretely, the optimization goal for learning latent actions with distributional hypothesis can be maximizing the following likelihood:

$$p(\mathbf{c}|\mathbf{x}) = p_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x})}(\mathbf{c}|\mathbf{z}) \tag{3.17}$$

where $q(\mathbf{z}|\mathbf{x})$ is a transformation function that maps a response $\mathbf{x}$ into latent embedding $\mathbf{z}$. Then the resulting latent action $\mathbf{z}$ can be used to infer the dialog context before and after the response $\mathbf{x}$ is used. In chapter 6, the proposed variational skip thought is an manifestation of this objective.

3.4.4 Downstream Tasks

Lastly, the ultimate evaluation metric for latent variable learning is if it can improve the performance on the end task, e.g. producing a system that can have coherent conversation with human users. In the context of this thesis, techniques to evaluate the performance of a E2E dialog agents can be divided into two categories:

Turn-level Evaluation

Given a corpus of dialog context-response $(\mathbf{c}, \mathbf{x})$ pairs, turn-level evaluation assesses the system performance by comparing a generated response $\tilde{\mathbf{x}}$ against the ground-truth response $\mathbf{x}$. To compare the similarity between $\mathbf{x}$ and $\tilde{\mathbf{x}}$, many metrics have been developed and used, e.g. BLEU score (Papineni et al., 2002), word embedding matching (Liu

et al., 2016) etc. The advantage of turn-level evaluation is that it is very easy to compute, good reproducibility and does not require expensive human evaluation. On the other hand, turn-level evaluation has many limitations, such as poor correlation with human judgement for open-domain conversations when only one reference response is presented (Liu et al., 2016). That said, turn-level evaluation still remains to be the mainstream evaluation method for E2E systems. This thesis will also show that its effectiveness can be further improved by designing better metrics, e.g. including multiple reference response at testing time.

Dialog-level Evaluation

Besides turn-level evaluation, a more challenging downstream task is dialog-level evaluation, which directly measures how well a dialog agent can converse. Often a dialog-level reward is defined to quantify if a dialog between a human and a machine is successful or not. For example, in task-oriented dialog domains, a dialog is successful if the system inform the correct information to the user as fast as possible (Williams and Young, 2007). Another example is for negotiation dialog agent, a dialog is successful if the dialog agent is able to win the deal after negotiating with the opponent. For open domain chatting, dialog-level evaluation is still an open-ended research challenge and prior study has explored to assess systems from engagement (Zhou et al., 2018), user satisfaction (Eskenazi et al., 2019) etc. Therefore, we only use dialog-level evaluation for tasks that have clear definition of success and use turn-level evaluation instead for tasks such as open-domain chatting.

3.5 Basic components

At last, given the above setup and learning objectives, our latent action framework involves in learning three basic components as shown in Figure 3.4. These three sets of parameters are:

1. a decoder network $p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{c})$ or $p_\theta(\mathbf{x}|\mathbf{z})$ for full or partial latent actions that is used to generate the response $\mathbf{x}$.
2. an encoder network $p_\pi(\mathbf{z}|\mathbf{c})$ that is used to predict the next latent action given the context. In later chapters, we also refer to it as the *policy network*, since it corresponds to the dialog policy that predict the next system action given the dialog context.
3. a recognition network $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c})$ that is used to map raw utterances into latent actions.

It is also worthwhile to note that the above three sets of parameters naturally correspond to the components in the classic frame-based dialog pipeline. Specifically, the decoder network corresponds to the NLG component, the recognition network resembles the NLU and the prior network resembles the DM. By having this analogy in mind, we are able to develop E2E dialog models that bridges the merits between E2E systems and

traditional frame-based systems.

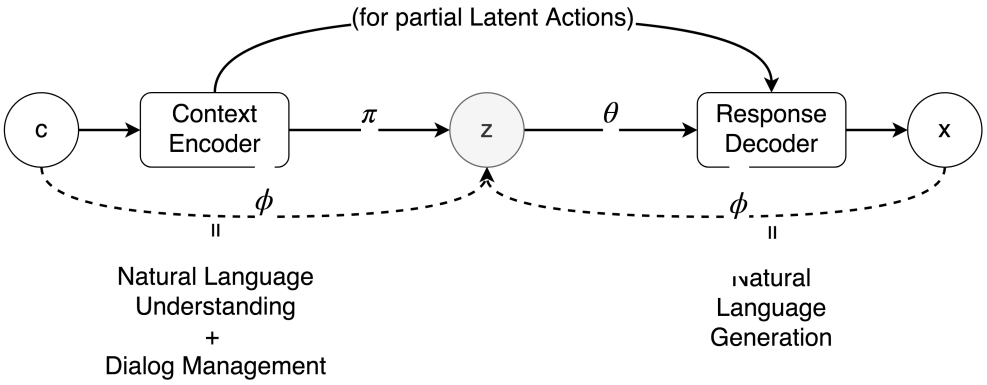

Figure 3.4: The basic model architecture of latent action framework. Dashed line indicate recognition networks. Solid line denote the networks for generation.

3.6 Why Latent Actions?

What are the advantages of creating a explicit representation of the dialog actions? One may argue that the standard E2E dialog models in theory can perfectly model the conditional distribution $p(\mathbf{x}|\mathbf{c})$ as more powerful generative models are developed and that is all we need for creating better dialog systems. We argue that only modelling the conditional distribution $p(\mathbf{x}|\mathbf{c})$ is not sufficient to create dialog systems and it is desired to intentionally factorize the generation process into $p(\mathbf{x}|\mathbf{c}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathbf{c})$ or $p(\mathbf{x}|\mathbf{c}) = p(\mathbf{x}|\mathbf{z}, \mathbf{c})p(\mathbf{z}|\mathbf{c})$. This factorization is necessary for the following reasons:

1. **Disentangle the hierarchical generation process of dialog response.** Open domain chatting can be extremely open-ended. Given the same dialog context, even the same speaker can lead the dialog towards different directions via different responses. Therefore, it is crucial to separate "what to say" from "how to say" and create a model that can model the distribution over next possible "moves" given a dialog context. This can be modeled by treating this distribution as a stochastic latent action. This latent action is then responsible for modeling the discourse-level intents while the decoder should only be responsible for mapping this intent to its surface form.
2. **Use latent actions as human-machine interface.** Another use case is to design a low-dimensional discrete latent variable that we can understand the "meaning" of each latent assignment. This provides a interface for us to obtain compact attributes about what the model is going to generate, and such symbolic interface opens doors for integrating with manual-rules or linguistic knowledge with a neural dialog system.
3. **Separate sentence-level representation from discourse-level for domain transfer.** Obtaining utterance data is far easier than collecting full conversations in a

new domain. Latent actions allow us to obtain a cross-domain embedding space where utterances with similar functions from different domains are encoded into adjacency in the latent space. This cross-domain latent space will make an E2E dialog model easier to adapt to a new domain, since all the model only needs to learn domain-specific dialog flow and does not need to re-learn utterance level representation.

4. **Induce a latent action space for dialog strategy optimization with reinforcement learning.** In frame-based dialog models, people often optimize the dialog policy via reinforcement with task-specific dialog-level rewards, e.g. completing the task as soon as possible, persuade a user to agree to an argument etc. This types of dialog policy optimization has only been done with actions that are manually crafted, which cannot be done for standard encoder-decoder models (e.g. high-level actions are not in the form of random variables, but distributively hidden in the hidden states of each decoding step). Naturally latent actions provides a similar action space that can be optimized via reinforcement learning, but at the same time completely free from manual action crafting.
5. **Better understanding of large dialog datasets.** Besides developing better dialog systems, it is also highly valuable to get more insights about a dialog dataset, which in turns can help developers to accumulate domain knowledge and further improve performance. As dialog datasets are getting increasingly large, it becomes harder to manually inspect a dialog dataset and summarize the major action group that both sides of the speaker can use. Our latent action framework provides a two birds, one stone solution that simultaneously creates a working E2E dialog system and offers a powerful recognition network. The recognition network is essentially a learned NLU that can map all the utterances in the data into meaningful semantic groups. Therefore, having an explicit action representation will be helpful to scale dialog analysis to larger dataset and extract precious information for further domain customization.

Chapter 4

Learning Methods for Latent Actions

This chapters addresses the question on how to train latent variable neural networks to induce meaningful latent representation from raw conversation data. Building upon the stochastic variational inference (SVI) framework, we propose a neural network architecture to model latent actions for dialog response generation. Moreover, we focus on mitigating the well-known *posterior collapse* issue when training latent variable models with powerful auto-regressive decoders (Bowman et al., 2015). Concretely, posterior collapse represents the situation where the posterior distribution $q(\mathbf{z}|\mathbf{x})$ is a constant, so that the value of latent variable is independent on the response $\mathbf{x}$. Then it is self-evident that fixing posterior collapse is an essential step to deliver what is promised in our proposed latent action framework.

We propose a set of general-purpose novel algorithms in order to better train latent variable models for dialog response representation learning. Our solution show superior performance compared to baseline methods and we discuss in details the practical implications and suggestions for using SVI methods for learning latent actions. At last, we present the methods in this chapters as general learning algorithms that are not tailored to specific usage for dialog systems, so that they are general-purpose and can benefit related NLP text generation tasks, e.g. abstractive summarization, story generation, machine translation etc. The proposed methods are adapted and used for specific improvement for dialog response generation in Chapters 5-8.

4.1 Foundations

Latent variable models can be trained by maximizing the marginalized likelihood, e.g. $p(\mathbf{x}|\mathbf{c})$ or $p(\mathbf{x})$. However, since the latent variable $\mathbf{z}$ is unobserved, we have to marginalize over $\mathbf{z}$ which is intractable when $\mathbf{z}$ is continuous or $\mathbf{z}$ is a large discrete variable. Use the full continuous latent action as an example:

$$p(\mathbf{x}|\mathbf{c}) = \int_{\mathbf{z}} p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathbf{c})d\mathbf{z} \tag{4.1}$$

The integration over $\mathbf{z}$ is intractable. Therefore, an approximation method is needed overcome this challenge and variational inference (Wainwright et al., 2008) is one of most powerful classes of algorithms to achieve this goal.

4.1.1 Stochastic Variational Inference

Variational inference suggests to maximize a lowerbound of $p(\mathbf{x}|\mathbf{z})$, aka evidence lowerbound (ELBO), instead of the true marginal likelihood to train latent variable models (Wainwright et al., 2008). The first step is to define a proposal distribution $q(\mathbf{z})$ to approximate the unknown true posterior distribution $p(\mathbf{z}|\mathbf{x}, \mathbf{c})$. Then ELBO can be derived by:

$$\begin{aligned}
\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x}, \mathbf{c})) &= -\int_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{z}|\mathbf{x}, \mathbf{c})}{q(\mathbf{z})} \\
&= -\int_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{z}, \mathbf{x}|\mathbf{c})}{q(\mathbf{z})p(\mathbf{x}|\mathbf{c})} \\
&= \underbrace{-\int_{\mathbf{z}} q(\mathbf{z}) \log \frac{p(\mathbf{z}, \mathbf{x}|\mathbf{c})}{q(\mathbf{z})}}_{-\mathcal{L}_{\mathrm{ELBO}}} + \int_{\mathbf{z}} q(\mathbf{z}) \log p(\mathbf{x}|\mathbf{c}) \\
&= -\mathcal{L}_{\mathrm{ELBO}} + \log p(\mathbf{x}|\mathbf{c}) \\
\log p(\mathbf{x}|\mathbf{c}) &= \mathcal{L}_{\mathrm{ELBO}} + \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}|\mathbf{x}, \mathbf{c})) \geq \mathcal{L}_{\mathrm{ELBO}}
\end{aligned} \tag{4.2}$$

Since the Kullback-Leibler (KL) divergence between $q(\mathbf{z})$ and $p(\mathbf{z}|\mathbf{x}, \mathbf{c})$ is non-negative, it is evident that $\mathcal{L}_{\mathrm{ELBO}}$ is a lowerbound of the log likelihood $\log p(\mathbf{x}|\mathbf{c})$ and the tightness of the bound depends on how well q can approximate the true posterior distribution. Therefore, we often design q to be $q(\mathbf{z}|\mathbf{x}, \mathbf{c})$ so that we use the best of our knowledge (based on $\mathbf{x}$ and $\mathbf{c}$) to predict $\mathbf{z}$. Then ELBO for full latent action can be written as:

$$\begin{aligned}
\mathcal{L}_{\mathrm{ELBO}} &= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}, \mathbf{c}) \log \frac{p(\mathbf{z}, \mathbf{x}|\mathbf{c})}{q(\mathbf{z}|\mathbf{x}, \mathbf{c})} \\
&= \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}, \mathbf{c}) \log p(\mathbf{x}|\mathbf{z}) + \int_{\mathbf{z}} q(\mathbf{z}|\mathbf{x}, \mathbf{c}) \frac{p(\mathbf{x}|\mathbf{c})}{q(\mathbf{z}|\mathbf{x}, \mathbf{c})} \\
&= \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x},\mathbf{c})}[\log p(\mathbf{x}|\mathbf{z}) + D_{KL}(q(\mathbf{z}|\mathbf{x}, \mathbf{c})\|p(\mathbf{z}|\mathbf{c}))]
\end{aligned} \tag{4.4}$$

Similarly, the ELBO for partial latent action can be written as:

$$\mathcal{L}_{\mathrm{ELBO}} = \mathbb{E}_{\mathbf{z}\sim q(\mathbf{z}|\mathbf{x},\mathbf{c})}[\log p(\mathbf{x}|\mathbf{z}, \mathbf{c}) + D_{KL}(q(\mathbf{z}|\mathbf{x}, \mathbf{c})\|p(\mathbf{z}|\mathbf{c}))] \tag{4.5}$$

where q is a proposal distribution is that is used to approximate the true but unknown posterior distribution $p(\mathbf{z}|\mathbf{c}, \mathbf{x})$.

The Reparametrization Trick

One challenge of learning latent variables in neural networks is that we cannot back-propagate through a stochastic node by default. To see this, given a latent variable

$\mathbf{z} \sim p_\theta(\mathbf{z})$ and p_θ is a parametric distribution of $\mathbf{z}$, then its derivatives over certain function f can be written as:

$$\bigtriangledown_\theta \, \mathbb{E}_{\mathbf{z} \sim p_\theta(\mathbf{z})}[f(\mathbf{z})] \tag{4.6}$$

Equation 4.6 shows the difficulty lies in evaluating the expectation over a distribution over the model parameters θ. Reparameterization is a technique used in varational autoencoders (Kingma and Welling, 2013) that provides a solution to this issue and offers low-variance gradients through a random variable for training deep neural network latent variable models. Specifically, the Reparametrization Trick is to rewrite the latent variable as:

$$\mathbf{z} = g_\theta(\epsilon) \quad \epsilon \sim p(\epsilon) \tag{4.7}$$

$$\bigtriangledown_\theta \, \mathbb{E}_{\mathbf{z} \sim p_\theta(\mathbf{z})}[f(\mathbf{z})] = \bigtriangledown_\theta \, \mathbb{E}_{\epsilon \sim p(\epsilon)}[f(g(\epsilon)] \tag{4.8}$$

where ϵ is a random number follows a noise distribution $p(\epsilon)$. The key observation here is that Equation 4.8 transforms the original gradient function so that it no longer depends on the model parameter θ, which allows us to easily have an unbiased estimate about the gradient via Monte Carlo:

$$\bigtriangledown_\theta \, \mathbb{E}_{\mathbf{z} \sim p_\theta(\mathbf{z})}[f(\mathbf{z})] = \frac{1}{N} \sum_i^N \bigtriangledown_\theta [f(g(\epsilon^i)] \tag{4.9}$$

For Gaussian latent variables, the noise distribution used for reparametrization is (Kingma and Welling, 2013):

$$\mathbf{z} = \mu + \sigma\epsilon \quad \epsilon \sim \mathcal{N}(0, 1) \tag{4.10}$$

For discrete latent variables, the Gumbel-Softmax Distribution is used to achieve the same goal (Jang et al., 2016; Maddison et al., 2016). Concretely, let the noise distribution follows Gumbel distribution:

$$\epsilon \sim -\log(-\log(u)) \quad u \sim \text{Uniform}[0, 1] \tag{4.11}$$

Jang et al (2016) shows that a continuous approximation for samples from a K-way categorical distributions can be expressed:

$$\frac{\exp((\log \alpha_i + \epsilon_i)/\tau)}{\sum_{j=1}^K \exp(\log(\alpha_j + \epsilon_j)/\tau)} \quad i = 1, 2...K \tag{4.12}$$

where τ is a temperature constant that is used to control how closely the sample approximates a true one-hot categorical distribution. When τ approaches 0, Eq 4.12 approaches one-hot distribution.

Thus using the Reparametrization Trick, both types of proposed latent actions can be trained efficiently via gradient-based optimization methods in neural networks. Also future improvements on reparametrization techniques from the machine learning community can be readily used in our proposed framework.

4.1.2 The Posterior Collapse Problem

Unfortunately, despite the success of creating generative models for images using standard VAEs (Kingma and Welling, 2013; Jang et al., 2016), simply training VAEs for text generation with auto-regressive decoders often leads to the *posterior collapse* issue (Bowman et al., 2015). Its symptom is defined as follows:

Definition: After training with ELBO, $\mathrm{D}_{\mathrm{KL}}(q\|p)$ is set to 0 for all data points, so that the recognition network outputs a almost constant $\mathbf{z}$ regardless of the input $\mathbf{x}$. As a results, the latent code $\mathbf{z}$ is ignored by the decoder.

In other words, when the decoder is a auto-regressive model that can leverage the language modeling information, the whole system is prone to overfit to a local optimal by not leveraging any information from $\mathbf{z}$, which easily decrease the loss by making the KL-divergence term small. To given an example, we trained a LSTM-based VAEs on Penn Tree Bank pre-processed by (Mikolov et al., 2010) using similar setup used in (Bowman et al., 2015) to demonstrate how the loss function evolves over training steps for both Gaussian latent variables as well as categorical latent variables in Figure 4.1.

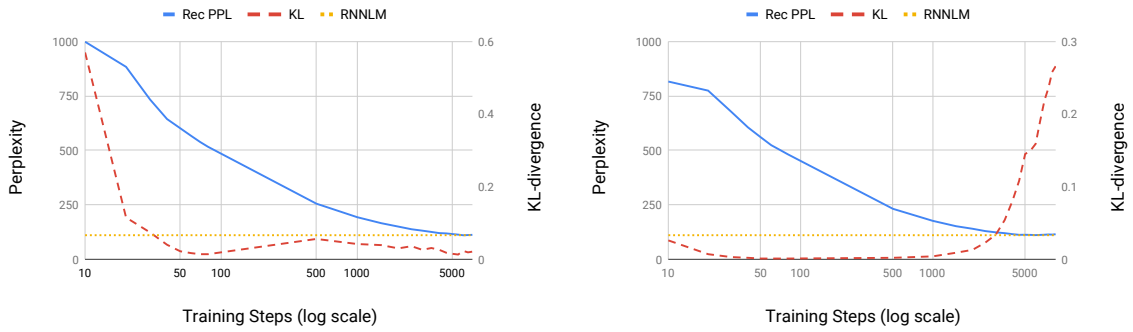

Figure 4.1: The evolution of reconstruction perplexity and KL-divergence for Gaussian latent variable (left) and categorical latent variable (right) on PennTree Bank test dataset. The yellow dotted line is the perplexity of a standard LSTM language model trained on the same data. Note that the horizontal axis is in log-scale.

Figure 4.1 shows that for Gaussian (continuous) VAEs, the KL-divergence drops down to 0 within 100 training steps. And after that although the reconstruction perplexity (PPL) continues to decrease after more training, the KL term never becomes larger again. Eventually, the model converges to a reconstruction perplexity that is equivalent to a LSTM language model trained on the same data, which strongly suggests that only the decoder (the language model) is being used. For discrete latent variables, the KL term is almost 0 even in the beginning of the training and the KL term does not increase until the very end of the training. Meanwhile, the reconstruction PPL is improved similarly to the continuous counterparts, and converges to the same PPL as the standard language model (LM). Again, this evidence suggests that the decoder is not utilizing any additional information from the latent $\mathbf{z}$ and predicts the next word purely based

on the context information at the decoder side. Note that the increase of KL in the discrete case is interesting since we expect at that point the recognition network begins to output meaningful $\mathbf{z}$. Unfortunately, a closer inspection shows that this is only due to overfitting, so that the recognition network begins to output a constant value of $\mathbf{z}$ instead of uniform prior, while the decoder still ignores $\mathbf{z}$.

The posterior collapse problem is destructive to latent action learning, since it undermines the basic assumption i.e. latent action contains salient features about the dialog responses. During the development of this dissertation, several novel techniques are proposed to mitigate the posterior collapse challenge for training VAEs for natural language generators. This section presents those techniques that are universally applicable to any latent variable neural text generator beyond dialog generation. In later chapters, we will apply these solutions without duplicated explanations and focus on chapter specific novelties, e.g. the action matching algorithm, knowledge-guided latent variable learning and etc.

4.2 Proposed Solutions

We conjecture that the reasons behind posterior collapse for text generation VAEs are two fold: (1) it is an optimization challenge since it is very easy for auto-regressive decoder to find low hanging fruits, e.g. local pattern on the decoder side, and it is less obvious on learning a global representation in the latent code (2) the standard ELBO may not be the best learning objectives if the main interest is in discovering a latent code that is highly correlated with the input. Based on these two hypothesis, two branches of solutions are developed. The first branch of the proposed solution is through multi-task learning by designing an auxiliary loss function that prevent the model from exploiting local patterns in the auto-regressive decoders. The second branch takes a dive into the evidence lowerbound and derivs a information maximization based objectives for discrete latent variables.

4.2.1 Non-autoregressive Auxiliary Loss

a. Bag-of-word Auxiliary Loss

We first propose a simple yet effective auxiliary objective: *bag-of-word loss*. The idea is to introduce an auxiliary loss that requires the decoder network to predict the bag-of-words in the response $\mathbf{x}$. Denote $\mathbf{x} = [w_1, ...w_T]$ be the list of T words in the response $\mathbf{x}$. Bag-of-words prediction assumes conditional independence property, which in terms force the latent variable to capture global information about the target response. In the VAE case and full latent action case, we add a separate decoder f_τ that predicts the probability of the words in $\mathbf{x}$ independently given a sample of $\mathbf{z}$ from the recognition

network, i.e.

$$\log p_\tau(\mathbf{x}_{bow}|\mathbf{z}) = \sum_{i=1}^{T} \log p_\tau(w_i|\mathbf{z}) = \sum_{i=1}^{T} \log \frac{e^{f_\tau(\mathbf{z})}}{\sum_j^V e^{f_\tau(\mathbf{z})}} \tag{4.13}$$

Similarly, for partial latent action, this separate decoder takes in both dialog context $\mathbf{c}$ and $\mathbf{z}$ as inputs and resulting into the following likelihood.

$$\log p_\tau(\mathbf{x}_{bow}|\mathbf{z}, \mathbf{c}) = \sum_{i=1}^{T} \log \frac{e^{f_\tau(\mathbf{z},\mathbf{c})}}{\sum_j^V e^{f_\tau(\mathbf{z},\mathbf{c})}} \tag{4.14}$$

In the end, we add the bag-of-word auxiliary objective into the standard ELBO objective and creates the following multi-tasking objective:

$$\mathcal{L}'(\theta, \phi, \tau; \mathbf{x}, \mathbf{c}) = \mathcal{L}_{\text{ELBO}} + \lambda \, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x},\mathbf{y})}[\log p_\tau(\mathbf{x}_{bow}|\mathbf{z}, \mathbf{c})] \tag{4.15}$$

In practice, f_τ can be implemented as a multi-layer perceptron (MLP). We will show that the bag-of-word loss in Equation 4.15 is very effective against the posterior collapse and it is also complementary to other techniques proposed in related work, e.g. the KL annealing technique (Bowman et al., 2015).

b. Response Selection Auxiliary Loss

An alternative auxiliary objective to encourage a meaningful latent $\mathbf{z}$ is through response selection. Normally, an auto-regressive decoder factorizes the conditional distribution $p(\mathbf{x}|\mathbf{c}, \mathbf{z})$ through a product of next word prediction:

$$p(\mathbf{x}|\mathbf{c}, \mathbf{z}) = \prod_{i}^{T} p(w_i|w_{<i}, \mathbf{c}, \mathbf{z}) \tag{4.16}$$

where T is the length of the response $\mathbf{x}$. Such a chain structure makes it easy for the decoder to ignore the latent action $\mathbf{z}$ due to the availability of previous words in the response $w_{<i}$ and strong temporal dependencies in natural language. On the contrary, the response selection loss tries to model the conditional distribution at sentence-level, i.e.:

$$p(\mathbf{x}|\mathbf{z}, \mathbf{c}) = \frac{e^{g_\tau(\mathbf{x})^T f_\tau(\mathbf{c},\mathbf{z})}}{\sum_i^A e^{g_\tau(\mathbf{x}_k)^T f_\tau(\mathbf{c},\mathbf{z})}} \tag{4.17}$$

where g_τ is a response encoder that maps candidate response $\mathbf{x}_i$ into distributed vectors and f_τ is a separate decoder that transforms $\mathbf{c}$ and $\mathbf{z}$ into vector representations.

However, it is intractable to enumerate through all possible responses in the denominator due to the vast space of natural language since A can be prohibitively large. Therefore, we approximate it via negative sampling (Mikolov and Zweig, 2012). That is we randomly sample K distracting responses from all the utterances from the training data and we modify the training objective to be:

$$\log p_\tau(\mathbf{x}|\mathbf{z},\mathbf{c}) = \log \frac{e^{g_\tau(\mathbf{x})^T f_\tau(\mathbf{c},\mathbf{z})}}{\sum_i^K e^{g_\tau(\mathbf{x}_k)^T f_\tau(\mathbf{c},\mathbf{z})}} \tag{4.18}$$

Similar to the bag-of-word loss, the response selection loss will be added to the original ELBO objective during training.

$$\begin{aligned}\mathcal{L}'(\theta,\phi,\tau;\mathbf{x},\mathbf{c}) &= \mathcal{L}(\theta,\phi;\mathbf{x},\mathbf{c}) \\ &+ \lambda\, \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x},\mathbf{y})}[\log p_\tau(\mathbf{x}|\mathbf{z},\mathbf{c})]\end{aligned} \tag{4.19}$$

4.2.2 Maximum Mutual Information

Anti-Information Nature of ELBO

We argue that the posterior collapse issue lies in ELBO and we offer a novel decomposition to understand its behavior. For simplicity, we temporally drop the parameter subscripts ϕ and θ for the recognition network and the decoder. First, instead of writing ELBO for a single data point, we write it as an expectation over a dataset:

$$\begin{aligned}\mathcal{L}_{\text{VAE}} = \mathbb{E}_{\mathbf{x}}[&\mathbb{E}_{q(\mathbf{z}|\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] \\ &- \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))]\end{aligned} \tag{4.20}$$

We can expand the KL term as Eq. 4.25 by following:

$$\begin{aligned}&\mathbb{E}_{\mathbf{x}}[\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))] = && (4.21)\\ &\mathbb{E}_{q(\mathbf{z}|\mathbf{x})p(\mathbf{x})}[\log(q(\mathbf{z}|\mathbf{x})) - \log(p(\mathbf{z}))] \\ &= -H(\mathbf{z}|\mathbf{x}) - \mathbb{E}_{q(\mathbf{z})}[\log(p(\mathbf{z}))] && (4.22)\\ &= -H(\mathbf{z}|\mathbf{x}) + H(\mathbf{z}) + \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z})) && (4.23)\\ &= I(\mathbf{z},\mathbf{x}) + \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z})) && (4.24)\end{aligned}$$

where $q(\mathbf{z}) = \mathbb{E}_{\mathbf{x}}[q(\mathbf{z}|\mathbf{x})]$ and $I(\mathbf{z},\mathbf{x}) = H(\mathbf{z}) - H(\mathbf{z}|\mathbf{x})$ is mutual information between $\mathbf{z}$ and $\mathbf{x}$ by definition. Then we rewrite ELBO as:

$$\begin{aligned}&\mathbb{E}_{\mathbf{x}}[\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))] = && (4.25)\\ &\qquad I(Z,X) + \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))\end{aligned}$$

$$\begin{aligned}\mathcal{L}_{\text{VAE}} = &\mathbb{E}_{q(\mathbf{z}|\mathbf{x})p(\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] \\ &- I(Z,X) - \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))\end{aligned} \tag{4.26}$$

where $q(\mathbf{z}) = \mathbb{E}_{\mathbf{x}}[q(\mathbf{z}|\mathbf{x})]$ and $I(Z,X)$ is the mutual information between Z and X. This expansion shows that the KL term in ELBO is trying to reduce the mutual information between latent variables and the input data, which explains why VAEs often ignore the latent variable, especially when equipped with powerful decoders.

VAE with Information Maximization and Batch Prior Regularization

A natural solution to correct the anti-information issue in Eq. 4.26 is to maximize both the data likelihood lowerbound and the mutual information between $\mathbf{z}$ and the input data:

$$\begin{aligned} &\mathcal{L}_{\text{VAE}} + I(Z, X) = \\ &\mathbb{E}_{q(\mathbf{z}|\mathbf{x})p(\mathbf{x})}[\log p(\mathbf{x}|\mathbf{z})] - \mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z})) \end{aligned} \tag{4.27}$$

Therefore, jointly optimizing ELBO and mutual information simply cancels out the information-discouraging term. Also, we can still sample from the prior distribution for generation because of $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))$. Eq. 4.27 is similar to the objectives used in adversarial autoencoders (Makhzani et al., 2015; Kim et al., 2017). Our derivation provides a theoretical justification to their superior performance. Notably, Eq. 4.27 arrives at the same loss function proposed in infoVAE (Zhao S et al., 2017). However, our derivation is different, offering a new way to understand ELBO behavior.

The remaining challenge is how to minimize $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))$, since $q(\mathbf{z})$ is an expectation over $q(\mathbf{z}|\mathbf{x})$. When $\mathbf{z}$ is continuous, prior work has used adversarial training (Makhzani et al., 2015; Kim et al., 2017) or moment matching based distance function, e.g. Maximum Mean Discrepancy (MMD) (Zhao S et al., 2017) to regularize $q(\mathbf{z})$. It turns out that minimizing $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))$ for discrete $\mathbf{z}$ is much simpler than its continuous counterparts. Let $\mathbf{x}_n$ be a sample from a batch of N data points. Then we have:

$$q(\mathbf{z}) \approx \frac{1}{N}\sum_{n=1}^{N} q(\mathbf{z}|\mathbf{x}_n) = q'(\mathbf{z}) \tag{4.28}$$

where $q'(\mathbf{z})$ is a mixture of softmax from the posteriors $q(\mathbf{z}|\mathbf{x}_n)$ of each $\mathbf{x}_n$. We can approximate $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))$ by:

$$\mathrm{D}_{\mathrm{KL}}(q'(\mathbf{z})\|p(\mathbf{z})) = \sum_{k=1}^{K} q'(\mathbf{z}=k)\log\frac{q'(\mathbf{z}=k)}{p(\mathbf{z}=k)} \tag{4.29}$$

We refer to Eq. 4.29 as Batch Prior Regularization (BPR). When N approaches infinity, $q'(\mathbf{z})$ approaches the true marginal distribution of $q(\mathbf{z})$. In practice, we only need to use the data from each mini-batch assuming that the mini batches are randomized. Last, BPR is fundamentally different from multiplying a coefficient < 1 to anneal the KL term in VAE (Bowman et al., 2015). This is because BPR is a non-linear operation *log_sum_exp*. For later discussion, we denote our discrete infoVAE with BPR as DI-VAE.

Autoregressive Prior using Recurrent Neural Networks

After the BPR model is trained, we can also train a separate RNN, e.g. LSTM network to model the real $q(\mathbf{z})$ from the data. This is because although we encourage each dimension of $\mathbf{z}$ to be conditionally independent given $\mathbf{x}$, it is possible that there are still correlation among them, especially for models with information maximization. To see that,

if there are only two types of data and we have 2 binary variables in $\mathbf{z}$. Then $\mathbf{z}_1 = [1, 0]$ and $\mathbf{z}_2 = [0, 1]$. Given this condition, our estimated $q(\mathbf{z}) \approx \frac{1}{N}\sum_{n=1}^{N} q(\mathbf{z}|\mathbf{x}_n) = [0.5, 0.5]$, which has 0 BPR with the uniform prior. However, if we directly sample from prior distribution, it has 25% to get $[1, 1]$ and $[0, 0]$, which are undefined in data. Thus, training a LSTM prior can solve this problem. To achieve this, we first train a discrete VAE with only BPR regularization. After the model is trained and froze, an LSTM network is trained to minimize the following objective:

$$\mathcal{L}(\pi) = \sum_{k=1}^{K} \mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}_k|\mathbf{x})\|p_\pi(\mathbf{z}_i|\mathbf{z}_{i<k})) \tag{4.30}$$

where π is the parameters of the LSTM prior network. After this RNN prior network is trained, we can obtain higher quality samples from $\mathbf{z} \sim p(\mathbf{z})$ compared to the naive $p(\mathbf{z})$ in which each latent variable is independent.

4.3 Related Work

Besides the solutions proposed above, there are several related methods are developed to address the posterior collapse issue in VAE text generator.

KL Annealing

A simple technique proposed in (Bowman et al., 2015), which gradually increases the weight of the KL term from 0 to 1 during training.

Word drop decoding

Bowman et al (2015) also proposed to setting a certain percentage of the target words to 0. Although this approach has shown negative impact on the performance on the final decoder.

Non-autoregressive Decoder

This approach replaces the typical RNN-based decoder by a non-autoregressive decoder that decodes sentence in an non-autoregressive manner (in parallel or semi autoregressive), and therefore the decoder is forced to utilize the latent variable better. Deconvolutional neural network decoders are used to replace LSTM and the authors show improvement on learning latent variables (Semeniuta et al., 2017). Zichao et al., (2017b) used Dilated Convolutions to achieve better VAE text generation for similar reasons.

More Flexible Prior Distribution

One belief about why the standard VAE fail for text generation is that the latent space should be highly multi-modal for natural language, but standard VAE assumes a simple uni-modal Gaussian distribution. Serban et al., (2016a) discretizes the continuous distributions into chunks and provides analytic solutions to learn continuous distribution that can be multi-modal. Gu et al.,(2018) takes a further step by using GAN training (Miyato et al., 2018) to learn an arbitrary complex distribution for the posterior and prior distribution for $\mathbf{z}$. They show that by relaxing the KL divergence between two simple isotropic Gaussian distribution in VAE, the model becomes easier to learn the multi-modal behavior of dialog responses generation.

In summary, posterior collapse in text VAE is far from completely solved and deeper understanding is required. As analyzed in (Xiao et al., 2018), there is a trade off between the KL divergence and the reconstruction quality. That is whenever the system encodes more information into the latent space, it inevitably leads to higher KL divergence between the posterior and prior. Our proposed methods will show strong performance in terms of enable the models to encode more knowledge into the latent space while keep the KL divergence reasonable. Furthermore, a novel evaluation metric is proposed to facilitate finding the optimal balance between inference and generation.

4.4 Experiments

Experiments in this chapter will focus on unconditional response modeling. This setting ignores the dialog context and use a VAE to model the dialog response only. This setting is also known as language modeling with latent variable models (Bowman et al., 2015). The results from unconditional response modeling can easily generalize to conditional cases, including partial latent action and full latent actions. Experiments for conditional cases are left to later chapters where we actually build systems for dialog applications.

4.4.1 Evaluation Metrics

Evaluation are conducted in two categories: inference performance and generation performance.

Inference Performance

Inference performance tests if the posterior network learn to output latent code $\mathbf{z}$ that capture salient information about the response $\mathbf{x}$. Note here a latent $\mathbf{z}$ is not only expected to contain as much information about $\mathbf{x}$ as possible, but also expect to generalize and focus on global semantics rather than local information. The following metrics serve as indicators for the inference performance:

- Reconstruction Perplexity (Rec PPL): given $\mathbf{z}$ sampled from the posterior network, how well can a decoder reconstruct the original response $\mathbf{x}$.

- Text classification: using the posterior network as an feature extractor for $\mathbf{x}$ and use $\mathbf{z}$ as input feature to simple linear classifier and test if we can reliably use $\mathbf{z}$ to classify the responses into the right label.

Generation Performance

Generation performance tests the quality of response that can be generated from latent code sampled from the prior distribution. Ideally, the decoder should be able to generate coherent and natural responses and any high likely samples from the prior distribution should have reasonable generation performance rather than sudden drop of generation quality. In other words, the valid set of latent code should concentrate under the prior distribution and result in a smooth latent space without "holes" that lead to undefined decoder behavior. The following metrics can be readily used to test if this holds true.

- KL-divergence: the KL-divergence $\mathrm{D}_{\mathrm{KL}}(Q\|P)$ between the proposed posterior distribution with the prior distribution. Smaller KL implies less exposure bias at testing time for the decoder.
- Qualitative Analysis on Response Generation: given latent $\mathbf{z}$ sampled from the prior distribution, we evaluate the text generation quality via manually inspect the generated responses.
- Discriminator-based Evaluation: we further propose a novel method to test the generation quality of a model. After a model is trained, we use the model to unconditionally generate N sentences. Then we sample an additional N samples from the actual data to create a 50:50 balanced real vs. fake dataset. Then an RNN-based binary classifier is trained to classify if a given sentence is real or fake. Therefore, for a perfect generator, this classifier should only be able to achieve 50% accuracy since it is completely fooled by the generation results (Goodfellow et al., 2014).

4.4.2 Dataset

Two datasets are used in the following experiments. Penn Treebank (PTB) (Marcus et al., 1993), is a well-known used dataset for language modeling. The dataset consists of 929,000 training words, 73,000 validation words, and 82,000 test words. As part of the pre-processing, words were lower-cased, numbers were replaced with N, newlines were replaced with EOS, and all other punctuation was removed. The vocabulary is the most frequent 10,000 words with the rest of the tokens replaced by an UNK token (Mikolov et al., 2010). Models are evaluated based on perplexity, which is the average per-word log-probability (lower is better).

The second dataset used is MultiWoz (Budzianowski et al., 2018), the largest multi-domain slot filling dialog dataset. Multi-Woz is a slot-filling dataset that contains 10,438 dialogs on 6 different domains, 115,424 turns and 1,520,970 word tokens. 8,438 dialogs are for training and 1,000 each are for validation and testing. All the system turns in

MultiWoz is also manually annotated with domain/dialog acts and entities. Note that there can be multiple domain/dialog acts frames for each responses, making it a multi-label classification problem. For later text classification experiments, we remove the entities leading to a label vocabulary size 31. The top-15 frequent labels are as follows. (Format is: domain-dialog act (frequency)).

1. general-reqmore(13.01%)
2. general-bye(8.4%)
3. Hotel-Inform(7.6%)
4. Restaurant-Inform(7.5%)
5. Train-Inform(6.7%)
6. Attraction-Inform(6.5%)
7. Booking-Inform(5.3%)
8. Train-Request(5.1%)
9. Booking-Book(4.9%)
10. general-welcome(4.4%)
11. Hotel-Request(3.0%)
12. Restaurant-Request(2.9%)
13. Train-OfferBook(2.8%)
14. Booking-Request(2.5%)
15. Train-OfferBooked(2.1%)

4.5 Results and Discussion

4.5.1 Compared Models

For unconditional response modeling on PTB and MultiWoz datasets, a variety of models are compared. For VAEs with continuous (Gaussian) latent variables, we compared:

- VAE: the standard VAE trained with ELBO using LSTM encoder and decoder.
- KLA: the standard VAE trained with KL annealing as proposed in (Bowman et al., 2015). We linearly increases the KL weight from 0 to 1 in the first 5000 batches.
- BOW: the standard VAE trained with the proposed bag-of-words auxiliary loss.
- SEL:the standard VAE trained with the proposed response selection auxiliary loss.
- BOW+KLA: apply KLA with bag-of-words at the same time.
- SEL+KLA: apply KLA with response selection at the same time.

For discrete latent variable VAEs, the above variations of models still apply. We use the *D-* prefix to indicate that the latent variable is discrete. In addition to that, for discrete latent variable, we also implemented the proposed batch prior regularization that maximize the mutual information between $\mathbf{x}$ and $\mathbf{z}$. In summary, all the compared models are shown in table 4.1.

The above models are all equipped with an LSTM encoder and LSTM decoder with 1 layer hidden states and a hidden state size of 512. The word embeddings are randomly initialized with dimension 200. For Gaussian latent variable, the latent dimension is set to 256. For discrete latent variable, there are 100 4-way categorical variables. The optimization is done via Adam (Kingma and Ba, 2014) with learning rate 1E-4, dropout rate 30% and gradient clipping at 3.0. All the parameters are uniformly initialized between [-0.08, 0.08]. For response selection loss, 300 negative responses are uniformly

Name	Type of Var	KLA	Aux Loss	Mutal Info
VAE	Gaussian	/	/	/
KLA	Gaussian	/	/	/
BOW	Gaussian	/	bag-of-word	/
SEL	Gaussian	/	response selection	/
BOW+KLA	Gaussian	Yes	bag-of-word	/
SEL+KLA	Gaussian	Yes	response selection	/
D-VAE	Categorical	/	/	/
D-KLA	Categorical	/	/	/
D-BOW	Categorical	/	bag-of-word	/
D-SEL	Categorical	/	response selection	/
D-BOW+KLA	Categorical	Yes	bag-of-word	/
D-SEL+KLA	Categorical	Yes	response selection	/
D-BPR(LSTM)	Categorical	/	/	Yes

Table 4.1: Compared models for unconditional response modeling.

sampled from all unique responses occur in the training set. The reported results are on test dataset using models selected by the best ELBO loss on the validation data.

4.5.2 Results for Inference

Table 4.2 and Table 4.3 show the main results for training all the compared models on Penn Treebank and MultiWoz. Based on the results, we can observe and draw the following insights.

Model	PPL	$\mathrm{D}_{\mathrm{KL}}(q\|\|p)$	Model	PPL	$\mathrm{D}_{\mathrm{KL}}(q\|\|p)$	$\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|\|p)$
RNNLM	110.5	/	/	/	/	/
VAE	110.7	0.009	D-VAE	111.0	0.1591	0.03
KLA	111.5	2.02	D-KLA	110.2	0.261	0.01
BOW	97.72	7.41	D-BOW	95.3	5.89	0.03
SEL	92.19	7.26	D-SEL	91.25	10.53	0.01
BOW+KLA	66.94	15.23	D-BOW+KLA	88.5	9.79	0.02
SEL+KLA	72.58	18.08	D-SEL+KLA	92.1	17.5	0.02
/	/	/	D-BPR(LSTM)	45.2	203.2 (80.3)	0.01

Table 4.2: The reconstruction perplexity, $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))$, $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))$ (discrete only) on Penn Treebank test set.

BOW and SEL both mitigate the posterior collapse problem: results of the KL-divergence and reconstruction perplexity on both datasets show that by adding bag-of-word or response selection loss to the original ELBO objectives enable the latent variable to encode substantial more information into the latent space, which results in a non-trivial KL-divergence and a much smaller reconstruction perplexity compared to

Model	PPL	$D_{KL}(q\|\|p)$	Model	PPL	$D_{KL}(q\|\|p)$	$D_{KL}(q(\mathbf{z})\|\|p)$
RNNLM	4.99	/	/	/	/	/
VAE	4.93	0.004	D-VAE	5.3	0.72	0.001
KLA	4.88	0.01	D-KLA	5.58	1.91	0.002
BOW	3.21	11.49	D-BOW	3.94	8.76	0.005
SEL	3.07	14.08	D-SEL	4.51	6.34	0.004
BOW+KLA	2.98	13.46	D-BOW+KLA	3.80	9.19	0.005
SEL+KLA	3.03	13.72	D-SEL+KLA	3.75	11.716	0.003
/	/	/	D-BPR (LSTM)	1.76	202.3 (61)	0.009

Table 4.3: The reconstruction perplexity, KL terms and mutual information (discrete only) on MultiWoz test set.

the vanilla VAEs or RNNLMs. Furthermore, the results also show that KLA are complementary to both proposed objectives, since we can observe the reconstruction perplexity both further improves by comparing BOW with BOW+KLA or SEL with SEL+KLA (at a cost of a larger KL divergence).

KLA alone falls short Only using KLA alone has very limited effects in terms of mitigating the posterior collapse challenge. Figure 4.2 visualizes the evolution of the KL cost. We can see that for the standard model, the KL cost crashes to 0 at the beginning of training and never recovers. On the contrary, the model with only KLA learns to encode substantial information in latent $\mathbf{z}$ when the KL cost weight is small. However, after the KL weight is increased to 1 (after 5000 batch), the model once again decides to ignore the latent $\mathbf{z}$ and falls back to the naive implementation. The model with BOW loss, however, consistently converges to a non-trivial KL cost even without KLA, which confirms the importance of BOW loss for training latent variable models with the RNN decoder. Similar trend can also be observed for response selection loss.

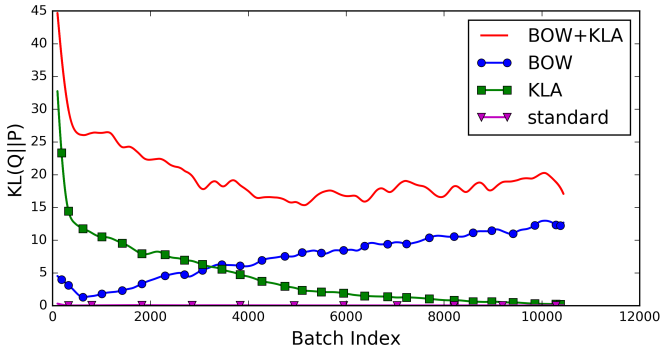

Figure 4.2: The value of the KL divergence during training with different setups on Penn Treebank.

Information maximization leads to very different system behavior as we can see the proposed D-BPR model achieve significantly lower reconstruction perplexity on both datasets, at an expected cost with very big KL-divergence. This is expected since

maximizing the mutual information leads to very peaky $q(\mathbf{z}|\mathbf{x})$, i.e. the posterior latent codes are almost one-hot vectors. This is because the modified information maximization objective only requires the marginal distribution of $\mathbf{z}$ to be similar to the prior, i.e. $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z})) \sim 0$. From both tables, we can see that D-BPR ensure that $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z})\|p(\mathbf{z}))$ is very small while allowing $\mathrm{D}_{\mathrm{KL}}(q(\mathbf{z}\mathbf{x})\|p(\mathbf{z}))$ to be very large. On the other hand, other ELBO based VAEs encourage both quantities to be as small as possible.

Therefore, it is evident that the model with information maximization encodes substantially more information into its latent code, but the remaining question is if it can still be used as a generative model by sampling from the prior distribution of unconditional response generation. This important question will be addressed shortly in Section 4.5.3.

Continuous latent variables encode more information moreover, by comparing results horizontally, we can observe that the continuous latent models often enjoy lower reconstruction perplexity with slightly larger KL-divergence. This is expected since the discrete version can be thought as quantified version of a continuous counterpart, i.e. discretize the original continuous variable into 3 categories in our case. Although it is tempting to draw conclusion that continuous latent variable is better than discrete variables, discrete latent variable models have unique properties that are useful for practical applications, including interpretability and safety. These two features will be extensively discussed in Chapter 6 and Chapter 8.

Using z for Text Classification

It is well-understood that autoencoder style feature extraction without regularization does not necessarily leads to robust and generalized features about the input since the model is pruned to create one-to-one mapping instead of learning features that can generalize (Vincent et al., 2008; Hill et al., 2016). This principle can also be applied to latent action learning and only a low reconstruction perplexity does not guarantee robust features about the responses are acquired by the model. Therefore, we further test the inference performance by using the trained recognition network as a feature extractor and trained a simple 2-layer multi-layer perceptron to predict the 31 dialog acts on MultiWoz dataset. Since each system response in MultiWoz may contains more than 1 dialog act, we created a multi-label classifier using Sigmoid output for each dialog act, and use F-1 score to measure the classifier's performance.

For models with Gaussian latent variables, the concatenation of mean vector μ and variance σ^2 is used as the feature and for Categorical latent variables, the posterior softmax output $q(\mathbf{z}|\mathbf{x})$ is used as the feature vectors. Therefore, the continuous latent models has a feature vector size of $256 \times 2 = 512$ and discrete models have a feature vectors of size $256 \times 3 = 768$.

Table 4.4 shows the results. As we can see, the baseline VAE and KLA models perform poorly in prediction F-1 scores. For all models with proposed methods, i.e. BOW, SEL or BPR achieve substantial performance improvement and validates that the resulting latent variable contain salient features about the responses. Through the results, we can also notice that BOW models in general perform better than SEL models. Also,

Model	F-1	Model	F-1
VAE	38.7%	D-VAE	19.7%
KLA	44.7%	D-KLA	50.1%
BOW	**81.0%**	D-BOW	75.4%
SEL	77.3%	D-SEL	60.7%
BOW+KLA	80.2%	D-BOW+KLA	70.5%
SEL+KLA	77.8%	D-SEL+KLA	74.0%
/	/	D-BPR	**77.7 %**

Table 4.4: F-1 score for predicting multi-label dialog acts using $\mathbf{z}$ as feature on MultiWoz test set. Bold-face number indicate statistically significant best results using Wilcoxon signed-rank test p-value < 0.01.

continuous models perform better than discrete ones.

4.5.3 Results for Generation from Prior

Given the effectiveness of learning meaningful posterior distribution, we now switch gear to evaluate if the model can generate coherent dialog responses given latent samples from the prior distribution. Since in the extreme case, a unregularized VAE (removing the KL term from ELBO) can encode and reconstruct the input perfectly, but cannot generate coherent samples by taking latent code drawing from prior distribution.

Results in Table 4.2 and Table 4.3 show that the proposed methods all lead to a larger KL-divergence in order to encode non-trivial information into the latent space. Meanwhile, although larger KL-divergence between the posterior and prior suggest that higher risk of generating incoherent responses, we argue that it is difficult to decide the optimal value of KL-divergence only based on its absolute value for the following reasons:

1. There is trade off between KL divergence and reconstruction perplexity and it is not easy to know the optimal balance point.
2. It is possible that the decoder can generalize and still generate high quality samples by using $\mathbf{z} \sim p(\mathbf{z})$ even the there is a non-trivial KL divergence.
3. The baseline VAE has KL=0 and it will always generate the same sentence if sampling from $\mathbf{z}$ and keep the decoder greedy. This obviously suggest bad generation behavior but not captured by the KL term.

Table 4.5 shows qualitative example to illustrate the limitations of depending on KL divergence to evaluate the generation performance of a model. We train 3 VAEs with the proposed BOW loss with 3 different weights applied to the auxiliary loss. When the weight is 0, the model degenerates to standard VAE where the posterior collapse. Therefore, only sampling from $p(\mathbf{z})$ leads to the same response output since we set the decoder to be greedy. As the weight increases to 1 and later to 2, results in the table show the reconstruction perplexity decreases and the KL-term increases as expected. Meanwhile, the generated samples become increasingly more diverse as the KL-divergence

Model	PPL	$D_{KL}(q\|p)$	Generated Samples
BOW (0)	4.99	0.0	I am looking for a train leaving on Thursday. - I am looking for a train leaving on Thursday. - I am looking for a train leaving on Thursday. - I am looking for a train leaving on Thursday. - I am looking for a train leaving on Thursday.
BOW (1)	2.98	13.46	- I need the address for the restaurant . - you are welcome. - what is the name of the college? - I need a taxi to take me to the restaurant and the restaurant . - I have made that reservation for you and your reference number is [number]
BOW (2)	2.7	17.33	- ok, you are booked at the [restaurant name] at [number] mill road city centre, and the phone number is [number]. - that would be fine. i need the hotel for [number] nights starting on Wednesday. - is there any in the moderate price range? - I will need a destination in the destination and your destination is [number] minutes - yes. I need a train that is leaving on Monday .

Table 4.5: Generation samples from BOW models with different λ weight to the bag-of-word loss. As the weight to BOW becomes higher, there are more information encoded into the latent space. Meanwhile, the generation performance decreases as the posterior distribution becomes increasingly more different from the prior distribution.

term becomes larger, but it is difficult to quantitatively judge if the generation quality decreases as the KL term increases. Therefore, although the KL term in ELBO correlates with the amount of information and variations encoded in the latent space, it is poor measurement for deciding on the actual generation quality.

Discriminator-based Evaluation

As a result, we propose to use a trained discriminator (described in Section 4.4.1) to classify if an input response is from real data or it is generated from the model. This has the advantages to automatically detect errors in outputs from VAEs as a generative

model, including:

1. Incoherence and grammatical errors: if the generated responses contains significantly more language errors than the real data, these erroneous patterns can be easily captured by the discriminator and used to predict which one is real vs. fake.
2. Posterior collapse and dullness: by training a classifier that classifies between real data and generated responses that only sample from the latent variable, this classifier can detect if posterior collapse happen. If the generator only generate the same or a few response all the time, then the classifier can utilize this pattern for prediction. It also tells us how much variations are captured in the latent space.

Model	$\text{Acc}_{\mathbf{x}\sim p(\mathbf{z})}$	$\text{Acc}_{\mathbf{x}\sim p(\mathbf{z})p(\mathbf{x}\|\mathbf{z})}$	Model	$\text{Acc}_{\mathbf{x}\sim p(\mathbf{z})}$	$\text{Acc}_{\mathbf{x}\sim p(\mathbf{z})p(\mathbf{x}\|\mathbf{z})}$
KLA	100%	62%	D-KLA	95.3%	**65.1%**
BOW+KLA	76.4%	65.7%	D-BOW+KLA	76.4%	75.9%
SEL+KLA	76.8%	**61.5%**	D-SEL+KLA	**72.3%**	**65.1%**
/	/	/	D-BPR	96.2%	95.1%
/	/	/	D-BPR+LSTM	79.8%	78.1%

Table 4.6: Accuracy of an RNN classifier for distinguishing between the generated text vs the real data. Lower the better and the ideal generator should have 50%, i.e. completely confuses the discriminator. The best results are in bold-face with statistical significance using 2-proportion z-test p-value < 0.01 compared to the second best systems in its column.

Table 4.4 summarizes the results on MultiWoz dataset. The classifier is trained on 25,120 pairs of real vs. fake sentencens generated from a trained VAE model. The results is reported on a hold-out test set that contains 3,140 sentence pairs. The classifier accuracy is reported for discriminating real data between responses sampled from $p(\mathbf{z})$ and sampled from both $p(\mathbf{z})$ and $p(\mathbf{x}|\mathbf{z})$. The results confirm that the baseline KLA methods can be detected 100% since it only generates the same output when only sample from $p(\mathbf{z})$. Furthermore, the proposed SEL and BOW auxiliary loss perform similarly for continuous latent variables, achieving about 76% detection rate. For discrete latent variables, the BOW loss performs worse than the SEL auxiliary loss. Eventually, the VAE+BPR with uniform prior performs very badly, resulting in about 95% detection rate. The following are some sample outputs form D-BPR:

- i am sorry when you finish is a place , but i need the price range and postcode .
- you are all set , i recommend ashley hotel in the south . can i get you too ?
- am sorry , any of those will be in the centre . can you please restate your needs ?
- hi , i need a cab to the restaurant and bar for you . is there anything else that

These responses contain utterance segments that clearly belong to multiple speakers or intents. This confirms our hypothesis that although $\text{D}_{\text{KL}}(q(\mathbf{z})\|p(\mathbf{z})) = 0$, sample from $p(\mathbf{z})$ will leads to specific combination of latent code that does not make sense as a posterior. On the other hand, with the learned LSTM prior distribution, the detection rate significantly decreases to about 80%.

A follow up study is to see if we can further improve the performance of BPR-LSTM models via latent space reshaping. Given the same model size, we can change the number of latent variables and the size of each variable. The hypothesis is by reducing the number of latent variables, we can reduce the chance for improbable latent code at generation time. Table 4.7 shows that by reshaping the latent space into smaller number of

# of Var	Latent Size	PPL	$\mathrm{D}_{\mathrm{KL}}(q\|\|p)$	$\mathrm{Acc}_{\mathbf{x}\sim p(\mathbf{z})}$	$\mathrm{Acc}_{\mathbf{x}\sim p(\mathbf{z})p(\mathbf{x}\|\mathbf{z})}$
256	3	2.31	61.4	79.8%	78.1%
64	6	2.35	45.3	75.3%	75.9%
32	12	2.64	29.7	73.4%	71.8%
16	24	2.65	25.4	**68.8%**	**67.0%**

Table 4.7: Reconstruction perplexity, KL-divergence, detection rate for D-BPR-LSTM with various latent size. The lower the detection rate, the better the generation quality. Bold-face number indicate statistically significant best results using 2 proportion z-test p-value < 0.01.

latent variables lead, D-BPR-LSTM can work really well in terms of both inference ability (i.e. low reconstruction PPL) and strong generation performance (i.e. low detection rate) despite that fact that its KL-divergence is still significantly higher than the ones of non-BPR discrete models.

Last but not least, Table 4.8 shows the detection rate can also be used to tune the auxiliary loss weight of a continuous BOW model. As the weight on BOW loss increases,

Aux Weight	PPL	$\mathrm{D}_{\mathrm{KL}}(q\|\|p)$	$\mathrm{Acc}_{\mathbf{x}\sim p(\mathbf{z})}$	p-value
0	4.99	0	100%	N/A
0.5	3.34	8.17	80.5%	< 0.01
1.0	2.98	13.46	76.4%	< 0.01
2.0	2.71	19.24	74.6%	0.1004
5.0	2.32	31.95	78.4%	< 0.01

Table 4.8: Detection rate of BOW models with various weights multiplied to the auxiliary loss. The lower the detection rate the better. The p-value shows statistical significance test for rejecting null hypothesis that the current detection rate is the same as the previous row using 2 proportion z-test. Therefore, besides the difference between weight=1.0 and 2.0, other differences are significant.

the KL-divergence monotonically increases as well, making it difficult to decide the best point of stopping. However, the detection rate showing a U-shape curve where the lowest detection rate suggests the optimal weight. When the weight is too small, the detection rate is high because the latent space encodes little information and the classifier can utilizes the dullness of the output to predict which response is fake. When the weight becomes too large, the information encoded in the latent space increase, but at the same time it increases the chance of ungrammatical generation. This also

allows the classifier to detect which sample is generated from the model. Thus, using the discriminator for evaluation provides a robust method to tune VAEs models.

4.6 Conclusions

In conclusion, this chapter presents a network architecture and a set of latent variables algorithms to learn latent variable neural networks that can be used to create latent action E2E dialog models. The contributions include (1) a description of network architecture that utilizes SVI, Reparametrization Trick to achieve the proposed latent action definitions. (2) two auxiliary objective function that mitigates the posterior collapse problem for training text VAEs (3) a mutual information maximization approach to learn discrete VAEs with batch prior regularization and auto-regressive prior network (4) a comprehensive analysis on the performance of the proposed methods for both inference and generation (5) a novel discriminator based evaluation measure to evaluate the generation performance of text VAEs. Experiments are conducted on both Penn Tree Bank and MultiWoz and results suggest that the proposed methods can significantly improves both the inference and generation performance of baseline models. The results from this chapter paves the foundation for introducing latent variables to improve end-to-end dialog systems.

Chapter 5

Latent Actions for Discourse-level Diversity

This chapter presents a type of partial latent action that is designed to separate the high-level intents from variations in surface realization. This in turn enables a generation-based dialog system to generate a diverse set of appropriate responses with different discourse-level intents meanings. We further improve the system by introducing a knowledge guidance mechanism that integrates linguistic knowledge into the learning process of the latent action. We propose a novel evaluation metric that concerns the one-to-many property to quantify an open-domain dialog performance. Experiments validate the superiority of the proposed methods in terms of both diversity and appropriateness compared to other baseline models. [1]

5.1 The Dull Response Problem

We are interested in building open-domain dialog agents that can chat with human users about any topics without particular purpose. As pointed out in Chapter 2, this task is categorized into chat-oriented dialog system and it is a extremely challenging objective because:

1. Since there is no limitation in domains, the action space of a chat-oriented dialog agent is infinite and is highly diverse in terms of content and surface realizations.
2. The system needs to model arbitrary multi-turn dialog history and extract salient semantic information that are relevant to the rest of the conversations and resolving references spanning over multiple turns.
3. The systems needs to be equipped with world-knowledge in order to appropriately introduce topic of discussion and respond with users' input with suitable responses.

All of the above challenges are far from solved even by the state-of-the-art technologies, but significant progress has been made due to the development of E2E generation-based

[1] The code and data are available at `https://github.com/snakeztc/NeuralDialog-CVAE`.

dialog systems. First generation based systems often uses a auto-regressive decoder to factorize the response generation process as a conditional language modeling task: $p(\mathbf{x}|\mathbf{c}) = \prod_i^T (p(w_i|w_{<i}, \mathbf{c}))$. At inference time, a well trained generation system can generate responses with an unbounded number of words, which gives the possibility of modeling the infinite action space in open-domain conversations. Second, the dialog encoder based on various neural network architectures (Cho et al., 2014) can encode variable length dialog history and learn to extract useful features, which matches with the second requirement. Last, generation-based dialog systems have been often trained on massive conversational datasets, e.g. movie subtitles (Li et al., 2016a), forum discussion (Lowe et al., 2015) and etc. Research has shown that language model type training in fact implicitly captures common sense knowledge into the model and makes the model to learn information that are reasonable in the real world (Trinh and Le, 2018).

However, early attempts in training large generation-based dialog models using encoder-decoder networks all suggest that the resulting system often generates generic and dull responses (e.g. *I don't know*), rather than meaningful and specific answers (Li et al., 2015a; Serban et al., 2016c). Follow up research further discovers that even training very deep models on massive amounts of data (8-layer attention encoder decoder network with more than 200 million training conversations), the resulting models still suffer from the dull response problem (Shao et al., 2017). Therefore, the issue lies in the model and training algorithm itself instead of the lack of training data or the incompetence of the model. Prior to this work, there have been many attempts to explain and solve this limitation, and they can be broadly divided into two categories: context enrichment and decoder diversification. The first type has focused on augmenting the input of encoder-decoder models with richer context information, in order to generate more specific responses. Li et al., (2016a) captured speakers' characteristics by encoding background information and speaking style into the distributed embeddings, which are used to re-rank the generated response from an encoder-decoder model. Xing et al., (2016) maintain topic encoding based on Latent Dirichlet Allocation (LDA) (Blei et al., 2003) of the conversation to encourage the model to output more topic coherent responses.

In the second category, researchers have been improving the architecture of encoder-decoder models. Li et al,. (2015a) proposed to optimize the standard encoder-decoder by maximizing the mutual information between input and output, which in turn reduces generic responses. This approach penalized unconditionally high frequency responses, and favored responses that have high conditional probability given the input. Wiseman and Rush (2016) focused on improving the decoder network by alleviating the biases between training and testing. They introduced a search-based loss that directly optimizes the networks for beam search decoding. The resulting model achieves better performance on word ordering, parsing and machine translation. Besides improving beam search, Li et al., (2016b) pointed out that the MLE objective of an encoder-decoder model is unable to approximate the real-world goal of the conversation. Thus, they initialized a encoder-decoder model with a MLE objective and leveraged reinforcement learning to fine tune the model by optimizing three heuristic rewards functions: infor-

mativity, coherence, and ease of answering.

5.1.1 The One-to-Many Nature in Response Generation

Building on the past work in dialog managers and encoder-decoder models, the key idea of this chapter is to model dialogs as a *one-to-many* problem at the discourse level. Previous studies indicate that there are many factors in open-domain dialogs that decide the next response, and it is non-trivial to extract all of them. Intuitively, given a similar dialog history (and other observed inputs), there may exist many valid responses (at the discourse level), each corresponding to a certain configuration of the latent variables that are not presented in the input. To uncover the potential responses, we strive to model a probabilistic distribution over groups of responses that have different semantics using a latent variable (Figure 5.1). This allows us to generate diverse responses by drawing samples from the learned distribution and reconstruct their words via a decoder neural network.

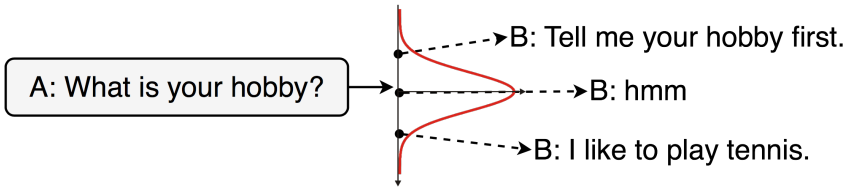

Figure 5.1: Given A's question, there exists many valid responses from B for different assumptions of the latent variables, e.g., B's hobby.

Specifically, our contributions are 4-fold: 1. We present a novel neural dialog model adapted from conditional variational autoencoders (CVAE) (Yan et al., 2015; Sohn et al., 2015), which introduces a latent variable that can capture discourse-level variations as described above. We propose Knowledge-Guided CVAE (kgCVAE), which enables easy integration of expert knowledge and results in performance improvement and model interpretability. 3. We develop a training method in addressing the difficulty of optimizing CVAE for natural language generation (Bowman et al., 2015). 4. We evaluate our models on human-human conversation data and yield promising results in: (a) generating appropriate and discourse-level diverse responses, and (b) showing that the proposed training method is more effective than the previous techniques.

5.2 Proposed Models

Following the $(\mathbf{c} - \mathbf{x})$ notation defined in Chapter 3, each dyadic conversation can be represented via three random variables: the dialog context $\mathbf{c}$ (context window size $k-1$), the response utterance $\mathbf{x}$ (the k^{th} utterance) and a latent variable $\mathbf{z}$, which is used to capture the latent distribution over the valid responses. Further, $\mathbf{c}$ is composed of the dialog history: the preceding k-1 utterances; conversational floor (1 if the utterance is from the same speaker of $\mathbf{x}$, otherwise 0) and meta features m (e.g. the topic).

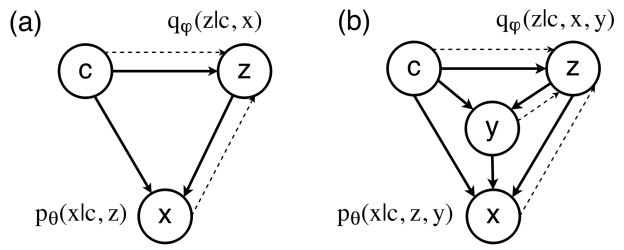

Figure 5.2: Graphical models of CVAE (a) and kgCVAE (b)

5.2.1 Partial Latent Action for Dialog Generation

We are interested in learning a latent action that represents the illocutionary force of the response, so that *partial latent action* is a suitable choice. We define the conditional distribution $p(\mathbf{x}, \mathbf{z}|\mathbf{c}) = p(\mathbf{x}|\mathbf{z}, \mathbf{c})p(\mathbf{z}|\mathbf{c})$ and our goal is to use deep neural networks to approximate $p(\mathbf{z}|\mathbf{c})$ and $p(\mathbf{x}|\mathbf{z}, \mathbf{c})$. We refer to $p_\pi(\mathbf{z}|\mathbf{c})$ as the *prior network* or *policy network* and $p_\theta(\mathbf{x}, |\mathbf{z}, \mathbf{c})$ as the *response decoder*. Then the generative process of $\mathbf{x}$ is (Figure 5.2 (a)):

1. Sample a latent variable $\mathbf{z}$ from the prior network $p_\pi(\mathbf{z}|\mathbf{c})$.
2. Generate $\mathbf{x}$ through the response decoder $p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{c})$.

Note that this formulation also makes this model a Conditional Variational Autoencoder (CVAE) (Sohn et al., 2015; Yan et al., 2015). CVAE can be efficiently trained with methods the SGVB framework described in Chapter 4 by maximizing the variational lower bound of the conditional log likelihood. We assume $\mathbf{z}$ follows multivariate Gaussian distribution with a diagonal covariance matrix and introduce a *recognition network* $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c})$ to approximate the true posterior distribution $p(\mathbf{z}|\mathbf{x}, \mathbf{c})$. Sohn and et al,. (2015) have shown that the variational lower bound can be written as:

$$\begin{aligned}\mathcal{L}(\theta, \pi, \phi; \mathbf{x}, \mathbf{c}) &= -\mathrm{D}_{\mathrm{KL}}[q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c})\|p_\pi(\mathbf{z}|\mathbf{c})] \\ &\quad + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x})}[\log p_\pi(\mathbf{x}|\mathbf{z}, \mathbf{c})] \\ &\leq \log p(\mathbf{x}|\mathbf{c})\end{aligned} \tag{5.1}$$

Figure 5.3 demonstrates an overview of our model. The utterance encoder is a bidirectional recurrent neural network (BRNN) (Schuster and Paliwal, 1997) with a gated recurrent unit (GRU) (Chung et al., 2014) to encode each utterance into fixed-size vectors by concatenating the last hidden states of the forward and backward RNN $u_i = [\vec{h_i}, \overleftarrow{h_i}]$. $\mathbf{x}$ is simply u_k. The context encoder is a 1-layer GRU network that encodes the preceding k-1 utterances by taking $u_{1:k-1}$ and the corresponding conversation floor as inputs. The last hidden state $h^{\mathbf{c}}$ of the context encoder is concatenated with meta features and $\mathbf{c} = [h^{\mathbf{c}}, m]$. Since we assume $\mathbf{z}$ follows isotropic Gaussian distribution, the recognition network $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c}) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$ and the prior network $p_\theta(\mathbf{z}|\mathbf{c}) \sim \mathcal{N}(\boldsymbol{\mu}', \boldsymbol{\sigma}'^2\mathbf{I})$, and then

we have:

$$\begin{bmatrix} \boldsymbol{\mu} \\ \log(\boldsymbol{\sigma}^2) \end{bmatrix} = W_r \begin{bmatrix} \mathbf{x} \\ \mathbf{c} \end{bmatrix} + b_r \tag{5.2}$$

$$\begin{bmatrix} \boldsymbol{\mu}' \\ \log(\boldsymbol{\sigma}'^2) \end{bmatrix} = \text{MLP}_p(\mathbf{c}) \tag{5.3}$$

We then use the Reparametrization Trick (Kingma and Welling, 2013) to obtain samples of $\mathbf{z}$ either from $\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$ predicted by the recognition network (training) or $\mathcal{N}(\mathbf{z}; \boldsymbol{\mu}', \boldsymbol{\sigma}'^2\mathbf{I})$ predicted by the prior network (testing). Finally, the response decoder is a 1-layer GRU network with initial state $s_0 = W_i[\mathbf{z}, \mathbf{c}] + b_i$. The response decoder then predicts the words in $\mathbf{x}$ sequentially.

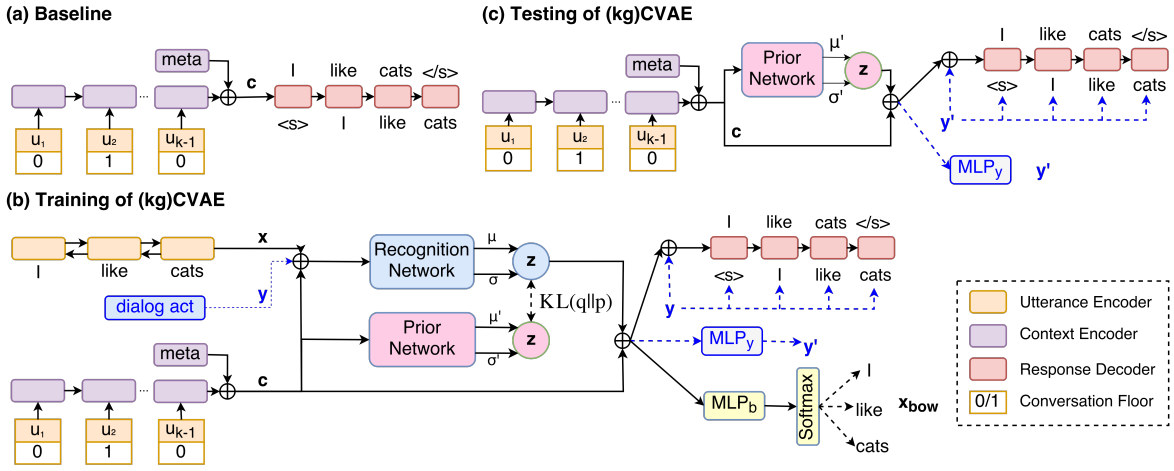

Figure 5.3: The neural network architectures for the baseline and the proposed CVAE/kgCVAE models. $\bigoplus$ denotes the concatenation of the input vectors. The dashed blue connections only appear in kgCVAE.

5.2.2 Knowledge-Guided CVAE (kgCVAE)

In practice, training CVAE is a challenging optimization problem and often requires large amount of data. On the other hand, past research in spoken dialog systems and discourse analysis has suggested that many linguistic cues capture crucial features in representing natural conversation. For example, dialog acts (Poesio and Traum, 1998) have been widely used in dialog managers (Litman and Allen, 1987; Raux et al., 2005; Zhao and Eskenazi, 2016) to represent the propositional function of the system. Therefore, we conjecture that it will be beneficial for the model to learn meaningful latent $\mathbf{z}$ if it is provided with explicitly extracted discourse features during the training.

In order to incorporate the linguistic features, such as dialog acts or topics into the basic CVAE model, we first denote the set of linguistic features as $\mathbf{y}$. Then we assume that the generation of $\mathbf{x}$ depends on $\mathbf{c}$, $\mathbf{z}$ and $\mathbf{y}$. $\mathbf{y}$ relies on $\mathbf{z}$ and $\mathbf{c}$ as shown in Figure 5.2.

Specifically, during training the initial state of the response decoder is $s_0 = W_i[\mathbf{z}, \mathbf{c}, \mathbf{y}]+b_i$ and the input at every step is $[e_t, \mathbf{y}]$ where e_t is the word embedding of t^{th} word in $\mathbf{x}$. In addition, there is an MLP to predict $\mathbf{y}' = \text{MLP}_{\mathbf{y}}(\mathbf{z}, \mathbf{c})$ based on $\mathbf{z}$ and $\mathbf{c}$. In the testing stage, the predicted $\mathbf{y}'$ is used by the response decoder instead of the oracle decoders. We denote the modified model as knowledge-guided CVAE (kgCVAE) and developers can add desired discourse features that they wish the latent variable $\mathbf{z}$ to capture. KgCVAE model is trained by maximizing:

$$\begin{aligned}\mathcal{L}(\theta, \pi, \phi, \tau; \mathbf{x}, \mathbf{c}, \mathbf{y}) = &- \mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c}, \mathbf{y}) \| p_\pi(\mathbf{z}|\mathbf{c})) \\ &+ \mathbf{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{c}, \mathbf{y})] \\ &+ \mathbf{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x},\mathbf{y})}[\log p_\tau(\mathbf{y}|\mathbf{z}, \mathbf{c})] \end{aligned} \tag{5.4}$$

Since now the reconstruction of $\mathbf{y}$ is a part of the loss function, kgCVAE can more efficiently encode $\mathbf{y}$-related information into $\mathbf{z}$ than discovering it only based on the surface-level $\mathbf{x}$ and $\mathbf{c}$. Another advantage of kgCVAE is that it can output a linguistic label (e.g. dialog act) along with the word-level responses, which allows easier interpretation of the model's outputs. Note that we only assume $\mathbf{y}$ is available at training time, and at testing time, predicted $\mathbf{y}$ will be used.

5.2.3 Optimization Challenges

Since our decoder ϕ is a auto-regressive LSTM, training both loss 5.1 and 5.4 suffer from the posterior collapse problem. Furthermore, since the latent $\mathbf{z}$ is defined by to a continuous variable, the proposed multi-task learning method is used to tackle the posterior collapse problem. Specifically, the *bag-of-word* auxiliary loss defined in Chapter 4 is used enforce the effectiveness of $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c})$. Note that for KgCVAE, we assume that even with the presence of linguistic feature $\mathbf{y}$ regarding $\mathbf{x}$, the prediction of $\mathbf{x}_{bow}$ still only depends on $\mathbf{z}$ and $\mathbf{c}$. Therefore, we have:

$$\begin{aligned}\mathcal{L}(\theta, \pi, \phi, \tau; \mathbf{x}, \mathbf{c}, \mathbf{y}) = &- \mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c}, \mathbf{y}) \| p_\theta(\mathbf{z}|\mathbf{c})) \\ &+ \mathbf{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x},\mathbf{y})}[\log p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{c}, \mathbf{y})] \\ &+ \mathbf{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x},\mathbf{y})}[\log p_\tau(\mathbf{y}|\mathbf{z}, \mathbf{c})] \\ &+ \mathbf{E}_{q_\phi(\mathbf{z}|\mathbf{c},\mathbf{x},\mathbf{y})}[\log p_\tau(\mathbf{x}_{bow}|\mathbf{z}, \mathbf{c})] \end{aligned} \tag{5.5}$$

5.2.4 Generalized Precision and Recall for Dialog Response Generation Evaluation

Automatically evaluating an open-domain generative dialog model is an open research challenge (Liu et al., 2016). Following our *one-to-many* hypothesis, we propose the following metrics. We assume that for a given dialog context $\mathbf{c}$, there exist $M_{\mathbf{c}}$ reference responses r_j, $j \in [1, M_{\mathbf{c}}]$. Meanwhile a model can generate N hypothesis responses h_i,

$i \in [1, N]$. The generalized response-level precision/recall for a given dialog context is:

$$\text{precision(c)} = \frac{\sum_{i=1}^{N} \max_{j \in [1, M_{\mathbf{c}}]} d(r_j, h_i)}{N} \quad \text{Appropriateness}$$

$$\text{recall(c)} = \frac{\sum_{j=1}^{M_{\mathbf{c}}} \max_{i \in [1, N]} d(r_j, h_i))}{M_{\mathbf{c}}} \quad \text{Diversity}$$

where $d(r_j, h_i)$ is a distance function which lies between 0 to 1 and measures the similarities between r_j and h_i. The final score is averaged over the entire test dataset and we report the performance with 3 types of distance functions in order to evaluate the systems from various linguistic points of view:

1. Smoothed Sentence-level BLEU (Chen and Cherry, 2014): BLEU is a popular metric that measures the geometric mean of modified n-gram precision with a length penalty (Papineni et al., 2002; Li et al., 2015a). We use BLEU-1 to 4 as our lexical similarity metric and normalize the score to 0 to 1 scale.
2. Cosine Distance of Bag-of-word Embedding: a simple method to obtain sentence embeddings is to take the average or extrema of all the word embeddings in the sentences (Forgues et al., 2014; Adi et al., 2016). The $d(r_j, h_i)$ is the cosine distance of the two embedding vectors. We used Glove embedding described in Section 5.3 and denote the average method as A-bow and extrema method as E-bow. The score is normalized to [0, 1].
3. Dialog Act Match: to measure the similarity at the discourse level, the same dialog-act tagger from 5.3 is applied to label all the generated responses of each model. We set $d(r_j, h_i) = 1$ if r_j and h_i have the same dialog acts, otherwise $d(r_j, h_i) = 0$.

Intuitively, assuming that the test set has high quality multiple reference responses collected for each dialog context, the proposed evaluation metric should correlate much better with a model's actual performance compared to existing metrics that only compare to a single reference response. We are glad to see that there are many research groups have begun to use our proposed generalized precision and recall for evaluating their response generation systems (Yang et al., 2017a; Gu et al., 2018; Gao et al., 2019; Liu et al., 2019).

5.3 Experiments

5.3.1 Dataset

We chose the Switchboard (SW) 1 Release 2 Corpus (Godfrey and Holliman, 1997) to evaluate the proposed models. SW has 2400 two-sided telephone conversations with manually transcribed speech and alignment. In the beginning of the call, a computer operator gave the callers recorded prompts that define the desired topic of discussion. There are 70 available topics. We randomly split the data into $2316/60/62$ dialogs for train/validate/test. The pre-processing includes (1) tokenize using the NLTK tokenizer (Bird et al., 2009); (2) remove non-verbal symbols and repeated words due to

false starts; (3) keep the top 10K frequent word types as the vocabulary. The final data have $207,833/5,225/5,481$ $(\mathbf{c}, \mathbf{x})$ pairs for train/validate/test. Furthermore, a subset of SW was manually labeled with dialog acts (Stolcke et al., 2000). We extracted dialog act labels based on the dialog act recognizer proposed in (Ribeiro et al., 2015). The features include the uni-gram and bi-gram of the utterance, and the contextual features of the last 3 utterances. We trained a Support Vector Machine (SVM) (Suykens and Vandewalle, 1999) with linear kernel on the subset of SW with human annotations. There are 42 types of dialog acts and the SVM achieved 77.3% accuracy on held-out data. Then the rest of SW data are labelled with dialog acts using the trained SVM dialog act recognizer.

5.3.2 Collection of Multiple Reference Responses

We collected multiple reference responses for each dialog context in the test set by information retrieval techniques combined with traditional a machine learning method. First, we encode the dialog history using Term Frequency-Inverse Document Frequency (TFIDF) (Salton and Buckley, 1988) weighted bag-of-words into vector representation h. Then we denote the topic of the conversation as t and denote f as the conversation floor, i.e. if the speakers of the last utterance in the dialog history and response utterance are the same $f = 1$ otherwise $f = 0$. Then we computed the similarity $d(\mathbf{c}_i, \mathbf{c}_j)$ between two dialog contexts using:

$$d(\mathbf{c}_i, \mathbf{c}_j) = \mathbb{1}(t_i = t_j)\mathbb{1}(t_i = t_j)\frac{h_i \cdot h_j}{||h_i||||h_j||} \tag{5.6}$$

Unlike past work (Sordoni et al., 2015), this similarity function only cares about the distance in the context and imposes no constraints on the response, therefore it is suitable for finding diverse responses regarding the same dialog context. Second, for each dialog context in the test set, we retrieved the 10 nearest neighbors from the training set and treated the responses from the training set as candidate reference responses. Third, we further sampled 240 context-responses pairs from 5481 pairs in the total test set and post-processed the selected candidate responses by two human computational linguistic experts who were told to give a binary label for each candidate response about whether the response is appropriate regarding its dialog context. The filtered lists then served as the ground truth to train our reference response classifier. For the next step, we extracted bigrams, part-of-speech bigrams and word part-of-speech pairs from both dialogue contexts and candidate reference responses with rare threshold for feature extraction being set to 20. Then L2-regularized logistic regression with 10-fold cross validation was applied as the machine learning algorithm. Cross validation accuracy on the human-labelled data was 71%. Finally, we automatically annotated the rest of the test set with this trained classifier and the resulting data were used for model evaluation.

5.3.3 Training Details

We trained with the following hyperparameters (according to the loss on the validate dataset): word embedding has size 200 and is shared everywhere. We initialize the

word embedding from Glove embeddings pre-trained on Twitter (Pennington et al., 2014). The utterance encoder has a hidden size of 300 for each direction. The context encoder has a hidden size of 600 and the response decoder has a hidden size of 400. The prior network and the MLP for predicting $\mathbf{y}$ both have 1 hidden layer of size 400 and $tanh$ non-linearity. The latent variable $\mathbf{z}$ has a size of 200. The context window k is 10. All the initial weights are sampled from a uniform distribution [-0.08, 0.08]. The mini-batch size is 30. The models are trained end-to-end using the Adam optimizer (Kingma and Ba, 2014) with a learning rate of 0.001 and gradient clipping at 5. We selected the best models based on the variational lower bound on the validate data. Finally, we use the BOW loss along with *KL annealing* (defined in Chapter 4) of 10,000 batches to achieve the best performance.

5.4 Results

5.4.1 Baseline Model

We compared three neural dialog models: a strong baseline model, CVAE, and kgCVAE.

The baseline model is an encoder-decoder neural dialog model without latent variables similar to (Serban et al., 2016b). The baseline model's encoder uses the same context encoder to encode the dialog history and the meta features as shown in Figure 7.2. The encoded context $\mathbf{c}$ is directly fed into the decoder networks as the initial state. The hyperparameters of the baseline are the same as the ones reported in Section 5.3.3 and the baseline is trained to minimize the standard cross entropy loss of the decoder RNN model without any auxiliary loss.

Also, to compare the diversity introduced by the stochasticity in the proposed latent variable versus the softmax of RNN at each decoding step, we generate N responses from the baseline by sampling from the softmax. For CVAE/kgCVAE, we sample N times from the latent $\mathbf{z}$ and only use greedy decoders so that the randomness comes entirely from the latent variable $\mathbf{z}$.

5.4.2 Quantitative Analysis

Following the information retrieval method described in Section 5.3.2, we gather 10 extra candidate reference responses/context from other conversations with the same topics. Then the 10 candidate references are filtered by two experts, which serve as the ground truth to train the reference response classifier. The result is 6.69 extra references in average per context. The average number of distinct reference dialog acts is 4.2. Table 5.1 shows the results.

The proposed models outperform the baseline in terms of recall in all the metrics with statistical significance. This confirms our hypothesis that generating responses with discourse-level diversity can lead to a more comprehensive coverage of the potential responses than promoting only word-level diversity. As for precision, we observed that the baseline has higher or similar scores compared to CVAE in all metrics, which

Metrics	Baseline	CVAE	kgCVAE
perplexity (KL)	35.4 (n/a)	20.2 (11.36)	16.02 (13.08)
BLEU-1 prec	0.405	0.372	**0.412**
BLEU-1 recall	0.336	0.381	**0.411**
BLEU-2 prec	0.300	0.295	**0.350**
BLEU-2 recall	0.281	0.322	**0.356**
BLEU-3 prec	0.272	0.265	**0.310**
BLEU-3 recall	0.254	0.292	**0.318**
BLEU-4 prec	0.226	0.223	**0.262**
BLEU-4 recall	0.215	0.248	**0.272**
A-bow prec	0.951	0.954	**0.961**
A-bow recall	0.935	0.943	**0.944**
E-bow prec	**0.827**	0.815	0.804
E-bow recall	0.801	**0.812**	0.807
DA prec	**0.736**	0.704	0.721
DA recall	0.514	**0.604**	0.598

Table 5.1: Performance of each model on automatic measures. The highest score in each row is in bold. Note that our BLEU scores are normalized to $[0, 1]$. A-bow/E-bow mean average/extreme bag-of-words word embedding distance. DA stands for dialog acts. Bold-face numbers indicate significantly better results compared to the baseline system with p-value < 0.01.

is expected since the baseline tends to generate the mostly likely and safe responses repeatedly in the N hypotheses. However, kgCVAE is able to achieve the highest precision and recall in the 4 metrics at the same time (BLEU1-4, A-BOW). One reason for kgCVAE's good performance is that the predicted dialog act label in kgCVAE can regularize the generation process of its RNN decoder by forcing it to generate more coherent and precise words. We further analyze the precision/recall of BLEU-4 by looking at the average score versus the number of distinct reference dialog acts. A low number of distinct dialog acts represents the situation where the dialog context has a strong constraint on the range of the next response (low entropy), while a high number indicates the opposite (high-entropy). Figure 5.4 shows that CVAE/kgCVAE achieves significantly higher recall than the baseline in higher entropy contexts. Also it shows that CVAE suffers from lower precision, especially in low entropy contexts. Finally, kgCVAE gets higher precision than both the baseline and CVAE in the full spectrum of context entropy.

5.4.3 Qualitative Analysis

Table 5.2 shows the outputs generated from the baseline and kgCVAE. In example 1, caller A begins with an open-ended question. The kgCVAE model generated highly diverse answers that cover multiple plausible dialog acts. Further, we notice that the

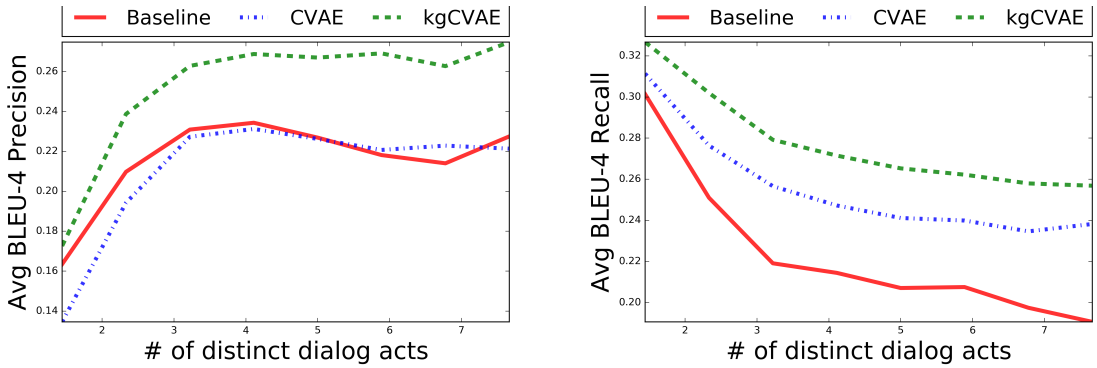

Figure 5.4: BLEU-4 precision/recall vs. the number of distinct reference dialog acts.

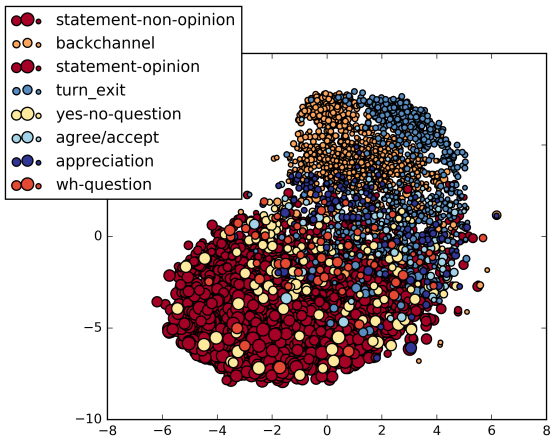

Figure 5.5: t-SNE visualization of the posterior $\mathbf{z}$ for test responses with top 8 frequent dialog acts. The size of circle represents the response length.

generated text exhibits similar dialog acts compared to the ones predicted separately by the model, implying the consistency of natural language generation based on $\mathbf{y}$. On the contrary, the responses from the baseline model are limited to local n-gram variations and share a similar prefix, i.e. "I'm". Example 2 is a situation where caller A is telling B stories. The ground truth response is a back-channel and the range of valid answers is more constrained than example 1 since B is playing the role of a listener. The baseline successfully predicts "uh-huh". The kgCVAE model is also able to generate various ways of back-channeling. This implies that the latent $\mathbf{z}$ is able to capture context-sensitive variations, i.e. in low-entropy dialog contexts modeling lexical diversity while in high-entropy ones modeling discourse-level diversity. Moreover, kgCVAE is occasionally able to generate more sophisticated grounding (sample 4) beyond a simple back-channel, which is also an acceptable response given the dialog context.

In addition, past work (Kingma and Welling, 2013) has shown that the recognition network is able to learn to cluster high-dimension data, so we conjecture that posterior $\mathbf{z}$ outputted from the recognition network should cluster the responses into meaningful

Example 1-Topic: Recycling **Context**: A: are they doing a lot of recycling out in Georgia? **Target-B** (statement): well at my workplace we have places for aluminium cans and we have a separate trash can for recyclable paper	
Baseline+Sampling	**kgCVAE+Greedy**
1. well I'm a graduate student and have two kids	1. (non-understand) pardon
2. well I was in last year and so we've had lots of recycling	2. (statement) oh you're not going to have a curbside pick up here
3. I'm not sure	3. (statement) okay I am sure about a recycling center
4. well I don't know I just moved here in new york	4. (yes-answer) yeah so

Example 2-Topic: Child Care **Context**: A: you know a private home to take their children to when they're young until they hit the preschool age and they **Target-B** (backchannel): uh-huh	
Baseline+Sampling	**kgCVAE+Greedy**
1. um - hum	1. (backchannel) uh-huh
2. yeah	2. (turn-exit) um-hum
3. um - hum	3. (backchannel) yeah
4. uh-huh	4. (statement) oh yeah I think that's part of the problem

Table 5.2: Generated responses from the baselines and kgCVAE in two examples. KgCVAE also provides the predicted dialog act for each response. The context only shows the last utterance due to space limit (the actual context window size is 10).

groups. Figure 5.5 visualizes the posterior $\mathbf{z}$ of responses in the test dataset in 2D space using t-SNE (Maaten and Hinton, 2008). We found that the learned latent space is highly correlated with the dialog act and length of responses, which confirms our assumption.

5.5 Conclusion

In conclusion, we identified the *one-to-many* nature of open-domain conversation and proposed two novel models that show superior performance in generating diverse and appropriate responses at the discourse level, which validates the effectiveness of latent actions. The current chapter addresses diversifying responses in respect to dialogue acts. This work is part of a larger research direction that targets leveraging both past linguistic findings and the learning power of deep neural networks to learn better representation of the latent factors in dialog. In turn, the output of this novel neural dialog model will be easier to explain and control by humans. In addition to dialog acts, we plan to apply our kgCVAE model to capture other different linguistic phenomena including sentiment, named entities,etc. Last but not least, this chapter shows that the

recognition network in our model can serve as the foundation for designing a data-driven dialog manager, which automatically discovers useful high-level intents. All of the above suggest that our latent action framework provides unique properties that previous state-of-the-art E2E dialog models that are not capable of.

Chapter 6

Discrete Latent Action for Model Interpretability

Like other deep learning models, an E2E generation-based dialog system cannot be easily interpreted by human developers. The primary reason is its intermediate representations are continuous hidden vectors that are automatically learned from data, which are indecipherable from human perspectives. Paradoxically, this property is at the same time the key feature that makes deep learning models powerful and scalable. The chapter presents a new type of latent action that are discrete random variables, which are possible for human to assign meanings. The proposed latent action can be learned from data in an unsupervised fashion and seamlessly integrated into a normal E2E generation-based dialog systems. Therefore, the resulting framework enjoys both the scalability of neural dialog system, while offers a window for human developers to peek into models' decision making mechanism (via the discrete latent actions). [1]

6.1 Towards Interpretable Generation

Classic dialog systems rely on developing a meaning representation to represent the utterances from both the machine and human users (Larsson and Traum, 2000; Bohus et al., 2007). The dialog manager of a conventional dialog system outputs the system's next action in a semantic frame that usually contains hand-crafted dialog acts and slot values (Williams and Young, 2007). Then a natural language generation module is used to generate the system's output in natural language based on the given semantic frame. This approach suffers from generalization to more complex domains because it soon become intractable to manually design a frame representation that covers all of the fine-grained system actions. The recently developed neural dialog system is one of the most prominent frameworks for developing dialog agents in complex domains. The basic model is based on encoder-decoder networks (Cho et al., 2014) and can learn to generate system responses without the need for hand-crafted meaning representations and other

[1] The code and data are available at `https://github.com/snakeztc/NeuralDialog-LAED`.

annotations.

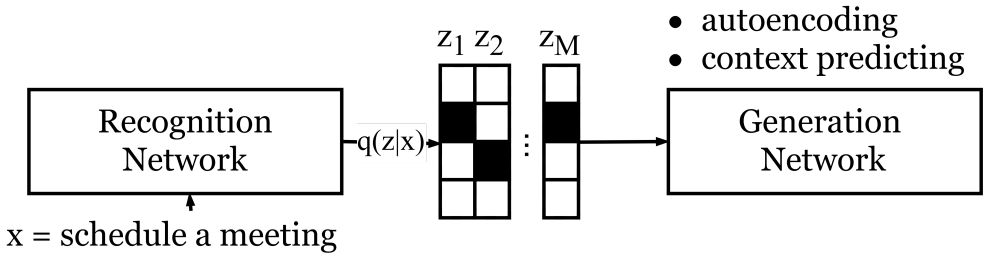

Figure 6.1: Our proposed models learn a set of discrete variables to represent sentences by either autoencoding or context prediction.

Although generative dialog models have advanced rapidly (Serban et al., 2016c; Li et al., 2016a; Zhao et al., 2017b), they cannot provide interpretable system actions as in the conventional dialog systems. This inability limits the effectiveness of generative dialog models in several ways. First, having interpretable system actions enables human to understand the behavior of a dialog system and better interpret the system intentions. Also, modeling the high-level decision-making policy in dialogs enables useful generalization and data-efficient domain adaptation (Gašić et al., 2010). Therefore, the motivation of this chapter is to develop an unsupervised neural recognition model that can discover interpretable meaning representations of utterances (denoted as *latent actions*) as a set of discrete latent variables from a large unlabelled corpus as shown in Figure 6.1. The discovered meaning representations will then be integrated with encoder decoder networks to achieve interpretable dialog generation while preserving all the merit of neural dialog systems.

We focus on learning discrete latent representations instead of dense continuous ones because discrete variables are easier to interpret (van den Oord et al., 2017) and can naturally correspond to categories in natural languages, e.g. topics, dialog acts etc. Despite the difficulty of learning discrete latent variables in neural networks, the recently proposed Gumbel-Softmax offers a reliable way to back-propagate through discrete variables (Maddison et al., 2016; Jang et al., 2016). However, we found a simple combination of sentence variational autoencoders (VAEs) (Bowman et al., 2015) and Gumbel-Softmax fails to learn meaningful discrete representations. We then highlight the anti-information limitation of the evidence lowerbound objective (ELBO) in VAEs and improve it by proposing Discrete Information VAE (DI-VAE) that maximizes the mutual information between data and latent actions. We further enrich the learning signals beyond auto encoding by extending Skip Thought (Kiros et al., 2015) to Discrete Information Variational Skip Thought (DI-VST) that learns sentence-level distributional semantics. Finally, an integration mechanism is presented that combines the learned latent actions with encoder decoder models.

The proposed systems are tested on several real-world dialog datasets. Experiments show that the proposed methods significantly outperform the standard VAEs and can discover meaningful latent actions from these datasets. Also, experiments confirm the

effectiveness of the proposed integration mechanism and show that the learned latent actions can control the sentence-level attributes of the generated responses and provide human-interpretable meaning representations.

6.2 Related Work

Our work is closely related to research in latent variable dialog models. The majority of models are based on Conditional Variational Autoencoders (CVAEs) (Serban et al., 2016c; Cao and Clark, 2017) with continuous latent variables to better model the response distribution and encourage diverse responses. Zhao et al., (2017b) further introduced dialog acts to guide the learning of the CVAEs. Discrete latent variables have also been used for task-oriented dialog systems (Wen et al., 2017), where the latent space is used to represent intention. The second line of related work is enriching the dialog context encoder with more fine-grained information than the dialog history. Li et al., (2016a) captured speakers' characteristics by encoding background information and speaking style into the distributed embeddings. Xing et al., (2016) maintain topic encoding based on Latent Dirichlet Allocation (LDA) (Blei et al., 2003) of the conversation to encourage the model to output more topic coherent responses.

The proposed method also relates to sentence representation learning using neural networks. Most work learns continuous distributed representations of sentences from various learning signals (Hill et al., 2016), e.g. the Skip Thought learns representations by predicting the previous and next sentences (Kiros et al., 2015). Another area of work focused on learning regularized continuous sentence representation, which enables sentence generation by sampling the latent space (Bowman et al., 2015; Kim et al., 2017). There is less work on discrete sentence representations due to the difficulty of passing gradients through discrete outputs. The recently developed Gumbel Softmax (Jang et al., 2016; Maddison et al., 2016) and vector quantization (van den Oord et al., 2017) enable us to train discrete variables. Notably, discrete variable models have been proposed to discover document topics (Miao et al., 2016) and semi-supervised sequence transaction (Zhou and Neubig, 2017)

Our work differs from these as follows: (1) we focus on learning interpretable variables; in prior research the semantics of latent variables are mostly ignored in the dialog generation setting. (2) we improve the learning objective for discrete VAEs and overcome the well-known posterior collapsing issue (Bowman et al., 2015; Chen et al., 2016a). (3) we focus on unsupervised learning of salient features in dialog responses instead of hand-crafted features.

6.3 Proposed Methods

Our formulation contains three random variables: the dialog context $\mathbf{c}$, the response $\mathbf{x}$ and the latent action $\mathbf{z}$. The context often contains the discourse history in the format of a list of utterances. The response is an utterance that contains a list of word tokens.

The latent action is a set of discrete variables that define high-level attributes of $\mathbf{x}$. Before introducing the proposed framework, we first identify two key properties that are essential in order for latent actions $\mathbf{z}$ to be *interpretable*:

1. $\mathbf{z}$ should capture salient sentence-level features about the response $\mathbf{x}$.
2. The meaning of latent symbols $\mathbf{z}$ should be independent of the context $\mathbf{c}$.

The first property is self-evident. The second can be explained: assume $\mathbf{z}$ contains a single discrete variable with K classes. Since the context $\mathbf{c}$ can be any dialog history, if the meaning of each class changes given a different context, then it is difficult to extract an intuitive interpretation by only looking at all responses with class $k \in [1, K]$. Therefore, the second property looks for latent actions that have *context-independent semantics* so that each assignment of $\mathbf{z}$ conveys the same meaning in all dialog contexts.

With the above definition of interpretable latent actions, the overall structure is illustrated in Figure 6.2. We introduce:

- A recognition network $R : q_\phi(\mathbf{z}|\mathbf{x})$ and a generation network $G : p_\phi$. The role of R is to map a sentence to the latent variable $\mathbf{z}$ and the generator G defines the learning signals that will be used to train $\mathbf{z}$'s representation. Notably, our recognition network R does not depend on the context $\mathbf{c}$ as has been the case in prior work (Serban et al., 2016c). The motivation of this design is to encourage $\mathbf{z}$ to capture context-independent semantics, which are further elaborated in Section 6.3.4.
- With the $\mathbf{z}$ learned by R and G, we then introduce an encoder decoder network $F : p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{c})$ and and a policy network $\pi : p_\pi(\mathbf{z}|\mathbf{c})$. At test time, given a context $\mathbf{c}$, the policy network and encoder decoder will work together to generate the next response via $\tilde{\mathbf{x}} = p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{c}) \quad \mathbf{z} \sim p_\pi(\mathbf{z}|\mathbf{c})$.

In short, R, G, F and π are the four components that comprise our proposed framework. The next section will first focus on developing R and G for learning interpretable $\mathbf{z}$ and then will move on to integrating R with F and π in Section 6.3.3.

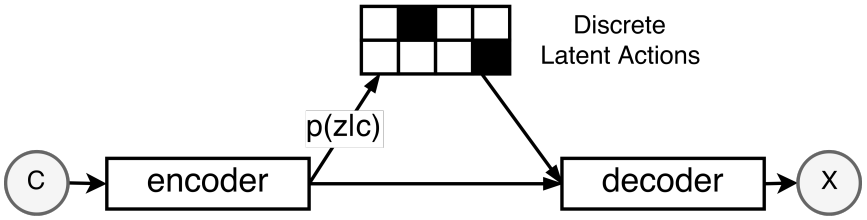

Figure 6.2: The network architecture for integrating the latent action into an encoder decoder model. Essentially it falls into a type of partial latent action, with the difference that the meaning of $\mathbf{z}$ is learned separately as a 2-step process.

6.3.1 Learning Sentence Representations from Auto-Encoding

Our baseline model is a sentence VAE with discrete latent space. We use an RNN as the recognition network to encode the response $\mathbf{x}$. Its last hidden state $h^R_{|\mathbf{x}|}$ is used to

represent $\mathbf{x}$. We define $\mathbf{z}$ to be a set of K-way categorical variables $\mathbf{z} = \{\mathbf{z}_1 ... \mathbf{z}_m ... \mathbf{z}_M\}$, where M is the number of variables. For each $\mathbf{z}_m$, its posterior distribution is defined as $q_{q_\phi}(\mathbf{z}_m|\mathbf{x}) = \text{Softmax}(W_q h^R_{|\mathbf{x}|} + b_q)$. During training, we use the Gumbel-Softmax Trick to sample from this distribution and obtain low-variance gradients. To map the latent samples to the initial state of the decoder RNN, we define $\{e_1 ... e_m ... e_M\}$ where $e_m \in \mathbb{R}^{K \times D}$ and D is the generator cell size. Thus the initial state of the generator is: $h^G_0 = \sum_{m=1}^{M} e_m(\mathbf{z}_m)$. Finally, the generator RNN is used to reconstruct the response given h^G_0. VAEs is trained to maximize the evidence lowerbound objective (ELBO) (Kingma and Welling, 2013). For simplicity, later discussion drops the subscript m in $\mathbf{z}_m$ and assumes a single latent $\mathbf{z}$. Since each $\mathbf{z}_m$ is independent, we can easily extend the results below to multiple variables. Our proposed auto-encoding model is the discrete BPR system described in Chapter 4. For later discussion, we denote our discrete infoVAE with BPR as DI-VAE.

6.3.2 Learning Sentence Representations from the Context

DI-VAE infers sentence representations by reconstruction of the input sentence. Past research in distributional semantics has suggested the meaning of language can be inferred from the adjacent context (Harris, 1954; Hill et al., 2016). The distributional hypothesis is especially applicable to dialog since the utterance meaning is highly contextual. For example, the dialog act is a well-known utterance feature and depends on dialog state (Austin, 1975; Stolcke et al., 2000). Thus, we introduce a second type of latent action based on sentence-level distributional semantics.

Skip thought (ST) is a powerful sentence representation that captures contextual information (Kiros et al., 2015). ST uses an RNN to encode a sentence, and then uses the resulting sentence representation to predict the previous and next sentences. Inspired by ST's robust performance across multiple tasks (Hill et al., 2016), we adapt our DI-VAE to Discrete Information Variational Skip Thought (DI-VST) to learn discrete latent actions that model distributional semantics of sentences. We use the same recognition network from DI-VAE to output $\mathbf{z}$'s posterior distribution $q_\phi(\mathbf{z}|\mathbf{x})$. Given the samples from $q_\phi(\mathbf{z}|\mathbf{x})$, two RNN generators are used to predict the previous sentence $\mathbf{x}_p$ and the next sentences $\mathbf{x}_n$. Finally, the learning objective is to maximize:

$$\mathcal{L}_{\text{DI-VST}} = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x}))}[\log(p_\phi(\mathbf{x}_n|\mathbf{z})p_\phi(\mathbf{x}_p|\mathbf{z}))] - \mathrm{D}_{\mathrm{KL}}(q_\phi(\mathbf{z})\|p(\mathbf{z})) \tag{6.1}$$

6.3.3 Integration with Encoder Decoders

We now describe how to integrate a given $q_\phi(\mathbf{z}|\mathbf{x})$ with an encoder decoder and a policy network. Let the dialog context $\mathbf{c}$ be a sequence of utterances. Then a dialog context encoder network can encode the dialog context into a distributed representation $h^e = F^e(\mathbf{c})$. The decoder F^d can generate the responses $\tilde{\mathbf{x}} = F^d(h^e, \mathbf{z})$ using samples from $q_\phi(\mathbf{z}|\mathbf{x})$. Meanwhile, we train π to predict the aggregated posterior $\mathbb{E}_{p(\mathbf{x}|\mathbf{c})}[q_\phi(\mathbf{z}|\mathbf{x})]$ from $\mathbf{c}$ via maximum likelihood training. This model is referred as Latent Action Encoder

Decoder (LAED) with the following objective.

$$\mathcal{L}_{\text{LAED}}(\theta, \pi; \mathbf{x}, \mathbf{z}, \mathbf{c}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})p(\mathbf{x},\mathbf{c})}[\log p_\pi(\mathbf{z}|\mathbf{c}) + \log p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{c})] \tag{6.2}$$

Also simply augmenting the inputs of the decoders with latent action does not guarantee that the generated response exhibits the attributes of the given action. Thus we use the controllable text generation framework (Hu et al., 2017) by introducing $\mathcal{L}_{\text{Attr}}$, which reuses the same recognition network $q_{q_\phi}(\mathbf{z}|\mathbf{x})$ as a fixed discriminator to penalize the decoder if its generated responses do not reflect the attributes in $\mathbf{z}$.

$$\mathcal{L}_{\text{Attr}}(\theta) = \mathbb{E}_{\mathbf{z}\sim q_\phi(\mathbf{z}|\mathbf{x})}[\log q_\phi(\mathbf{z}|\tilde{\mathbf{x}})] \tag{6.3}$$

Since it is not possible to propagate gradients through the discrete outputs at F^d at each word step, we use a deterministic continuous relaxation (Hu et al., 2017) by replacing output of F^d with the probability of each word. Let o_t be the normalized probability at step $t \in [1, |\mathbf{x}|]$, the inputs to q_{q_ϕ} at time t are then the sum of word embeddings weighted by o_t, i.e. $h_t^R = \text{RNN}(h_{t-1}^R, \mathbf{E}o_t)$ and $\mathbf{E}$ is the word embedding matrix. Finally this loss is combined with $\mathcal{L}_{\text{LAED}}$ and a hyperparameter λ to have Attribute Forcing LAED.

$$\mathcal{L}_{\text{attrLAED}} = \mathcal{L}_{\text{LAED}} + \lambda\mathcal{L}_{\text{Attr}} \tag{6.4}$$

6.3.4 Relationship with Conditional VAEs

It is not hard to see $\mathcal{L}_{\text{LAED}}$ is closely related to the objective of CVAEs for dialog generation (Serban et al., 2016c; Zhao et al., 2017b), which is:

$$\mathcal{L}_{\text{CVAE}} = \mathbb{E}_q[\log p(\mathbf{x}|\mathbf{z}, \mathbf{c})] - \text{D}_{\text{KL}}(q(\mathbf{z}|\mathbf{x}, \mathbf{c})\|p(\mathbf{z}|\mathbf{c})) \tag{6.5}$$

Despite their similarities, we highlight the key differences that prohibit CVAE from achieving interpretable dialog generation. First $\mathcal{L}_{\text{CVAE}}$ encourages $I(\mathbf{x}, \mathbf{z}|\mathbf{c})$ (Agakov, 2005), which learns $\mathbf{z}$ that capture context-dependent semantics. More intuitively, $\mathbf{z}$ in CVAE is trained to generate $\mathbf{x}$ via $p(\mathbf{x}|\mathbf{z}, \mathbf{c})$ so the meaning of a learned $\mathbf{z}$ can only be interpreted along with its context $\mathbf{c}$. Therefore this violates our goal of learning context-independent semantics. Our methods learn $q_{q_\phi}(\mathbf{z}|\mathbf{x})$ that only depends on $\mathbf{x}$ and trains q_{q_ϕ} separately to ensure the semantics of $\mathbf{z}$ are interpretable standalone.

6.4 Experiments and Results

The proposed methods are evaluated on four datasets. The first corpus is Penn Treebank (PTB) (Marcus et al., 1993) used to evaluate sentence VAEs (Bowman et al., 2015). We used the version pre-processed by Mikolov (Mikolov et al., 2010). The second dataset is the Stanford Multi-Domain Dialog (SMD) dataset that contains 3,031 human-Woz, task-oriented dialogs collected from 3 different domains (navigation, weather and scheduling) (Eric and Manning, 2017b). The other two datasets are chat-oriented data: Daily

Dialog (DD) and Switchboard (SW) (Godfrey and Holliman, 1997), which are used to test whether our methods can generalize beyond task-oriented dialogs but also to to open-domain chatting. DD contains 13,118 multi-turn human-human dialogs annotated with dialog acts and emotions. (Li et al., 2017). SW has 2,400 human-human telephone conversations that are annotated with topics and dialog acts. SW is a more challenging dataset because it is transcribed from speech which contains complex spoken language phenomenon, e.g. hesitation, self-repair etc.

6.4.1 Comparing Discrete Sentence Representation Models

The first experiment used PTB and DD to evaluate the performance of the proposed methods in learning discrete sentence representations. We implemented DI-VAE and DI-VST using GRU-RNN (Chung et al., 2014) and trained them using Adam (Kingma and Ba, 2014). Besides the proposed methods, the following baselines are compared. **Unregularized models**: removing the KL$(q|p)$ term from DI-VAE and DI-VST leads to a simple discrete autoencoder (DAE) and discrete skip thought (DST) with stochastic discrete hidden units. **ELBO models**: the basic discrete sentence VAE (DVAE) or variational skip thought (DVST) that optimizes ELBO with regularization term KL$(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))$. We found that standard training failed to learn informative latent actions for either DVAE or DVST because of the posterior collapse. Therefore, KL-annealing (Bowman et al., 2015) and bag-of-word loss (Zhao et al., 2017b) are used to force these two models learn meaningful representations. We also include the results for VAE with continuous latent variables reported on the same PTB (Zhao et al., 2017b). Additionally, we report the perplexity from a standard GRU-RNN language model (Zaremba et al., 2014).

The evaluation metrics include reconstruction perplexity (PPL), KL$(q(\mathbf{z})\|p(\mathbf{z}))$ and the mutual information between input data and latent variables $I(\mathbf{x}, \mathbf{z})$. Intuitively a good model should achieve low perplexity and KL distance, and simultaneously achieve high $I(\mathbf{x}, \mathbf{z})$. The discrete latent space for all models are M=20 and K=10. Mini-batch size is 30.

Dom	Model	PPL	KL$(q\|\|p)$	$I(\mathbf{x}, \mathbf{z})$
PTB	RNNLM	116.22	-	-
	VAE	73.49	15.94*	-
	DAE	66.49	2.20	0.349
	DVAE	70.84	0.315	0.286
	DI-VAE	**52.53**	**0.133**	**1.18**
DD	RNNLM	31.15	-	-
	DST	$\mathbf{x}_p$:28.23 $\mathbf{x}_n$:28.16	0.588	**1.359**
	DVST	$\mathbf{x}_p$:30.36 $\mathbf{x}_n$:30.71	**0.007**	0.081
	DI-VST	$\mathbf{x}_p$:**28.04** $\mathbf{x}_n$:**27.94**	0.088	1.028

Table 6.1: Results for various discrete sentence representations. The KL for VAE is KL$(q(\mathbf{z}|\mathbf{x})\|p(\mathbf{z}))$ instead of KL$(q(\mathbf{z})\|p(\mathbf{z}))$ (Zhao et al., 2017b). $\mathbf{x}_p$ and $\mathbf{x}_n$ are the perplexity for predicting the previous and next utterances.

Table 6.1 shows that all models achieve better perplexity than an RNNLM, which shows they manage to learn meaningful $q(\mathbf{z}|\mathbf{x})$. First, for auto-encoding models, DI-VAE is able to achieve the best results in all metrics compared other methods. We found DAEs quickly learn to reconstruct the input but they are prone to overfitting during training, which leads to lower performance on the test data compared to DI-VAE. Also, since there is no regularization term in the latent space, $q(\mathbf{z})$ is very different from the $p(\mathbf{z})$ which prohibits us from generating sentences from the latent space. As for DVAEs, it achieves zero $I(\mathbf{x}, \mathbf{z})$ in standard training and only manages to learn some information when training with KL-annealing and bag-of-word loss. On the other hand, our methods achieve robust performance without the need for additional processing. Similarly, the proposed DI-VST is able to achieve the lowest PPL and similar KL compared to the strongly regularized DVST. Interestingly, although DST is able to achieve the highest $I(\mathbf{x}, \mathbf{z})$, but PPL is not further improved. These results confirm the effectiveness of the proposed BPR in terms of regularizing $q(\mathbf{z})$ while learning meaningful posterior $q(\mathbf{z}|\mathbf{x})$.

In order to understand BPR's sensitivity to batch size N, a follow-up experiment varied the batch size from 2 to 60 (If N=1, DI-VAE is equivalent to DVAE). Figure 6.3

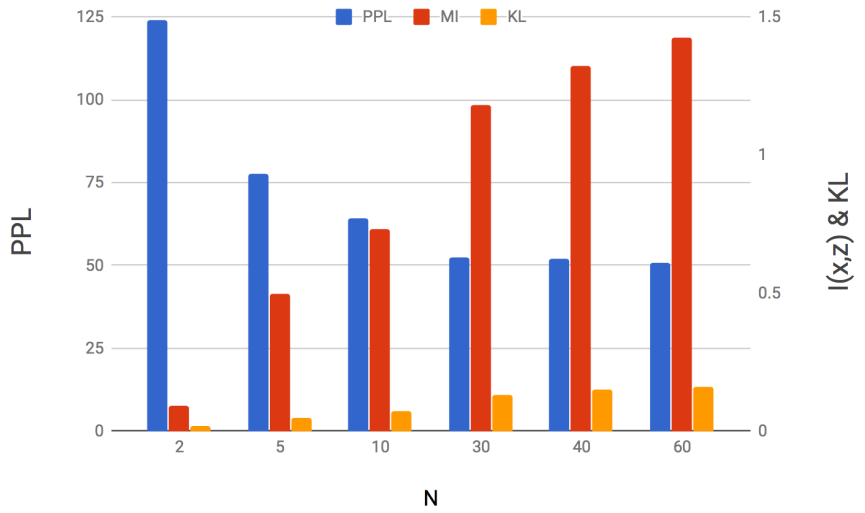

Figure 6.3: Perplexity and $I(\mathbf{x}, \mathbf{z})$ on PTB by varying batch size N. BPR works better for larger N.

show that as N increases, perplexity, $I(\mathbf{x}, \mathbf{z})$ monotonically improves, while KL$(q\|p)$ only increases from 0 to 0.159. After $N > 30$, the performance plateaus. Therefore, using mini-batch is an efficient trade-off between $q(\mathbf{z})$ estimation and computation speed.

The last experiment in this section investigates the relation between representation learning and the dimension of the latent space. We set a fixed budget by restricting the maximum number of modes to be about 1000, i.e. $K^M \approx 1000$ (Note again K is the size of each categorical latent dimension and M is the number of dimensions). We then vary the latent space size and report the same evaluation metrics. Table 6.2 shows that models with multiple small latent variables perform significantly better than those with

large and few latent variables.

K, M	K^M	PPL	KL$(q\|\|p)$	$I(\mathbf{x}, \mathbf{z})$
1000, 1	1000	75.61	0.032	0.335
10, 3	1000	71.42	0.071	0.607
4, 5	1024	68.43	0.088	0.809

Table 6.2: DI-VAE on PTB with different latent dimensions under the same budget.

6.4.2 Interpreting Latent Actions

The next question is to interpret the meaning of the learned latent action symbols. To achieve this, the latent action of an utterance $\mathbf{x}_n$ is obtained from a greedy mapping: $a_n = \underset{k}{\text{argmax}}\, q_{q_\phi}(\mathbf{z} = k|\mathbf{x}_n)$. We set M=3 and K=5, so that there are at most 125 different latent actions, and each $\mathbf{x}_n$ can now be represented by a_1-a_2-a_3, e.g. "How are you?" $\rightarrow$ 1-4-2. Assuming that we have access to manually clustered data according to certain classes (e.g. dialog acts), it is unfair to use classic cluster measures (Vinh et al., 2010) to evaluate the clusters from latent actions. This is because the uniform prior $p(\mathbf{z})$ evenly distributes the data to all possible latent actions, so that it is expected that frequent classes will be assigned to several latent actions. Thus we utilize the *homogeneity* metric (Rosenberg and Hirschberg, 2007). B y definition, homogeneity measures if each latent action contains only members of a single class. We tested this on SW and DD, which contain human annotated features and we report the latent actions' homogeneity w.r.t these features in Table 6.3.

	SW		**DD**	
	Act	Topic	Act	Emotion
DI-VAE	0.48	0.08	0.18	0.09
DI-VST	0.33	0.13	0.34	0.12

Table 6.3: Homogeneity results (bounded [0, 1]).

On DD, results show DI-VST works better than DI-VAE in terms of creating actions that are more coherent for emotion and dialog acts. The results are interesting on SW since DI-VST performs worse on dialog acts than DI-VAE. One reason is that the dialog acts in SW are more fine-grained (42 acts) than the ones in DD (5 acts) so that distinguishing utterances based on words in $\mathbf{x}$ is more important than the information in the neighbouring utterances.

We then apply the proposed methods to SMD which has no manual annotation and contains task-oriented dialogs. Two experts are shown 5 randomly selected utterances from each latent action and are asked to give an action name that can describe as many of the utterances as possible. Then an Amazon Mechanical Turk study is conducted to evaluate whether other utterances from the same latent action match these titles.

5 workers see the action name and a different group of 5 utterances from that latent action. They are asked to select all utterances that belong to the given actions, which tests the homogeneity of the utterances falling in the same cluster. Negative samples are included to prevent random selection. Table 6.4 shows that both methods work well and DI-VST achieved better homogeneity than DI-VAE.

Model	Expert Agree	Worker κ	Match Rate
DI-VAE	85.6%	0.52	71.3%
DI-VST	93.3%	0.48	74.9%

Table 6.4: Human evaluation results on judging the homogeneity of latent actions in SMD.

Since DI-VAE is trained to reconstruct its input and DI-VST is trained to model the context, they group utterances in different ways. For example, DI-VST would group "Can I get a restaurant", "I am looking for a restaurant" into one action where DI-VAE may denote two actions for them. Finally, Table 6.4.2 shows sample annotation results, which show cases of the different types of latent actions discovered by our models.

Model	**Action**	**Sample utterance**
DI-VAE	scheduling	- sys: okay, scheduling a yoga activity with Tom for the 8th at 2pm. - sys: okay, scheduling a meeting for 6 pm on Tuesday with your boss to go over the quarterly report.
	requests	- usr: find out if it 's supposed to rain - usr: find nearest coffee shop
DI-VST	ask schedule info	- usr: when is my football activity and who is going with me? - usr: tell me when my dentist appointment is?
	requests	- usr: how about other coffee? - usr: 11 am please

Table 6.5: Example latent actions discovered in SMD using our methods.

6.4.3 Dialog Response Generation with Latent Actions

Finally we implement an LAED as follows. The encoder is a hierarchical recurrent encoder (Serban et al., 2016c) with bi-directional GRU-RNNs as the utterance encoder and a second GRU-RNN as the discourse encoder. The discourse encoder output its last hidden state $h^e_{|\mathbf{x}|}$. The decoder is another GRU-RNN and its initial state of the decoder is obtained by $h^d_0 = h^e_{|\mathbf{x}|} + \sum_{m=1}^{M} e_m(\mathbf{z}_m)$, where $\mathbf{z}$ comes from the recognition network of the proposed methods. The policy network π is a 2-layer multi-layer perceptron (MLP) that models $p_\pi(\mathbf{z}|h^e_{|\mathbf{x}|})$. We use up to the previous 10 utterances as the dialog context

and denote the LAED using DI-VAE latent actions as AE-ED and the one uses DI-VST as ST-ED.

First we need to confirm whether an LAED can generate responses that are consistent with the semantics of a given $\mathbf{z}$. To answer this, we use a pre-trained recognition network R to check if a generated response carries the attributes in the given action. We generate dialog responses on a test dataset via $\tilde{\mathbf{x}} = F(\mathbf{z} \sim \pi(\mathbf{c}), \mathbf{c})$ with greedy RNN decoding. The generated responses are passed into R and we measure *attribute consistency rate* by counting $\tilde{\mathbf{x}}$ as correct if $\mathbf{z} = \underset{k}{\operatorname{argmax}}\, q_{q_\phi}(k|\tilde{\mathbf{x}})$.

Domain	AE-ED	+L_{attr}	ST-ED	+L_{attr}
SMD	93.5%	**94.8%**	91.9%	**93.8%**
DD	88.4%	**93.6%**	78.5%	**86.1%**
SW	84.7%	**94.6%**	57.3%	**61.3%**

Table 6.6: Results for attribute consistency rate with and without attribute loss. P-value < 0.01 using Mcnemar's test compared to models w/o L_{attr}.

Table 6.6 shows our generated responses are highly consistent with the given latent actions. Also, latent actions from DI-VAE achieve higher attribute consistency rate than the ones from DI-VST, because $\mathbf{z}$ from auto-encoding is explicitly trained for $\mathbf{x}$ reconstruction. Adding $\mathcal{L}_{attr}$ is effective in forcing the decoder to take $\mathbf{z}$ into account during its generation, which helps the most in more challenging open-domain chatting data, e.g. SW and DD. The consistency rate of ST-ED on SW is worse than the other two datasets. The reason is that SW contains many short utterances that can be either a continuation of the same speaker or a new turn from the other speaker, whereas the responses in the other two domains are always followed by a different speaker. The more complex context pattern in SW may require special treatment. We leave it for future work.

The second experiment checks if the policy network π is able to predict the right latent action given just the dialog context. We report both accuracy, i.e. $\underset{k}{\operatorname{argmax}}\, q_{q_\phi}(k|\mathbf{x}) = \underset{k'}{\operatorname{argmax}}\, p_\pi(k'|\mathbf{c})$ and perplexity of $p_\pi(\mathbf{z}|\mathbf{c})$. The perplexity measure is more useful for open domain dialogs because decision-making in complex dialogs is often one-to-many given a similar context (Zhao et al., 2017b).

	SMD	
AE-ED	3.045 (51.5% sys 52.4% usr 50.5%)	
ST-ED	1.695 (75.5% sys 82.1% usr 69.2%)	
	DD	SW
AE-ED	4.47 (35.8%)	4.46 (31.68%)
ST-ED	3.89 (47.5%)	3.68 (33.2%)

Table 6.7: Performance of policy network. $\mathcal{L}_{attr}$ is included in training. The reported numbers are in the format of perplexity (accuracy).

Table 6.7 shows the prediction scores on the three dialog datasets. These scores provide useful insights to understand the complexity of a dialog dataset. For example, accuracy on open-domain chatting is harder than the task-oriented SMD data. Also, it is intuitive that predicting system actions is easier than predicting user actions on SMD. Also, in general the prediction scores for ST-ED are higher the ones for AE-ED. The reason is related to our previous discussion about the granularity of the latent actions. Since latent actions from DI-VST mainly model the the type of utterances used in certain types of context, it is easier for the policy network to predict latent actions from DI-VST. Therefore, choosing the type of latent actions is a design choice and depends on the type of interpretability that is needed. We finish with an example generated from the two variants of LAED on SMD as shown in Table 6.8. Given a dialog context, our systems are able to output a probability distribution over different latent actions that have interpretable meaning along with their natural language realizations.

c	**usr**: Where does my friend live?	
Model	Action	Generated Responses
AE-ED	give loc info $p_\pi(\mathbf{z}\|\mathbf{c})$=0.34	- Tom is at 753 University Ave, and a road block. - Comfort Inn is at 7 miles away.
	give user info $p_\pi(\mathbf{z}\|\mathbf{c})$=0.22	- Your home address is 5671 barringer street. - Your home is at 10 ames street.
ST-ED	give loc info $p_\pi(\mathbf{z}\|\mathbf{c})$=0.93	- Jill's house is 8 miles away at 347 Alta Mesa Ave. - Jill lives at 347 Alta Mesa Ave.

Table 6.8: Interpretable dialog generation on SMD with top probable latent actions. AE-ED predicts more fine-grained but more error-prone actions.

6.5 Conclusion and Future Work

This chapter presents a novel unsupervised framework that enables the discovery of discrete latent actions and interpretable dialog response generation. Our main contributions reside in the two sentence representation models DI-VAE and DI-VST, and their integration with the encoder decoder models. Experiments show the proposed methods outperform strong baselines in learning discrete latent variables and showcase the effectiveness of interpretable dialog response generation. Our findings also suggest promising future research directions, including learning better context-based latent actions and using reinforcement learning to adapt policy networks. We believe that this work is an important step forward towards creating generative dialog models that can not only generalize to large unlabelled datasets in complex domains but also be explainable to human users.

Chapter 7

Cross-Domain Latent Action for Zero-shot Generalization

There are unlimited application domains where a dialog system can be useful. Thus collecting new datasets for every possible domain is not only tedious but also nearly impractical. Nonetheless, E2E generation based systems requires at least thousands of conversations for training to reach minimal performance, and this put generation-based system into a place that are difficult to be used in the practise. How to create more data-efficient E2E system is thus one of the crucial research challenge.

Now imagine a scenario where a human has learned to ride a bike and now requires to learn to ride motorcycles. Human is able to transfer a lot of the prior knowledge from the bike experience, e.g. holding the handles to keep balance, since they are similar actions that are shared between these two tasks. Similar knowledge transfer can be expected in human-human conversations. Imagine an customer support operator is transferred from the clothing department to shoe department. Although this operator now should ask very different questions and give completely different self-introduction about his/her role, human operator requires no extra "training" for this type of adaptation. It is also evident that the human operator can establish connections between similar "actions" between these two domains, so that they use the new available actions in the shoe domain using their knowledge from the clothing department.

This chapter is an execution of this intuition by proposing cross-domain latent actions, which learns a latent alignment between actions from different domains and enable a model that is trained on old domain to directly operate in a new domain without the needs to retrain on dialogs from the new domain. [1]

7.1 The Challenge of Domain Generalization

The generation-based end-to-end dialog model (GEDM) is one of the most powerful methods of learning dialog agents from raw conversational data in both chat-oriented and task-oriented domains (Serban et al., 2016c; Wen et al., 2016a; Zhao et al., 2017a).

[1]The code and data are available at `https://github.com/snakeztc/NeuralDialog-ZSDG`.

Its base model is an encoder-decoder network (Cho et al., 2014) that uses an encoder network to encode the dialog context and generate the next response via a decoder network. Yet prior work in GEDMs has overlooked an important issue, i.e. the data scarcity problem. In fact, the data scarcity problem is extremely common in most dialog applications due to the wide range of potential domains that dialog systems can be applied to. To the best of our knowledge, current GEDMs are data-hungry and have only been successfully applied to domains with abundant training material. This limitation prohibits the possibility of using the GEDMs for rapid prototyping in new domains and is only useful for domains with large datasets.

The key idea of this chapter lies in developing domain descriptions that can capture domain-specific information and a new type of GEDM model that can generalize to a new domain based on the domain description. Humans exhibit incredible efficiency in achieving this type of adaptation. Remember that a customer service agent in the shoe department is transferred to the clothing department. After reading some relevant instructions and documentation, this agent can immediately begin to deal with clothes-related calls without the need for any example dialogs. We also argue that it is more efficient and natural for domain experts to express their knowledge in terms of domain descriptions rather than example dialogs. This is because creating example dialogs involves writing down imagined dialog exchanges that can be shared across multiple domains and are not relevant to the unique proprieties of a specific domain. However, current state-of-the-art GEDMs are not designed to incorporate such knowledge and are therefore incapable of adapting their behavior to unseen domains.

This chapter introduces the use of *zero-shot dialog generation* (ZSDG) in order to enable GEDMs to generalize to unseen situations using minimal dialog data. Building on zero-shot classification (Palatucci et al., 2009), we formalize ZSDG as a learning problem where the training data contains dialog data from source domains along with domain descriptions from both the source and target domains. Then at testing time, ZSDG models are evaluated on the target domain, where no training dialogs were available. We approach ZSDG by first discovering a dialog policy network that can be shared between the source and target domains. The output from this policy is distributed vectors which are referred to as *latent actions*. Then, in order to transform the latent actions from any domain back to natural language utterances, a novel Action Matching (AM) algorithm is proposed that learns a cross-domain latent action space that models the semantics of dialog responses. This in turns enables the GEDM to generate responses in the target domains even when it has never observed full dialogs in them.

Finally the proposed methods and baselines are evaluated on two dialog datasets. The first one is a new synthetic dialog dataset generated by SimDial, which was developed for this study. SimDial enables us to easily generate task-oriented dialogs in a large number of domains, and provides a test bed to evaluate different ZSDG approaches. We further test our methods on a recently released multi-domain human-human corpus (Eric and Manning, 2017b) to validate whether performance can generalize to real-world conversations. Experimental results show that our methods are effective in incorporating knowledge from domain descriptions and achieve strong ZSDG performance.

7.2 Related Work

Perhaps the most closely related topic is zero-shot learning (ZSL) for image classification (Larochelle et al., 2008), which has focused on classifying unseen labels. A common approach is to represent the labels as attribute values instead of class indexes (Palatucci et al., 2009). As a result, at test time, the model can first predict the semantic attributes in the input, then make the final prediction by comparing the predicted attributes with the candidate labels' attributes. More recent work (Socher et al., 2013; Romera-Paredes and Torr, 2015) improved on this idea by learning parametric models, e.g. neural networks, to map the label and input data into a joint embedding space and then make predictions. Besides classification, prior art has explored the notion of task generalization in robotics, so that a robot can execute a new task that was not mentioned in training (Oh et al., 2017; Duan et al., 2017). In this case, a task is described by a demonstration or a sequence of instructions, and the system needs to learn to break down the instructions into previously learned skills. Also generating out-of-vocabulary (OOV) words from recurrent neural networks (RNNs) can be seen as a form of ZSL, where the OOV words are unseen labels. Prior work has used delexicalized tags (Zhao et al., 2017a) and copy-mechanism (Gu et al., 2016; Merity et al., 2016; Elsahar et al., 2018) to enable RNN output words that are not in its vocabulary.

Finally, ZSL has been applied to individual components in the dialog system pipeline. Chen et al. (Chen et al., 2016b) developed an intent classifier that can predict new intent labels that are not included in the training data. Bapna et al. (Bapna et al., 2017) extended that idea to the slot-filling module to track novel slot types. Both papers leverage a natural language description for the label (intent or slot-type) in order to learn a semantic embedding of the label space. Then, given any new labels, the model can still make predictions. There has also been extensive work on learning domain-adaptable dialog policy by first training a dialog policy on previous domains and testing the policy on a new domain. Gasic et al. (Gasic and Young, 2014) used the Gaussian Process with cross-domain kernel functions. The resulting policy can leverage experience from other domains to make educated decisions in a new one.

In summary, past ZSL research in the dialog domain has mostly focused on the individual modules in a pipeline-based dialog system. We believe our proposal is the first step in exploring the notion of adapting an entire end-to-end dialog system to new domains for domain generalization.

7.3 Problem Formulation

We begin by formalizing zero-shot dialog generation (ZSDG). Generative dialog models take a dialog context $\mathbf{c}$ as input and then generate the next response $\mathbf{x}$. ZSDG uses the term *domain* to describe the difference between training and testing data. Let $D = D_s \bigcup D_t$ be a set of domains, where D_s is a set of source domains, D_t is a set of target domains and $D_s \cap D_t = \emptyset$. During training, we are given a set of samples $\{\mathbf{c}^{(n)}, \mathbf{x}^{(n)}, d^{(n)}\} \sim p_{\text{source}}(\mathbf{c}, \mathbf{x}, d)$ drawn from the *source domains*. During testing, a ZSDG

model will be given a dialog context $\mathbf{c}$ and a domain d drawn from the *target domains* and must generate the correct response $\mathbf{x}$. Moreover, ZSDG assumes that every domain d has its own domain description $\phi(d)$ that is available at training for both source and target domains. The primary goal is to learn a generative dialog model $\mathcal{F} : C \times D \rightarrow X$ that can perform well in a target domain, by relating the unseen target domain description to the seen descriptions of the source domains. Our secondary goal is that $\mathcal{F}$ should perform similarly to a model that is designed to operate solely in the source domains. In short, the problem of ZSDG can be summarized as:

$$\begin{aligned}
\text{Train Data: } &\{\mathbf{c}, \mathbf{x}, d\} \sim p_{\text{source}}(\mathbf{c}, \mathbf{x}, d) \\
&\{\phi(d)\}, d \in D \\
\text{Test Data: } &\{\mathbf{c}, \mathbf{x}, d\} \sim p_{\text{target}}(\mathbf{c}, \mathbf{x}, d) \\
\text{Goal: } &\mathcal{F} : C \times D \rightarrow X
\end{aligned}$$

7.4 Proposed Method

7.4.1 Seed Responses as Domain Descriptions

The design of the domain description ϕ is a crucial factor that decides whether robust performance in the target domains is achievable. This paper proposes *seed response* (SR) as a general-purpose domain description that can readily be applied to different dialog domains. SR needs for the developers to provide a list of example responses that the model can generate in this domain. SR's assumption is that a dialog model can discover analogies between responses from different domains, so that its dialog policy trained on source domains can be reused in the target domain. Without losing generality, SR_d defines $\phi(d)$ as $\{\mathbf{x}^{(i)}, a^{(i)}, d\}_{\text{seed}}$ for domain d, where $\mathbf{x}$ is a seed response and a is its annotations. Annotations are domain-general salient features that are shared across source and target domains, and help the system in infer the relationship amongst responses from different domains. This may be difficult to achieve using only words in $\mathbf{x}$, e.g. two domains with distinct word distributions. For example, in a task-oriented weather domain, a seed response can be: *The weather in New York is raining* and the annotation is a semantic frame that contains domain general dialog acts and slot arguments, i.e. *[Inform, loc=New York, type=rain]*. The number of seed responses is often much smaller than the number of potential responses in the domain so it is best for SR to cover more responses that are unique to this domain. SRs assume that there is a discourse-level pattern that can be shared between the source and target domains, so that a system only needs sentence-level knowledge to adapt to the target. This assumption holds in many slot-filling dialog domains and it is easy to provide utterances in the target domain that are analogies to the ones from the source domains.

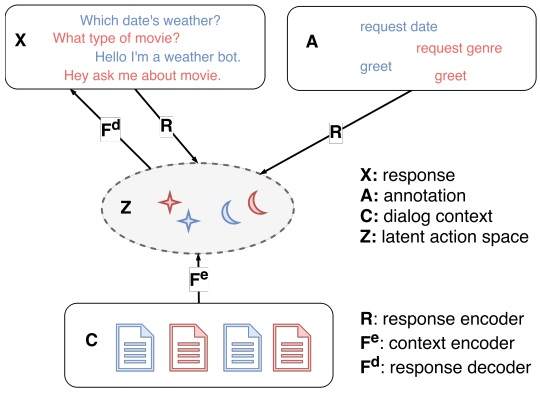

Figure 7.1: An overview of our Action Matching framework that looks for a latent action space Z shared by the response, annotation and predicted latent action from F^e.

7.4.2 Action Matching Encoder-Decoder

Figure 7.1 shows an overview of the model we use to tackle ZSDG. The base model is a standard encoder-decoder F where an encoder F^e maps $\mathbf{c}$ and d into a distributed representation $\mathbf{z_c} = F^e(\mathbf{c}, d)$ and the decoder F^d generates the response $\mathbf{x}$ given $\mathbf{z_c}$. We denote the embedding space that $\mathbf{z_c}$ resides in as the *latent action* space. We follow the KB-as-an-environment approach (Zhao and Eskenazi, 2016) where the generated $\mathbf{x}$ include both system verbal utterances and API queries that interface with back-end databases. This base model has been proven to be effective in human interactive evaluation for task-oriented dialogs (Zhao et al., 2017a).

We have two high-level goals: (1) learn a cross-domain F that can be reused in all source domains and potentially shared with target domains as well. (2) create a mechanism to incorporate knowledge from the domain descriptions into F so that it can generate novel responses when tested on the target domains. To achieve the first goal, we combine $\mathbf{c}$ and d by appending d as a special word token at the beginning of every utterance in $\mathbf{c}$. This simple approach performs well and enables the context encoder to take the domain into account when processing later word tokens. Also, this context domain integration can easily scale to dealing with a large number of domains. Then we encourage F to discover reusable dialog policy by training the same encoder decoder on dialog data generated from multiple source domains at the same time, which is a form of multi-task learning (Collobert and Weston, 2008). We achieve the second goal by projecting the response $\mathbf{x}$ from all domains into the same latent action space Z. Since $\mathbf{x}$ alone may not be sufficient to infer its semantics, we rely on their annotations a to learn meaningful semantic representations. Let $\mathbf{z_x}$ and $\mathbf{z}_a$ be the projected latent actions from $\mathbf{x}$ and a. Our method encourages $\mathbf{z}^{d_1}_{\mathbf{x}_1} \approx \mathbf{z}^{d_2}_{\mathbf{x}_2}$ when $\mathbf{z}^{d_1}_{a_1} \approx \mathbf{z}^{d_2}_{a_2}$. Moreover, for a given $\mathbf{z}$ from any domain, we ensure that the decoder F^d can generate the corresponding response $\mathbf{x}$ by training on both SR_d for $d \in D$ and source dialogs.

Specifically, we propose the Action Matching (AM) training procedure. We first introduce a recognition network R that can encode $\mathbf{x}$ and a into $\mathbf{z_x} = R(\mathbf{x}, d)$ and $\mathbf{z}_a = R(a, d)$ respectively. During training, the model receives two types of data. The

first type is domain description data in the form of $\{\mathbf{x}, a, d\}_{seed}$ for each domain. The second type of data is source domain dialog data in the form of $\{\mathbf{c}, \mathbf{x}, d\}$. For the first type of data, we update the parameters in q_ϕ and F^d by minimizing the following loss function:

$$\mathcal{L}_{\text{dd}}(F^d, R) = -\log p_{p_\theta}(\mathbf{x}|q_\phi(a, d)) + \lambda \mathbb{D}[q_\phi(\mathbf{x}, d)\|q_\phi(a, d)] \tag{7.1}$$

where λ is a constant hyperparameter and $\mathbb{D}$ is a distance function, e.g. mean square error (MSE), that measures the closeness of two input vectors. The first term in $\mathcal{L}_{\text{dd}}$ trains the decoder F^d to generate the response $\mathbf{x}$ given $\mathbf{z}_a = R(a, d)$ from all domains. The second term in $\mathcal{L}_{\text{dd}}$ enforces the recognition network R to encode a response and its annotation to nearby vectors in the latent action space from all domains, i.e. $\mathbf{z}^d_{\mathbf{x}} \approx \mathbf{z}^d_a$ for $d \in D$.

Moreover, just optimizing $\mathcal{L}_{\text{dd}}$ does not ensure that the $\mathbf{z}_{\mathbf{c}}$ predicted by the encoder F^e will be related to the $\mathbf{z}_{\mathbf{x}}$ or $\mathbf{z}_a$ encoded by the recognition network q_ϕ. So when we receive the second type of data (source dialogs), we add a second term to the standard maximum likelihood objective to train F and q_ϕ.

$$\mathcal{L}_{\text{dialog}}(F, R) = -\log p_{p_\theta}(\mathbf{x}|F^e(\mathbf{c}, d)) + \lambda \mathbb{D}(q_\phi(\mathbf{x}, d)\|F^e(\mathbf{c}, d)) \tag{7.2}$$

The second term in $\mathcal{L}_{\text{dialog}}$ completes the loop by encouraging $\mathbf{z}^d_{\mathbf{c}} \approx \mathbf{z}^d_{\mathbf{x}}$, which resembles the regularization term used in variational autoencoders (Kingma and Welling, 2013). Assuming that annotation a provides a domain-agnostic semantic representation of $\mathbf{x}$, then F trained on source domains can begin to operate in the target domains as well. During training, our AM algorithm alternates between these two types of data and optimizes $\mathcal{L}_{\text{dd}}$ or $\mathcal{L}_{\text{dialog}}$ accordingly. The resulting models effectively learn a latent action space that is shared by the the response annotation a, response $\mathbf{x}$ and predicted latent action based on $\mathbf{c}$ in all domains. AM training is summarized in Algorithm 1.

Algorithm 1: Action Matching Training

```
Initialize weights of F^e, F^d, R;
Data = {c, x, d} ⋃ {x, a, d}_seed
while batch ~ Data do
    if batch in the form {c, x, d} then
        Backpropagate loss L_dialog
    else
        Backpropagate loss L_dd
    end
end
```

7.4.3 Architecture Details

We implement an AMED for later experiments as follows:

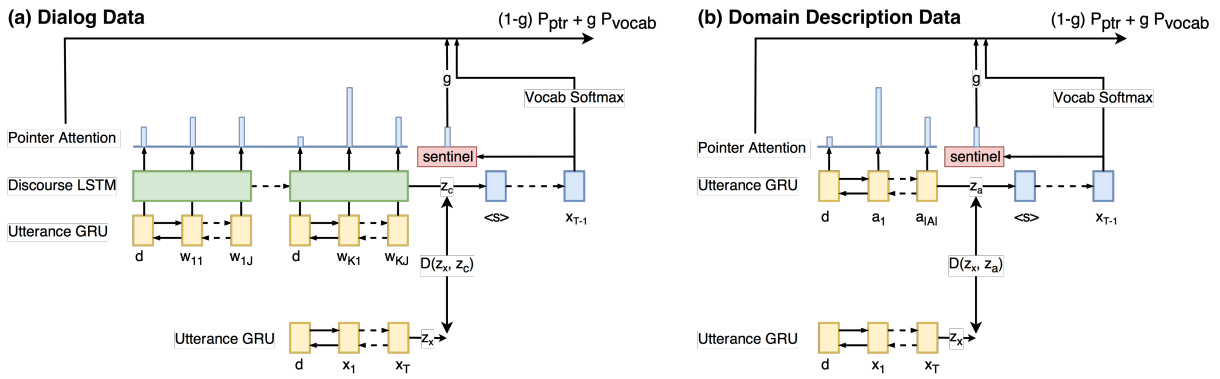

Figure 7.2: Visual illustration of our AM encoder decoder with copy mechanism (Merity et al., 2016). Note that AM can also be used with RNN decoders without the copy functionality.

Distance Functions: In this study, we assume that the latent actions are deterministic distributed vectors. Thus MSE is used: $\mathbb{D}(\mathbf{z}, \hat{\mathbf{z}}) = \frac{1}{L}\sum_l^L(\mathbf{z}_l - \hat{\mathbf{z}}_l)^2$, where L is the dimension size of the latent actions. Also, L_{dialog} and L_{dd} use the same distance function.

Recognition Networks: we use a bidirectional GRU-RNN (Cho et al., 2014) as q_ϕ to obtain utterance-level embedding. Since both $\mathbf{x}$ and a are sequences of word tokens, we combine them with the domain tag by appending the domain tag in the beginning of the original word sequence, i.e. $\{\mathbf{x}, d\}$ or $\{a, d\} = [d, w_1, ...w_J]$, where J is the length of the word sequence. Then the R will encode $[d, w_1, ...w_J]$ into hidden outputs in forward and backward directions, $[(\overrightarrow{h_0}, \overleftarrow{h_J}), ...(\overrightarrow{h_J}, \overleftarrow{h_0})]$. We use the concatenation of the last hidden states from each direction, i.e. $\mathbf{z_x}$ or $\mathbf{z}_a = [\overrightarrow{h_J}, \overleftarrow{h_J}]$ as utterance-level embedding for $\mathbf{x}$ or a respectively.

Dialog Encoders: a hierarchical recurrent encoder (HRE) is used to encode the dialog context, which handles long contexts better than non-hierarchical ones (Li et al., 2015b). HRE first uses an utterance encoder to encode every utterance in the dialog and then uses a discourse-level LSTM-RNN to encode the dialog context by taking output from the utterance encoder as input. Instead of introducing a new utterance encoder, we reuse the recognition network R described above as the utterance encoder, which serves the purpose perfectly. Another advantage is that using $\mathbf{z_x}$ predicted by R as input enables the discourse-level encoder to use knowledge from latent actions as well. Our discourse-level encoder is a 1-layer LSTM-RNN (Hochreiter and Schmidhuber, 1997), which takes in a list of output $[\mathbf{z}_1, \mathbf{z}_2..\mathbf{z}_K]$ from R and encodes them into $[v_1, v_2, ...v_K]$, where K is the number of utterances in the context. The last hidden state v_K is used as the predicted latent action $\mathbf{z_c}$.

Response Decoders: we experiment with two types of LSTM-RNN decoders. The first is an RNN decoder with an attention mechanism (Luong et al., 2015), enabling the decoder to dynamically look up information from the context. Specifically, we flatten the dialog context into a sequence of words $[w_{11}, ...w_{1J}...w_{KJ}]$. Using output from the

R and the discourse-level LSTM-RNN, each word here is represented by $m_{kj} = h_{kj} + W_v v_k$. Let the hidden state of the decoder at step t be s_t, then our attention mechanism computes the Softmax output via:

$$\alpha_{kj,t} = \text{softmax}(m_{kj}^T \tanh(W_\alpha s_t)) \tag{7.3}$$

$$\widetilde{s_t} = \sum_{kj} \alpha_{kj,t} m_{kj} \tag{7.4}$$

$$p_{\text{vocab}}(w_t|s_t) = \text{softmax}(\text{MLP}(s_t, \widetilde{s_t})) \tag{7.5}$$

The second type is the LSTM-RNN with a copy mechanism that can directly copy words from the context as output (Gu et al., 2016). Such a mechanism has already exhibited strong performance in task-oriented dialogs (Eric and Manning, 2017a) and is well suited for generating OOV word tokens (Elsahar et al., 2018). We implemented the Pointer Sentinel Mixture Model (PSM) (Merity et al., 2016) as our copy decoder. PSM defines the generation of the next word as a mixture of probabilities from either the Softmax output from the decoder LSTM or the attention Softmax for words in the context: $p(w_t|s_t) = g p_{\text{vocab}}(w_t|s_t) + (1-g) p_{\text{ptr}}(w_t|s_t)$, where g is the mixture weight computed from a sentinel vector u with s_t.

$$p_{\text{ptr}}(w_t|s_t) = \sum_{kj \in I(w,\mathbf{x})} \alpha_{kj,t} \tag{7.6}$$

$$g = \text{softmax}(u^T \tanh(W_\alpha s_i)) \tag{7.7}$$

7.5 Datasets for ZSDG

Two dialog datasets were used for evaluation.

7.5.1 SimDial Data

We developed SimDial[2], which is a multi-domain dialog generator that can generate realistic conversations for slot-filling domains with configurable complexity. Compared to other synthetic dialog corpora used to test GEDMs, e.g. bAbI (Dodge et al., 2015), SimDial data is significantly more challenging. First since SimDial simulates communication noise, the dialogs that are generated can be very long (more than 50 turns) and the simulated agent can carry out error recovery strategies to correctly infer the users' goals. This challenges end-to-end models to model long dialog contexts. SimDial also simulates spoken language phenomena, e.g. self-repair, hesitation. Prior work (Eshghi et al., 2017) has shown that this type of utterance-level noise deteriorates end-to-end dialog system performance.

[2] `https://github.com/snakeztc/SimDial`

Data Details

SimDial was used to generate dialogs for 6 domains: restaurant, movie, bus, restaurant-slot, restaurant-style and weather. For each domain, 900/100/500 dialogs were generated for training, validation and testing. On average, each dialog had 26 utterances and each utterance had 12.8 word tokens. The total vocabulary size was 651. We split the data such that the training data included dialogs from the restaurant, bus and weather domains and the test data included the restaurant, movie, restaurant-slot and restaurant style domains. This setup evaluates a ZSDG system from the following perspectives:

Restaurant (in domain): evaluation on the restaurant test data checks if a dialog model is able to maintain its performance on the source domains. **Restaurant-slot (unseen slots)**: restaurant-slot has the same slot types and natural language generation (NLG) templates as the restaurant domain, but has a completely different slot vocabulary, i.e. different location names and cuisine types. Thus this is designed to evaluate a model that can generalize to unseen slot values. **Restaurant-style (unseen NLG)**: restaurant-style has the same slot type and vocabulary as restaurant, but its NLG templates are completely different, e.g. "which cuisine type?" $\rightarrow$ "please tell me what kind of food you prefer". This part tests whether a model can learn to adapt to generate novel utterances with similar semantics. **Movie (new domain)**: movie has completely different NLG templates and structure and shares few common traits with the source domains at the surface level. Movie is the hardest task in the SimDial data, which challenges a model to correctly generate next responses that are semantically different from the ones in source domains.

Finally, we obtain SRs as domain descriptions by randomly selecting 100 unique utterances from each domain. The response annotation is a response's internal semantic frame used by the SimDial generator. For example, "I believe you said Boston. Where are you going?" $\rightarrow$ [implicit-confirm loc=Boston; request location].

7.5.2 Stanford Multi-Domain Dialog Data

The second dataset is the Stanford multi-domain dialog (SMD) dataset (Eric and Manning, 2017b) of 3031 human-human dialogs in three domains: weather, navigation and scheduling. One speaker plays the role of a driver. The other plays the car's AI assistant and talks to the driver to complete tasks, e.g. setting directions on a GPS. Average dialog length is 5.25 utterances; vocabulary size is 1601. We use SMD to validate whether our proposed methods generalize to human-generated dialogs. We generate SR by randomly selecting 150 unique utterances for each domain. An expert annotates the seed utterances with dialog acts and entities. For example "All right, I've set your next dentist appointment for 10am. Anything else?" $\rightarrow$ [ack; inform goal event=dentist appointment time=10am ; request needs]. Finally, in order to formulate a ZSDG problem, we use a leave-one-out approach with two domains as source domains and the third one as the target domain, which results in 3 possible configurations.

In order to effectively evaluate the proposed models' ability to generalize to new domains, multi-domain conversation dataset is needed. Unfortunately, currently avail-

able dataset for both task-oriented and social-oriented dialogs are not designed to collect conversations for different domain but similar task. The closest data we have is from dialog state tracking challenge (DSTC) since 2013, which is mostly task-oriented dialogs collected from various systems, including Let's Go Bus Information System (DSTC-1) (Raux et al., 2005), Cambridge Restaurant Recommendation System (DSTC2-3) (Young, 2006). However, this dataset is not ideal for two main reasons: 1) the number of domain is only 2, whereas we wish to train the models on a larger number of domains (more than 5) to test the limit of domain generalization 2) the data is collected from different hand-crafted dialog managers, which may not be complex enough to test the expressive power of generative encoder-decoder models. Therefore, we develop two new corpora, one synthetic and one real-world, that are designed to be used as the benchmark of learning generative dialog models from multiple domains. The following two sections describe them in details.

7.5.3 SimDial: A Multi-domain Dialog Generator

Collecting large conversational data is a tedious task via human. Therefore, using simulated data has been a common approach as the initial test bed for evaluating and training dialog system (Dodge et al., 2015). SimDial is a configurable domain-agnostic synthetic conversation generator that can generate arbitrary number of conversations with various number of noisy condition for any domains. The overall architecture of SimDial is shown in Figure 7.3 To generate conversation data using SimDial, the developers needs

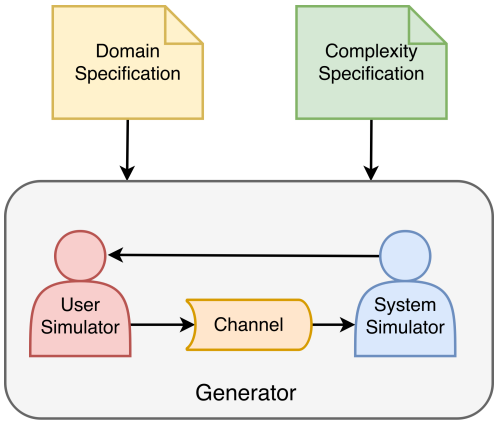

Figure 7.3: Overall Architecture of SimDial Data Generator

to provide two specifications: *domain specification* (DS) and *complexity specification* (CS). DS decides which slot-filling domain the generator is targeting at, which is equivalent to the concept of domain representation defined above. CS on the other hand defines how complex the user simulator will be from three perspectives:

1. Environmental Complexity: defines the noise level of communication channel, i.e. imitating ASR errors.

	Environment	Interaction			
	ASR error	Hesitation	Self Restart	Self Correct	Don't Care
Clean	0.0	0.0	0.0	0.0	0.0
Noisy	0.3	0.4	0.1	0.2	0.1
	Propostion				
	Yes/No Question	Mix Reject	Two-slots	Don't Care	Two Goals
Clean	0.0	0.0	0.0	0.0	0.0
Noisy	0.4	0.5	0.3	0.1	0.4

Table 7.1: Complexity Specifications for clean and noisy conditions

.

2. Propositional Complexity: including how many patterns the user simulator will choose to express and receive information.
3. Interaction Complexity: defines how much lexical-level linguistic variations that the user simulator will take.

Now we define two complexity spec that will be used in the rest of this chapter. The first one is Clean which introduces 0 noise and the second one is Noisy, which we introduce non-zero noise in all of the above three dimension. Details numbers are shown in Table 7.1.

Given DS and CS the generator will initialize an instance of user simulator and an instance of system simulator. The two simulator will converse with each at semantic level (i.e. using dialog acts and slot values). We assume that user can understand the system perfectly, while the users output is passed through a noisy channel which will introduce uncertainty and errors. Therefore, the system will need to carry out error-handling strategies, e.g. grounding to successfully complete the dialogs. The following are two example dialog with clean CS and noisy CS respectively. (The number before every user utterance is the simulated ASR confidence score).

Clean Complexity Spec
Sys*: What can I do for you?*
User(1.0)*: Hi. Recommend a restaurant.*
Sys*: Which place?*
User(1.0)*: I am at Pittsburgh*
Sys*: What kind of food do you like?*
User(1.0)*: Chinese food.*
Sys*: Restaurant xx is a good choice. Anything else?*
User(1.0)*: What kind of parking options does it have?*
Sys*: It has street parking?*
User(1.0)*: Thanks. Bye.*

Noisy Complexity Spec
Sys*: What can I do for you?*
User (0.54)*: Hello robot. I need to find a restaurant.*

***Sys**: Which city are you interested in?*
***User (0.75)**: Boston. Oh no, at uhm Seattle actually. And I like Chinese food.*
***Sys**: I believe you said Seattle and Chinese food. Restaurant xx is a nice place. Anything else?*
***User (0.6)**: Not done yet. Is it open now?*
***Sys**: Yes, it is right now. How can I help?*
***User (0.69)**: No more questions. Thanks.*
***Sys**: Bye.*

As we can see, the construction of SimDial is identical to our prior assumption of transferring knowledge among domains, since the generator is essentially a domain-agnostic function that can generate arbitrary dialogs just based on the information of the DS. Thus if any proposed models in fact is able to learn the desired domain-agnostic model, it should achieve very good results on data generated from SimDial, since the data is actually generated from such a distribution. This allows us to rapidly test and benchmark domain-agnostic dialog models without worrying about the fact the data is generated from more complex distribution.

7.6 Experiments and Results

The baseline models include 1. hierarchical recurrent encoder with attention decoder (+Attn) (Serban et al., 2016c). 2. hierarchical recurrent encoder with copy decoder (Merity et al., 2016) (+Copy), which has achieved very good performance on task-oriented dialogs (Eric and Manning, 2017a). We then augment both baseline models with the proposed cross-domain AM training procedure and denote them as +Attn+AM and +Copy+AM.

Evaluating generative dialog systems is challenging since the model can generate free-form responses. Fortunately, we have access to the internal semantic frames of the SimDial data, so we use the automatic measures used in (Zhao et al., 2017a) that employ four metrics to quantify the performance of a task-oriented dialog model. **BLEU** is the corpus-level BLEU-4 between the generated response and the reference ones (Papineni et al., 2002). **Entity $\mathbf{F_1}$** checks if a generated response contains the correct entities (slots) in the reference response. **Act $\mathbf{F_1}$** measures whether the generated responses reflect the dialog acts in the reference responses, which compensates for BLEU's limitation of looking for exact word choices. A one-vs-rest support vector machine (Scholkopf and Smola, 2001) with bi-gram features is trained to tag the dialogs in a response. **KB $\mathbf{F_1}$** checks all the key words in a KB query that the system issues to the KB backend. Finally, we introduce **BEAK** $= \sqrt[4]{\text{bleu} \times \text{ent} \times \text{act} \times \text{kb}}$, the geometric mean of these four scores, to quantify a system's overall performance. Meanwhile, since the oracle dialog acts and KB queries are not provided in the SMD data (Eric and Manning, 2017b), we only report BLEU and entity F_1 results on SMD.

7.6.1 Main Results

In domain	+Attn	+Copy	+Attn +AM	+Copy +AM
BLEU	59.1	70.4	67.7	70.1
Entity	69.2	70.5	74.1	**79.9**
Act	94.7	92.0	94.1	**95.1**
KB	94.7	96.1	95.2	**97.0**
BEAK	77.2	81.3	81.9	**84.7**
Unseen Slot	+Attn	+Copy	+Attn +AM	+Copy +AM
BLEU	24.9	45.6	47.9	**68.5**
Entity	56.0	68.0	53.1	**74.6**
Act	90.9	91.8	86.0	**94.5**
KB	78.1	89.6	81.0	**95.3**
BEAK	56.1	71.1	64.8	**82.3**
Unseen NLG	+Attn	+Copy	+Attn +AM	+Copy +AM
BLEU	15.8	36.9	43.5	**70.1**
Entity	61.7	68.9	63.8	**72.9**
Act	91.5	92.2	89.3	**95.2**
KB	66.2	94.6	93.1	**97.0**
BEAK	49.3	65.9	69.3	**82.9**
New domain	+Attn	+Copy	+Attn +AM	+Copy +AM
BLEU	13.5	24.6	36.7	**54.6**
Entity	23.1	40.8	23.3	**52.6**
Act	82.3	85.5	84.8	**88.5**
KB	43.5	67.1	67.0	**88.2**
BEAK	32.5	48.8	46.8	**68.8**

Table 7.2: Evaluation results on test dialogs from SimDial Data. Bold values indicate the statistically significant best performance.

Table 7.2 shows results on the SimDial data. Although the standard +Attn model achieves good performance in the source domains, it doesn't generalize to target domains, especially for entity F_1 in the unseen-slot domain, BLEU score in the unseen-NLG domain, and all new domain metrics. The +Copy model has better, although still limited, generalization to target domains. The main benefit of the +Copy model is its ability to directly copy and output words from the context, reflected in its strong entity F_1 in the unseen slot domain. However, +Copy can't generalize to new domains where utterances are novel, e.g. the unseen NLG or the new domain. However, our AM algorithm substantially improves performance of both decoders (Attn and Copy). Results show that the proposed AM algorithm is complementary to decoders with a copy mechanism: HRED+Copy+AM model has the best performance on all target domains. In the easier unseen-slot and unseen-NLG domains, the resulting ZSDG system achieves a BEAK of about 82, close to the in-domain BEAK performance (84.7). Even in the new domain (movie), our model achieves a BEAK of 67.2, 106% relative improvement w.r.t +Attn and 38.8% relative improvement w.r.t +Copy. Moreover, our AM method also im-

proves performance on in-domain dialogs, suggesting that AM exploits the knowledge encoded in the domain description and improves the models' generalization.

Navigate	Oracle	+Attn	+Copy	+Copy +AM
BLEU	13.4	0.9	5.4	**5.9**
Entity	19.3	2.6	4.7	**14.3**
Weather	Oracle	+Attn	+Copy	+Copy +AM
BLEU	18.9	4.8	4.4	**8.1**
Entity	51.9	0.0	16.3	**31.0**
Schedule	Oracle	+Attn	+Copy	+Copy +AM
BLEU	20.9	3.0	3.8	**7.9**
Entity	47.3	0.4	17.1	**36.9**

Table 7.3: Evaluation on SMD data. The bold domain title is the one that was excluded from training. Bold values indicate the statistically significant best performance.

Table 7.3 summarizes the results on the SMD data. We also report the oracle performance, obtained by training +Copy on the full dataset. The AM algorithm can significantly improve Entity F_1 and BLEU from the two baseline models. +Copy+AM also achieves competitive performance in terms of Entity F_1 compared to the oracle scores, despite the fact that no target domain data was used in training.

Type	**Reference**	**+Attn**	**+Copy**	**+Copy+AM**
General Utts	See you next time.	Goodbye.	See you next time.	See you next time.
Unseen Slots	Do you mean romance movie?	Do you mean Chinese food.	Do you mean **romance** food?	Do you mean **romance** movie?
Unseen Utts	Movie 55 is a great movie.	Bus 12 can take you there.	Bus **55** can take you there.	Movie **55** is a great movie.

Table 7.4: Three types of responses and generation results (tested on the new movie domain). The text in bold is the output directly copied from the context by the copy decoder.

7.6.2 Model Analysis

Various types of performance improvement were also studied. Figure 7.4 shows the breakdown of the BLEU score according to the dialog acts of reference responses. Models with +Copy decoder can improve performance for all dialog acts except for the *greet* act, which occurs at the beginning of a dialog. In this case, the +Copy decoder has no context to copy and thus cannot generate any novel responses. This is one limitation of +Copy decoder since in real interactive testing with humans, each system utterance must be generated from the model instead of copied from the context. However, models with AM training learn to generate novel utterances based on knowledge from the SR, so +Copy+AM can generate responses at the beginning of a dialog.

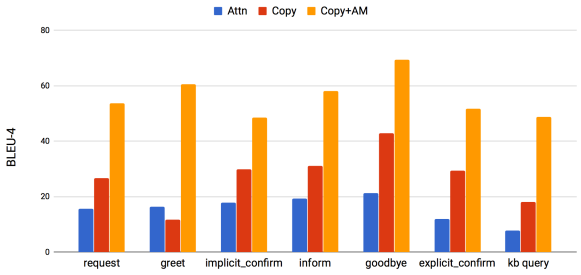

Figure 7.4: Breakdown BLEU scores on the new domain test set from SimDial.

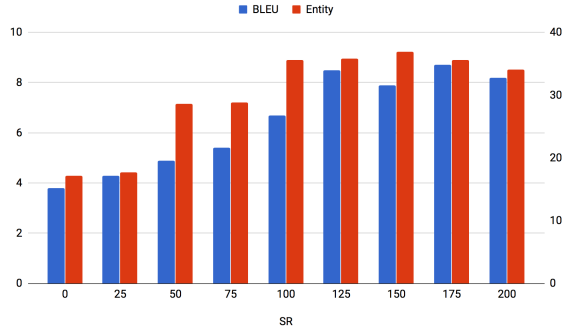

Figure 7.5: Performance on the schedule domain from SMD while varying the size of SR.

A qualitative analysis was conducted to summarize typical responses from these models. Table 7.4 shows three types of typical situations in the SimDial data. The first type is **general utterance** utterances, e.g. "See you next time" that appear in all domains. All three models correctly generate them in the ZSDG setting. The second type is utterances with **unseen slots**. For example, explicit confirm "Do you mean xx?". +Attn fails in this situation since the new slot values are not in its vocabulary. +Copy still performs well since it learns to copy entity-like words from the context, but the overall sentence is often incorrect, e.g. "Do you mean romance food". The last one is **unseen utterance** where both +Attn and +Copy fail. The two baseline models can still generate responses with correct dialog acts, but the output words are in the source domains. Only the models trained with AM are able to infer that "Movie xx is a great movie" serves a function similar to "Bus xx can take you there", and generates responses using the correct words from the target domain.

Finally we investigate how the the size of SR affects AM performance. Figure 7.5 shows results in the SMD schedule domain. The number of seed responses varies from 0 to 200. Performance in the target domains is positively correlated with the number of seed responses. We also observe that the model achieves sufficient SR performance at 100, compared to the ones trained on all of the 200 seed responses. This suggests that the amount of seeding needed by SR is relatively small, which shows the practicality of using SR as a domain description.

7.7 Conclusion and Future Work

This chapter introduces ZSDG, dealing with neural dialog systems' domain generalization ability. We formalize the ZSDG problem and propose an Action Matching framework that discovers cross-domain latent actions. We present a new simulated multi-domain dialog dataset, SimDial, to benchmark the ZSDG models. Our assessment validates the AM framework's effectiveness and the AM encoder decoders perform well in the ZSDG setting.

From a latent action point of view, the proposed AM algorithm shows that (1) shareable latent action enables knowledge transfer across domains at utterance level. This includes knowledge on both understanding (feeding latent action into the context encoder as input) and decision-making (generating responses given latent actions in the new domain). In practices, there are many dialog applications that share similar discourse-level structure, but very different utterances, e.g. recommendation dialog systems for various categories of products. Thus there is a huge value for utterance-level knowledge transfer (2) the proposed AM algorithm demonstrates that latent action enables a GEDM to be trained with more than end-to-end maximum likelihood estimation. The introduction of latent action allows system designers to assign inductive bias to the purposes these latent variables in the model (e.g. domain matching in the proposed AM), and this creates opportunities for extra auxiliary loss signals and bringing novel improvement.

ZSDG also provides promising future research questions. How can we reduce the annotation cost of learning the latent alignment between actions in different domains? How can we create ZSDG for new domains where the discourse-level patterns are significantly different? What are other potential domain description formats? In summary, solving ZSDG is an important step for future general-purpose conversational agents.

Chapter 8

Optimizing Dialog Strategy with Latent Action Reinforcement Learning

So far, we have presented three different types of latent actions that can be induced from supervised learning. After the supervised learning stage, the model will be used directly. This chapter takes a step beyond supervised learning and moves towards improving the dialog policy according to task-level rewards with reinforcement learning. For E2E generation dialog systems, this is often done via policy gradients methods (Williams, 1992) at each word output in the decoder network. This approach has shown to be effective in improving language generation quality when the reward signals is on evaluating the quality of language (Yu et al., 2017a; Zhong et al., 2017). Unfortunately, the reward function for dialog system is often calculated from higher level signals, e.g. user satisfaction, task success etc. Thus direct application of these reward functions for a generation-based dialog systems has been shown in previous research to make the model stop generating natural language (Lewis et al., 2017; Das et al., 2017). As a result, reinforcement learning for generation-based dialog models with is notoriously difficult to train.

We argue that the solution is to optimize the dialog policy over high-level actions and learn new conversational strategy about discourse-level plans, instead of sentence level variations. Using latent action becomes a natural choice, since previous chapters have already shown that they can be trained to represent high-level actions for dialog systems. In this chapter, we present a set of experiments on how to apply reinforcement learning with latent actions as the action space for E2E dialog models. Moreover, we propose a novel evaluation measure, Language Constrained Reward (LCR) curve, to expliclty quantify the trade-off between task-level rewards and language generation quality. Experimental results on two real-world dialog tasks show that latent action reinforcement learning performs significantly better than the conventional word-level approaches and does not suffer from many of the long lasting challenges. [1].

[1]The code and data are available at `https://github.com/snakeztc/NeuralDialog-LaRL`

8.1 Reinforcement Learning for End-to-end Generation-based Dialog Models

Optimizing dialog strategies in multi-turn dialog models is the cornerstone of building dialog systems that more efficiently solve real-world challenges, e.g. providing information (Young, 2006), winning negotiations (Lewis et al., 2017), improving engagement (Li et al., 2016b) etc. A classic solution employs reinforcement learning (RL) to learn a dialog policy that models the optimal action distribution conditioned on the dialog state (Williams and Young, 2007). However, since there are infinite human language possibilities, an enduring challenge has been to define what the action space is. For traditional modular systems, the action space is defined by hand-crafted semantic representations such as dialog acts and slot-values (Raux et al., 2005; Chen et al., 2013) and the goal is to obtain a dialog policy that chooses the best hand-crafted action at each dialog turn. But it is limited because it can only handle simple domains whose entire action space can be captured by hand-crafted representations (Walker, 2000; Su et al., 2017). This cripples a system's ability to handle conversations in complex domains.

Conversely, end-to-end (E2E) dialog systems have removed this limit by directly learning a response generation model conditioned on the dialog context using neural networks (Vinyals and Le, 2015; Sordoni et al., 2015). To apply RL to E2E systems, the action space is typically defined as the entire vocabulary; every response output word is considered to be an action selection step (Li et al., 2016b), which we denote as the word-level RL. Word-level RL, however, has been shown to have several major limitations in learning dialog strategies. The foremost one is that direct application of word-level RL leads to degenerate behavior: the response decoder deviates from human language and generates utterances that are incomprehensible (Lewis et al., 2017; Das et al., 2017; Kottur et al., 2017). A second issue is that since a multi-turn dialog can easily span hundreds of words, word-level RL suffers from credit assignment over a long horizon, leading to slow and sub-optimal convergence (Kaelbling et al., 1996; He et al., 2018).

This chapter proposes Latent Action Reinforcement Learning (LaRL), a novel framework that overcomes the limitations of word-level RL for E2E dialog models, marrying the benefits of a traditional modular approach in an unsupervised manner. The key idea is to develop E2E models that can invent their own discourse-level actions. These actions must be expressive enough to capture response semantics in complex domains, thus decoupling the discourse-level decision-making process from natural language generation. Then any RL technique can be applied to this induced action space in the place of word-level output. We propose a flexible latent variable dialog framework and investigate several approaches to inducing latent action space from natural conversational data. We further propose (1) a novel training objective that outperforms the typical evidence lower bound used in dialog generation (Zhao et al., 2017b) and (2) an attention mechanism for integrating discrete latent variables in the decoder to better model long responses. We test our proposed approach on two datasets, a negotiation domain DealOrNoDeal (Lewis et al., 2017) and a multi-domain slot-filling MultiWoz (Budzianowski et al., 2018). The experiments are carefully designed in order to

answer two research key questions: (1) what are the advantages of LaRL over Word-level RL? (2) what are the effective methods that can induce this latent action space?

8.2 Related Work

Prior RL research in modular dialog management has focused on policy optimization over hand-crafted action spaces in task-oriented domains (Walker, 2000; Young et al., 2007). A dialog manager is formulated as a Partially Observable Markov Decision Process (POMDP) (Young et al., 2013), where the dialog state is estimated via dialog state tracking models from the raw dialog context (Lee, 2013; Henderson et al., 2014; Ren et al., 2018). RL techniques are then used to find the optimal dialog policy (Gasic and Young, 2014; Su et al., 2017; Williams et al., 2017). Recent deep-learning modular dialog models have also explored joint optimization over dialog policy and state tracking to achieve stronger performance (Wen et al., 2016a; Zhao and Eskenazi, 2016; Liu and Lane, 2017).

A related line of work is reinforcement learning for E2E dialog systems. Due to the flexibility of encoder-decoder dialog models, prior work has applied reinforcement learning to more complex domains and achieved higher dialog-level rewards, such as open-domain chatting (Li et al., 2016b; Serban et al., 2017a), negotiation (Lewis et al., 2017), visual dialogs (Das et al., 2017), grounded dialog (Mordatch and Abbeel, 2017) etc. As discussed in Section 1, these methods consider the output vocabulary at every decoding step to be the action space; they suffer from limitations such as deviation from natural language and sub-optimal convergence.

Finally, research in latent variable dialog models is closely related to our work, which strives to learn meaningful latent variables for E2E dialog systems. Prior work has shown that learning with latent variables leads to benefits like diverse response decoding in Chapter 5, interpretable decision-making in Chapter 6 and zero-shot domain transfer in Chapter 7. Our work differs from prior work for two reasons: (1) latent action in previous work was only auxiliary, small-scale and mostly learned in a supervised or semi-supervised setting. This work focuses on unsupervised learning of latent variables and learns variables that are expressive enough to capture the entire action space by itself. (2) to our best knowledge, our work is the first comprehensive study of the use of latent variables for RL policy optimization in dialog systems.

8.3 Baseline Approach

E2E response generation can be treated as a conditional language generation task which uses neural encoder-decoders (Cho et al., 2014) to model the conditional distribution $p(\mathbf{x}|\mathbf{c})$ where $\mathbf{c}$ is the observed dialog context and $\mathbf{x}$ is the system's response to the context. The format of the dialog context is domain dependent. It can vary from textual raw dialog history (Vinyals and Le, 2015) to visual and textual context (Das et al., 2017). Training with RL usually has 2 steps: supervised pre-training and policy gradient re-

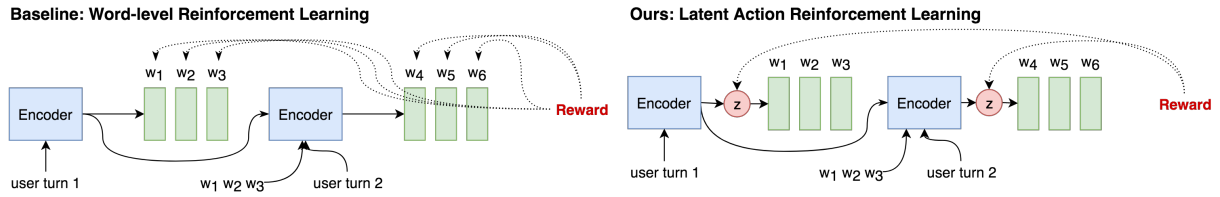

Figure 8.1: High-level comparison between word-level and latent-action reinforcement learning in a sample multi-turn dialog. Dashed line denotes places where policy gradients from task rewards are applied to the model.

inforcement learning (Williams and Zweig, 2016; Dhingra et al., 2017; Li et al., 2016b). Specifically, the supervised learning step maximizes the log likelihood on the training dialogs, where θ is the model parameter:

$$\mathcal{L}_{SL}(\theta) = \mathbb{E}_{\mathbf{x},\mathbf{c}}[\log p_\theta(\mathbf{x}|\mathbf{c})] \tag{8.1}$$

Then the following RL step uses policy gradients methods, such as the REINFORCE algorithm (Williams, 1992) to update the model parameters with respect to task-dependent goals. We assume that we have an environment that the dialog agent can interact with and that there is a turn-level reward r_t at every turn t of the dialog. We can then write the expected discounted return under a dialog model θ as $J(\theta) = \mathbb{E}[\sum_0^T \gamma^t r_t]$, where $\gamma \in [0, 1]$ is the discounting factor and T is the length of the dialog. Often a baseline function b is used to reduce the variance of the policy gradient (Greensmith et al., 2004), leading to $R_t = \sum_{k=0}^{T-t} \gamma^k (r_{t+k} - b)$.

Word-level Reinforcement Learning: as shown in Figure 8.1, the baseline approach treats every output word as an action step and its policy gradient is:

$$\nabla_\theta J(\theta) = \mathbb{E}_\theta[\sum_{t=0}^{T} \sum_{j=0}^{U_t} R_{tj} \nabla_\theta \log p_\theta(w_{tj}|w_{<tj}, \mathbf{c}_t)] \tag{8.2}$$

where U_t is the number of tokens in the response at turn t and j is the word index in the response. It is evident that Eq 8.2 has a very large action space, i.e. $|V|$ and a long learning horizon, i.e. TU. Prior work has found that the direct application of Eq 8.2 leads to divergence of the decoder. The common solution is to alternate with supervised learning with Eq 8.2 at a certain ratio (Lewis et al., 2017). We denote this ratio as RL:SL=A:B, which means for every A policy gradient updates, we run B supervised learning updates. We use RL:SL=off for the case where only policy gradients are used and no supervised learning is involved.

8.4 Latent Action Reinforcement Learning

We now describe the proposed LaRL framework. As shown in Figure 8.1, a latent variable $\mathbf{z}$ is introduced in the response generation process. We follow the definition of full

latent action that is defined in Chapter 3. The conditional distribution is factorized into $p(\mathbf{x}|\mathbf{c}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathbf{c})$ and the generative story is: (1) given a dialog context $\mathbf{c}$ we first sample a latent action $\mathbf{z}$ from $p_\pi(\mathbf{z}|\mathbf{c})$ and (2) generate the response $\mathbf{x}$ based on $\mathbf{z}$ via $p_\theta(\mathbf{x}|\mathbf{z})$, where p_π is the dialog encoder network and p_θ is the response decoder network. Given the above setup, LaRL treats the latent variable $\mathbf{z}$ as its action space instead of outputting words in response $\mathbf{x}$. We can now apply REINFORCE in the latent action space:

$$\nabla_\theta J(\pi) = \mathbb{E}_\theta[\sum_{t=0}^{T} R_t \log p_\pi(\mathbf{z}|\mathbf{c}_t)] \tag{8.3}$$

Compared to Eq 8.2, LaRL differs by:

- Shortens the horizon from TU to T.
- Latent action space is designed to be low-dimensional, much smaller than V.
- The policy gradient only updates the policy network p_π and the decoder p_θ stays intact.

These properties reduce the difficulties for dialog policy optimization and decouple high-level decision-making from natural language generation. The p_π are responsible for choosing the best latent action given a context $\mathbf{c}$ while p_θ is only responsible for transforming $\mathbf{z}$ into the surface-form words. Our formulation also provides a flexible framework for experimenting with various types of model learning methods. In this chapter, we focus on two key aspects: the type of latent variable $\mathbf{z}$ and optimization methods for learning $\mathbf{z}$ in the supervised pre-training step.

8.4.1 Types of Latent Actions

We explore both types of latent variables defined in Chapter 3: continuous Gaussian distribution (Serban et al., 2016c) and multivariate categorical distribution (Zhao et al., 2018). These two types are both compatible with our LaRL framework and can be defined as follows:

Gaussian Latent Actions follow M dimensional multivariate Gaussian distribution with a diagonal covariance matrix, i.e. $\mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$. Let the policy encoder network p_π consist of two parts: a context encoder $\mathcal{F}$, a neural network that encodes the dialog context $\mathbf{c}$ into a vector representation h, and a feed forward network π that projects h into $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. The process is defined as follows:

$$h = \mathcal{F}(\mathbf{c}) \tag{8.4}$$

$$\begin{bmatrix} \boldsymbol{\mu} \\ \log(\boldsymbol{\sigma}^2) \end{bmatrix} = \pi(h) \tag{8.5}$$

$$p(\mathbf{x}|\mathbf{z}) = p_\theta(\mathbf{z}) \quad \mathbf{z} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I}) \tag{8.6}$$

where the sampled $\mathbf{z}$ is used as the initial state of the decoder for response generation. Also we use $p_\theta(\mathbf{z}|\mathbf{c}) = \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}, \boldsymbol{\sigma}^2\mathbf{I})$ to compute the policy gradient update in Eq 8.3.

Categorical Latent Actions are M independent K-way categorical random variables. Each $\mathbf{z}_m$ has its own token embeddings to map latent symbols into vector space $\mathbf{E}_m \in \mathbb{R}^{K \times D}$ where $m \in [1, M]$ and D is the embedding size. Thus M latent actions can represent exponentially, K^M, unique combinations, making it expressive enough to model dialog acts in complex domains. Similar to Gaussian Latent Actions, we have

$$h = \mathcal{F}(\mathbf{c}) \tag{8.7}$$

$$p(Z_m|\mathbf{c}) = \text{softmax}(\pi_m(h)) \tag{8.8}$$

$$p(\mathbf{x}|\mathbf{z}) = p_\theta(\mathbf{E}_{1:M}(\mathbf{z}_{1:M})) \quad \mathbf{z}_m \sim p(Z_m|\mathbf{c}) \tag{8.9}$$

For the computing policy gradient in Eq 8.3, we have $p_\theta(\mathbf{z}|\mathbf{c}) = \prod_{m=1}^{M} p(Z_m = \mathbf{z}_m|\mathbf{c})$

Unlike Gaussian latent actions, a matrix $\mathbb{R}^{M \times D}$ comes after the embedding layers $\mathbf{E}_{1:M}(\mathbf{z}_{1:M})$, whereas the decoder's initial state is a vector of size $\mathbb{R}^D$. Previous work integrated this matrix with the decoder by summing over the latent embeddings, i.e. $\mathbf{x} = p_{\theta_d}(\sum_1^M \mathbf{E}_m(\mathbf{z}_m))$, denoted as **Summation Fusion** for later discussion (Zhao et al., 2018). A limitation of this method is that it could lose fine-grained order information in each latent dimension and have issues with long responses that involve multiple dialog acts. Therefore, we propose a novel method, **Attention Fusion**, to combine categorical latent actions with the decoder. We apply the attention mechanism (Luong et al., 2015) over latent actions as the following. Let i be the step index during decoding. Then we have:

$$\alpha_{mi} = \text{softmax}(h_i^T \mathbf{W}_a \mathbf{E}_m(\mathbf{z}_m)) \tag{8.10}$$

$$\mathbf{c}_i = \sum_{m=1}^{M} \alpha_{mi} \mathbf{E}_m(\mathbf{z}_m) \tag{8.11}$$

$$\widetilde{h_i} = \tanh(\mathbf{W}_s \begin{bmatrix} h_i \\ \mathbf{c}_i \end{bmatrix}) \tag{8.12}$$

$$p(w_i|h_i, \mathbf{c}_i) = \text{softmax}(W_o \widetilde{h_i}) \tag{8.13}$$

The decoder's next state is updated by $h_{i+1} = \text{RNN}(h_i, w_{i+1}), \widetilde{h_i})$ and h_0 is computed via summation-fusion. Thus attention fusion lets the decoder focus on different latent dimensions at each generation step.

8.4.2 Optimization Approaches

Full ELBO: Now given a training dataset $\{\mathbf{x}, \mathbf{c}\}$, our base optimization method is via stochastic variational inference by maximizing the evidence lowerbound (ELBO), a lowerbound on the data log likelihood:

$$\mathcal{L}_{full}(\theta, \pi, \phi) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x},\mathbf{c})}[p_\theta(\mathbf{x}|\mathbf{z}) - \text{D}_{\text{KL}}[q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c})||p_\pi(\mathbf{z}|\mathbf{c})]] \tag{8.14}$$

where $q_\gamma(\mathbf{z}|\mathbf{x}, \mathbf{c})$ is a neural network that is trained to approximate the posterior distribution $q(\mathbf{z}|\mathbf{x}, \mathbf{c})$ and $p(\mathbf{z}|\mathbf{c})$ and $p(\mathbf{x}|\mathbf{z})$ are achieved by $\mathcal{F}$, π and p_{θ_d}. For Gaussian latent

actions, we use the reparametrization trick (Kingma and Welling, 2013) to backpropagate through Gaussian latent actions and the Gumbel-Softmax (Jang et al., 2016) to backpropagate through categorical latent actions.

Lite ELBO: a major limitation is that Full ELBO can suffer from exposure bias at latent space, i.e. the decoder only sees $\mathbf{z}$ sampled from $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c})$ and never experiences $\mathbf{z}$ sampled from $p_\pi(\mathbf{z}|\mathbf{c})$, which is always used at testing time. Therefore, in this chapter, we propose a simplified ELBO for encoder-decoder models with stochastic latent variables:

$$\mathcal{L}_{lite}(\theta, \pi) = \mathbb{E}_{p(\mathbf{z}|\mathbf{c})}[p_\theta(\mathbf{x}|\mathbf{z}) - \beta \mathrm{D}_{\mathrm{KL}}[p_\pi(\mathbf{z}|\mathbf{c}))\|p(\mathbf{z})]] \tag{8.15}$$

Essentially this simplified objective sets the posterior network the same as our encoder, i.e. $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{c}) = p_\pi(\mathbf{z}|\mathbf{c})$, which makes the KL term in Eq 8.14 zero and removes the issue of exposure bias. But this leaves the latent spaces unregularized and our experiments show that if we only maximize $\mathbb{E}_{p_\pi(\mathbf{z}|\mathbf{c})}\, p_\theta(\mathbf{x}|\mathbf{z})]$ there is overfitting. For this, we add the additional regularization term $\beta \mathrm{D}_{\mathrm{KL}}[p_\pi(\mathbf{z}|\mathbf{c}))\|p(\mathbf{z})]$ that encourages the posterior to be similar to certain prior distributions and β is a hyper-parameter between 0 and 1. We set the $p(\mathbf{z})$ for categorical latent actions to be uniform, i.e. $p(\mathbf{z}) = 1/K$, and set the prior for Gaussian latent actions to be $\mathcal{N}(\mathbf{0}, \mathbf{I})$, which we will show that these design choices are effective.

8.4.3 Language Constrained Reward (LCR) curve for Evaluation

It is notoriously difficult to automatically quantify the performance of RL-based neural generation systems because it is possible for a model to achieve high task reward and yet not generate human language (Das et al., 2017). On the hand, although human-in-the-loop evaluation is the best metric to assess a system, it is expensive and hard to reproduce. Therefore, we propose a novel measure, the Language Constrained Reward (LCR) curve as an additional robust measure. The basic idea is to use an ROC-style curve to visualize the tradeoff between achieving higher reward and being faithful to human language. Specifically, at each checkpoint i over the course of RL training, we record two measures: (1) the PPL of a given model on the test data $p_i = \mathrm{PPL}(\theta_i)$ and (2) this model's average cumulative task reward in the test environment R_i^t. After RL training is complete, we create a 2D plot where the x-axis is the maximum PPL allowed, and the y-axis is the best achievable reward within the PPL budget in the testing environments:

$$\mathbf{y} = \max_i R_i^t \quad \text{subject to} \quad p_i < x \tag{8.16}$$

As a result, a perfect model should lie in the upper left corner whereas a model that sacrifices language quality for higher reward will lie in the lower right corner. Our results will show that the LCR curve is an informative and robust measure for model comparison.

8.5 Experiment Settings

8.5.1 DealOrNoDeal Corpus and RL Setup

DealOrNoDeal is a negotiation dataset that contains 5805 dialogs based on 2236 unique scenarios (Lewis et al., 2017). We hold out 252 scenarios for testing environment and randomly sample 400 scenarios from the training set for validation. The results are evaluated from 4 perspectives: Perplexity (PPL), Reward, Agreement and Diversity. PPL helps us to identify which model produces the most human-like responses, while Reward and Agreement evaluate the model's negotiation strength. Concretely, Agreement Diversity indicates whether the model discovers a novel discourse-level strategy or just repeats dull responses to compromise with the opponent. We closely follow the original paper and use the same reward function and baseline calculation. At last, to have a fair comparison, all the compared models shared the identical judge model and user simulator, which are a standard hierarchical encoder-decoder model trained with Maximum Likelihood Estimation (MLE).

8.5.2 Multi-Woz Corpus and a Novel RL Setup

Multi-Woz is a slot-filling dataset that contains 10438 dialogs on 6 different domains. 8438 dialogs are for training and 1000 each are for validation and testing. Since no prior user simulator exists for this dataset, for a fair comparison with the previous state-of-the-art we focus on the Dialog-Context-to-Text Generation task proposed in (Budzianowski et al., 2018). This task assumes that the model has access to the ground-truth dialog belief state and is asked to generate the next response at every system turn in a dialog. The results are evaluated from 3 perspectives: BLEU, Inform Rate and Success Rate. The BLEU score checks the response-level lexical similarity, while Inform and Success Rate measure whether the model gives recommendations and provides all the requested information at dialog-level. Current state-of-the-art systems struggle in this task and MLE models only achieve 60% success (Budzianowski et al., 2018). To transform this task into an RL task, we propose a novel extension to the original task as follows:

1. For each RL episode, randomly sample a dialog from the training set
2. Run the model on every system turn, and do not alter the original dialog context at every turn given the generated responses.
3. Compute Success Rate based on the generated responses in this dialog.
4. Compute policy gradient using Eq 8.3 and update the parameters.

This setup creates a variant RL problem that is similar to the Contextual Bandits (Langford and Zhang, 2008), where the goal is to adjust its parameters to generate responses that yield better Success Rate. One may argue that our setup creates a slightly simpler problem compared to the typical RL setting where the dialog agent optimizes its policy via interacting with real/simulated users. However, the proposed setups has huge advantages in reproducibility because it does not depend on external user simulation

component. Moreover, our results show that this problem is challenging and that word-level RL falls short.

8.6 Results: Latent Actions or Words?

We have created 6 different variations of latent action dialog models under our LaRL framework. To demonstrate the advantages of LaRL, during the RL training step, we

Model	Var Type	Loss	Integration
Gauss	Gaussian	$\mathcal{L}_{full}$	/
Cat	Categorical	$\mathcal{L}_{full}$	sum
AttnCat	Categorical	$\mathcal{L}_{full}$	attn
LiteGauss	Gaussian	$\mathcal{L}_{lite}$	/
LiteCat	Categorical	$\mathcal{L}_{lite}$	sum
LiteAttnCat	Categorical	$\mathcal{L}_{lite}$	attn

Table 8.1: All proposed variations of LaRL models.

set RL:SL=off for all latent action models, while the baseline word-level RL models are free to tune RL:SL for best performance. For latent variable models, their perplexity is estimated via Monte Carlo $p(\mathbf{x}|\mathbf{c}) \approx \mathbb{E}_{p(\mathbf{z}|\mathbf{c})}[p(\mathbf{x}|\mathbf{z})p(\mathbf{z}|\mathbf{c})]$. For the sake of clarity, this section only compares the best performing latent action models to the best performing word-level models and focuses on the differences between them. A detailed comparison of the 6 latent space configurations is addressed in Section 8.7.

8.6.1 DealOrNoDeal

The baseline system is a hierarchical recurrent encoder-decoder (HRED) model (Serban et al., 2016b) that is tuned to reproduce results from (Lewis et al., 2017). Word-level RL is then used to fine-tune the pre-trained model with RL:SL=4:1. On the other hand, the best performing latent action model is LiteCat. Best models are chosen based on performance on the validation environment.

	PPL	Reward	Agree%	Diversity
Baseline	5.23	3.75	59	109
LiteCat	5.35	2.65	41	58
Baseline +RL	8.23	7.61	86	5
LiteCat +RL	**6.14**	7.27	87	**202**

Table 8.2: Results calculated over the entire test set of DealOrNoDeal. Diversity is measured by the number of unique responses the model used in all scenarios from the test data. Bold-face numbers show statistically significant better results by comparing LiteCat+RL vs. Baseline+RL with p-value < 0.01.

The results are summarized in Table 8.2 and Figure 8.2 shows the LCR curves for the baseline with the two best models plus LiteCat and baseline without RL:SL. From Table 8.2, it appears that the word-level RL baseline performs better than LiteCat in terms of rewards. However, Figure 8.2 shows that the two LaRL models achieve strong task rewards with a much smaller performance drop in language quality (PPL), whereas the word-level model can only increase its task rewards by deviating significantly from natural language.

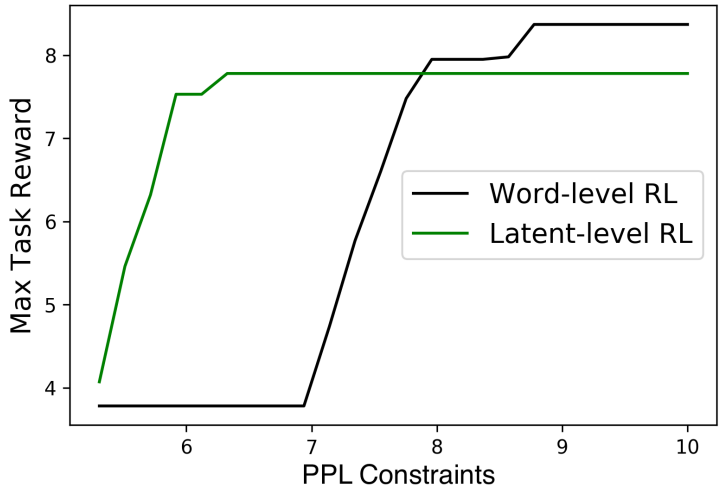

Figure 8.2: LCR curves on DealOrNoDeal dataset.

Closer analysis shows the word-level baseline severely overfits to the user simulator. The caveat is that the word-level models have in fact discovered a loophole in the simulator by insisting on 'hat' and 'ball' several times and the user model eventually yields to agree to the deal. This is reflected in the diversity measure, which is the number of unique responses that a model uses in all 200 testing scenarios. As shown in Figure 8.3, after RL training, the diversity of the baseline model drops to only 5. It is surprising that the agent can achieve high reward with a well-trained HRED user simulator using only 5 unique utterances. On the contrary, LiteCat increases its response diversity after RL training from 58 to 202, suggesting that LiteCat discovers novel discourse-level strategies in order to win the negotiation instead of exploiting local loopholes in the same user simulator. Our qualitative analysis confirms this when we observe that our LiteCat model is able to use multiple strategies in negotiation, e.g. elicit preference question, request different offers, insist on key objects etc. This is shown in Table 8.7 and Table 8.8, which contains example dialogs from word-level and latent-level models.

8.6.2 MultiWoz

For MultiWoz, we reproduce results from (Budzianowski et al., 2018) as the baseline. After RL training, the best LaRL model is LiteAttnCat and the best word-level model is word RL:SL=off. Table 8.3 shows that LiteAttnCat is on par with the baseline in the su-

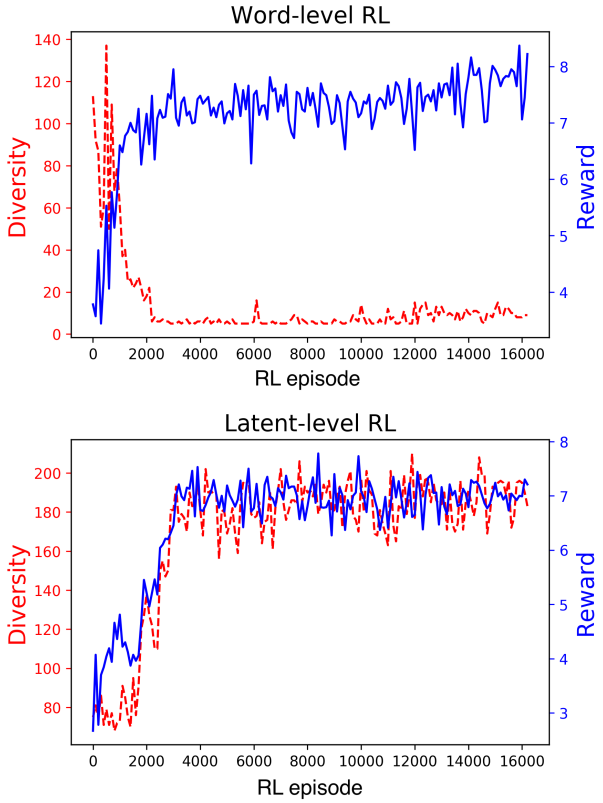

Figure 8.3: Response diversity and task reward learning curve over the course of RL training for both word RL:SL=4:1 (left) and LiteCat (right).

pervised learning step, showing that multivariate categorical latent variables alone are powerful enough to match with continuous hidden representations for modeling dialog actions. For performance after RL training, LiteAttnCat achieves near-human performance in terms of success rate and inform rate, obtaining 18.24% absolute improvement over the MLE-based state-of-the-art (Budzianowski et al., 2018). More importantly, perplexity only slightly increases from 4.05 to 5.22. On the other hand, the word-level RL's success rate also improves to 79%, but the generated responses completely deviate from natural language, increasing perplexity from 3.98 to 17.11 and dropping BLEU from 18.9 to 1.4.

Figure 8.4 shows the LCR curves for MultiWoz, with a trend similar to the previous section: the word-level models can only achieve task reward improvement by sacrificing their response decoder PPL. Figure 8.4 also shows the LCR curve for the baseline trained with RL:SL=100:1, hoping that supervised learning can force the model to conform to natural language. While PPL and BLEU are indeed improved, it also limits final reward performance. The latent-level models, on the contrary, do not suffer from this tradeoff. We also observe that LiteAttnCat consistently outperforms LiteCat on MultiWoz, confirming the effectiveness of Attention Fusion for handling long dialog responses with multiple entities and dialog acts. Lastly, Table 8.4 qualitatively exhibits

	PPL	BLEU	Inform	Success
Human	/	/	90%	82.3%
Baseline	3.98	18.9	71.33%	60.96%
LiteAttnCat	4.05	19.1	67.98%	57.36%
Baseline +RL	17.11	1.4	80.5%	79.07%
LiteAttnCat +RL	**5.22**	**12.8**	**82.78%**	79.2%

Table 8.3: Main results on MultiWoz test set. RL models are chosen based on performance on the validation set. Bold-face numbers indicate that the LiteAttnCat+RL significantly better than the baseline+RL with p-value < 0.01.

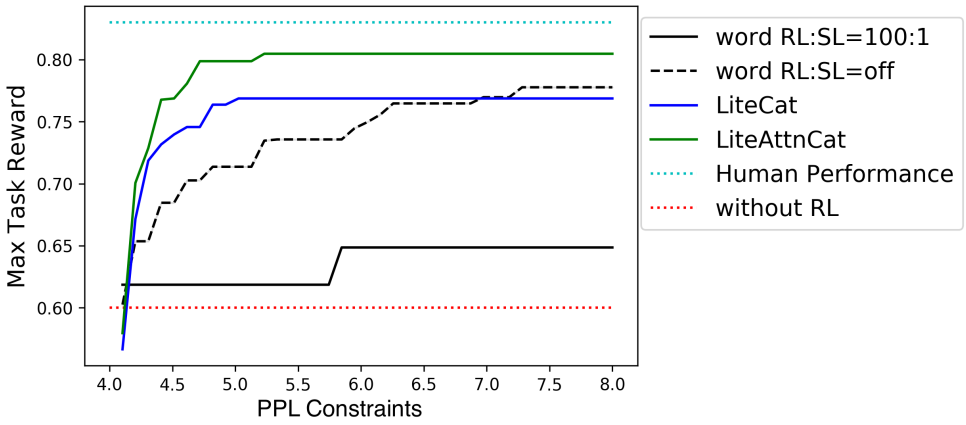

Figure 8.4: LCR curves on the MultiWoz dataset.

the generation differences between the two approaches. The RL:SL=off model learns to continuously output entities to fool the evaluation script for high success rate, whereas LiteCatAttn learns to give more information while maintaining the language quality.

8.7 Model Analysis

We compare the 6 variants of latent action models on DealOrNoDeal and MultiWoz. Table 8.5 shows performance of the models that are pre-trained only with supervised learning. Figure 8.5 shows LCR curves for the 3 models pre-trained with $\mathcal{L}_{lite}$ and fine-tuned with policy gradient reinforcement learning. The following are the main findings based on these results.

$\mathcal{L}_{lite}$ outperforms $\mathcal{L}_{full}$ as a pre-train objective.

Table 8.5 shows that models with $\mathcal{L}_{full}$ fall behind their Lite counterparts on PPL and BLEU. We attribute this to the exposure bias in the latent space, i.e. the decoder is

Context	**Sys** I have [value_count] trains matching your request . Is there a specific day and time you would like to travel? **Usr** I would like to leave on [value_day] and arrive by [value_time].
Model	**Generated Response**
word RL:SL=off	[train_id] is leaving [value_place] on [value_day] on [value_day] on [train_id] [train_id] [value_count] [train_id] leaving ...
word RL:SL=100	[train_id] leaves at [value_time] . would you like me to book you a ticket ?
LiteAttnCat	[train_id] leaves [value_place] at [value_time] and arrives in [value_place] at [value_time]. Would you like me to book that for you ?

Table 8.4: Example responses from baselines and LiteCatAttn on MultiWoz. The baseline system word RL:SL=off deviates from natural language by generating repetitive entities to get higher success rate. On the contrary, LiteAttnCat learns to produce more informative responses while maintaining grammatical correctness.

not trained to consider the discrepancy between the posterior network and actual dialog policy network. Meanwhile, the full models tend to enjoy higher diversity at pre-training, which agrees with the diversity-promoting effect observed in prior research (Zhao et al., 2017b). However, our previous discussion on Figure 8.3 shows that Lite models are able to increase their response diversity in order to win more in negotiation through RL training. This is fundamentally different from diversity in pre-training, since diversity in LaRL is optimized to improve task reward, rather than to better model the original data distribution. Table 8.6 shows the importance of latent space regularization. When β is 0, both LiteCat and LiteGauss reach suboptimal policies with final reward that are much smaller than the regularized versions ($\beta = 0.01$). The reason behind this is that the unregularized pre-trained policy has very low entropy, which prohibits sufficient exploration in the RL stage.

Categorical latent actions outperform Gaussian latent actions.

Models with discrete actions consistently outperform models with Gaussian ones. This is surprising since continuously distributed representations are a key reason for the success of deep learning in natural language processing. Our finding suggests that (1) multivariate categorical distributions are powerful enough to model complex natural dialog responses semantics, and can achieve on par results with Gaussian or non-stochastic continuous representations. (2) categorical variables are a better choice to serve as action spaces for reinforcement learning. Figure 8.5 shows that Lite(Attn)Cat easily achieves strong rewards while LiteGauss struggles to improve its reward. Also, applying REINFORCE on Gaussian latent actions is unstable and often leads to model divergence. We suspect the reason for this is the unbounded nature of continuous latent space: RL

Deal	PPL	Reward	Agree%	Diversity
Baseline	3.23	3.75	59	109
Gauss	110K	2.71	43	176
LiteGauss	5.35	4.48	65	91
Cat	80.41	3.9	62	115
AttnCat	118.3	3.23	51	145
LiteCat	5.35	2.67	41	58
LiteAttnCat	5.25	3.69	52	75
MultiWoz	PPL	BLEU	Inform%	Succ%
Baseline	3.98	18.9	71.33	60.96
Gauss	712.3	7.54	60.5	23.0
LiteGauss	4.06	19.3	56.46	48.06
Cat	7.07	13.7	54.15	42.04
AttnCat	12.01	12.6	63.9	45.8
LiteCat	4.10	19.1	61.56	49.15
LiteAttnCat	4.05	19.1	67.97	57.36

Table 8.5: Comparison of 6 model variants with only supervised learning training.

β	0.0	0.01	β	0.0	0.01
LiteCat	4.23	**7.27**	LiteGauss	4.83	**6.67**

Table 8.6: Average rewards over the entire test environments on DealOrNoDeal with various β. The differences are statistically significant with $p < 0.01$.

exploration in the continuous space may lead to areas in the manifold that are not covered in supervised training, which causes undefined decoder behavior given $\mathbf{z}$ in these unknown areas.

8.8 Conclusion and Discussion

In conclusion, this chapter proposes a latent variable action space for RL in E2E dialog agents. We present a general framework with a regularized ELBO objective and attention fusion for discrete variables. The methods are assessed on two dialog tasks and analyzed using the proposed LCR curve. Results show our models achieve superior performance and create a new state-of-the-art success rate on MultiWoz. Results show that LaRL is significantly more effective than word-level RL for learning dialog policies and it does not lead to incomprehensible language generation. Our models achieve 18.2% absolute improvement over the previous state-of-the-art on MultiWoz and discover novel and diverse negotiation strategies on DealOrNoDeal. Besides strong empirical improvement, our model analysis reveals novel insights, e.g. it is crucial to reduce the exposure bias in the latent action space and discrete latent actions are more suitable than continuous ones to serve as action spaces for RL dialog agents. These ex-

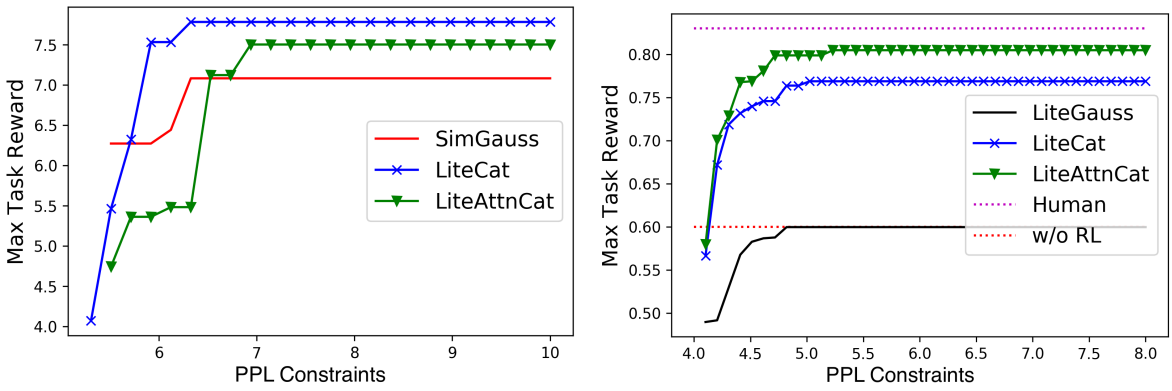

Figure 8.5: LCR curves on DealOrNoDeal and MultiWoz. Models with $\mathcal{L}_{full}$ are not included because their PPLs are too poor to compare to the Lite models.

tensive analyses enable us to gain insight on how to properly train latent variables that can serve as the action spaces for dialog agents. This work is situated in the approach concerning practical latent variables in dialog agents, being able to create action abstraction in an unsupervised manner. We believe that our findings are a first step that in this promising research direction and can bring decades of reinforcement learning research for classic frame-based dialog management to the development of better end-to-end dialog models.

Scenario 1
1 book, 1 hat and 3 balls available.
Value Function
Agent: book=1, hat=6, ball=1.
User: book=1, hat=6, ball=1.
Dialog
Agent: I'd like the hat and 2 balls.
User: Deal.
Agent: SELECTION
Outcome: Agreement
Agent: 1 hat and 2 balls; User: 1 book and 1 ball
Agent reward = 8; User reward = 2
Scenario 2
4 books, 1 hat and 1 ball available.
Value Function
Agent: book=0, hat=7, ball=3;
User: book=1, hat=6, ball=0.
Dialog
User: I can offer you 1 hat and 2 books.
Agent: I need the hat and ball.
User: That won't work for me. I can offer you one book and the hat.
Agent: I need the hat and ball.
User: I have to have the hat.
Agent: I need the hat and ball.
User: Okay.
Agent: SELECTION
Outcome: Agreement
Agent: 1 hat and 1 ball; User: 4 books
Agent reward = 10; User reward = 4

Table 8.7: Example dialogs between baseline with the user model. Agent is trained with word-level policy gradient and the user is a supervised pre-trained model.

Scenario 1
2 books, 2 hats and 1 ball available.
Value Function
Agent: book=3, hat=0, ball=4;
User: book=2, hat=3, ball=0.
Dialog
Agent: Can i have the books and one ball?
User: I would like the books and the hats then.
Agent: I would like the ball and books.
User: Deal.
Agent: SELECTION
Outcome: Agreement
Agent: 2 books and 1 ball
User: 2 hats
Agent reward = 10, User reward = 6
Scenario 2
2 books, 1 hat and 3 ball available.
Value Function
Agent: book=0, hat=4, ball=2;
User: book=3, hat=1, ball=1.
Dialog
Agent: I would like the hat and balls.
User: I cannot accept that. I need the hat and books.
Agent: Can I have the hat and balls, you get the rest?
User: Okay, deal.
Agent: SELECTION.
Outcome: Agreement
Agent: 1 hat and 3 ball
User: 2 books
Agent reward = 10, Simulator reward = 6

Table 8.8: Example dialogs between LiteCat and the user model. Agent is trained with latent-level policy gradient and the user is a supervised pre-trained model.

Chapter 9

Conclusions and Future Work

9.1 Overview

In summary, this dissertation presents a new family of E2E dialog systems based on the proposed latent actions and provides a complete framework starting from problem definition, to learning algorithms, and from high-level motivations to complete implementations that solve real-world challenges. With one unified latent action framework, we show that it can substantially improve current state-of-the-art E2E dialog systems for four challenging problems: diversity, interpretability, transferability and strategy optimization. Table 9.1 shows a summary on how our solutions can seamlessly fit into the proposed latent action E2E framework and can be understood as special cases of the overall architecture.

Challenge	Action Type	Var Type	Recognition Network	Learning Objective
Diversity	Partial	Continuous	$q(\mathbf{z}\|\mathbf{x}, \mathbf{c})$	ELBO + Dialog Acts
Interpretability	Partial	Discrete	$q(\mathbf{z}\|\mathbf{x})$	ELBO or Distributional Semantics
Transferability	Full	Continuous	$q(\mathbf{z}\|\mathbf{x})$ and $q(\mathbf{z}\|a)$	ELBO + Seed Matching
Strategy	Full	Both	$q(\mathbf{z}\|\mathbf{x}, \mathbf{c})$ or $q(\mathbf{z}\|\mathbf{c})$	ELBO + RL

Table 9.1: A summary of the proposed latent actions for solving the real-world challenges of current E2E dialog systems.

Moreover, several intriguing insights can by drawn from the results of this thesis. First of all, in order to scale up dialog systems to more complicated domains, it is extremely useful for developing dialog systems that can be trained end-to-end with no constraints from hand designed representation. The models proposed in this work are able to easily create dialog models in multiple datasets, ranging from task-oriented (i.e. MultiWoz, Stanford Multi-domain, DealOrNoDeal) to open-domain chatting (i.e. Switchboard, Daily Dialog) without the needs for heavy hand engineering. Whereas this result is infeasible to achieve with traditional modular-based dialog systems. Therefore, we focus on developing unsupervised methods (e.g. training techniques in Chap-

ter 6) or semi-supervised methods (e.g. seed responses in Chapter 7) so that no data or only a subset of the data needs to be manually annotated with linguistic labels during training. And at testing time, our models take only the raw dialog context as input without demanding external labels.

Second, having an explicit abstraction of actions, i.e. latent actions, can naturally solve many real-world dialog challenges for E2E dialog models. We argue that this is because the latent action architecture matches closer to the actual underlying process for a human to respond in a conversation. Experiment results from this study show that the inductive bias on network structure is beneficial to improve the generalization ability of the resulting systems. Other fields of study, e.g. the convolution filters in Convolutional Neural Networks (LeCun et al., 1995), also suggest that specialized architectures that are inspired by the natural process are important to create intelligent systems that can learn faster and better. Therefore, it is a promising direction to develop future E2E dialog models that are designed to resemble closer to the actual dialog processing process. On the other hand, one may argue that recent success in pre-training very large neural networks that are extremely flexible (e.g. BERT Transformers (Devlin et al., 2018)) is suggesting a opposite directions. We believe that pre-training very large models on gigantic dataset is in fact a promising way to discover these specialized network architecture automatically from data if such closely related data is available. This may leads to a novel approach to establish useful network architectures in a more data-driven fashion.

Third, the introduction of latent variables in an E2E neural networks has created an new opportunity for encoding additional knowledge into an E2E models without breaking their scalability. For example, the Knowledge-guided CVAE in Chapter 5 is a way to use linguistic knowledge to guide what information should be kept in the latent action space. The seed response annotation in Chapter 7 is another example to control the latent space via prior knowledge about utterances from different domains. We impose these knowledge via auxiliary loss directly at the latent variables instead of the final response outputs, which encourages further division of responsibilities and abstractions. Furthermore, the success of discrete latent actions in Chapter 8 demonstrates a different types of knowledge encoding where we discretize the action space so that it becomes safer for the agent to do exploration in the reinforcement learning stage. All of the above examples show that combining latent variables with E2E neural networks is an effective approach to integrate prior knowledge into deep learning-based systems.

At last, the advantages of the proposed latent action E2E dialog systems are summarized in the following. It is superior than hand-crafted frame-based dialog systems in terms of:

- Our framework can model complex dialog domain without constraints to hand-crafted representations and achieve much more natural and intelligent conversations with human users.
- Our framework can scale to larger dataset and can be trained directly on conversational data.

ALso, the latent action framework surpasses current E2E dialog systems in terms of:

- Our framework can separate the intention-level decision-making from word-level response generation as a hierarchical generation process. Discrete latent action can be used to further improve model explainability, which is infeasible for current E2E systems.
- Our framework can enable injection of linguistic labels (e.g. dialog acts) to guide the learning of models.
- Our framework can allow domain adaptation at utterance level via cross-domain latent actions, which in turn achieving a zero-shot generalization under certain conditions (i.e. similar discourse pattern).
- Our framework is easier to fine tune dialog strategy via policy gradient from a pre-trained model and achieve substantially better converged performance.

9.2 Contributions by Chapters

In this thesis, each chapter makes the following contributions.

Chapter 3 lays out the base of this thesis and highlights the four basic research questions that we strive to answer: (1) how to define latent actions? (2) how to evaluate latent actions (3) how to learn latent actions and (4) how to use latent actions. The rest of the chapter provides detailed answers to the first two fundamental questions, including the definition of full/partial latent actions and the types of studied random variables and an overview of how latent actions can be used.

Chapter 4 describes three novel learning algorithms for improving the performance of variational autoencoders for text modeling, including bag-of-word loss, response selection loss, batch prior regularization with and without autoregressive prior. This chapter also conducts detailed experiments on two datasets, Penn Treebank and MultiWoz to assess the performance of the proposed methods compared to previous baselines. A novel discriminator-based evaluation method is proposed in the experiment part to more robustly tune VAE models that balance between the trade-off between inference and generation.

Chapter 5 highlights the one-to-many nature of open-domain chatting. Besides applying the proposed partial continuous latent actions (aka CVAE) to solve the problem, an additional knowledge-guided CVAE is proposed to further improve the performance on response precision. Further, a novel generalized recall/precision evaluation metric is designed to explicitly test an open-domain response system's ability to generate diverse yet appropriate responses. Experiment results show that the proposed method achieves significant improvement over baselines and confirms that CVAE/kgCVAE is able to model the one-to-many properties in open-domain chatting. This work was published at ACL 2017 (Zhao et al., 2017b).

Chapter 6 first defines the conditions that make a latent action interpretable to human users. Then besides using the proposed BPR model to learn latent action via auto-encoding, an alternative context based variational skip thought is proposed to learn distributional semantic style latent actions. Also, an integration training objective is

proposed to combine the resulting latent action into any encoder-decoder E2E dialog models. Experiments are conducted on three different datasets and results validate the effectiveness of the proposed approach, resulting in the first step towards explainable E2E dialog models. This work was published at ACL 2018 (Zhao et al., 2018).

Chapter 7 proposes a new task named Zero Shot Dialog Generation for advancing E2E dialog models to generalize to new domains with minimal data. A novel action matching algorithm is proposed to learn latent actions that are aligned cross-domain. SimDial, a new dataset is also developed to benchmark the performance of ZSDG systems. The proposed algorithms significantly improve the zero-shot performance for E2E models and also suggest a promising future research venue. This work was published at SIGDIAL 2018 and awarded Best Paper Award (Zhao and Eskenazi, 2018).

Chapter 8 extends the work to reinforcement learning by first proposing to use latent actions as the action space for reinforcement learning optimization. A novel variant of ELBO, LiteELBO is proposed to mitigate the exposure bias problem at the testing time. Moreover, a novel evaluation method named Language Constrained Reward (LCR) curve is developed to visually quantify the trade-off between task-level rewards and language generation for various approaches. Extensive experiments on two datasets show that Latent Action Reinforcement Learning results into much more stable and better dialog strategy learning compared the standard word-level policy gradient fine-tuning for E2E dialog models. The results also discover that discrete latent actions are far more suitable than continuous latent actions when used as a reinforcement learning action space. This work will be published at NAACL 2019 (Zhao et al., 2019).

9.3 Open Source Software

For reproducibility and a more open environment of scientific research, we also release code and data at GitHub for all the methods that are developed in this thesis. The code repositories can be found at:

1. Chapter 5: `https://github.com/snakeztc/NeuralDialog-CVAE`
2. Chapter 6: `https://github.com/snakeztc/NeuralDialog-LAED`
3. Chapter 7: `https://github.com/snakeztc/NeuralDialog-ZSDG`
4. SimDial: `https://github.com/snakeztc/SimDial`
5. Chapter 8 `https://github.com/snakeztc/NeuralDialog-LaRL`

9.4 Summary of Comparative Results

We stress on both creating novel evaluation metrics and obtaining new state-of-the-art results compared to prior research when existing evaluation metrics are convincing. We argue that both aspects are important, especially considering the current state of dialog research. In the last five years, dialog research is undertaking a fundamental transformation where the entire field of research is advancing at a unprecedented speed.

As a result, many of previous benchmark tasks, e.g. intent classification and dialog state tracking, have become less relevant in the setting of the emerging E2E dialog models. In fact, many of the popular evaluation metrics for response generation systems are borrowed from other fields of research and some of them are questionable and require improvement. One example is BLEU (Papineni et al., 2002), which is adapted from the machine translation community. Research has shown that BLEU is sub-optimal to evaluate open-domain chatting system (Liu et al., 2016). Therefore, this thesis strives for finding new evaluation metrics that are designed for dialog systems, which in turns creating a foundation for future research. In the meantime, our proposed methods are compared to prior art as much as possible when qualified evaluation metrics exist. The comparative results are summarized as follows:

Novel Evaluation Metrics

In Chapter 5, we proposes a novel generalized precision/recall metric that can automatically assess an open-domain chatting systems in terms of both response appropriateness and diversity. The multiple references are validated by two human experts. We showed that the proposed CVAE and kgCVAE are able to achieve the state-of-the-art results on Switchboard Corpus (Godfrey et al., 1992) at the time of publication, comparing to the best model at that time (Serban et al., 2017b). After that, there are many research groups have begun to use our proposed generalized precision and recall for evaluating their response generation systems (Yang et al., 2017a; Gu et al., 2018; Gao et al., 2019; Liu et al., 2019).

In Chapter 6, we propose a novel task that aims to improve the interpretability of a neural dialog system. To our best knowledge, it was the first step in this direction. Despite of the difficulty to automatically quantify the performance of interpretability for a neural dialog system, we propose homogeneity and action prediction accuracy to assess the proposed methods on two popular dialog chatting datasets, including Daily Dialog (Li et al., 2017) and Switchboard (Godfrey et al., 1992). We also use human evaluation based on expert and crowd annotations to test the performance on Stanford Multi-domain dataset.

In Chapter 8 we propose a novel language constrained reward (LCR) curve to visualize the trade-off between language quality versus dialog-level rewards. The experiment results show that LCR curves are more informative compared to a single point score such as negotiation success rate. We believe this metric can be beneficial for future research that centered around reinforcement learning and E2E text generation system.

Comparative Results using Existing Metrics

In the meantime, our proposed methods are able to achieve the state-of-the-art results in a number of existing metrics.

In Chapter 4 the proposed anti-posterior collapse techniques enable a text VAE to achieve better inference and generation performance compared to the best text VAEs at the time of publication on PennTree Bank (PTB) language modeling dataset (Bowman

et al., 2015). We used three types of evaluation metrics, including ELBO (reconstruction perplexity and KL distance), classifier-based evaluation for inference ability and discriminator-based evaluation for generation quality. We did not provide comparison in terms of perplexity with other state-of-the-art language modeling results on PTB because it is a known issue that latent variable model can only provide a lowerbound on the likelihood and cannot provide exact likelihood on the observed data, which prohibits us to compute a comparable perplexity.

In Chapter 7 our systems are able to achieve better performance compared to the state-of-the-art generative task-oriented E2E dialog models on Stanford Multi-domain Dialog dataset in terms of BLEU and entity F-1 in non-zero-shot setting (Eric and Manning, 2017a). The proposed zero-shot dialog generation is a new task, so that our results set the baseline for future research to tackle the challenge of zero-shot dialog generation.

In Chapter 8 our systems are able to achieve better results on DealOrNoDeal compared to the original paper with their single point metrics, e.g. Reward and Agreement Rate (Lewis et al., 2017). We further show that the proposed LCR curve is a more meaningful assessment method, because it provides a quantitative measurement for the relationship between language quality and dialog-level reward. On the MultiWoz dataset, the proposed latent RL systems are able to achieve 18.3% absolute improvement on success rate and 14.8% absolute improvement on inform rate, compared to the previous state-of-the-art systems (Budzianowski et al., 2018).

In summary, we provide solid comparison between the proposed latent action dialog systems with the previous best E2E dialog models and show statistically significant improvement. Moreover, we propose novel evaluation metrics that are designed to better quantify dialog system performance, which we believe will create a healthier environment for future E2E dialog research.

9.5 Future Research Directions

The presented framework also suggests many promising future research directions. Some of them are:

- **Latent Variables beyond Dialog Actions**. In this work, we mainly focus on creating latent variables that correspond to the actions of dialog agents. There are a number of other important variables that can be also modeled as latent variables. These variables are also often not observed in the raw dialog corpora so that they have to be inferred. For example, the persona of the system or the users can be modelled as a dialog-level latent variable that influences the topics and style of the conversation. Another example can be the users' mental state, including their satisfaction about the systems, sentiment, and goals, is another essential factor that if it is appropriately captured, can significantly improve the capability of automated dialog systems. Meanwhile, studying these variables can be more challenging because it is unclear what kind of unsupervised learning signals can enforce the latent variable to focus on these aspects. Perhaps semi-supervised learning or extensive inductive bias is needed to make the learning meaningful. That said, there

can still exist novel yet to be found learning objectives enabling the models to create levels of abstractions, where these critical aspects can be discovered in an unsupervised manner.

- **Disentanglement and Compositionality**. Although the proposed multivariate latent code are designed to be disentangled and should capture the compositionality of natural language, there is still much room for improvement. First of all, there is no standard dataset or evaluation metric yet to quantify the ability of neural dialog models to handle compositionality. This metric needs to test from both inference and generation perspectives. That is, for inference, can the model generalize to understand a combination of two smaller utterances; and for the generation, can the model generalize to output responses that combine smaller utterances observed in the training data. Besides than that, better algorithms and models are needed to train models in a way that encourages compositional behavior over simply memorizing the pattern.
- **Better Modeling of Propositional Content**. So far the proposed latent action has been mainly focused on modeling the intents and has not tried to explicitly model the content in the response, e.g. fine-grained entities that are mentioned. These entities are extremely important for task-based dialog systems, because they need to use these entities to search in external databases or back-end API lookup. There remains many interesting research directions on how to encode these entities into the latent actions because they are challenging for a number of reasons. First, the vocabulary of entities are huge and there is high chance for encountering out-of-vocabulary entities at testing time. How to develop a OOV robust representation that can allow the latent action to encode the entity information while staying compact and low dimension remains to be solved.
- **Quantify Model Uncertainty**. Measuring uncertainty is a crucial task for any intelligent agent. Uncertainty measure can contain detecting out-of-domain user utterances, systems' confidence in its current decisions, and inherited uncertainty of the data itself. Quantifying these aspects are advantageous because it can improve the robustness of a dialog agent at the testing, improve its learning speed since it can actively choose to gather data in the most uncertain situations etc. Latent variable provides a natural solution to these problems because it is often equipped with uncertainty measure by itself, such as the variance for continuous variables and entropy for discrete variables. Yet these features are under-explored in this thesis and promising research can be done to fully utilize Bayesian learning to advance systems performance in uncertainty measurement.
- **Extension to End-to-end Retrieval Dialog Systems**. This thesis is dedicated to generation-based dialog systems due to its potential to generalize to new response and complex domains. In the mean time, retrieval-based dialog models is another important class of systems that is widely deployed in real world applications. For a retrieval-based system, it also suffers from many challenges that can be improved via latent action learning. For example, a retrieval dialog systems

would rank many similar responses high given a context because they share similar words and often should belong to the same latent action. Then if the goal is to extract N-best responses with diverse intents, it becomes challenging to extract the N pivots responses from the full response ranking. Given our knowledge from this thesis, it is natural to use latent action to separate the intentions from lexical variations in the ranking and enable the models to output N-best responses easily. Moreover, hybrid dialog systems that combine generation based and retrieval based dialog systems have become increasingly powerful. One can think of the retrieved response as a non-parametric representation of the "latent action", which can be further fine-tuned by the generation based decoder. How would this approach compared to the parametric latent action approach proposed in this thesis? These topics all suggest fruitful research directions.

- **Extension to Other Text Generation Tasks**. Beside dialog systems, there are many other NLP text generation tasks that require long-term planning and abstraction over high-level ideas, such as story generation, caption generation, document summarization etc. We believe the methods proposed in this thesis can also benefit research in these related areas.

Bibliography

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207* . 2

Felix Vsevolodovich Agakov. 2005. *Variational Information Maximization in Stochastic Environments*. Ph.D. thesis, University of Edinburgh. 6.3.4

James F Allen, Donna K Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent. 2001. Toward conversational human-computer interaction. *AI magazine* 22(4):27. 1.1

John Austin. 1962. How to do things with words. . (document), 2.1.1

John Langshaw Austin. 1975. *How to do things with words*. Oxford university press. 6.3.2

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* . 2.2.1

Rafael E Banchs and Haizhou Li. 2012. Iris: a chat-oriented dialogue system based on the vector space model. In *Proceedings of the ACL 2012 System Demonstrations*. Association for Computational Linguistics, pages 37–42. 2.1.3

Ankur Bapna, Gokhan Tur, Dilek Hakkani-Tur, and Larry Heck. 2017. Towards zero-shot frame semantic parsing for domain scaling. *arXiv preprint arXiv:1707.02363* . 2.2.3, 7.2

Andrew G Barto, Richard S Sutton, and Christopher JCH Watkins. 1989. Learning and sequential decision making. In *Learning and computational neuroscience*. Citeseer. 1.1

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python*. " O'Reilly Media, Inc.". 5.3.1

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research* 3(Jan):993–1022. 2.1.4, 5.1, 6.2

Dan Bohus, Antoine Raux, Thomas K Harris, Maxine Eskenazi, and Alexander I Rudnicky. 2007. Olympus: an open-source framework for conversational spoken language interface research. In *Proceedings of the workshop on bridging the gap: Academic and industrial research in dialog technologies*. Association for Computational Linguistics, pages 32–39. 6.1

Dan Bohus and Alexander I Rudnicky. 2003. Ravenclaw: Dialog management using hierarchical task decomposition and an expectation agenda. *Computer Speech and Lan-*

guage . 2.1.2

Antoine Bordes and Jason Weston. 2016. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683* . 2.1.3

Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. 2015. Generating sentences from a continuous space. *arXiv preprint arXiv:1511.06349* . 2.2.2, 4, 4.1.2, 4.2.1, 4.2.2, 4.3, 4.3, 4.4, 4.5.1, 5.1.1, 6.1, 6.2, 6.4, 6.4.1, 9.4

Paweł Budzianowski, Tsung-Hsien Wen, Bo-Hsiang Tseng, Iñigo Casanueva, Stefan Ultes, Osman Ramadan, and Milica Gasic. 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. 3.2, 4.4.2, 8.1, 8.5.2, 8.6.2, 8.6.2, 9.4

Kris Cao and Stephen Clark. 2017. Latent variable dialogue models and their diversity. *arXiv preprint arXiv:1702.05962* . 2.1.4, 6.2

Boxing Chen and Colin Cherry. 2014. A systematic comparison of smoothing techniques for sentence-level bleu. *ACL 2014* page 362. 1

Xi Chen, Diederik P Kingma, Tim Salimans, Yan Duan, Prafulla Dhariwal, John Schulman, Ilya Sutskever, and Pieter Abbeel. 2016a. Variational lossy autoencoder. *arXiv preprint arXiv:1611.02731* . 6.2

Yun-Nung Chen, Dilek Hakkani-Tür, and Xiaodong He. 2016b. Zero-shot learning of intent embeddings for expansion by convolutional deep structured semantic models. In *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*. IEEE, pages 6045–6049. 2.2.3, 7.2

Yun-Nung Chen, William Yang Wang, and Alexander I Rudnicky. 2013. Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*. IEEE, pages 120–125. 8.1

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* . 1.1, 2.1.4, 2.2.1, 3.2, 5.1, 6.1, 7.1, 7.4.3, 8.3

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. 2014. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555* . 5.2.1, 6.4.1

Herbert H Clark. 1996. *Using language*. Cambridge university press. 2.1.1

Herbert H Clark, Susan E Brennan, et al. 1991. Grounding in communication. *Perspectives on socially shared cognition* 13(1991):127–149. 2.1.1

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*. ACM, pages 160–167. 7.4.2

Pavel Curtis. 1992. Mudding: Social phenomena in text-based virtual realities. *High noon on the electronic frontier: Conceptual issues in cyberspace* pages 347–374. 2.2.4

Abhishek Das, Satwik Kottur, José MF Moura, Stefan Lee, and Dhruv Batra. 2017. Learning cooperative visual dialog agents with deep reinforcement learning. In *Computer Vision (ICCV), 2017 IEEE International Conference on*. IEEE, pages 2970–2979. 8, 8.1, 8.2, 8.3, 8.4.3

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* . 9.1

Bhuwan Dhingra, Lihong Li, Xiujun Li, Jianfeng Gao, Yun-Nung Chen, Faisal Ahmed, and Li Deng. 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)*. volume 1, pages 484–495. 2.1.2, 2.1.4, 8.3

Jesse Dodge, Andreea Gane, Xiang Zhang, Antoine Bordes, Sumit Chopra, Alexander Miller, Arthur Szlam, and Jason Weston. 2015. Evaluating prerequisite qualities for learning end-to-end dialog systems. *arXiv preprint arXiv:1511.06931* . 7.5.1, 7.5.3

Yan Duan, Marcin Andrychowicz, Bradly Stadie, Jonathan Ho, Jonas Schneider, Ilya Sutskever, Pieter Abbeel, and Wojciech Zaremba. 2017. One-shot imitation learning. *arXiv preprint arXiv:1703.07326* . 2.2.3, 7.2

Hady Elsahar, Christophe Gravier, and Frederique Laforest. 2018. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. *arXiv preprint arXiv:1802.06842* . 7.2, 7.4.3

Mihail Eric and Christopher D Manning. 2017a. A copy-augmented sequence-to-sequence architecture gives good performance on task-oriented dialogue. *arXiv preprint arXiv:1701.04024* . 2.1.4, 2.2.1, 7.4.3, 7.6, 9.4

Mihail Eric and Christopher D Manning. 2017b. Key-value retrieval networks for task-oriented dialogue. *arXiv preprint arXiv:1705.05414* . 2.1.4, 6.4, 7.1, 7.5.2, 7.6

Akiko Eriguchi, Kazuma Hashimoto, and Yoshimasa Tsuruoka. 2016. Tree-to-sequence attentional neural machine translation. *arXiv preprint arXiv:1603.06075* . 2.2.1

Arash Eshghi, Igor Shalyminov, and Oliver Lemon. 2017. Bootstrapping incremental dialogue systems from minimal data: the generalisation power of dialogue grammars. *arXiv preprint arXiv:1709.07858* . 7.5.1

Maxine Eskenazi, Shikib Mehri, Evgeniia Razumovskaia, and Tiancheng Zhao. 2019. Beyond turing: Intelligent agents centered on the user. *arXiv preprint arXiv:1901.06613* . 3.4.4

Gabriel Forgues, Joelle Pineau, Jean-Marie Larchevêque, and Réal Tremblay. 2014. Bootstrapping dialog systems with word embeddings. In *NIPS, Modern Machine Learning and Natural Language Processing Workshop*. 2

Xiang Gao, Sungjin Lee, Yizhe Zhang, Chris Brockett, Michel Galley, Jianfeng Gao, and

Bill Dolan. 2019. Jointly optimizing diversity and relevance in neural response generation. *arXiv preprint arXiv:1902.11205* . 5.2.4, 9.4

M Gašić, F Jurčíček, Simon Keizer, François Mairesse, Blaise Thomson, Kai Yu, and Steve Young. 2010. Gaussian processes for fast policy optimisation of pomdp-based dialogue managers. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*. Association for Computational Linguistics, pages 201–204. 2.2.4, 6.1

Milica Gasic and Steve Young. 2014. Gaussian processes for pomdp-based dialogue manager optimization. *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 22(1):28–40. 2.2.3, 7.2, 8.2

James R Glass, Timothy J Hazen, and I Lee Hetherington. 1999. Real-time telephone-based speech recognition in the jupiter domain. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*. IEEE, volume 1, pages 61–64. 2.1.2

John J Godfrey and Edward Holliman. 1997. Switchboard-1 release 2. *Linguistic Data Consortium, Philadelphia* . 5.3.1, 6.4

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*. IEEE, volume 1, pages 517–520. 3.2, 9.4

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. pages 2672–2680. 4.4.1

Arthur C Graesser, Shulan Lu, George Tanner Jackson, Heather Hite Mitchell, Mathew Ventura, Andrew Olney, and Max M Louwerse. 2004. Autotutor: A tutor with dialogue in natural language. *Behavior Research Methods, Instruments, & Computers* 36(2):180–192. 2.1.3

Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. 2004. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research* 5(Nov):1471–1530. 8.3

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor OK Li. 2016. Incorporating copying mechanism in sequence-to-sequence learning. *arXiv preprint arXiv:1603.06393* . 2.2.1, 7.2, 7.4.3

Xiaodong Gu, Kyunghyun Cho, Jungwoo Ha, and Sunghun Kim. 2018. Dialogwae: Multimodal response generation with conditional wasserstein auto-encoder. *arXiv preprint arXiv:1805.12352* . 4.3, 5.2.4, 9.4

Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *Transactions of the Association of Computational Linguistics* 6:437–450. 2.1.5

Zellig S Harris. 1954. Distributional structure. *Word* 10(2-3):146–162. 3.4.3, 6.3.2

Matthew Hausknecht and Peter Stone. 2015. Deep recurrent q-learning for partially observable mdps. *arXiv preprint arXiv:1507.06527* . 2.2.4

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. 2018. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. pages 2333–2343. 8.1

Mikael Henaff, Jason Weston, Arthur Szlam, Antoine Bordes, and Yann LeCun. 2016. Tracking the world state with recurrent entity networks. *arXiv preprint arXiv:1612.03969* . 2.1.4

Matthew Henderson, Blaise Thomson, and Steve Young. 2014. 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. 8.2

Felix Hill, Kyunghyun Cho, and Anna Korhonen. 2016. Learning distributed representations of sentences from unlabelled data. *arXiv preprint arXiv:1602.03483* . 4.5.2, 6.2, 6.3.2

Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. 2012. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine* 29(6):82–97. 1.1

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780. 2.2.4, 7.4.3

Eduard H Hovy. 1990. Pragmatics and natural language generation. *Artificial Intelligence* 43(2):153–197. 1.1

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning*. pages 1587–1596. 6.3.3, 6.3.3

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144* . 4.1.1, 4.1.1, 4.1.2, 6.1, 6.2, 8.4.2

Jiwoon Jeon, W Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, pages 84–90. 2.1.3

Shafiq Joty, Giuseppe Carenini, and Chin-Yew Lin. 2011. Unsupervised modeling of dialog acts in asynchronous conversations. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*. 3, page 1807. 1.1

Leslie Pack Kaelbling, Michael L Littman, and Andrew W Moore. 1996. Reinforcement learning: A survey. *Journal of artificial intelligence research* 4:237–285. 8.1

Henry A Kautz and James F Allen. 1986. Generalized plan recognition. In *AAAI*. 3237, page 5. 2.1.1

Yoon Kim, Kelly Zhang, Alexander M Rush, Yann LeCun, et al. 2017. Adversarially regularized autoencoders for generating discrete structures. *arXiv preprint arXiv:1706.04223* . 4.2.2, 6.2

Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* . 4.5.1, 5.3.3, 6.4.1

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* . 2.2.2, 4.1.1, 4.1.1, 4.1.2, 5.2.1, 5.4.3, 6.3.1, 7.4.2, 8.4.2

Ryan Kiros, Yukun Zhu, Ruslan R Salakhutdinov, Richard Zemel, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Skip-thought vectors. In *Advances in neural information processing systems*. pages 3294–3302. 3.4.3, 6.1, 6.2, 6.3.2

Satwik Kottur, José Moura, Stefan Lee, and Dhruv Batra. 2017. Natural language does not emerge naturallyin multi-agent dialog. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. pages 2962–2967. 8.1

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360* . 1.1

John Langford and Tong Zhang. 2008. The epoch-greedy algorithm for multi-armed bandits with side information. In *Advances in neural information processing systems*. pages 817–824. 8.5.2

Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. 2008. Zero-data learning of new tasks. In *AAAI*. volume 1, page 3. 2.2.3, 7.2

Staffan Larsson, Peter Bohlin, Johan Bos, and David Traum. 1999. Trindikit manual. Technical report, Tech. rept. Deliverable. 2.1.2

Staffan Larsson and David R Traum. 2000. Information state and dialogue management in the trindi dialogue move engine toolkit. *Natural language engineering* 6(3&4):323–340. 2.1.1, 6.1

Yann LeCun, Yoshua Bengio, et al. 1995. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks* 3361(10):1995. 9.1

Cheongjae Lee, Sangkeun Jung, Seokhwan Kim, and Gary Geunbae Lee. 2009. Example-based dialog modeling for practical multi-domain dialog system. *Speech Communication* 51(5):466–484. 2.1.3

Sungjin Lee. 2013. Structured discriminative model for dialog state tracking. In *Proceedings of the SIGDIAL 2013 Conference*. pages 442–451. 8.2

Wenqiang Lei, Xisen Jin, Min-Yen Kan, Zhaochun Ren, Xiangnan He, and Dawei Yin. 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)*. volume 1, pages 1437–1447. 2.1.4

Mike Lewis, Denis Yarats, Yann N Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning for negotiation dialogues. *arXiv preprint arXiv:1706.05125* . 8, 8.1, 8.2, 8.3, 8.5.1, 8.6.1, 9.4

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2015a. A diversity-promoting objective function for neural conversation models. *arXiv preprint arXiv:1510.03055* . 2.1.4, 5.1, 1

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016a. A persona-based neural conversation model. *arXiv preprint arXiv:1603.06155* . 2.1.4, 5.1, 6.1, 6.2

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. 2015b. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057* . 7.4.3

Jiwei Li, Will Monroe, Alan Ritter, and Dan Jurafsky. 2016b. Deep reinforcement learning for dialogue generation. *arXiv preprint arXiv:1606.01541* . 2.1.4, 5.1, 8.1, 8.2, 8.3

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957* . 6.4, 9.4

Diane J Litman and James F Allen. 1987. A plan recognition model for subdialogues in conversations. *Cognitive science* 11(2):163–200. 1.1, 2.1.1, 5.2.2

Bing Liu and Ian Lane. 2017. An end-to-end trainable neural network model with belief tracking for task-oriented dialog. *arXiv preprint arXiv:1708.05956* . 8.2

Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023* . 3.4.4, 5.2.4, 9.4

Xiaodong Liu, Jianfeng Gao, Asli Celikyilmaz, Lawrence Carin, et al. 2019. Cyclical annealing schedule: A simple approach to mitigating kl vanishing. *arXiv preprint arXiv:1903.10145* . 5.2.4, 9.4

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. 2015. The ubuntu dialogue corpus: A large dataset for research in unstructured multi-turn dialogue systems. *arXiv preprint arXiv:1506.08909* . 5.1

Ryan Thomas Lowe, Nissan Pow, Iulian Vlad Serban, Laurent Charlin, Chia-Wei Liu, and Joelle Pineau. 2017. Training end-to-end dialogue systems with the ubuntu dialogue corpus. *Dialogue & Discourse* 8(1):31–65. 2.1.3

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025* . 2.2.1, 2.2.1, 7.4.3, 8.4.1

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research* 9(Nov):2579–2605. 5.4.3

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. 2016. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712* . 4.1.1, 6.1, 6.2

Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2018. Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. *arXiv preprint arXiv:1804.08217* . 2.1.4

Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. 2015. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644* . 4.2.2

Mitchell P Marcus, Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a large annotated corpus of english: The penn treebank. *Computational linguistics* 19(2):313–330. 4.4.2, 6.4

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843* . (document), 2.2.1, 7.2, 7.2, 7.4.3, 7.6

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. 2015. Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)* 23(3):530–539. 2.1.2

Yishu Miao, Lei Yu, and Phil Blunsom. 2016. Neural variational inference for text processing. In *International Conference on Machine Learning*. pages 1727–1736. 6.2

Tomas Mikolov, Martin Karafiát, Lukas Burget, Jan Cernockỳ, and Sanjeev Khudanpur. 2010. Recurrent neural network based language model. In *Interspeech*. volume 2, page 3. 1.1, 4.1.2, 4.4.2, 6.4

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119. 3.4.3

Tomas Mikolov and Geoffrey Zweig. 2012. Context dependent recurrent neural network language model. *SLT* 12:234–239. 4.2.1

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. 2018. Spectral normalization for generative adversarial networks. *arXiv preprint arXiv:1802.05957* . 4.3

Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533. 2.2.4

George E Monahan. 1982. State of the arta survey of partially observable markov decision processes: theory, models, and algorithms. *Management Science* 28(1):1–16. 2.2.4

Igor Mordatch and Pieter Abbeel. 2017. Emergence of grounded compositional language in multi-agent populations. *arXiv preprint arXiv:1703.04908* . 8.2

Karthik Narasimhan, Tejas Kulkarni, and Regina Barzilay. 2015. Language understanding for text-based games using deep reinforcement learning. *arXiv preprint arXiv:1506.08941* . 2.2.4

Lasguido Nio, Sakriani Sakti, Graham Neubig, Tomoki Toda, and Satoshi Nakamura. 2014. Improving the robustness of example-based dialog retrieval using recursive neural network paraphrase identification. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, pages 306–311. 2.1.3

Hyungjong Noh, Seonghan Ryu, Donghyeon Lee, Kyusong Lee, Cheongjae Lee, and Gary Geunbae Lee. 2012. An example-based approach to ranking multiple dialog

states for flexible dialog management. *IEEE Journal on Selected Topics in Signal Processing* 6(8):943–958. 2.1.3

Junhyuk Oh, Satinder Singh, Honglak Lee, and Pushmeet Kohli. 2017. Zero-shot task generalization with multi-task deep reinforcement learning. *arXiv preprint arXiv:1706.05064* . 2.2.3, 7.2

Mark Palatucci, Dean Pomerleau, Geoffrey E Hinton, and Tom M Mitchell. 2009. Zero-shot learning with semantic output codes. In *Advances in neural information processing systems*. pages 1410–1418. 2.2.3, 7.1, 7.2

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pages 311–318. 3.4.4, 1, 7.6, 9.4

Ronald Parr and Stuart J Russell. 1998. Reinforcement learning with hierarchies of machines. In *Advances in neural information processing systems*. pages 1043–1049. 1.1

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–43. 5.3.3

Massimo Poesio and David Traum. 1998. Towards an axiomatization of dialogue acts. In *Proceedings of the Twente Workshop on the Formal Semantics and Pragmatics of Dialogues (13th Twente Workshop on Language Technology*. Citeseer. 5.2.2

Owen Rambow, Srinivas Bangalore, and Marilyn Walker. 2001. Natural language generation in dialog systems. In *Proceedings of the first international conference on Human language technology research*. Association for Computational Linguistics, pages 1–4. 1.1

Antoine Raux, Brian Langner, Dan Bohus, Alan W Black, and Maxine Eskenazi. 2005. Lets go public! taking a spoken dialog system to the real world. In *in Proc. of Interspeech 2005*. Citeseer. 1.1, 2.1.2, 5.2.2, 7.5.2, 8.1

Liliang Ren, Kaige Xie, Lu Chen, and Kai Yu. 2018. Towards universal dialogue state tracking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. pages 2780–2786. 8.2

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082* . 2.2.2

Eugénio Ribeiro, Ricardo Ribeiro, and David Martins de Matos. 2015. The influence of context on dialogue act recognition. *arXiv preprint arXiv:1506.00839* . 5.3.1

Bernardino Romera-Paredes and Philip Torr. 2015. An embarrassingly simple approach to zero-shot learning. In *International Conference on Machine Learning*. pages 2152–2161. 2.2.3, 7.2

Andrew Rosenberg and Julia Hirschberg. 2007. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 6.4.2

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. pages 627–635. 1

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information processing & management* 24(5):513–523. 5.3.2

Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. 2015. Prioritized experience replay. *arXiv preprint arXiv:1511.05952* . 2.2.4

Bernhard Scholkopf and Alexander J Smola. 2001. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press. 7.6

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681. 5.2.1

John R Searle, Ferenc Kiefer, Manfred Bierwisch, et al. 1980. *Speech act theory and pragmatics*, volume 10. Springer. 1.1

Stanislau Semeniuta, Aliaksei Severyn, and Erhardt Barth. 2017. A hybrid convolutional variational autoencoder for text generation. *arXiv preprint arXiv:1702.02390* . 4.3

Iulian V Serban, II Ororbia, G Alexander, Joelle Pineau, and Aaron Courville. 2016a. Piecewise latent variables for neural variational text processing. *arXiv preprint arXiv:1612.00377* . 4.3

Iulian V Serban, Chinnadhurai Sankar, Mathieu Germain, Saizheng Zhang, Zhouhan Lin, Sandeep Subramanian, Taesup Kim, Michael Pieper, Sarath Chandar, Nan Rosemary Ke, et al. 2017a. A deep reinforcement learning chatbot. *arXiv preprint arXiv:1709.02349* . 8.2

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2015. Building end-to-end dialogue systems using generative hierarchical neural network models. *arXiv preprint arXiv:1507.04808* . 2.1.4, 2.2.1

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. 2016b. Building end-to-end dialogue systems using generative hierarchical neural network models. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI-16)*. 5.4.1, 8.6.1

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron Courville, and Yoshua Bengio. 2016c. A hierarchical latent variable encoder-decoder model for generating dialogues. *arXiv preprint arXiv:1605.06069* . 2.1.4, 5.1, 6.1, 6.2, 6.3, 6.3.4, 6.4.3, 7.1, 7.6, 8.4.1

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017b. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*. pages 3295–3301. 9.4

Louis Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating high-quality and informative conversation responses with sequence-to-sequence models. *arXiv preprint arXiv:1701.03185* . 5.1

Richard Socher, Milind Ganjoo, Christopher D Manning, and Andrew Ng. 2013. Zero-

shot learning through cross-modal transfer. In *Advances in neural information processing systems*. pages 935–943. 7.2

Richard Socher, Cliff C Lin, Chris Manning, and Andrew Y Ng. 2011. Parsing natural scenes and natural language with recursive neural networks. In *Proceedings of the 28th international conference on machine learning (ICML-11)*. pages 129–136. 1.1

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems*. pages 3483–3491. 2.2.2, 5.1.1, 5.2.1

Yiping Song, Rui Yan, Xiang Li, Dongyan Zhao, and Ming Zhang. 2016. Two are better than one: An ensemble of retrieval-and generation-based dialog systems. *arXiv preprint arXiv:1610.07149* . 2.1.5

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. 2015. A neural network approach to context-sensitive generation of conversational responses. *arXiv preprint arXiv:1506.06714* . 1.1, 5.3.2, 8.1

Andreas Stolcke, Noah Coccaro, Rebecca Bates, Paul Taylor, Carol Van Ess-Dykema, Klaus Ries, Elizabeth Shriberg, Daniel Jurafsky, Rachel Martin, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics* 26(3):339–373. 5.3.1, 6.3.2

Pei-Hao Su, Paweł Budzianowski, Stefan Ultes, Milica Gasic, and Steve Young. 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. 8.1, 8.2

Pei-Hao Su, Milica Gasic, Nikola Mrksic, Lina Rojas-Barahona, Stefan Ultes, David Vandyke, Tsung-Hsien Wen, and Steve Young. 2016. Continuously learning neural dialogue management. *arXiv preprint arXiv:1606.02689* . 2.1.2

Richard S Sutton and Andrew G Barto. 1998. *Introduction to reinforcement learning*. MIT Press. 2.2.4

Johan AK Suykens and Joos Vandewalle. 1999. Least squares support vector machine classifiers. *Neural processing letters* 9(3):293–300. 5.3.1

David R Traum. 1999. Computational models of grounding in collaborative systems. In *Psychological Models of Communication in Collaborative Systems-Papers from the AAAI Fall Symposium*. pages 124–131. 2.1.1

Trieu H Trinh and Quoc V Le. 2018. A simple method for commonsense reasoning. *arXiv preprint arXiv:1806.02847* . 5.1

Aaron van den Oord, Oriol Vinyals, et al. 2017. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*. pages 6309–6318. 6.1, 6.2

Hado Van Hasselt, Arthur Guez, and David Silver. 2015. Deep reinforcement learning with double q-learning. *arXiv preprint arXiv:1509.06461* . 2.2.4

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. 2008.

Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th international conference on Machine learning*. ACM, pages 1096–1103. 4.5.2

Nguyen Xuan Vinh, Julien Epps, and James Bailey. 2010. Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance. *Journal of Machine Learning Research* 11(Oct):2837–2854. 6.4.2

Oriol Vinyals and Quoc Le. 2015. A neural conversational model. *arXiv preprint arXiv:1506.05869* . 1.1, 2.1.4, 2.2.1, 2.2.1, 8.1, 8.3

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. 2015. Show and tell: A neural image caption generator. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 3156–3164. 2.2.1

Martin J Wainwright, Michael I Jordan, et al. 2008. Graphical models, exponential families, and variational inference. *Foundations and Trends® in Machine Learning* 1(1–2):1–305. 4.1, 4.1.1

Marilyn A. Walker. 2000. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research* pages 387–416. 2.2.4, 8.1, 8.2

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, David Vandyke, and Steve Young. 2016a. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562* . 2.1.2, 7.1, 8.2

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Lina M Rojas-Barahona, Pei-Hao Su, David Vandyke, and Steve Young. 2016b. Multi-domain neural network language generation for spoken dialogue systems. *arXiv preprint arXiv:1603.01232* . 2.2.3

Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve Young. 2017. Latent intention dialogue models. *arXiv preprint arXiv:1705.10229* . 6.2

Jason Williams, Antoine Raux, Deepak Ramachandran, and Alan Black. 2013. The dialog state tracking challenge. In *Proceedings of the SIGDIAL 2013 Conference*. pages 404–413. 2.1.2

Jason Williams and Steve Young. 2003. Using wizard-of-oz simulations to bootstrap reinforcement-learning-based dialog management systems. In *Proceedings of the 4th SIGDIAL Workshop on Discourse and Dialogue*. 1.1

Jason D Williams, Kavosh Asadi, and Geoffrey Zweig. 2017. Hybrid code networks: practical and efficient end-to-end dialog control with supervised and reinforcement learning. *arXiv preprint arXiv:1702.03274* . 2.1.4, 8.2

Jason D Williams and Steve Young. 2007. Partially observable markov decision processes for spoken dialog systems. *Computer Speech & Language* 21(2):393–422. 2.2.4, 3.4.4, 6.1, 8.1

Jason D Williams and Geoffrey Zweig. 2016. End-to-end lstm-based dialog control optimized with supervised and reinforcement learning. *arXiv preprint arXiv:1606.01269*

. 2.1.2, 8.3

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256. 8, 8.3

Sam Wiseman and Alexander M Rush. 2016. Sequence-to-sequence learning as beam-search optimization. *arXiv preprint arXiv:1606.02960* . 2.1.4, 5.1

Yijun Xiao, Tiancheng Zhao, and William Yang Wang. 2018. Dirichlet variational autoencoder for text modeling. *arXiv preprint arXiv:1811.00135* . 4.3

Chen Xing, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2016. Topic augmented neural response generation with a joint attention mechanism. *arXiv preprint arXiv:1606.08340* . 2.1.4, 5.1, 6.2

Chen Xing, Wei Wu, Yu Wu, Ming Zhou, Yalou Huang, and Wei-Ying Ma. 2017. Hierarchical recurrent attention network for response generation. *arXiv preprint arXiv:1701.07149* . 2.1.4

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*. volume 14, pages 77–81. 1.1

Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. 2015. Attribute2image: Conditional image generation from visual attributes. *arXiv preprint arXiv:1512.00570* . 2.2.2, 5.1.1, 5.2.1

Zhilin Yang, Zihang Dai, Ruslan Salakhutdinov, and William W Cohen. 2017a. Breaking the softmax bottleneck: a high-rank rnn language model. *arXiv preprint arXiv:1711.03953* . 5.2.4, 9.4

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017b. Improved variational autoencoders for text modeling using dilated convolutions. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, pages 3881–3890. 4.3

Stephanie Young, Jost Schatzmann, Karl Weilhammer, and Hui Ye. 2007. The hidden information state approach to dialog management. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*. IEEE, volume 4, pages IV–149. 2.1.2, 8.2

Steve Young, Milica Gašić, Blaise Thomson, and Jason D Williams. 2013. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE* 101(5):1160–1179. 8.2

Steve J Young. 2006. Using pomdps for dialog management. In *SLT*. pages 8–13. 1.1, 2.1.2, 7.5.2, 8.1

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017a. Seqgan: Sequence generative adversarial nets with policy gradient. In *Thirty-First AAAI Conference on Artificial Intelligence*. 8

Zhou Yu, Alan W Black, and Alexander I Rudnicky. 2017b. Learning conversational

systems that interleave task and non-task content. *arXiv preprint arXiv:1703.00099* . 2.1.5

Wojciech Zaremba, Ilya Sutskever, and Oriol Vinyals. 2014. Recurrent neural network regularization. *arXiv preprint arXiv:1409.2329* . 6.4.1

Ran Zhao, Alexandros Papangelis, and Justine Cassell. 2014. Towards a dyadic computational model of rapport management for human-virtual agent interaction. In *International Conference on Intelligent Virtual Agents*. Springer, pages 514–527. 2.1.5

Tiancheng Zhao and Maxine Eskenazi. 2016. Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. *arXiv preprint arXiv:1606.02560* . 2.1.2, 2.1.4, 5.2.2, 7.4.2, 8.2

Tiancheng Zhao and Maxine Eskenazi. 2018. Zero-shot dialog generation with cross-domain latent actions. *arXiv preprint arXiv:1805.04803* . 9.2

Tiancheng Zhao, Kyusong Lee, and Maxine Eskenazi. 2018. Unsupervised discrete sentence representation learning for interpretable neural dialog generation. *arXiv preprint arXiv:1804.08069* . 8.4.1, 8.4.1, 9.2

Tiancheng Zhao, Allen Lu, Kyusong Lee, and Maxine Eskenazi. 2017a. Generative encoder-decoder models for task-oriented spoken dialog systems with chatting capability. *arXiv preprint arXiv:1706.08476* . 2.1.4, 2.1.5, 7.1, 7.2, 7.4.2, 7.6

Tiancheng Zhao, Kaige Xie, and Maxine Eskenazi. 2019. Rethinking action spaces for reinforcement learning in end-to-end dialog agents with latent variable models. *arXiv preprint arXiv:1902.08858* . 9.2

Tiancheng Zhao, Ran Zhao, and Maxine Eskenazi. 2017b. Learning discourse-level diversity for neural dialog models using conditional variational autoencoders. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 654–664. (document), 2.1.4, 6.1, 6.2, 6.3.4, 6.4.1, 6.1, 6.4.3, 8.1, 8.7, 9.2

Shengjia Zhao S, Jiaming Song, and Stefano Ermon. 2017. Infovae: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262* . 4.2.2

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *arXiv preprint arXiv:1709.00103* . 2.2.1, 8

Chunting Zhou and Graham Neubig. 2017. Multi-space variational encoder-decoders for semi-supervised labeled sequence transduction. *arXiv preprint arXiv:1704.01691* . 6.2

Li Zhou, Jianfeng Gao, Di Li, and Heung-Yeung Shum. 2018. The design and implementation of xiaoice, an empathetic social chatbot. *arXiv preprint arXiv:1812.08989* . 1.1, 3.4.4

Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. 2016. Multi-view response selection for human-computer conversation. In *EMNLP*. 2.1.3