arXiv:1809.08267v3 [cs.CL] 10 Sep 2019

Neural Approaches to Conversational AI

Question Answering, Task-Oriented Dialogues and Social Chatbots

Jianfeng Gao
Microsoft Research
jfgao@microsoft.com

Michel Galley
Microsoft Research
mgalley@microsoft.com

Lihong Li
Google Brain
lihong@google.com

Abstract

The present paper surveys neural approaches to conversational AI that have been developed in the last few years. We group conversational systems into three categories: (1) question answering agents, (2) task-oriented dialogue agents, and (3) chatbots. For each category, we present a review of state-of-the-art neural approaches, draw the connection between them and traditional approaches, and discuss the progress that has been made and challenges still being faced, using specific systems and models as case studies.[1]

---

[1] We are grateful to the anonymous reviewers, Chris Brockett, Asli Celikyilmaz, Yu Cheng, Bill Dolan, Pascale Fung, Zhe Gan, Sungjin Lee, Jinchao Li, Xiujun Li, Bing Liu, Andrea Madotto, Rangan Majumder, Alexandros Papangelis, Olivier Pietquin, Chris Quirk, Alan Ritter, Paul Smolensky, Alessandro Sordoni, Yang Song, Hisami Suzuki, Wei Wei, Tal Weiss, Kun Yuan, and Yizhe Zhang for their helpful comments and suggestions on earlier versions of this paper.

Contents

1 **Introduction** **5**
1.1 Who Should Read this Paper? . . . 6
1.2 Dialogue: What Kinds of Problems? . . . 6
1.3 A Unified View: Dialogue as Optimal Decision Making . . . 8
1.4 The Transition of NLP to Neural Approaches . . . 9

2 **Machine Learning Background** **11**
2.1 Machine Learning Basics . . . 11
2.2 Deep Learning . . . 13
2.2.1 Foundations . . . 13
2.2.2 Two Examples . . . 15
2.3 Reinforcement Learning . . . 16
2.3.1 Foundations . . . 17
2.3.2 Basic Algorithms . . . 17
2.3.3 Exploration . . . 19

3 **Question Answering and Machine Reading Comprehension** **20**
3.1 Knowledge Base . . . 20
3.2 Semantic Parsing for KB-QA . . . 21
3.3 Embedding-based Methods . . . 22
3.4 Multi-Step Reasoning on KB . . . 22
3.4.1 Symbolic Methods . . . 23
3.4.2 Neural Methods . . . 24
3.4.3 Reinforcement Learning based Methods . . . 25
3.5 Conversational KB-QA Agents . . . 26
3.6 Machine Reading for Text-QA . . . 28
3.7 Neural MRC Models . . . 30
3.7.1 Encoding . . . 30
3.7.2 Reasoning . . . 31
3.7.3 Training . . . 33
3.8 Conversational Text-QA Agents . . . 33

4 Task-oriented Dialogue Systems 35
4.1 Overview 35
4.2 Evaluation and User Simulation 38
4.2.1 Evaluation Metrics 38
4.2.2 Simulation-Based Evaluation 38
4.2.3 Human-based Evaluation 40
4.2.4 Other Evaluation Techniques 41
4.3 Natural Language Understanding and Dialogue State Tracking 41
4.3.1 Natural Language Understanding 41
4.3.2 Dialogue State Tracking 43
4.4 Dialogue Policy Learning 44
4.4.1 Deep RL for Policy Optimization 44
4.4.2 Efficient Exploration and Domain Extension 45
4.4.3 Composite-task Dialogues 46
4.4.4 Multi-domain Dialogues 47
4.4.5 Integration of Planning and Learning 48
4.4.6 Reward Function Learning 48
4.5 Natural Language Generation 49
4.6 End-to-end Learning 51
4.7 Further Remarks 52

5 Fully Data-Driven Conversation Models and Social Bots 53
5.1 End-to-End Conversation Models 53
5.1.1 The LSTM Model 54
5.1.2 The HRED Model 54
5.1.3 Attention Models 55
5.1.4 Pointer-Network Models 55
5.2 Challenges and Remedies 56
5.2.1 Response Blandness 56
5.2.2 Speaker Consistency 57
5.2.3 Word Repetitions 59
5.2.4 Further Challenges 59
5.3 Grounded Conversation Models 59
5.4 Beyond Supervised Learning 60
5.5 Data 62
5.6 Evaluation 63
5.7 Open Benchmarks 64

6 Conversational AI in Industry 65
6.1 Question Answering Systems 65

6.1.1 Bing QA 65
6.1.2 Satori QA 66
6.1.3 Customer Support Agents 66
6.2 Task-oriented Dialogue Systems (Virtual Assistants) 67
6.3 Chatbots 69

7 Conclusions and Research Trends **72**

Chapter 1

Introduction

Developing an intelligent dialogue system[1] that not only emulates human conversation, but also answers questions on topics ranging from latest news about a movie star to Einstein's theory of relativity, and fulfills complex tasks such as travel planning, has been one of the longest running goals in AI. The goal has remained elusive until recently. We are now observing promising results both in academia sindustry, as large amounts of conversational data become available for training, and the breakthroughs in deep learning (DL) and reinforcement learning (RL) are applied to conversational AI.

Conversational AI is fundamental to natural user interfaces. It is a rapidly growing field, attracting many researchers in the Natural Language Processing (NLP), Information Retrieval (IR) and Machine Learning (ML) communities. For example, SIGIR 2018 has created a new track of Artificial Intelligence, Semantics, and Dialog to bridge research in AI and IR, especially targeting Question Answering (QA), deep semantics and dialogue with intelligent agents.

Recent years have seen the rise of a small industry of tutorials and survey papers on deep learning and dialogue systems. Yih et al. (2015b, 2016); Gao (2017) reviewed deep learning approaches for a wide range of IR and NLP tasks, including dialogues. Chen et al. (2017e) presented a tutorial on dialogues, with a focus on task-oriented agents. Serban et al. (2015; 2018) surveyed public dialogue datasets that can be used to develop conversational agents. Chen et al. (2017b) reviewed popular deep neural network models for dialogues, focusing on supervised learning approaches. The present work substantially expands the scope of Chen et al. (2017b); Serban et al. (2015) by going beyond data and supervised learning to provide what we believe is the first survey of neural approaches to conversational AI, targeting NLP and IR audiences.[2] Its contributions are:

- We provide a comprehensive survey of the neural approaches to conversational AI that have been developed in the last few years, covering QA, task-oriented and social bots with a unified view of optimal decision making.
- We draw connections between modern neural approaches and traditional approaches, allowing us to better understand why and how the research has evolved and to shed light on how we can move forward.
- We present state-of-the-art approaches to training dialogue agents using both supervised and reinforcement learning.

---

[1] "Dialogue systems" and "conversational AI" are often used interchangeably in the scientific literature. The difference is reflective of different traditions. The former term is more general in that a dialogue system might be purely rule-based rather than AI-based.

[2] One important topic of conversational AI that we do not cover is Spoken Language Understanding (SLU). SLU systems are designed to extract the meaning from speech utterances and their application are vast, ranging from voice search in mobile devices to meeting summarization. The present work does encompass many Spoken Dialogue Systems – for example Young et al. (2013) – but does not focus on components related to speech. We refer readers to Tur and De Mori (2011) for a survey of SLU.

- We sketch out the landscape of conversational systems developed in the research community and released in industry, demonstrating via case studies the progress that has been made and the challenges that we are still facing.

1.1 Who Should Read this Paper?

This paper is based on tutorials given at the SIGIR and ACL conferences in 2018 (Gao et al., 2018a,b), with the IR and NLP communities as the primary target audience. However, audiences with other backgrounds (such as machine learning) will also find it an accessible introduction to conversational AI with numerous pointers, especially to recently developed neural approaches.

We hope that this paper will prove a valuable resource for students, researchers, and software developers. It provides a unified view, as well as a detailed presentation of the important ideas and insights needed to understand and create modern dialogue agents that will be instrumental to making world knowledge and services accessible to millions of users in ways that seem natural and intuitive.

This survey is structured as follows:

- The rest of this chapter introduces dialogue tasks and presents a unified view in which open-domain dialogue is formulated as an optimal decision making process.
- Chapter 2 introduces basic mathematical tools and machine learning concepts, and reviews recent progress in the deep learning and reinforcement learning techniques that are fundamental to developing neural dialogue agents.
- Chapter 3 describes question answering (QA) agents, focusing on neural models for knowledge-base QA and machine reading comprehension (MRC).
- Chapter 4 describes task-oriented dialogue agents, focusing on applying deep reinforcement learning to dialogue management.
- Chapter 5 describes social chatbots, focusing on fully data-driven neural approaches to end-to-end generation of conversational responses.
- Chapter 6 gives a brief review of several conversational systems in industry.
- Chapter 7 concludes the paper with a discussion of research trends.

1.2 Dialogue: What Kinds of Problems?

Fig. 1.1 shows a human-agent dialogue during the process of making a business decision. The example illustrates the kinds of problems a dialogue system is expected to solve:

- **question answering**: the agent needs to provide concise, direct answers to user queries based on rich knowledge drawn from various data sources including text collections such as Web documents and pre-compiled knowledge bases such as sales and marketing datasets, as the example shown in Turns 3 to 5 in Fig. 1.1.
- **task completion**: the agent needs to accomplish user tasks ranging from restaurant reservation to meeting scheduling (e.g., Turns 6 to 7 in Fig. 1.1), and to business trip planning.
- **social chat**: the agent needs to converse seamlessly and appropriately with users — like a human as in the Turing test — and provide useful recommendations (e.g., Turns 1 to 2 in Fig. 1.1).

One may envision that the above dialogue can be collectively accomplished by a set of agents, also known as *bots*, each of which is designed for solving a particular type of task, e.g., QA bots, task-completion bots, social chatbots. These bots can be grouped into two categories, *task-oriented* and *chitchat*, depending on whether the dialogue is conducted to assist users to achieve specific tasks, e.g., obtain an answer to a query or have a meeting scheduled.

Most of the popular personal assistants in today's market, such as Amazon Alexa, Apple Siri, Google Home, and Microsoft Cortana, are task-oriented bots. These can only handle relatively simple tasks, such as reporting weather and requesting songs. An example of a chitchat dialogue bot is Microsoft

Turn	Dialogue
1	*usr:* Good morning!
	agt: Good morning, I hope you had a great weekend.
2	*usr:* not really.
	agt: You sound depressed. What can I do for you?
3	*usr:* I'd like to know where sales are lagging behind our forecast?
	agt: The worst region is [country], where sales are 15% below projections.
4	*usr:* Do you know why?
	agt: The forecast for [product] growth was overly optimistic.
5	*usr:* How can we turn this around?
	agt: Here are the 10 customers in [country] with the most growth potential, per our CRM model.
6	*usr:* Can you set up a meeting with the CTO of [company]?
	agt: Yes, I've set up a meeting with [person name] for next month when you are in [location].
7	*usr:* Thanks!

Figure 1.1: A human-agent dialogue during the process of making a business decision. (*usr*: user, *agt*: agent) The dialogue consists of multiple segments of different types. Turns 1 and 2 are a social chat segment. Turns 3 to 5 are a QA segment. Turns 6 and 7 are a task-completion segment.

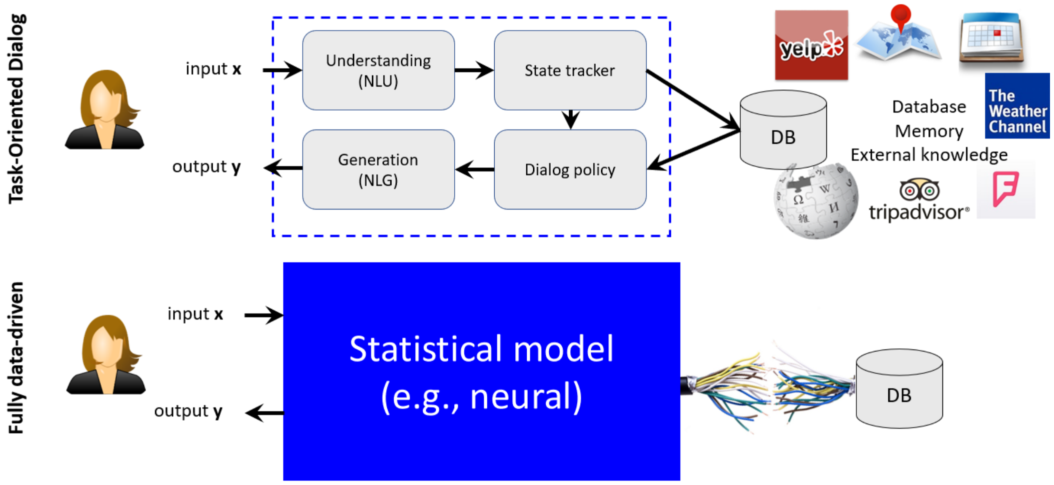


Figure 1.2: Two architectures of dialogue systems for (Top) traditional task-oriented dialogue and (Bottom) fully data-driven dialogue.

XiaoIce. Building a dialogue agent to fulfill complex tasks as in Fig. 1.1 remains one of the most fundamental challenges for the IR and NLP communities, and AI in general.

A typical task-oriented dialogue agent is composed of four modules, as illustrated in Fig. 1.2 (Top): (1) a Natural Language Understanding (NLU) module for identifying user intents and extracting associated information; (2) a state tracker for tracking the dialogue state that captures all essential information in the conversation so far; (3) a dialogue policy that selects the next action based on the current state; and (4) a Natural Language Generation (NLG) module for converting agent actions to natural language responses. In recent years there has been a trend towards developing fully data-driven systems by unifying these modules using a deep neural network that maps the user input to the agent output directly, as shown in Fig. 1.2 (Bottom).

Most task-oriented bots are implemented using a modular system, where the bot often has access to an external database on which to inquire about information to accomplish the task (Young et al., 2013; Tur and De Mori, 2011). Social chatbots, on the other hand, are often implemented using a unitary (non-modular) system. Since the primary goal of social chatbots is to be AI companions to humans with an emotional connection rather than completing specific tasks, they are often developed to mimic human conversations by training DNN-based response generation models on large amounts of human-human conversational data (Ritter et al., 2011; Sordoni et al., 2015b; Vinyals and Le, 2015; Shang et al., 2015). Only recently have researchers begun to explore how to ground the chitchat in world knowledge (Ghazvininejad et al., 2018) and images (Mostafazadeh et al., 2017) so as to make the conversation more contentful and interesting.

1.3 A Unified View: Dialogue as Optimal Decision Making

The example dialogue in Fig. 1.1 can be formulated as a decision making process. It has a natural hierarchy: a top-level process selects what agent to activate for a particular subtask (e.g., answering a question, scheduling a meeting, providing a recommendation or just having a casual chat), and a low-level process, controlled by the selected agent, chooses primitive actions to complete the subtask.

Such hierarchical decision making processes can be cast in the mathematical framework of *options* over Markov Decision Processes (MDPs) (Sutton et al., 1999b), where options generalize primitive actions to higher-level actions. In a traditional MDP setting, an agent chooses a primitive action at each time step. With options, the agent can choose a “multi-step” action which for example could be a sequence of primitive actions for completing a subtask.

If we view each option as an action, both top- and low-level processes can be naturally captured by the reinforcement learning framework. The dialogue agent navigates in a MDP, interacting with its environment over a sequence of discrete steps. At each step, the agent observes the current state, and chooses an action according to a policy. The agent then receives a reward and observes a new state, continuing the cycle until the episode terminates. The goal of dialogue learning is to find optimal policies to maximize expected rewards. Table 1.1 formulates an sample of dialogue agents using this unified view of RL, where the state-action spaces characterize the complexity of the problems, and the rewards are the objective functions to be optimized.

The unified view of hierarchical MDPs has already been applied to guide the development of some large-scale open-domain dialogue systems. Recent examples include Sounding Board [3], a social chatbot that won the 2017 Amazon Alexa Prize, and Microsoft XiaoIce [4], arguably the most popular social chatbot that has attracted more than 660 million users worldwide since its release in 2014. Both systems use a hierarchical dialogue manager: a master (top-level) that manages the overall conversation process, and a collection of skills (low-level) that handle different types of conversation segments (subtasks).

The reward functions in Table 1.1, which seem contradictory in CPS (e.g., we need to minimize CPS for efficient task completion but maximize CPS for improving user engagement), suggest that we have to balance the long-term and short-term gains when developing a dialogue system. For example, XiaoIce is a social chatbot optimized for user engagement, but is also equipped with more than 230 skills, most of which are QA and task-oriented. XiaoIce is optimized for *expected* CPS which corresponds a long-term, rather than a short-term, engagement. Although incorporating many task-oriented and QA skills can reduce CPS in the short term since these skills help users accomplish tasks *more efficiently* by minimizing CPS, these new skills establish XiaoIce as an efficient and trustworthy personal assistant, thus strengthening the emotional bond with human users in the long run.

Although RL provides a unified ML framework for building dialogue agents, applying RL requires training the agents by interacting with real users, which can be expensive in many domains. Hence, in practice, we often use a hybrid approach that combines the strengths of different ML methods. For example, we might use imitation and/or supervised learning methods (if there is a large amount of human-human conversational corpus) to obtain a reasonably good agent before applying RL to

---

[3] `https://sounding-board.github.io/`

[4] `https://www.msxiaobing.com/`

Table 1.1: Reinforcement Learning for Dialogue. CPS stands for Conversation-turns Per Session, and is defined as the average number of conversation-turns between the bot and the user in a conversational session.

dialogue	**state**	**action**	**reward**
QA	understanding of user query intent	clarification questions or answers	relevance of answer, (min) CPS
task-oriented	understanding of user goal	dialogue-act and slot/value	task success rate, (min) CPS
chitchat	conversation history and user intent	responses	user engagement, measured in CPS
top-level bot	understanding of user top-level intent	options	user engagement, measured in CPS

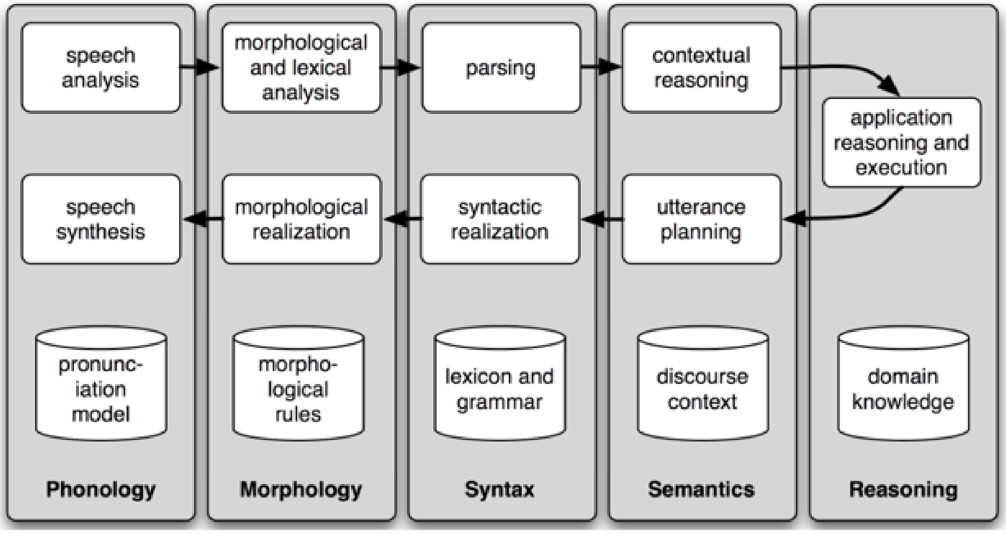


Figure 1.3: Traditional NLP Component Stack. Figure credit: Bird et al. (2009).

continue improving it. In the paper, we will survey these ML approaches and their use for training dialogue systems.

1.4 The Transition of NLP to Neural Approaches

Neural approaches are now transforming the field of NLP and IR, where symbolic approaches have been dominating for decades.

NLP applications differ from other data processing systems in their use of language knowledge of various levels, including phonology, morphology, syntax, semantics and discourse (Jurafsky and Martin, 2009). Historically, much of the NLP field has organized itself around the architecture of Fig. 1.3, with researchers aligning their work with one component task, such as morphological analysis or parsing. These tasks can be viewed as resolving (or realizing) natural language ambiguity (or diversity) at different levels by mapping (or generating) a natural language sentence to (or from) a series of human-defined, unambiguous, symbolic representations, such as Part-Of-Speech (POS) tags, context free grammar, first-order predicate calculus. With the rise of data-driven and statistical approaches, these components have remained and have been adapted as a rich source of engineered features to be fed into a variety of machine learning models (Manning et al., 2014).

Neural approaches do not rely on any human-defined symbolic representations but learn in a *task-specific* neural space where task-specific knowledge is *implicitly* represented as semantic concepts using low-dimensional continuous vectors. As Fig. 1.4 illustrates, neural methods in NLP tasks (e.g., machine reading comprehension and dialogue) often consist of three steps: (1) *encoding* symbolic

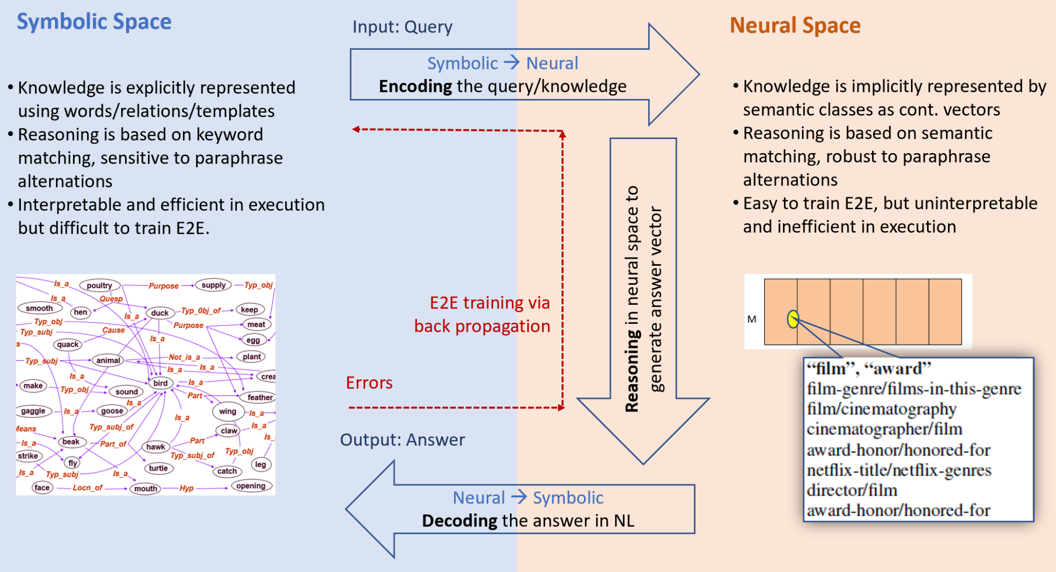

Symbolic Space
Input: Query
Neural Space
Symbolic → Neural
Encoding the query/knowledge
• Knowledge is explicitly represented using words/relations/templates
• Reasoning is based on keyword matching, sensitive to paraphrase alternations
• Interpretable and efficient in execution but difficult to train E2E.
• Knowledge is implicitly represented by semantic classes as cont. vectors
• Reasoning is based on semantic matching, robust to paraphrase alternations
• Easy to train E2E, but uninterpretable and inefficient in execution
Reasoning in neural space to generate answer vector
E2E training via back propagation
Errors
M
"film", "award"
film-genre/films-in-this-genre
film/cinematography
cinematographer/film
award-honor/honored-for
netflix-title/netflix-genres
director/film
award-honor/honored-for
Output: Answer
Neural → Symbolic
Decoding the answer in NL

Figure 1.4: Symbolic and Neural Computation.

user input and knowledge into their neural semantic representations, where semantically related or similar concepts are represented as vectors that are close to each other; (2) *reasoning* in the neural space to generate a system response based on input and system state; and (3) *decoding* the system response into a natural language output in a symbolic space. Encoding, reasoning and decoding are implemented using neural networks of different architectures, all of which may be stacked into a deep neural network trained in an end-to-end fashion via back propagation.

End-to-end training results in tighter coupling between the end application and the neural network architecture, lessening the need for traditional NLP component boundaries like morphological analysis and parsing. This drastically flattens the technology stack of Fig. 1.3, and substantially reduces the need for feature engineering. Instead, the focus has shifted to carefully tailoring the increasingly complex architecture of neural networks to the end application.

Although neural approaches have already been widely adopted in many AI tasks, including image processing, speech recognition and machine translation (e.g., Goodfellow et al., 2016), their impact on conversational AI has come somewhat more slowly. Only recently have we begun to observe neural approaches establish state-of-the-art results on an array of conversation benchmarks for both component tasks and end applications and, in the process, sweep aside the traditional component-based boundaries that have defined research areas for decades. This symbolic-to-neural shift is also reshaping the conversational AI landscape by opening up new tasks and user experiences that were not possible with older techniques. One reason for this is that neural approaches provide a consistent representation for many modalities, capturing linguistic and non-linguistic (e.g., image and video (Mostafazadeh et al., 2017)) features in the same modeling framework.

There are also works on hybrid methods that combine the strengths of both neural and symbolic approaches e.g., (Mou et al., 2016; Liang et al., 2016). As summarized in Fig. 1.4, neural approaches can be trained in an end-to-end fashion and are robust to paraphrase alternations, but are weak in terms of execution efficiency and explicit interpretability. Symbolic approaches, on the other hand, are difficult to train and sensitive to paraphrase alternations, but are more interpretable and efficient in execution.

Chapter 2

Machine Learning Background

This chapter presents a brief review of the deep learning and reinforcement learning technologies that are most relevant to conversational AI in later chapters.

2.1 Machine Learning Basics

Mitchell (1997) defines machine learning broadly to include any computer program that improves its performance at some task T, measured by P, through experiences E.

Dialogue, as summarized in Table 1.1, is a well-defined learning problem with T, P, and E specified as follows:

- T: perform conversations with a user to fulfill the user's goal.
- P: cumulative reward defined in Table 1.1.
- E: a set of dialogues, each of which is a sequence of user-agent interactions.

As a simple example, a single-turn QA dialogue agent might improve its performance *as measured by accuracy or relevance of its generated answers* at *the QA task*, through experiences of *human-labeled question-answer pairs*.

A common recipe of building an ML agent using *supervised learning* (SL) consists of a dataset, a model, a cost function (a.k.a. loss function) and an optimization procedure.

- The dataset consists of (x, y^*) pairs, where for each input x, there is a ground-truth output y^*. In QA, x consists of an input question and the documents from which an answer is generated, and y^* is the desired answer provided by a knowledgeable external supervisor.
- The model is typically of the form $y = f(x; \theta)$, where f is a function (e.g., a neural network) parameterized by θ that maps input x to output y.
- The cost function is of the form $L(y^*, f(x; \theta))$. $L(.)$ is often designed as a smooth function of error, and is differentiable w.r.t. θ. A commonly used cost function that meets these criteria is the *mean squared error* (MSE), defined as

  $$\frac{1}{M}\sum_{i=1}^{M}(y_i^* - f(x_i; \theta))^2 \,.$$

- The optimization can be viewed as a search algorithm to identify the best θ that minimize $L(.)$. Given that L is differentiable, the most widely used optimization procedure for deep learning is mini-batch Stochastic Gradient Descent (SGD) which updates θ after each batch as

  $$\theta \leftarrow \theta - \frac{\alpha}{N}\sum_{i=1}^{N}\nabla_\theta L(y_i^*, f(x_i; \theta)) \,, \tag{2.1}$$

  where N is the batch size and α the learning rate.

Common Supervised Learning Metrics. Once a model is trained, it can be tested on a *hold-out* dataset to have an estimate of its generalization performance. Suppose the model is $f(\cdot;\theta)$, and the hold-out set contains N data points: $\mathcal{D} = \{(x_1, y_1^*), (x_2, y_2^*), \ldots, (x_N, y_N^*)\}$.

The first metric is the aforementioned mean squared error that is appropriate for regression problems (i.e., y_i^* is considered real-values):

$$\mathrm{MSE}(f) := \frac{1}{N}\sum_{i=1}^{N}(y_i^* - f(x_i;\theta))^2 \,.$$

For classification problems, y_i^* takes values from a finite set viewed as categories. For simplicity, assume $y_i^* \in \{+1, -1\}$ here, so that an example (x_i, y_i^*) is called positive (or negative) if y_i^* is $+1$ (or -1). The following metrics are often used:

- Accuracy: the fraction of examples for which f predicts correctly:

  $$\mathrm{ACCURACY}(f) := \frac{1}{N}\sum_{i=1}^{N}\mathbf{1}(f(x_i;\theta) = y_i^*)\,,$$

  where $\mathbf{1}(E)$ is 1 if expression E is true and 0 otherwise.

- Precision: the fraction of correct predictions among examples that are predicted by f to be positive:

  $$\mathrm{PRECISION}(f) := \frac{\sum_{i=1}^{N}\mathbf{1}(f(x_i;\theta) = y_i^* \text{ AND } y_i^* = +1)}{\sum_{i=1}^{N}\mathbf{1}(f(x_i;\theta) = +1)}\,.$$

- Recall: the fraction of positive examples that are correctly predicted by f:

  $$\mathrm{RECALL}(f) := \frac{\sum_{i=1}^{N}\mathbf{1}(f(x_i;\theta) = y_i^* \text{ AND } y_i^* = +1)}{\sum_{i=1}^{N}\mathbf{1}(y_i^* = +1)}\,.$$

- F1 Score: the harmonic mean of precision and recall:

  $$\mathrm{F1}(f) := \frac{2 \times \mathrm{ACCURACY}(f) \times \mathrm{RECALL}(f)}{\mathrm{ACCURACY}(f) + \mathrm{RECALL}(f)}\,.$$

Other metrics are also widely used, especially for complex tasks beyond binary classification, such as the BLEU score (Papineni et al., 2002).

Reinforcement Learning. The above SL recipe applies to prediction tasks on a fixed dataset. However, in interactive problems such as dialogues[1], it can be challenging to obtain examples of desired behaviors that are both correct and representative of all the states in which the agent has to act. In unexplored territories, the agent has to learn how to act by interacting with an unknown environment on its own. This learning paradigm is known as *reinforcement learning* (RL), where there is a feedback loop between the agent and the external environment. In other words, while *SL learns from previous experiences* provided by a knowledgeable external supervisor, *RL learns by experiencing* on its own. RL differs from SL in several important respects (Sutton and Barto, 2018; Mitchell, 1997)

- **Exploration-exploitation tradeoff.** In RL, the agent needs to collect reward signals from the environment. This raises the question of which experimentation strategy results in more effective learning. The agent has to *exploit* what it already knows in order to obtain high rewards, while having to *explore* unknown states and actions in order to make better action selections in the future.

---

[1] As shown in Table 1.1, dialogue learning is formulated as RL where the agent learns a policy π that in each dialogue turn chooses an appropriate action a from the set $\mathcal{A}$, based on dialogue state s, so as to achieve the greatest cumulative reward.

- **Delayed reward and temporal credit assignment.** In RL, training information is not available in the form of (x, y^*) as in SL. Instead, the environment provides only delayed rewards as the agent executes a sequence of actions. For example, we do not know whether a dialogue succeeds in completing a task until the end of the session. The agent, therefore, has to determine which of the actions in its sequence are to be credited with producing the eventual reward, a problem known as *temporal credit assignment*.
- **Partially observed states.** In many RL problems, the observation perceived from the environment at each step, e.g., user input in each dialogue turn, provides only partial information about the entire state of the environment based on which the agent selects the next action. Neural approaches learn a deep neural network to represent the state by encoding all information observed at the current and past steps, e.g., all the previous dialogue turns and the retrieval results from external databases.

A central challenge in both SL and RL is *generalization*, the ability to perform well on unseen inputs. Many learning theories and algorithms have been proposed to address the challenge with some success by, e.g., seeking a good tradeoff between the amount of available training data and the model capacity to avoid underfitting and overfitting. Compared to previous techniques, neural approaches provide a potentially more effective solution by leveraging the representation learning power of deep neural networks, as we will review in the next section.

2.2 Deep Learning

Deep learning (DL) involves training neural networks, which in their original form consisted of a single layer (i.e., the perceptron) (Rosenblatt, 1957). The perceptron is incapable of learning even simple functions such as the logical XOR, so subsequent work explored the use of “deep” architectures, which added hidden layers between input and output (Rosenblatt, 1962; Minsky and Papert, 1969), a form of neural network that is commonly called the multi-layer perceptron (MLP), or deep neural networks (DNNs). This section introduces some commonly used DNNs for NLP and IR. Interested readers are referred to Goodfellow et al. (2016) for a comprehensive discussion.

2.2.1 Foundations

Consider a text classification problem: labeling a text string (e.g., a document or a query) by a domain name such as “sport” and “politics”. As illustrated in Fig. 2.1 (Left), a classical ML algorithm first maps a text string to a vector representation $\mathbf{x}$ using a set of hand-engineered features (e.g., word and character n-grams, entities, and phrases etc.), then learns a linear classifier with a softmax layer to compute the distribution of the domain labels $\mathbf{y} = f(\mathbf{x}; \mathbf{W})$, where $\mathbf{W}$ is a matrix learned from training data using SGD to minimize the misclassification error. The design effort is focused mainly on feature engineering.

Instead of using hand-designed features for $\mathbf{x}$, DL approaches jointly optimize the feature representation and classification using a DNN, as exemplified in Fig. 2.1 (Right). We see that the DNN consists of two halves. The top half can be viewed as a linear classifier, similar to that in the classical ML model in Fig. 2.1 (Left), except that its input vector $\mathbf{h}$ is not based on hand-engineered features but is learned using the bottom half of the DNN, which can be viewed as a feature generator optimized jointly with the classifier in an end-to-end fashion. Unlike classical ML, the effort of designing a DL classifier is mainly on optimizing DNN architectures for effective representation learning.

For NLP tasks, depending on the type of linguistic structures that we hope to capture in the text, we may apply different types of neural network (NN) layer structures, such as convolutional layers for local word dependencies and recurrent layers for global word sequences. These layers can be combined and stacked to form a deep architecture to capture different semantic and context information at different abstract levels. Several widely used NN layers are described below:

Word Embedding Layers. In a symbolic space each word is represented as a one-hot vector whose dimensionality n is the size of a pre-defined vocabulary. The vocabulary is often large; e.g., $n > 100K$. We apply a word embedding model, which is parameterized by a linear projection matrix $\mathbf{W}_e \in \mathbb{R}^{n \times m}$, to map each one-hot vector to a m-dimensional real-valued vector $(m \ll n)$

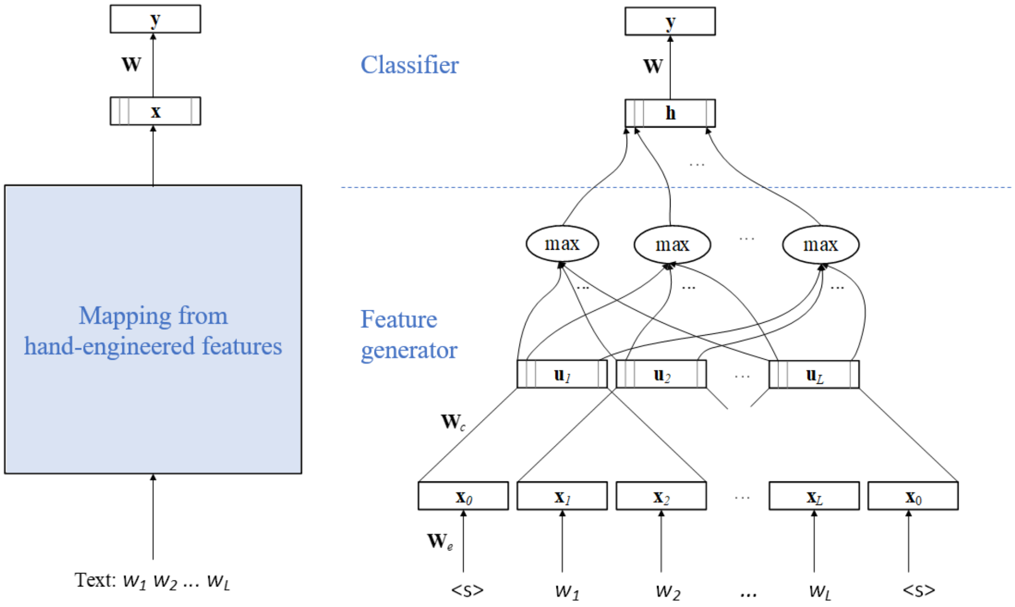


Figure 2.1: Flowcharts of classic machine learning (Left) and deep learning (Right). A convolutional neural network is used as an example for deep learning.

in a neural space where the embedding vectors of the words that are more semantically similar are closer to each other.

Fully Connected Layers. They perform linear projections as $\mathbf{W}^\intercal \mathbf{x}$.[2] We can stack multiple fully connected layers to form a deep feed-forward NN (FFNN) by introducing a nonlinear *activation function* g after each linear projection. If we view a text as a Bag-Of-Words (BOW) and let $\mathbf{x}$ be the sum of the embedding vectors of all words in the text, a deep FFNN can extract highly nonlinear features to represent hidden semantic topics of the text at different layers, e.g., $\mathbf{h}^{(1)} = g\left(\mathbf{W}^{(1)\intercal}\mathbf{x}\right)$ at the first layer, and $\mathbf{h}^{(2)} = g\left(\mathbf{W}^{(2)\intercal}\mathbf{h}^{(1)}\right)$ at the second layer, and so on, where $\mathbf{W}$'s are trainable matrices.

Convolutional-Pooling Layers. An example of convolutional neural networks (CNNs) is shown in Fig. 2.1 (Right). A convolutional layer forms a local feature vector, denoted $\mathbf{u}_i$, of word w_i in two steps. It first generates a contextual vector $\mathbf{c}_i$ by concatenating the word embedding vectors of w_i and its surrounding words defined by a fixed-length window. It then performs a projection to obtain $\mathbf{u}_i = g\left(\mathbf{W}_c^\intercal \mathbf{c}_i\right)$, where $\mathbf{W}_c$ is a trainable matrix and g is an activation function. Then, a pooling layer combines the outputs $\mathbf{u}_i, i = 1...L$ into a single global feature vector $\mathbf{h}$. For example, in Fig. 2.1, the *max pooling* operation is applied over each "time" i of the sequence of the vectors computed by the convolutional layer to obtain $\mathbf{h}$, where each element is computed as $h_j = \max_{1 \le i \le L} u_{i,j}$. Another popular pooling function is *average pooling*.

Recurrent Layers. An example of recurrent neural networks (RNNs) is shown in Fig. 2.2. RNNs are commonly used for sentence embedding where we view a text as a sequence of words rather than a BOW. They map the text to a dense and low-dimensional semantic vector by sequentially and recurrently processing each word, and mapping the subsequence up to the current word into a low-dimensional vector as $\mathbf{h}_i = \text{RNN}(\mathbf{x}_i, \mathbf{h}_{i-1}) := g\left(\mathbf{W}_{ih}^\intercal \mathbf{x}_i + \mathbf{W}_r^\intercal \mathbf{h}_{i-1}\right)$, where $\mathbf{x}_i$ is the word embedding of the i-th word in the text, $\mathbf{W}_{ih}$ and $\mathbf{W}_r$ are trainable matrices, and $\mathbf{h}_i$ is the semantic representation of the word sequence up to the i-th word.

---

[2] We often omit the bias terms for simplifying notations in this paper.

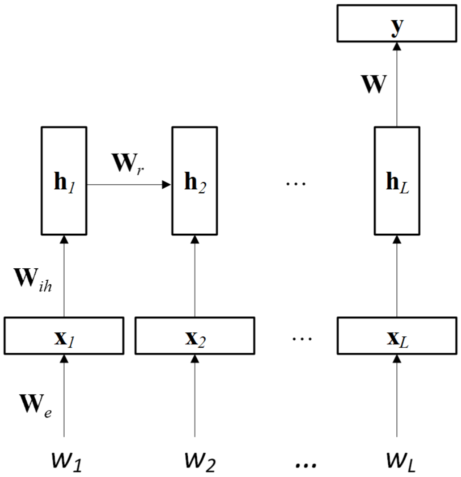


Figure 2.2: An example of recurrent neural networks.

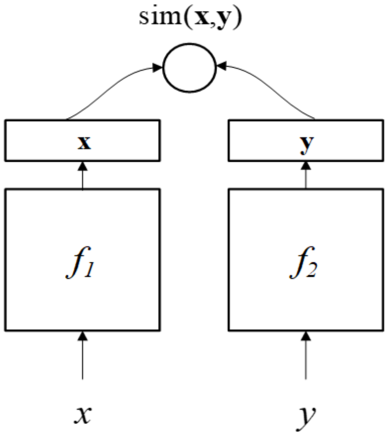


Figure 2.3: The architecture of DSSM.

2.2.2 Two Examples

This section gives a brief description of two examples of DNN models, designed for the ranking and text generation tasks, respectively. They are composed of the NN layers described in the last section.

DSSM for Ranking. In a ranking task, given an input query x, we want to rank all its candidate answers $y \in \mathcal{Y}$, based on a similarity scoring function $\text{sim}(x, y)$. The task is fundamental to many IR and NLP applications, such as query-document ranking, answer selection in QA, and dialogue response selection.

DSSM stands for Deep Structured Semantic Models (Huang et al., 2013; Shen et al., 2014), or more generally, Deep Semantic Similarity Model (Gao et al., 2014b). DSSM is a deep learning model for measuring the semantic similarity of a pair of inputs (x, y) [3]. As illustrated in Fig. 2.3, a DSSM consists of a pair of DNNs, f_1 and f_2, which map inputs x and y into corresponding vectors in a common low-dimensional semantic space. Then the similarity of x and y is measured by the cosine distance of the two vectors. f_1 and f_2 can be of different architectures depending on x and y. For

---

[3] DSSM can be applied to a wide range of tasks depending on the definition of (x, y). For example, (x, y) is a query-document pair for Web search ranking (Huang et al., 2013; Shen et al., 2014), a document pair in recommendation (Gao et al., 2014b), a question-answer pair in QA (Yih et al., 2015a), a sentence pair of different languages in machine translation (Gao et al., 2014a), and an image-text pair in image captioning (Fang et al., 2015) and so on.

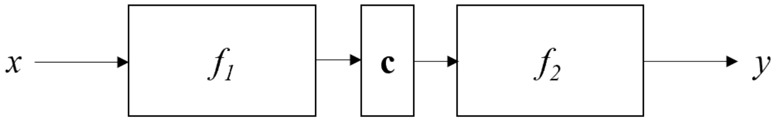


Figure 2.4: The architecture of seq2seq.

example, to compute the similarity of an image-text pair, f_1 can be a deep convolutional NN and f_2 an RNN.

Let θ be the parameters of f_1 and f_2. θ is learned to identify the most effective feature representations of x and y, optimized directly for end tasks. In other words, we learn a hidden semantic space, parameterized by θ, where the semantics of distance between vectors in the space is defined by the task or, more specifically, the training data of the task. For example, in Web document ranking, the distance measures the query-document relevance, and θ is optimized using a pair-wise rank loss. Consider a query x and two candidate documents y^+ and y^-, where y^+ is more relevant than y^- to x. Let $\text{sim}_\theta(x, y)$ be the similarity of x and y in the semantic space parameterized by θ as

$$\text{sim}_\theta(x, y) = \cos(f_1(x), f_2(y)).$$

We want to maximize $\Delta = \text{sim}_\theta(x, y^+) - \text{sim}_\theta(x, y^-)$. We do so by optimizing a smooth loss function

$$L(\Delta; \theta) = \log\left(1 + \exp\left(-\gamma\Delta\right)\right), \tag{2.2}$$

where γ is a scaling factor, using SGD of Eqn. 2.1.

Seq2Seq for Text Generation. In a text generation task, given an input text x, we want to generate an output text y. This task is fundamental to applications such as machine translation and dialogue response generation.

Seq2seq stands for the sequence-to-sequence architecture (Sutskever et al., 2014), which is also known as the encoder-decoder architecture (Cho et al., 2014b). Seq2Seq is typically implemented based on sequence models such as RNNs or gated RNNs. Gate RNNs, such as Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and the networks based on Gated Recurrent Unit (GRU) (Cho et al., 2014b), are the extensions of RNN in Fig. 2.2, and are often more effective in capturing long-term dependencies due to the use of gated cells that have paths through time that have derivatives neither vanishing nor exploding. We will illustrate in detail how LSTM is applied to end-to-end conversation models in Sec. 5.1.

Seq2seq defines the probability of generating y conditioned on x as $P(y|x)$ [4]. As illustrated in Fig. 2.4, a seq2seq model consists of (1) an input RNN or encoder f_1 that encodes input sequence x into context vector **c**, usually as a simple function of its final hidden state; and (2) an output RNN or decoder f_2 that generates output sequence y conditioned on **c**. x and y can be of different lengths. The two RNNs, parameterized by θ, are trained jointly to minimize the loss function over all the pairs of (x, y) in training data

$$L(\theta) = \frac{1}{M} \sum_{i=1\ldots M} \log -P_\theta(y_i|x_i)\,. \tag{2.3}$$

2.3 Reinforcement Learning

This section reviews reinforcement learning to facilitate discussions in later chapters. For a comprehensive treatment of this topic, interested readers are referred to existing textbooks and reviews, such as Sutton and Barto (2018); Kaelbling et al. (1996); Bertsekas and Tsitsiklis (1996); Szepesvári (2010); Wiering and van Otterlo (2012); Li (2018).

---

[4] Similar to DSSM, seq2seq can be applied to a variety of generation tasks depending on the definition of (x, y). For example, (x, y) is a sentence pair of different languages in machine translation (Sutskever et al., 2014; Cho et al., 2014b), an image-text pairs in image captioning (Vinyals et al., 2015b) (where f_1 is a CNN), and message-response pairs in dialogue (Vinyals and Le, 2015; Li et al., 2016a).

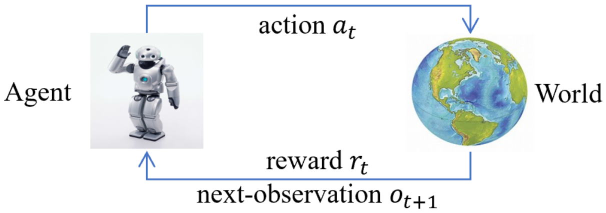


Figure 2.5: Interaction between an RL agent and the external environment.

2.3.1 Foundations

Reinforcement learning (RL) is a learning paradigm where an intelligent agent learns to make optimal decisions by interacting with an initially unknown environment (Sutton and Barto, 2018). Compared to supervised learning, a distinctive challenge in RL is to learn without a teacher (that is, without supervisory labels). As we will see, this will lead to algorithmic considerations that are often unique to RL.

As illustrated in Fig. 2.5, the agent-environment interaction is often modeled as a discrete-time Markov decision process, or MDP (Puterman, 1994), described by a five-tuple $M = \langle \mathcal{S}, \mathcal{A}, P, R, \gamma \rangle$:

- $\mathcal{S}$ is a possibly infinite set of states the environment can be in;
- $\mathcal{A}$ is a possibly infinite set of actions the agent can take in a state;
- $P(s'|s, a)$ gives the transition probability of the environment landing in a new state s' after action a is taken in state s;
- $R(s, a)$ is the average reward immediately received by the agent after taking action a in state s; and
- $\gamma \in (0, 1]$ is a discount factor.

The intersection can be recorded as a trajectory $(s_1, a_1, r_1, \ldots)$, generated as follows: at step $t = 1, 2, \ldots$,

- the agent observes the environment's current state $s_t \in \mathcal{S}$, and takes an action $a_t \in \mathcal{A}$;
- the environment transitions to a next-state s_{t+1}, distributed according to the transition probabilities $P(\cdot|s_t, a_t)$;
- associated with the transition is an immediate reward $r_t \in \mathbb{R}$, whose average is $R(s_t, a_t)$.

Omitting the subscript, each step results in a tuple (s, a, r, s') that is called a *transition*. The goal of an RL agent is to maximize the long-term reward by taking optimal actions (to be defined soon). Its action-selection policy, denoted by π, can be deterministic or stochastic. In either case, we use $a \sim \pi(s)$ to denote selection of action by following π in state s. Given a policy π, the value of a state s is the average discounted long-term reward from that state:

$$V^\pi(s) := \mathbb{E}[r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots | s_1 = s, a_i \sim \pi(s_i), \forall i \geq 1] \,.$$

We are interested in optimizing the policy so that V^π is maximized for *all* states. Denote by π^* an optimal policy, and V^* its corresponding value function (also known as the optimal value function). In many cases, it is more convenient to use another form of value function called the Q-function:

$$Q^\pi(s, a) := \mathbb{E}[r_1 + \gamma r_2 + \gamma^2 r_3 + \cdots | s_1 = s, a_1 = a, a_i \sim \pi(s_i), \forall i > 1] \,,$$

which measures the average discounted long-term reward by first selecting a in state s and then following policy π thereafter. The optimal Q-function, corresponding to an optimal policy, is denoted by Q^*.

2.3.2 Basic Algorithms

We now describe two popular classes of algorithms, exemplified by Q-learning and policy gradient, respectively.

Q-learning. The first family is based on the observation that an optimal policy can be immediately retrieved if the optimal Q-function is available. Specifically, the optimal policy can be determined by

$$\pi^*(s) = \arg\max_a Q^*(s, a)\,.$$

Therefore, a large family of RL algorithms focuses on learning $Q^*(s, a)$, and are collectively called *value function-based* methods.

In practice, it is expensive to represent $Q(s, a)$ by a table, one entry for each distinct (s, a), when the problem at hand is large. For instance, the number of states in the game of Go is larger than 2×10^{170} (Tromp and Farnebäck, 2006). Hence, we often use compact forms to represent Q. In particular, we assume the Q-function has a predefined parametric form, parameterized by some vector $\theta \in \mathbb{R}^d$. An example is linear approximation:

$$Q(s, a; \theta) = \phi(s, a)^{\mathrm{T}}\theta\,,$$

where $\phi(s, a)$ is a d-dimensional hand-coded feature vector for state-action pair (s, a), and θ is the corresponding coefficient vector to be learned from data. In general, $Q(s, a; \theta)$ may take different parametric forms. For example, in the case of Deep Q-Network (DQN), $Q(s, a; \theta)$ takes the form of deep neural networks, such as multi-layer perceptrons and convolutional networks (Tesauro, 1995; Mnih et al., 2015), recurrent network (Hausknecht and Stone, 2015; Li et al., 2015), etc. More examples will be seen in later chapters. Furthermore, it is possible to represent the Q-function in a non-parametric way, using decision trees (Ernst et al., 2005) or Gaussian processes (Engel et al., 2005), which is outside of the scope of this introductory section.

To learn the Q-function, we modify the parameter θ using the following update rule, after observing a state transition (s, a, r, s'):

$$\theta \leftarrow \theta + \alpha \underbrace{\left(r + \gamma \max_{a'} Q(s', a'; \theta) - Q(s, a; \theta)\right)}_{\text{“temporal difference”}} \nabla_\theta Q(s, a; \theta)\,. \tag{2.4}$$

The above update is known as Q-learning (Watkins, 1989), which applies a small change to θ, controlled by the step-size parameter α and computed from the *temporal difference* (Sutton, 1988).

While popular, in practice, Q-learning can be quite unstable and requires many samples before reaching a good approximation of Q^*. Two modifications are often helpful. The first is *experience replay* (Lin, 1992), popularized by Mnih et al. (2015). Instead of using an observed transition to update θ just *once* using Eqn. 2.4, one may store it in a replay buffer, and periodically sample transitions from it to perform Q-learning updates. This way, every transition can be used multiple times, thus increasing sample efficiency. Furthermore, it helps stabilize learning by preventing the data distribution from changing too quickly over time when updating parameter θ.

The second is a two-network implementation (Mnih et al., 2015), an instance of the more general fitted value iteration algorithm (Munos and Szepesvári, 2008). Here, the learner maintains an extra copy of the Q-function, called the *target network*, parameterized by θ_{target}. During learning, θ_{target} is fixed and is used to compute temporal difference to update θ. Specifically, Eqn. 2.4 now becomes:

$$\theta \leftarrow \theta + \alpha \underbrace{\left(r + \gamma \max_{a'} Q(s', a'; \theta_{\text{target}}) - Q(s, a; \theta)\right)}_{\text{temporal difference with a target network}} \nabla_\theta Q(s, a; \theta)\,. \tag{2.5}$$

Periodically, θ_{target} is updated to be θ, and the process continues.

There have been a number of recent improvements to the basic Q-learning described above, such as dueling Q-network (Wang et al., 2016), double Q-learning (van Hasselt et al., 2016), and a provably convergent SBEED algorithm (Dai et al., 2018b).

Policy Gradient. The other family of algorithms tries to optimize the policy directly, without having to learn the Q-function. Here, the policy itself is directly parameterized by $\theta \in \mathbb{R}^d$, and $\pi(s; \theta)$ is often a distribution over actions. Given any θ, the policy is naturally evaluated by the average long-term reward it gets in a trajectory of length H, $\tau = (s_1, a_1, r_1, \ldots, s_H, a_H, r_H)$:[5]

$$J(\theta) := \mathbb{E}\left[\sum_{t=1}^{H} \gamma^{t-1} r_t | a_t \sim \pi(s_t;\theta)\right] .$$

If it is possible to estimate the gradient $\nabla_\theta J$ from sampled trajectories, one can do stochastic gradient ascent[6] to maximize J:

$$\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)\,, \tag{2.6}$$

where α is again a stepsize parameter.

One such algorithm, known as REINFORCE (Williams, 1992), estimates the gradient as follows. Let τ be a length-H trajectory generated by $\pi(\cdot;\theta)$; that is, $a_t \sim \pi(s_t;\theta)$ for every t. Then, a stochastic gradient based on this single trajectory is given by

$$\nabla_\theta J(\theta) = \sum_{t=1}^{H-1} \gamma^{t-1} \left(\nabla_\theta \log \pi(a_t|s_t;\theta) \sum_{h=t}^{H} \gamma^{h-t} r_h\right) . \tag{2.7}$$

REINFORCE may suffer high variance in practice, as its gradient estimate depends directly on the sum of rewards along the entire trajectory. Its variance may be reduced by the use of an estimated value function of the current policy, often referred to as the critic in actor-critic algorithms (Sutton et al., 1999a; Konda and Tsitsiklis, 1999):

$$\nabla_\theta J(\theta) = \sum_{t=1}^{H-1} \gamma^{t-1} \left(\nabla_\theta \log \pi(a_t|s_t;\theta) \hat{Q}(s_t, a_t, h)\right) , \tag{2.8}$$

where $\hat{Q}(s, a, h)$ is an estimated value function for the current policy $\pi(s;\theta)$ that is used to approximate $\sum_{h=t}^{H} \gamma^{h-t} r_h$ in Eqn. 2.7. $\hat{Q}(s, a, h)$ may be learned by standard temporal difference methods (similar to Q-learning), but many variants exist. Moreover, there has been much work on methods to compute the gradient $\nabla_\theta J$ more effectively than Eqn. 2.8. Interested readers can refer to a few related works and the references therein for further details (Kakade, 2001; Peters et al., 2005; Schulman et al., 2015a,b; Mnih et al., 2016; Gu et al., 2017; Dai et al., 2018a; Liu et al., 2018b).

2.3.3 Exploration

So far we have described basic algorithms for updating either the value function or the policy, when transitions are given as input. Typically, an RL agent also has to determine how to select actions to collect desired transitions for learning. Always selecting the action ("exploitation") that seems best is problematic, as not selecting a novel action (that is, underrepresented, or even absent, in data collected so far), known as "exploration", may result in the risk of not seeing outcomes that are potentially better. Balancing exploration and exploitation efficiently is one of the unique challenges in reinforcement learning.

A basic exploration strategy is known as ϵ-greedy. The idea is to choose the action that looks best with high probability (for exploitation), and a random action with small probability (for exploration). In the case of DQN, suppose θ is the current parameter of the Q-function, then the action-selection rule for state s is given as follows:

$$a_t = \begin{cases} \arg\max_a Q(s_t, a; \theta) & \text{with probability } 1 - \epsilon \\ \text{random action} & \text{with probability } \epsilon\,. \end{cases}$$

In many problems this simple approach is effective (although not necessarily optimal). A further discussion is found in Sec. 4.4.2.

---

[5] We describe policy gradient in the simpler bounded-length trajectory case, although it can be extended to problems when the trajectory length is unbounded (Baxter and Bartlett, 2001; Baxter et al., 2001).

[6] Stochastic gradient *ascent* is simply stochastic gradient *descent* on the negated objective function.

Chapter 3

Question Answering and Machine Reading Comprehension

Recent years have witnessed an increasing demand for conversational Question Answering (QA) agents that allow users to query a large-scale Knowledge Base (KB) or a document collection in natural language. The former is known as KB-QA agents and the latter text-QA agents. KB-QA agents are more flexible and user-friendly than traditional SQL-like systems in that users can query a KB interactively without composing complicated SQL-like queries. Text-QA agents are much easier to use in mobile devices than traditional search engines, such as Bing and Google, in that they provide concise, direct answers to user queries, as opposed to a ranked list of relevant documents.

It is worth noting that multi-turn, conversational QA is an emerging research topic, and is not as well-studied as single-turn QA. Many papers reviewed in this chapter are focused on the latter. However, single-turn QA is an indispensable building block for all sorts of dialogues (e.g., chitchat and task-oriented), deserving our full attention if we are to develop real-world dialogue systems.

In this chapter, we start with a review of KB and symbolic approaches to KB-QA based on semantic parsing. We show that a symbolic system is hard to scale because the keyword-matching-based, query-to-answer inference used by the system is inefficient for a very large KB, and is not robust to paraphrasing. To address these issues, neural approaches are developed to represent queries and KB using continuous semantic vectors so that the inference can be performed at the semantic level in a compact neural space. We also describe the typical architecture of multi-turn, conversational KB-QA agents, using a movie-on-demand agent as an example, and review several conversational KB-QA datasets developed recently.

We then discuss neural text-QA agents. The heart of these systems is a neural Machine Reading Comprehension (MRC) model that generates an answer to an input question based on a (set of) passage(s). After reviewing popular MRC datasets and TREC text-QA open benchmarks, we describe the technologies developed for state-of-the-art MRC models along two dimensions: (1) the methods of encoding questions and passages as vectors in a neural space, and (2) the methods of performing reasoning in the neural space to generate the answer. We also describe the architecture of multi-turn, conversational text-QA agents, and the way MRC tasks and models are extended to conversational QA.

3.1 Knowledge Base

Organizing the world's facts and storing them in a structured database, large scale Knowledge Bases (KB) like DBPedia (Auer et al., 2007), Freebase (Bollacker et al., 2008) and Yago (Suchanek et al., 2007) have become important resources for supporting open-domain QA.

A typical KB consists of a collection of subject-predicate-object triples (s, r, t) where $s, t \in \mathcal{E}$ are entities and $r \in \mathcal{R}$ is a predicate or relation. A KB in this form is often called a Knowledge Graph

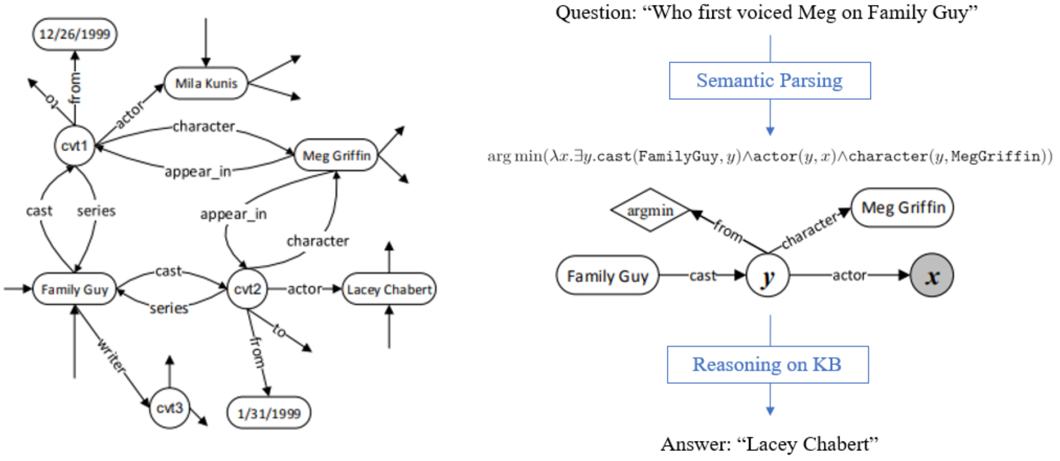


Figure 3.1: An example of semantic parsing for KB-QA. (Left) A subgraph of Freebase related to the TV show `Family Guy`. (Right) A question, its logical form in λ-calculus and query graph, and the answer. Figures adapted from Yih et al. (2015a).

(KG) due to its graphical representation, i.e., the entities are nodes and the relations the directed edges that link the nodes.

Fig. 3.1 (Left) shows a small subgraph of Freebase related to the TV show `Family Guy`. Nodes include some names, dates and special Compound Value Type (CVT) entities.[1] A directed edge describes the relation between two entities, labeled by a predicate.

3.2 Semantic Parsing for KB-QA

Most state-of-the-art symbolic approaches to KB-QA are based on semantic parsing, where a question is mapped to its formal meaning representation (e.g., logical form) and then translated to a KB query. The answers to the question can then be obtained by finding a set of paths in the KB that match the query and retrieving the end nodes of these paths (Richardson et al., 1998; Berant et al., 2013; Yao and Van Durme, 2014; Bao et al., 2014; Yih et al., 2015b).

We take the example used in Yih et al. (2015a) to illustrate the QA process. Fig. 3.1 (Right) shows the logical form in λ-calculus and its equivalent graph representation, known as *query graph*, of the question "Who first voiced Meg on Family Guy?". Note that the query graph is grounded in Freebase. The two entities, `MegGriffin` and `FamilyGuy`, are represented by two rounded rectangle nodes. The circle node y means that there should exist an entity describing some casting relations like the character, actor and the time she started the role. y is grounded in a CVT entity in this case. The shaded circle node x is also called the *answer node*, and is used to map entities retrieved by the query. The diamond node $\arg\min$ constrains that the answer needs to be the earliest actor for this role. Running the query graph without the aggregation function against the graph as in Fig. 3.1 (Left) will match both `LaceyChabert` and `MilaKunis`. But only `LaceyChabert` is the correct answer as she started this role earlier (by checking the from property of the grounded CVT node).

Applying a symbolic KB-QA system to a very large KB is challenging for two reasons:

- **Paraphrasing in natural language**: This leads to a wide variety of semantically equivalent ways of stating the same question, and in the KB-QA setting, this may cause mismatches between the natural language questions and the label names (e.g., predicates) of the nodes and edges used in the KB. As in the example of Fig. 3.1, we need to measure how likely the predicate used in the question matches that in Freebase, such as "Who first *voiced* Meg on Family Guy?" vs. `cast-actor`. Yih et al. (2015a) proposed to use a learned DSSM

---

[1] CVT is not a real-world entity, but is used to collect multiple fields of an event or a special relationship.

described in Sec. 2.2.2, which is conceptually an embedding-based method we will review in Sec. 3.3.

- **Search complexity**: Searching all possible multi-step (compositional) relation paths that match complex queries is prohibitively expensive because the number of candidate paths grows exponentially with the path length. We will review symbolic and neural approaches to multi-step reasoning in Sec. 3.4.

3.3 Embedding-based Methods

To address the paraphrasing problem, embedding-based methods map entities and relations in a KB to continuous vectors in a neural space; see, e.g., Bordes et al. (2013); Socher et al. (2013); Yang et al. (2015); Yih et al. (2015b). This space can be viewed as a hidden semantic space where various expressions with the same semantic meaning map to the same continuous vector.

Most KB embedding models are developed for the Knowledge Base Completion (KBC) task: predicting the existence of a triple (s, r, t) that is not seen in the KB. This is a simpler task than KB-QA since it only needs to predict whether a fact is true or not, and thus does not suffer from the search complexity problem.

The bilinear model is one of the basic KB embedding models (Yang et al., 2015; Nguyen, 2017). It learns a vector $\mathbf{x}_e \in \mathbb{R}^d$ for each entity $e \in \mathcal{E}$ and a matrix $\mathbf{W}_r \in \mathbb{R}^{d\times d}$ for each relation $r \in \mathcal{R}$. The model scores how likely a triple (s, r, t) holds using

$$\text{score}(s, r, t; \theta) = \mathbf{x}_s^\top \mathbf{W}_r \mathbf{x}_t. \tag{3.1}$$

The model parameters θ (i.e., the embedding vectors and matrices) are trained on pair-wise training samples in a similar way to that of the DSSM described in Sec. 2.2.2. For each positive triple (s, r, t) in the KB, denoted by x^+, we construct a set of negative triples x^- by corrupting s, t, or r. The training objective is to minimize the pair-wise rank loss of Eqn. 2.2, or more commonly the margin-based loss defined as

$$L(\theta) = \sum_{(x^+,x^-)\in\mathcal{D}} \left[\gamma + \text{score}(x^-;\theta) - \text{score}(x^+;\theta)\right]_+,$$

where $[x]_+ := \max(0, x)$, γ is the margin hyperparameter, and $\mathcal{D}$ the training set of triples.

These basic KB models have been extended to answer multi-step relation queries, as known as *path queries*, e.g., "Where did Tad Lincoln's parents live?" (Toutanova et al., 2016; Guu et al., 2015; Neelakantan et al., 2015). A path query consists of an initial anchor entity s (e.g., `TadLincoln`), followed by a sequence of relations to be traversed $(r_1, ..., r_k)$ (e.g., `(parents, location)`). We can use vector space compositions to combine the embeddings of individual relations r_i into an embedding of the path $(r_1, ..., r_k)$. The natural composition of the bilinear model of Eqn. 3.1 is matrix multiplication. Thus, to answer how likely a path query (q, t) holds, where $q = (s, r_1, ..., r_k)$, we would compute

$$\text{score}(q, t) = \mathbf{x}_s^\top \mathbf{W}_{r_1} ... \mathbf{W}_{r_k} \mathbf{x}_t. \tag{3.2}$$

These KB embedding methods are shown to have good generalization performance in terms of validating unseen facts (e.g., triples and path queries) given an existing KB. Interested users are referred to Nguyen (2017) for a detailed survey of embedding models for KBC.

3.4 Multi-Step Reasoning on KB

Knowledge Base Reasoning (KBR) is a subtask of KB-QA. As described in Sec. 3.2, KB-QA is performed in two steps: (1) semantic parsing translates a question into a KB query, then (2) KBR traverses the query-matched paths in a KB to find the answers.

To reason over a KB, for each relation $r \in \mathcal{R}$, we are interested in learning a set of first-order logical rules in the form of *relational paths*, $\pi = (r_1, ..., r_k)$. For the KBR example in Fig. 3.2, given the question "What is the citizenship of Obama?", its translated KB query in the form of subject-predicate-object triple is `(Obama, citizenship, ?)`. Unless the triple `(Obama, citizenship,`

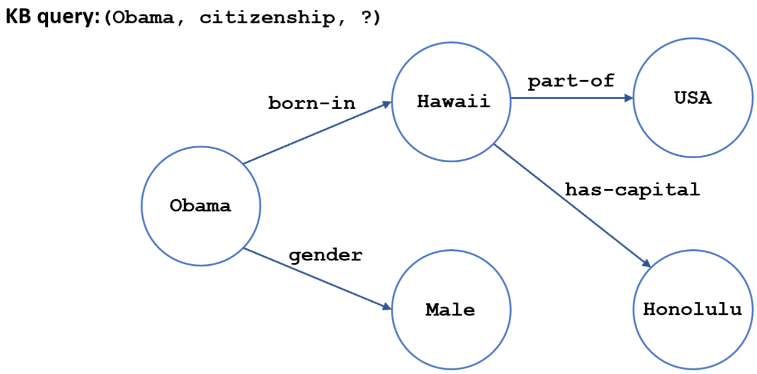


Figure 3.2: An example of knowledge base reasoning (KBR). We want to identify the answer node `USA` for a KB query `(Obama, citizenship, ?)`. Figure adapted from Shen et al. (2018).

Table 3.1: A sample of relational paths learned by PRA. For each relation, its top-2 PRA paths are presented, adapted from Lao et al. (2011).

ID	PRA Path # Comment
	`athlete-plays-for-team`
1	`(athlete-plays-in-league, league-players, athlete-plays-for-team)`
	# teams with many players in the athlete's league
2	`(athlete-plays-in-league, league-teams, team-against-team)`
	# teams that play against many teams in the athlete's league
	`stadium-located-in-city`
1	`(stadium-home-team,team-home-stadium,stadium-located-in-city)`
	# city of the stadium with the same team
2	`(latitude-longitude,latitude-longitude-of, stadium-located-in-city)`
	# city of the stadium with the same location
	`team-home-stadium`
1	`(team-plays-in-city,city-stadium)`
	# stadium located in the same city with the query team
2	`(team-member,athlete-plays-for-team,team-home-stadium)`
	# home stadium of teams which share players with the query
	`team-plays-in-league`
1	`(team-plays-sport,players,athlete-players-in-league)`
	# the league that the query team's members belong to
2	`(team-plays-against-team,team-players-in-league)`
	# the league that query team's competing team belongs to

`USA)` is explicitly stored in the KB,[2] a multi-step reasoning procedure is needed to deduce the answer from the paths that contain relevant triples, such as `(Obama, born-in, Hawaii)` and `(Hawaii, part-of, USA)`, using the learned relational paths such as `(born-in, part-of)`.

Below, we describe three categories of multi-step KBR methods. They differ in whether reasoning is performed in a discrete symbolic space or a continuous neural space.

3.4.1 Symbolic Methods

The Path Ranking Algorithm (PRA) (Lao and Cohen, 2010; Lao et al., 2011) is one of the primary symbolic approaches to learning relational paths in large KBs. PRA uses random walks with restarts to perform multiple bounded depth-first search to find relational paths. Table 3.1 shows a sample of

---

[2] As pointed out by Nguyen (2017), even very large KBs, such as Freebase and DBpedia, which contain billions of fact triples about the world, are still far from complete.

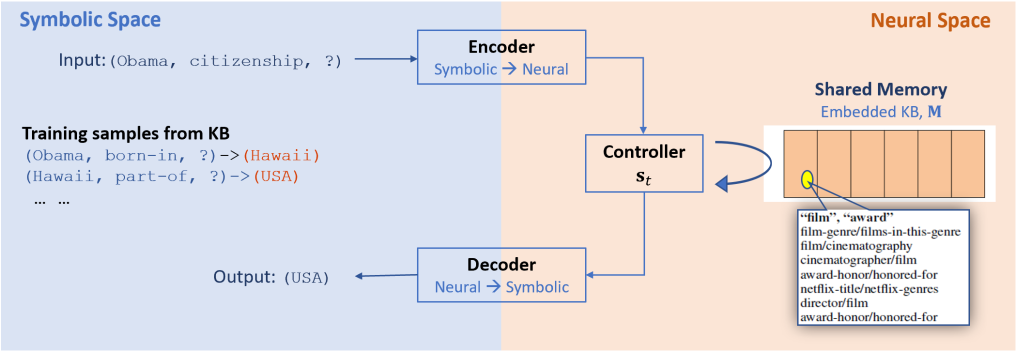


Figure 3.3: An overview of the neural methods for KBR (Shen et al., 2017a; Yang et al., 2017a). The KB is embedded in neural space as matrix $\mathbf{M}$ that is learned to store compactly the connections between related triples (e.g., the relations that are semantically similar are stored as a cluster). The controller is designed to adaptively produce lookup sequences in $\mathbf{M}$ and decide when to stop, and the encoder and decoder are responsible for the mapping between the symbolic and neural spaces.

relational paths learned by PRA. A relational path is a sequence $\pi = (r_1, ..., r_k)$. An instance of the relational path is a sequence of nodes $e_1, ..., e_{k+1}$ such that (e_i, r_i, e_{i+1}) is a valid triple.

During KBR, given a query $q = (s, r, ?)$, PRA selects the set of relational paths for r, denoted by $\mathcal{B}_r = \{\pi_1, \pi_2, ...\}$, then traverses the KB according to the query and $\mathcal{B}_r$, and scores each candidate answer t using a linear model

$$\text{score}(q, t) = \sum_{\pi \in \mathcal{B}_r} \lambda_\pi P(t|s, \pi) \,, \tag{3.3}$$

where λ_π's are the learned weights, and $P(t|s, \pi)$ is the probability of reaching t from s by a random walk that instantiates the relational path π, also known as a *path constrained random walk*.

Because PRA operates in a fully discrete space, it does not take into account semantic similarities among relations. As a result, PRA can easily produce millions of categorically distinct paths even for a small path length, which not only hurts generalization but makes reasoning prohibitively expensive. To reduce the number of relational paths that need to be considered in KBR, Lao et al. (2011) used heuristics (e.g., requiring that a path be included in PRA only if it retrieves at least one target entity in the training data) and added an L_1 regularization term in the loss function for training the linear model of Eqn. 3.3. Gardner et al. (2014) proposed a modification to PRA that leverages the KB embedding methods, as described in Sec. 3.3, to collapse and cluster PRA paths according to their relation embeddings.

3.4.2 Neural Methods

Implicit ReasoNet (IRN) (Shen et al., 2016, 2017a) and Neural Logic Programming (Neural LP) (Yang et al., 2017a) are proposed to perform multi-step KBR in a neural space and achieve state-of-the-art results on popular benchmarks. The overall architecture of these methods is shown in Fig. 3.3, which can be viewed as an instance of the neural approaches illustrated in Fig. 1.4 (Right). In what follows, we use IRN as an example to illustrate how these neural methods work. IRN consists of four modules: encoder, decoder, shared memory, and controller, as in Fig. 3.3.

Encoder and Decoder These two modules are task-dependent. Given an input query $(s, r, ?)$, the encoder maps s and r, respectively, into their embedding vectors and then concatenates the two vectors to form the initial hidden state vector $\mathbf{s}_0$ of the controller. The use of vectors rather than matrices for relation representations is inspired by the bilinear-diag model (Yang et al., 2015), which restricts the relation representations to the class of diagonal matrices.

The decoder outputs a prediction vector $\mathbf{o} = \tanh(\mathbf{W}_o^\top \mathbf{s}_t + \mathbf{b}_o)$, a nonlinear projection of state $\mathbf{s}$ at time t, where $\mathbf{W}_o$ and $\mathbf{b}_o$ are the weight matrix and bias vector, respectively. In KBR, we can map the answer vector $\mathbf{o}$ to its answer node (entity) o in the symbolic space based on L_1 distance as $o = \arg\min_{e \in \mathcal{E}} \|\mathbf{o} - \mathbf{x}_e\|_1$, where $\mathbf{x}_e$ is the embedding vector of entity e.

Shared Memory The shared memory $\mathbf{M}$ is differentiable, and consists of a list of vectors $\{\mathbf{m}_i\}_{1\le i\le|\mathbf{M}|}$ that are randomly initialized and updated through back-propagation in training. $\mathbf{M}$ stores a compact version of KB optimized for the KBR task. That is, each vector represents a concept (a cluster of relations or entities) and the distance between vectors represents the semantic relatedness of these concepts. For example, the system may fail to answer the question `(Obama, citizenship, ?)` even if it finds the relevant facts in $\mathbf{M}$, such as `(Obama, born-in, Hawaii)` and `(Hawaii, part-of, USA)`, because it does not know that `bore-in` and `citizenship` are semantically related relations. In order to correct the error, $\mathbf{M}$ needs to be updated using the gradient to encode the piece of new information by moving the two relation vectors closer to each other in the neural space.

Controller The controller is implemented as an RNN. Given initial state $\mathbf{s}_0$, it uses attention to iteratively lookup and fetch information from $\mathbf{M}$ to update the state $\mathbf{s}_t$ at time t according to Eqn. 3.4, until it decides to terminate the reasoning process and calls the decoder to generate the output.

$$\begin{aligned} a_{t,i} &= \frac{\exp\left(\lambda\cos(\mathbf{W}_1^\top\mathbf{m}_i, \mathbf{W}_2^\top\mathbf{s}_t)\right)}{\sum_k \exp\left(\lambda\cos(\mathbf{W}_1^\top\mathbf{m}_k, \mathbf{W}_2^\top\mathbf{s}_t)\right)}, \\ \mathbf{x}_t &= \sum_i^{|\mathbf{M}|} a_{t,i}\mathbf{m}_i, \\ \mathbf{s}_{t+1} &= g(\mathbf{W}_3^\top\mathbf{s}_t + \mathbf{W}_4^\top\mathbf{x}_t), \end{aligned} \tag{3.4}$$

where $\mathbf{W}$'s are learned projection matrices, λ a scaling factor and g a nonlinear activation function.

The reasoning process of IRN can be viewed as a Markov Decision Process (MDP), as illustrated in Sec. 2.3.1. The step size in the information lookup and fetching sequence of Eqn. 3.4 is not given by training data, but is decided by the controller on the fly. More complex queries need more steps. Thus, IRN learns a stochastic policy to get a distribution over termination and prediction actions by the REINFORCE algorithm (Williams, 1992), which is described in Sec. 2.3.2 and Eqn. 2.7. Since all the modules of IRN are differentiable, IRN is an end-to-end differentiable neural model whose parameters, including the embedded KB matrix $\mathbf{M}$, can be jointly optimized using SGD on the training samples derived from a KB, as shown in Fig. 3.3.

As outlined in Fig. 1.4, neural methods operate in a continuous neural space, and do not suffer from the problems associated with symbolic methods. They are robust to paraphrase alternations because knowledge is implicitly represented by semantic classes via continuous vectors and matrices. They also do not suffer from the search complexity issue even with complex queries (e.g.path queries) and a very large KB because they reason over a compact representation of a KB (e.g., the matrix $\mathbf{M}$ in the shared memory in IRN) rather than the KB itself.

One of the major limitations of these methods is the lack of interpretability. Unlike PRA which traverses the paths in the graph explicitly as Eqn. 3.3, IRN does not follow explicitly any path in the KB during reasoning but performs lookup operations over the shared memory iteratively using the RNN controller with attention, each time using the revised internal state $\mathbf{s}$ as a query for lookup. It remains challenging to recover the symbolic representations of queries and paths (or first-order logical rules) from the neural controller. See (Shen et al., 2017a; Yang et al., 2017a) for some interesting preliminary results of interpretation of neural methods.

3.4.3 Reinforcement Learning based Methods

DeepPath (Xiong et al., 2017), MINERVA (Das et al., 2017b) and M-Walk (Shen et al., 2018) are among the state-of-the-art methods that use RL for multi-step reasoning over a KB. They use a policy-based agent with continuous states based on KB embeddings to traverse the knowledge graph to identify the answer node (entity) for an input query. The RL-based methods are as robust as the neural methods due to the use of continuous vectors for state representation, and are as interpretable as symbolic methods because the agents explicitly traverse the paths in the graph.

We formulate KBR as an MDP defined by the tuple $(\mathcal{S}, \mathcal{A}, R, \mathcal{P})$, where $\mathcal{S}$ is the continuous state space, $\mathcal{A}$ the set of available actions, $\mathcal{P}$ the state transition probability matrix, and R the reward function. Below, we follow M-Walk to describe these components in detail. We denote a KB as graph $\mathcal{G}(\mathcal{E}, \mathcal{R})$ which consists a collection of entity nodes $\mathcal{E}$ and the relation edges $\mathcal{R}$ that link the nodes. We denote a KB query as $q = (e_0, r, ?)$, where e_0 and r are the given source node and relation, respectively, and ? the answer node to be identified.

States Let s_t denote the state at time t, which encodes information of all traversed nodes up to t, all the previous selected actions and the initial query q. s_t can be defined recursively as follows:

$$\begin{aligned} s_0 &:= \{q, \mathcal{R}_{e_0}, \mathcal{E}_{e_0}\}, \\ s_t &= s_{t-1} \cup \{a_{t-1}, e_t, \mathcal{R}_{e_t}, \mathcal{E}_{e_t}\}, \end{aligned} \tag{3.5}$$

where $a_t \in \mathcal{A}$ is the action selected by the agent at time t, e_t is the currently visited node, $\mathcal{R}_{e_t} \in \mathcal{R}$ is the set of all the edges connected to e_t, and $\mathcal{E}_{e_t} \in \mathcal{E}$ is the set of all the nodes connected to e_t. Note that in RL-based methods, s_t is represented as a continuous vector using e.g., a RNN in M-Walk and MINERVA or a MLP in DeepPath.

Actions Based on s_t, the agent selects one of the following actions: (1) choosing an edge in $\mathcal{E}_{e_t}$ and moving to the next node $e_{t+1} \in \mathcal{E}$, or (2) terminating the reasoning process and outputting the current node e_t as a prediction of the answer node e_T.

Transitions The transitions are deterministic. As shown in Fig. 3.2, once action a_t is selected, the next node e_{t+1} and its associated $\mathcal{E}_{e_{t+1}}$ and $\mathcal{R}_{e_{t+1}}$ are known.

Rewards We only have the terminal reward of $+1$ if e_T is the correct answer, and 0 otherwise.

Policy Network The policy $\pi_\theta(a|s)$ denotes the probability of selecting action a given state s, and is implemented as a neural network parameterized by θ. The policy network is optimized to maximize $\mathbb{E}[V_\theta(s_0)]$, which is the long-term reward of starting from s_0 and following the policy π_θ afterwards. In KBR, the policy network can be trained using RL, such as the REINFORCE method, from the training samples in the form of triples (e_s, r, e_t) extracted from a KB. To address the reward sparsity issue (i.e., the reward is only available at the end of a path), Shen et al. (2018) proposed to use Monte Carlo Tree Search to generate a set of simulated paths with more positive terminal rewards by exploiting the fact that all the transitions are deterministic for a given knowledge graph.

3.5 Conversational KB-QA Agents

All of the KB-QA methods we have described so far are based on *single-turn* agents which assume that users can compose in one shot a complicated, compositional natural language query that can uniquely identify the answer in the KB.

However, in many cases, it is unreasonable to assume that users can construct compositional queries without prior knowledge of the structure of the KB to be queried. Thus, conversational KB-QA agents are more desirable because they allow users to query a KB interactively without composing complicated queries.

A conversational KB-QA agent is useful for many interactive KB-QA tasks such as movie-on-demand, where a user attempts to find a movie based on certain attributes of that movie, as illustrated by the example in Fig. 3.4, where the movie DB can be viewed as an entity-centric KB consisting of entity-attribute-value triples.

In addition to the core KB-QA engine which typically consists of a semantic parser and a KBR engine, a conversational KB-QA agent is also equipped with a Dialogue Manager (DM) which tracks the dialogue state and decides what question to ask to effectively help users navigate the KB in search of an entity (movie). The high-level architecture of the conversational agent for movie-on-demand is illustrated in Fig. 3.5. At each turn, the agent receives a natural language utterance u_t as input, and selects an action $a_t \in \mathcal{A}$ as output. The action space $\mathcal{A}$ consists of a set of questions, each for requesting the value of an attribute, and an action of informing the user with an ordered list of retrieved entities. The agent is a typical task-oriented dialogue system of Fig. 1.2 (Top), consisting of (1) a *belief tracker* module for resolving coreferences and ellipsis in user utterances using conversation context, identifying user intents, extracting associated attributes, and tracking the dialogue state; (2) an interface with the KB to query for relevant results (i.e., the *Soft-KB Lookup* component, which can be implemented using the KB-QA models described in the previous sections,

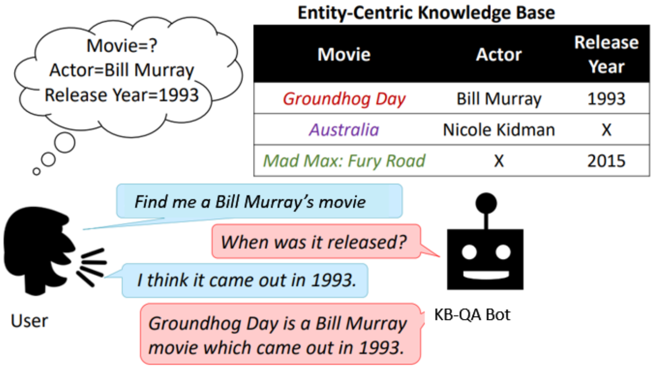


Figure 3.4: An interaction between a user and a multi-turn KB-QA agent for the movie-on-demand task. Figure credit: Dhingra et al. (2017).

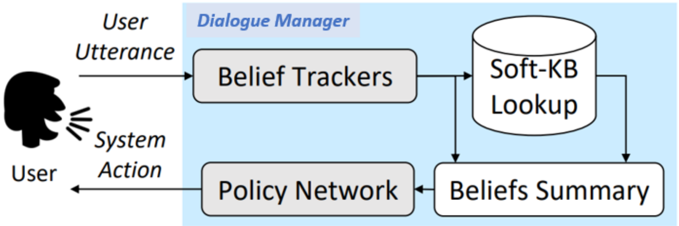


Figure 3.5: An overview of a conversational KB-QA agent. Figure credit: Dhingra et al. (2017).

except that we need to form the query based on the dialogue history captured by the belief tracker, not just the current user utterance, as described in Suhr et al. (2018)); (3) a *beliefs summary* module to summarize the state into a vector; and (4) a *dialogue policy* which selects the next action based on the dialogue state. The policy can be either programmed (Wu et al., 2015) or trained on dialogues (Wen et al., 2017; Dhingra et al., 2017).

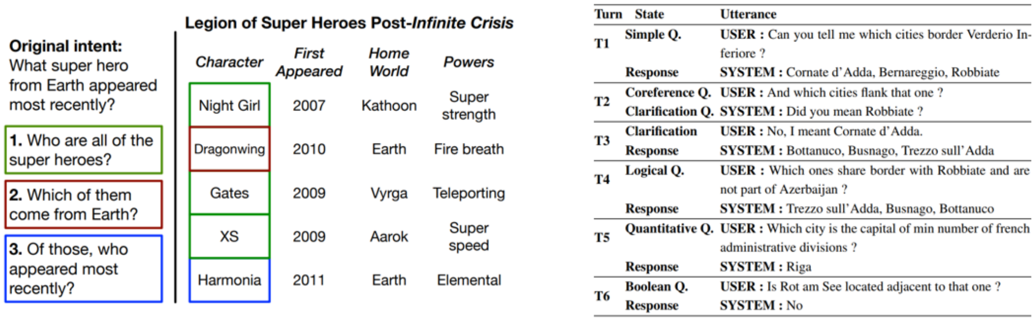
Wu et al. (2015) presented an Entropy Minimization Dialogue Management (EMDM) strategy. The agent always asks for the value of the attribute with maximum entropy over the remaining entries in the KB. EMDM is proved optimal in the absence of language understanding errors. However, it does not take into account the fact that some questions are easy for users to answer, whereas others are not. For example, in the movie-on-demand task, the agent could ask users to provide the movie release ID which is unique to each movie but is often unknown to regular users.

Dhingra et al. (2017) proposed KB-InfoBot – a fully neural end-to-end multi-turn dialogue agent for the movie-on-demand task. The agent is trained entirely from user feedback. It does not suffer from the problem of EMDM, and always asks users easy-to-answer questions to help search in the KB. Like all KB-QA agents, KB-InfoBot needs to interact with an external KB to retrieve real-world knowledge. This is traditionally achieved by issuing a symbolic query to the KB to retrieve entries based on their attributes. However, such symbolic operations break the differentiability of the system and prevent end-to-end training of the dialogue agent. KB-InfoBot addresses this limitation by replacing symbolic queries with an induced posterior distribution over the KB that indicates which entries the user is interested in. The induction can be achieved using the neural KB-QA methods described in the previous sections. Experiments show that integrating the induction process with RL leads to higher task success rate and reward in both simulations and against real users [3].

Recently, several datasets have been developed for building conversational KB-QA agents. Iyyer et al. (2017) collected a Sequential Question Answering (SQA) dataset via crowd sourcing by leveraging WikiTableQuestions (WTQ (Pasupat and Liang, 2015)), which contains highly compositional questions associated with HTML tables from Wikipedia. As shown in the example in Fig. 3.6 (Left),

---

[3] It remains to be verified whether the method can deal with large-scale KBs with millions of entities.



Original intent:
What super hero from Earth appeared most recently?

1. Who are all of the super heroes?
2. Which of them come from Earth?
3. Of those, who appeared most recently?

Legion of Super Heroes Post-*Infinite Crisis*

Character	*First Appeared*	*Home World*	*Powers*
Night Girl	2007	Kathoon	Super strength
Dragonwing	2010	Earth	Fire breath
Gates	2009	Vyrga	Teleporting
XS	2009	Aarok	Super speed
Harmonia	2011	Earth	Elemental

Turn	State	Utterance
T1	**Simple Q.**	**USER :** Can you tell me which cities border Verderio Inferiore ?
	Response	**SYSTEM :** Cornate d'Adda, Bernareggio, Robbiate
T2	**Coreference Q.**	**USER :** And which cities flank that one ?
	Clarification Q.	**SYSTEM :** Did you mean Robbiate ?
T3	**Clarification**	**USER :** No, I meant Cornate d'Adda.
	Response	**SYSTEM :** Bottanuco, Busnago, Trezzo sull'Adda
T4	**Logical Q.**	**USER :** Which ones share border with Robbiate and are not part of Azerbaijan ?
	Response	**SYSTEM :** Trezzo sull'Adda, Busnago, Bottanuco
T5	**Quantitative Q.**	**USER :** Which city is the capital of min number of french administrative divisions ?
	Response	**SYSTEM :** Riga
T6	**Boolean Q.**	**USER :** Is Rot am See located adjacent to that one ?
	Response	**SYSTEM :** No

Figure 3.6: The examples from two conversational KB-QA datasets. (Left) An example question sequence created from a compositional question intent in the SQA dataset. Figure credit: Iyyer et al. (2017). (Right) An example dialogue from the CSQA dataset. Figure credit: Saha et al. (2018).

each crowd sourcing task contains a long, complex question originally from WTQ as the question *intent*. The workers are asked to compose a sequence of simpler but inter-related questions that lead to the final intent. The answers to the simple questions are subsets of the cells in the table.

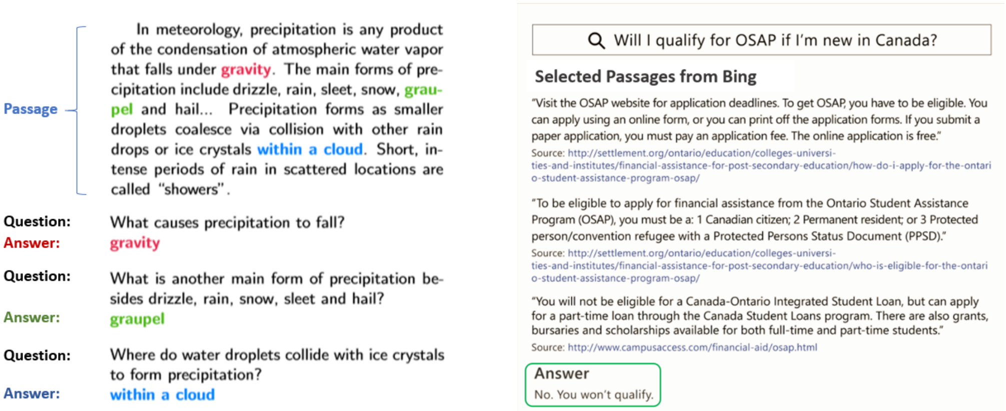
Saha et al. (2018) presented a dataset consisting of 200K QA dialogues for the task of Complex Sequence Question Answering (CSQA). CSQA combines two sub-tasks: (1) answering factoid questions through complex reasoning over a large-scale KB, and (2) learning to converse through a sequence of coherent QA pairs. As the example in Fig. 3.6 (Right) shows, CSQA calls for a conversational KB-QA agent that combines many technologies described in this chapter, including (1) parsing complex natural language queries (Sec. 3.2), (2) using conversation context to resolve coreferences and ellipsis in user utterances like the belief tracker in Fig. 3.5, (3) asking for clarification questions for ambiguous queries, like the dialogue manager in Fig. 3.5, and (4) retrieving relevant paths in the KB to answer questions (Sec. 3.4).

3.6 Machine Reading for Text-QA

Machine Reading Comprehension (MRC) is a challenging task: the goal is to have machines read a (set of) text passage(s) and then answer any question about the passage(s). The MRC model is the core component of text-QA agents.

The recent big progress on MRC is largely due to the availability of a multitude of large-scale datasets that the research community has created over various text sources such as Wikipedia (WikiReading (Hewlett et al., 2016), SQuAD (Rajpurkar et al., 2016), WikiHop (Welbl et al., 2017), DRCD (Shao et al., 2018)), news and other articles (CNN/Daily Mail (Hermann et al., 2015), NewsQA (Trischler et al., 2016), RACE (Lai et al., 2017), ReCoRD (Zhang et al., 2018d)), fictional stories (MCTest (Richardson et al., 2013), CBT (Hill et al., 2015), NarrativeQA (Kočisky et al., 2017)), science questions (ARC (Clark et al., 2018)), and general Web documents (MS MARCO (Nguyen et al., 2016), TriviaQA (Joshi et al., 2017), SearchQA (Dunn et al., 2017), DuReader (He et al., 2017b)).

SQuAD. This is the MRC dataset released by the Stanford NLP group. It consists of 100K questions posed by crowdworkers on a set of Wikipedia articles. As shown in the example in Fig. 3.7 (Left), the MRC task defined on SQuAD involves a question and a passage, and aims to find an answer span in the passage. For example, in order to answer the question "what causes precipitation to fall?", one might first locate the relevant part of the passage "precipitation ... falls under gravity", then reason that "under" refers to a cause (not location), and thus determine the correct answer: "gravity". Although the questions with span-based answers are more constrained than the real-world questions users submit to Web search engines such as Google and Bing, SQuAD provides a rich diversity of question and answer types and became one of the most widely used MRC datasets in the research community.



Passage

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

Question: What causes precipitation to fall?
Answer: gravity

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

Question: Where do water droplets collide with ice crystals to form precipitation?
Answer: within a cloud

Will I qualify for OSAP if I'm new in Canada?

Selected Passages from Bing

"Visit the OSAP website for application deadlines. To get OSAP, you have to be eligible. You can apply using an online form, or you can print off the application forms. If you submit a paper application, you must pay an application fee. The online application is free."
Source: http://settlement.org/ontario/education/colleges-universities-and-institutes/financial-assistance-for-post-secondary-education/how-do-i-apply-for-the-ontario-student-assistance-program-osap/

"To be eligible to apply for financial assistance from the Ontario Student Assistance Program (OSAP), you must be a: 1 Canadian citizen; 2 Permanent resident; or 3 Protected person/convention refugee with a Protected Persons Status Document (PPSD)."
Source: http://settlement.org/ontario/education/colleges-universities-and-institutes/financial-assistance-for-post-secondary-education/who-is-eligible-for-the-ontario-student-assistance-program-osap/

"You will not be eligible for a Canada-Ontario Integrated Student Loan, but can apply for a part-time loan through the Canada Student Loans program. There are also grants, bursaries and scholarships available for both full-time and part-time students."
Source: http://www.campusaccess.com/financial-aid/osap.html

Answer
No. You won't qualify.

Figure 3.7: The examples from two MRC datasets. (Left) Question-answer pairs for a sample passage in the SQuAD dataset, adapted from Rajpurkar et al. (2016). Each of the answers is a text span in the passage. (Right) A question-answer pair for a set of passages in the MS MARCO dataset, adapted from Nguyen et al. (2016). The answer, if there is one, is human generated.

MS MARCO. This is a large scale real-world MRC dataset, released by Microsoft, aiming to address the limitations of other academic datasets. For example, MS MARCO differs from SQuAD in that (1) SQuAD consists of the questions posed by crowdworkers while MS MARCO is sampled from the real user queries; (2) SQuAD uses a small set of high quality Wikipedia articles while MS MARCO is sampled from a large amount of Web documents, (3) MS MARCO includes some unanswerable queries[4] and (4) SQuAD requires identifying an answer span in a passage while MS MARCO requires generating an answer (if there is one) from multiple passages that may or may not be relevant to the given question. As a result, MS MARCO is far more challenging, and requires more sophisticated reading comprehension skills. As shown in the example in Fig. 3.7 (Right), given the question "will I qualify for OSAP if I'm new in Canada", one might first locate the relevant passage that includes: "you must be a 1 Canadian citizen; 2 permanent resident; or 3 protected person..." and reason that being new to the country is usually the opposite of being a citizen, permanent resident etc., thus determine the correct answer: "no, you won't qualify".

In addition, TREC[5] also provides a series of text-QA benchmarks:

The automated QA track. This is one of the most popular tracks in TREC for many years, up to year 2007 (Dang et al., 2007; Agichtein et al., 2015). It has focused on the task of providing automatic answers for human questions. The track primarily dealt with factual questions, and the answers provided by participants were extracted from a corpus of News articles. While the task evolved to model increasingly realistic information needs, addressing question series, list questions, and even interactive feedback, a major limitation remained: the questions did not directly come from real users, in real time.

The LiveQA track. This track started in 2015 (Agichtein et al., 2015), focusing on answering user questions in real time. Real user questions, i.e., fresh questions submitted on the Yahoo Answers (YA) site that have not yet been answered, were sent to the participant systems, which provided an answer in real time. Returned answers were judged by TREC editors on a 4-level Likert scale. LiveQA revived this popular QA track which has been frozen for several years, attracting significant attention from the QA research community.

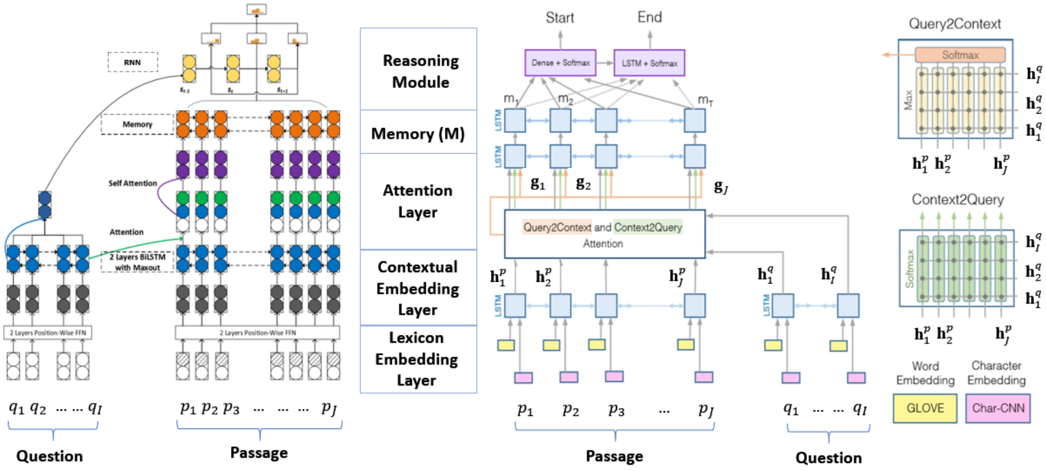


Figure 3.8: Two examples of state of the art neural MRC models. (Left) The Stochastic Answer Net (SAN) model. Figure credit: Liu et al. (2018d). (Right) The BiDirectional Attention Flow (BiDAF) model. Figure credit: Seo et al. (2016).

3.7 Neural MRC Models

The description in this section is based on the state of the art models developed on SQuAD, where given a question $Q = (q_1, ..., q_I)$ and a passage $P = (p_1, ..., p_J)$, we need to locate an answer span $A = (a_{start}, a_{end})$ in P.

In spite of the variety of model structures and attention types (Chen et al., 2016a; Xiong et al., 2016; Seo et al., 2016; Shen et al., 2017c; Wang et al., 2017b), a typical neural MRC model performs reading comprehension in three steps, as outlined in Fig. 1.4: (1) *encoding* the symbolic representation of the questions and passages into a set of vectors in a neural space; (2) *reasoning* in the neural space to identify the answer vector (e.g., in SQuAD, this is equivalent to ranking and re-ranking the embedded vectors of all possible text spans in P); and (3) *decoding* the answer vector into a natural language output in the symbolic space (e.g., this is equivalent to mapping the answer vector to its text span in P). Since the decoding module is straightforward for SQuAD models, we will focus on encoding and reasoning below.

Fig. 3.8 illustrate two examples of neural MRC models. BiDAF (Seo et al., 2016) is among the most widely used state of the art MRC baseline models in the research community, and SAN (Liu et al., 2018d) is the best documented MRC model on the SQuAD1.1 leaderboard[6] as of Dec. 19, 2017.

3.7.1 Encoding

Most MRC models encode questions and passages through three layers: a lexicon embedding layer, a contextual embedding layer, and an attention layer, as reviewed below.

Lexicon Embedding Layer. This extracts information from Q and P at the word level and normalizes for lexical variants. It typically maps each word to a vector space using a pre-trained word embedding model, such as word2vec (Mikolov et al., 2013) or GloVe (Pennington et al., 2014), such that semantically similar words are mapped to the vectors that are close to each other in the neural space (also see Sec. 2.2.1). Word embedding can be enhanced by concatenating each word embedding vector with other linguistic embeddings such as those derived from characters, Part-Of-Speech (POS) tags, and named entities etc. Given Q and P, the word embeddings for the tokens in Q is a matrix $\mathbf{E}^q \in \mathbb{R}^{d \times I}$ and tokens in P is $\mathbf{E}^p \in \mathbb{R}^{d \times J}$, where d is the dimension of word embeddings.

---

[4] SQuAD v2 (Rajpurkar et al., 2018) also includes unanswerable queries.

[5] `https://trec.nist.gov/data/qamain.html`

[6] `https://rajpurkar.github.io/SQuAD-explorer/`

Contextual Embedding Layer. This utilizes contextual cues from surrounding words to refine the embedding of the words. As a result, the same word might map to different vectors in a neural space depending on its context, such as "*bank* of a river" vs. " *bank* of America". This is typically achieved by using a Bi-directional Long Short-Term Memory (BiLSTM) network,[7] an extension of RNN of Fig. 2.2. As shown in Fig. 3.8, we place two LSTMs in both directions, respectively, and concatenate the outputs of the two LSTMs. Hence, we obtain a matrix $\mathbf{H}^q \in \mathbb{R}^{2d \times I}$ as a contextually aware representation of Q and a matrix $\mathbf{H}^p \in \mathbb{R}^{2d \times J}$ as a contextually aware representation of P.

ELMo (Peters et al., 2018) is one of the state of the art contextual embedding models. It is based on deep BiLSTM. Instead of using only the output layer representations of BiLSTM, ELMo combines the intermediate layer representations in the BiLSTM, where the combination weights are optimized on task-specific training data.

BERT (Devlin et al., 2018) differs from ELMo and BiLSTM in that it is designed to pre-train deep bidirection representations by *jointly* conditioning on both left and right context in all layers. The pre-trained BERT representations can be fine-tuned with just one additional output layer to create state of the art models for a wide range of NLP tasks, including MRC.

Since an RNN/LSTM is hard to train efficiently using parallel computing, Yu et al. (2018) presents a new contextual embedding model which does not require an RNN: Its encoder consists exclusively of convolution and self-attention, where convolution models local interactions and self-attention models global interactions. Such a model can be trained an order of magnitude faster than an RNN-based model on GPU clusters.

Attention Layer. This couples the question and passage vectors and produces a set of query-aware feature vectors for each word in the passage, and generates the working memory $\mathbf{M}$ over which reasoning is performed. This is achieved by summarizing information from both $\mathbf{H}^q$ and $\mathbf{H}^p$ via the *attention* process[8] that consists of the following steps:

1. Compute an attention score, which signifies which query words are most relevant to each passage word: $s_{ij} = \text{sim}_{\theta_s}(\mathbf{h}^q_i, \mathbf{h}^p_j) \in \mathbb{R}$ for each $\mathbf{h}^q_i$ in $\mathbf{H}^q$, where sim_{θ_s} is the similarity function e.g., a bilinear model, parameterized by θ_s.
2. Compute the normalized attention weights through softmax: $\alpha_{ij} = \exp(s_{ij})/\sum_k \exp(s_{kj})$.
3. Summarize information for each passage word via $\hat{\mathbf{h}}^p_j = \sum_i \alpha_{ij}\mathbf{h}^q_i$. Thus, we obtain a matrix $\hat{\mathbf{H}}^p \in \mathbb{R}^{2d \times J}$ as the question-aware representation of P.

Next, we form the working memory $\mathbf{M}$ in the neural space as $\mathbf{M} = f_\theta(\hat{\mathbf{H}}^p, \mathbf{H}^p)$, where f_θ is a function of fusing its input matrices, parameterized by θ. f_θ can be an arbitrary trainable neural network. For example, the fusion function in SAN includes a concatenation layer, a self-attention layer and a BiLSTM layer. BiDAF computes attentions in two directions: from passage to question $\hat{\mathbf{H}}^q$ as well as from question to passage $\hat{\mathbf{H}}^p$. The fusion function in BiDAF includes a layer that concatenates three matrices $\mathbf{H}^p$, $\hat{\mathbf{H}}^p$ and $\hat{\mathbf{H}}^q$, and a two-layer BiLSTM to encode for each word its contextual information with respect to the entire passage and the query.

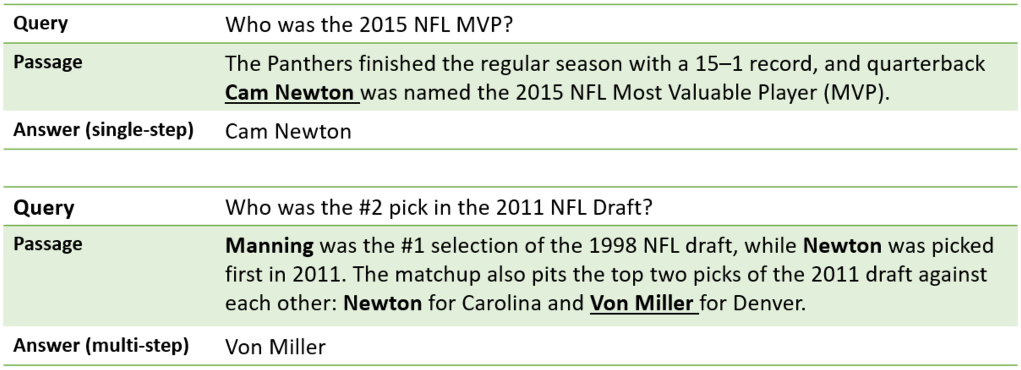
3.7.2 Reasoning

MRC models can be grouped into different categories based on how they perform reasoning to generate the answer. Here, we distinguish single-step models from multi-step models.

---

[7] Long Short-Term Memory (LSTM) networks are an extension for recurrent neural networks (RNNs). The units of an LSTM are used as building units for the layers of a RNN. LSTMs enable RNNs to remember their inputs over a long period of time because LSTMs contain their information in a gated cell, where gated means that the cell decides whether to store or delete information based on the importance it assigns to the information. The use of BiLSTM for contextual embedding is suggested by Melamud et al. (2016); McCann et al. (2017).

[8] Interested readers may refer to Table 1 in Huang et al. (2017) for a summarized view on the attention process used in several state of the art MRC models.



Query	Who was the 2015 NFL MVP?
Passage	The Panthers finished the regular season with a 15–1 record, and quarterback **Cam Newton** was named the 2015 NFL Most Valuable Player (MVP).
Answer (single-step)	Cam Newton

Query	Who was the #2 pick in the 2011 NFL Draft?
Passage	**Manning** was the #1 selection of the 1998 NFL draft, while **Newton** was picked first in 2011. The matchup also pits the top two picks of the 2011 draft against each other: **Newton** for Carolina and **Von Miller** for Denver.
Answer (multi-step)	Von Miller

Figure 3.9: (Top) A human reader can easily answer the question by reading the passage only once. (Bottom) A human reader may have to read the passage multiple times to answer the question.

Single-Step Reasoning. A single-step reasoning model matches the question and document only once and produces the final answers. We use the single-step version of SAN[9] in Fig. 3.8 (Left) as an example to describe the reasoning process. We need to find the answer span (i.e., the start and end points) over the working memory $\mathbf{M}$. First, a summarized question vector is formed as

$$\mathbf{h}^q = \sum_i \beta_i \mathbf{h}_i^q, \tag{3.6}$$

where $\beta_i = \exp(\mathbf{w}^\top \mathbf{h}_i^q) / \sum_k \exp(\mathbf{w}^\top \mathbf{h}_k^q)$, and $\mathbf{w}$ is a trainable vector. Then, a bilinear function is used to obtain the probability distribution of the start index over the entire passage by

$$\mathbf{p}^{(start)} = \text{softmax}(\mathbf{h}^{q\top} \mathbf{W}^{(start)} \mathbf{M}), \tag{3.7}$$

where $\mathbf{W}^{(start)}$ is a weight matrix. Another bilinear function is used to obtain the probability distribution of the end index, incorporating the information of the span start obtained by Eqn. 3.7, as

$$\mathbf{p}^{(end)} = \text{softmax}([\mathbf{h}^q; \sum_j p_j^{(start)} \mathbf{m}_j]^\top \mathbf{W}^{(end)} \mathbf{M}), \tag{3.8}$$

where the semicolon mark ; indicates the vector or matrix concatenation operator, $p_j^{(start)}$ is the probability of the j-th word in the passage being the start of the answer span, $\mathbf{W}^{(end)}$ is a weight matrix, and $\mathbf{m}_j$ is the j-th vector of $\mathbf{M}$.

Single-step reasoning is simple yet efficient and the model parameters can be trained using the classical back-propagation algorithm, thus it is adopted by most of the systems (Chen et al., 2016b; Seo et al., 2016; Wang et al., 2017b; Liu et al., 2017; Chen et al., 2017a; Weissenborn et al., 2017; Hu et al., 2017). However, since humans often solve question answering tasks by re-reading and re-digesting the document multiple times before reaching the final answer (this may be based on the complexity of the questions and documents, as illustrated by the examples in Fig. 3.9), it is natural to devise an iterative way to find answers as multi-step reasoning.

Multi-Step Reasoning. Multi-step reasoning models are pioneered by Hill et al. (2015); Dhingra et al. (2016); Sordoni et al. (2016); Kumar et al. (2016), who used a pre-determined fixed number of reasoning steps. Shen et al. (2017b,c) showed that multi-step reasoning outperforms single-step ones and dynamic multi-step reasoning further outperforms the fixed multi-step ones on two distinct MRC datasets (SQuAD and MS MARCO). But the dynamic multi-step reasoning models have to be trained using RL methods, e.g., policy gradient, which are tricky to implement due to the instability

---

[9] This is a special version of SAN where the maximum number of reasoning steps $T = 1$. SAN in Fig. 3.8 (Left) uses $T = 3$.

issue. SAN combines the strengths of both types of multi-step reasoning models. As shown in Fig. 3.8 (Left), SAN (Liu et al., 2018d) uses a fixed number of reasoning steps, and generates a prediction at each step. During decoding, the answer is based on the average of predictions in all steps. During training, however, SAN drops predictions via *stochastic dropout*, and generates the final result based on the average of the remaining predictions. Albeit simple, this technique significantly improves the robustness and overall accuracy of the model. Furthermore, SAN can be trained using back-propagation which is simple and efficient.

Taking SAN as an example, the multi-step reasoning module computes over T memory steps and outputs the answer span. It is based on an RNN, similar to IRN in Fig. 3.5. It maintains a state vector, which is updated on each step. At the beginning, the initial state $\mathbf{s}_0$ is the summarized question vector computed by Eqn. 3.6. At time step $t \in \{1, 2, \ldots, T\}$, the state is defined by $\mathbf{s}_t = \text{RNN}(\mathbf{s}_{t-1}, \mathbf{x}_t)$, where $\mathbf{x}_t$ contains retrieved information from memory using the previous state vector as a query via the attention process: $\mathbf{M}$: $\mathbf{x}_t = \sum_j \gamma_j \mathbf{m}_j$ and $\gamma = \text{softmax}(\mathbf{s}_{t-1}{}^\top \mathbf{W}^{(att)} \mathbf{M})$, where $\mathbf{W}^{(att)}$ is a trainable weight matrix. Finally, a bilinear function is used to find the start and end points of answer spans at each reasoning step t, similar to Eqn. 3.7 and 3.8:

$$\mathbf{p}_t^{(start)} = \text{softmax}(\mathbf{s}_t{}^\top \mathbf{W}^{(start)} \mathbf{M}), \tag{3.9}$$

$$\mathbf{p}_t^{(end)} = \text{softmax}([\mathbf{s}_t; \sum_j p_{t,j}^{(start)} \mathbf{m}_j]^\top \mathbf{W}^{(end)} \mathbf{M}), \tag{3.10}$$

where $p_{t,j}^{(start)}$ is the j-th value of the vector $\mathbf{p}_t^{(start)}$, indicating the probability of the j-th passage word being the start of the answer span at reasoning step t.

3.7.3 Training

A neural MRC model can be viewed as a deep neural network that includes all component modules (e.g., the embedding layers and reasoning engines) which by themselves are also neural networks. Thus, it can be optimized on training data in an end-to-end fashion via back-propagation and SGD, as outlined in Fig. 1.4. For SQuAD models, we optimize model parameters θ by minimizing the loss function defined as the sum of the negative log probabilities of the ground truth answer span start and end points by the predicted distributions, averaged over all training samples:

$$L(\theta) = -\frac{1}{|\mathcal{D}|} \sum_i^{|\mathcal{D}|} \left(\log \left(p^{(start)}_{y_i^{(start)}} \right) + \log \left(p^{(end)}_{y_i^{(end)}} \right) \right), \tag{3.11}$$

where $\mathcal{D}$ is the training set, $y_i^{(start)}$ and $y_i^{(end)}$ are the true start and end of the answer span of the i-th training sample, respectively, and p_k the k-th value of the vector $\mathbf{p}$.

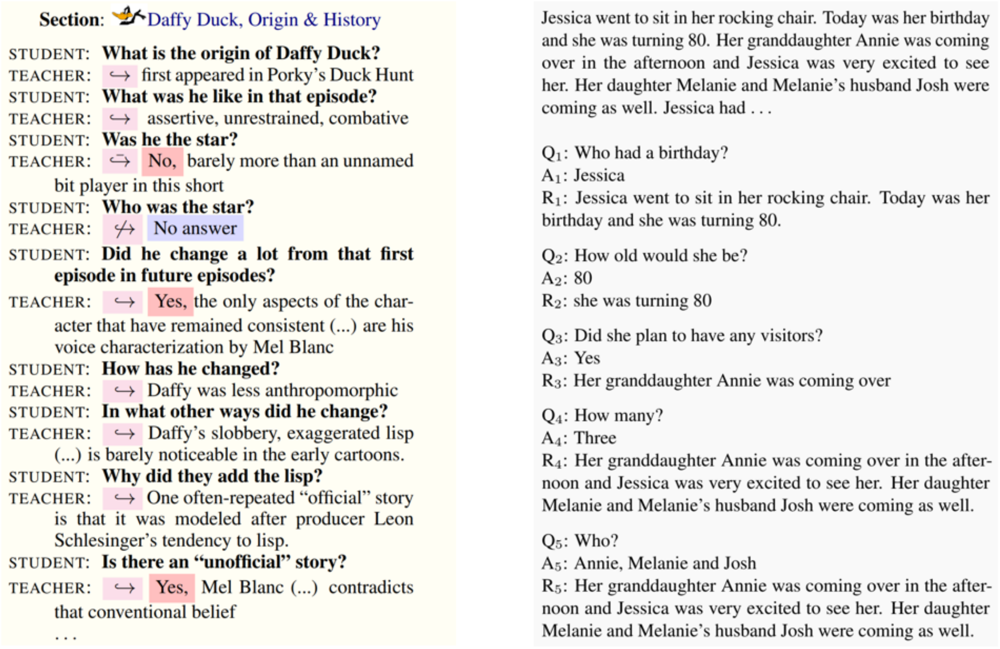
3.8 Conversational Text-QA Agents

While all the neural MRC models described in Sec. 3.7 assume a single-turn QA setting, in reality, humans often ask questions in a conversational context (Ren et al., 2018a). For example, a user might ask the question “when was California founded?”, and then depending on the received answer, follow up by “who is its governor?” and “what is the population?”, where both refer to “California” mentioned in the first question. This incremental aspect, although making human conversations succinct, presents new challenges that most state-of-the-art single-turn MRC models do not address directly, such as referring back to conversational history using coreference and pragmatic reasoning[10] (Reddy et al., 2018).

A conversational text-QA agent uses a similar architecture to Fig. 3.5, except that the *Soft-KB Lookup* module is replaced by a text-QA module which consists of a search engine (e.g., Google

---

[10] Pragmatic reasoning is defined as “the process of finding the intended meaning(s) of the given, and it is suggested that this amounts to the process of inferring the appropriate context(s) in which to interpret the given” (Bell, 1999). The analysis by Jia and Liang (2017); Chen et al. (2016a) revealed that state of the art neural MRC models, e.g., developed on SQuAD, mostly excel at matching questions to local context via lexical matching and paragraphing, but struggle with questions that require reasoning.



Section: Daffy Duck, Origin & History

STUDENT: **What is the origin of Daffy Duck?**
TEACHER: $\hookrightarrow$ first appeared in Porky's Duck Hunt
STUDENT: **What was he like in that episode?**
TEACHER: $\hookrightarrow$ assertive, unrestrained, combative
STUDENT: **Was he the star?**
TEACHER: $\overset{=}{\hookrightarrow}$ No, barely more than an unnamed bit player in this short
STUDENT: **Who was the star?**
TEACHER: $\not\hookrightarrow$ No answer
STUDENT: **Did he change a lot from that first episode in future episodes?**
TEACHER: $\hookrightarrow$ Yes, the only aspects of the character that have remained consistent (...) are his voice characterization by Mel Blanc
STUDENT: **How has he changed?**
TEACHER: $\hookrightarrow$ Daffy was less anthropomorphic
STUDENT: **In what other ways did he change?**
TEACHER: $\hookrightarrow$ Daffy's slobbery, exaggerated lisp (...) is barely noticeable in the early cartoons.
STUDENT: **Why did they add the lisp?**
TEACHER: $\hookrightarrow$ One often-repeated "official" story is that it was modeled after producer Leon Schlesinger's tendency to lisp.
STUDENT: **Is there an "unofficial" story?**
TEACHER: $\hookrightarrow$ Yes, Mel Blanc (...) contradicts that conventional belief
. . .

Jessica went to sit in her rocking chair. Today was her birthday and she was turning 80. Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie's husband Josh were coming as well. Jessica had . . .

Q_1: Who had a birthday?
A_1: Jessica
R_1: Jessica went to sit in her rocking chair. Today was her birthday and she was turning 80.

Q_2: How old would she be?
A_2: 80
R_2: she was turning 80

Q_3: Did she plan to have any visitors?
A_3: Yes
R_3: Her granddaughter Annie was coming over

Q_4: How many?
A_4: Three
R_4: Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie's husband Josh were coming as well.

Q_5: Who?
A_5: Annie, Melanie and Josh
R_5: Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie's husband Josh were coming as well.

Figure 3.10: The examples from two conversational QA datasets. (Left) A QA dialogue example in the QuAC dataset. The student, who does not see the passage (section text), asks questions. The teacher provides answers in the form of text spans and dialogue acts. These acts include (1) whether the student should $\hookrightarrow$, could $\overset{=}{\hookrightarrow}$, or should not $\not\hookrightarrow$ ask a follow-up; (2) affirmation (Yes / No), and, when appropriate, (3) No answer. Figure credit: Choi et al. (2018). (Right) A QA dialogue example in the CoQA dataset. Each dialogue turn contains a question (Q_i), an answer (A_i) and a rationale (R_i) that supports the answer. Figure credit: Reddy et al. (2018).

or Bing) that retrieves relevant passages for a given question, and an MRC model that generates the answer from the retrieved passages. The MRC model needs to be extended to address the aforementioned challenges in the conversation setting, henceforth referred to as a *conversational MRC model*.

Recently, several datasets have been developed for building conversational MRC models. Among them are CoQA (Conversational Question Answering (Reddy et al., 2018)) and QuAC (Question Answering in Context (Choi et al., 2018)), as shown in Fig. 3.10. The task of conversational MRC is defined as follows. Given a passage P, the conversation history in the form of question-answer pairs $\{Q_1, A_1, Q_2, A_2, ..., Q_{i-1}, A_{i-1}\}$ and a question Q_i, the MRC model needs to predict the answer A_i.

A conversational MRC model extends the models described in Sec. 3.7 in two aspects. First, the encoding module is extended to encode not only P and A_i but also the conversation history. Second, the reasoning module is extended to be able to generate an answer (via pragmatic reasoning) that might not overlap P. For example, Reddy et al. (2018) proposed a reasoning module that combines the text-span MRC model of DrQA (Chen et al., 2017a) and the generative model of PGNet (See et al., 2017). To generate a free-form answer, DrQA first points to the answer evidence in text (e.g., R5 in Fig. 3.10 (Right)), and PGNet generates the an answer (e.g., A5) based on the evidence.

Chapter 4

Task-oriented Dialogue Systems

This chapter focuses on task-oriented dialogue systems that assist users in solving a task. Different from applications where the user seeks an answer or certain information (previous chapter), dialogues covered here are often for completing a task, such as making a hotel reservation or booking movie tickets. Furthermore, compared to chatbots (next chapter), these dialogues often have a specific goal to achieve, and are typically domain dependent.

While task-oriented dialogue systems have been studied for decades, they have quickly gaining increasing interest in recent years, both in the research community and in industry. This chapter focuses on the foundation and algorithmic aspects, while industrial applications are discussed in Chapter 6. Furthermore, we restrict ourselves to dialogues where user input is in the form of raw text, not spoken language, but many of the techniques and discussions in this chapter can be adapted to spoken dialogues systems.

The chapter is organized as follows. It starts with an overview of basic concepts, terminology, and a typical architecture for task-oriented dialogue systems. Second, it reviews several representative approaches to dialogue system evaluation. This part is critical in the development cycle of dialogue systems, but is largely orthogonal to the concrete techniques used to build them. The next three sections focus on each of three main components in a typical dialogue system, with an emphasis on recent, neural approaches. Finally, we review several recent works on end-to-end dialogue systems, which are enabled by recent advances at the intersection of deep learning and reinforcement learning, with further discussions and pointers provided in the last section.

4.1 Overview

We start with a brief overview of task-oriented dialogue systems, focusing on aspects that facilitate later discussions. For more information and historical developments, readers are referred to the textbook of Jurafsky and Martin (2018, Chapters 25 & 26).

Slot-filling Dialogues

Throughout the chapter, we focus on a relatively simple yet important class of dialogues that involve filling in values for a predefined set of *slots* before a task can be successfully completed. These dialogues, known as *slot-filling* or *form-filling*, find a number of uses in practice. Table 4.1 gives an example conversation between a user and a dialogue system. To successfully complete a transaction to book tickets, the system must collect necessary information by conversing with the user.

Slot-filling dialogue may be applied to many other *domains*, such as movie (as shown in the example above), restaurant, air ticket booking, etc. For each domain, a set of *slots* are defined by domain experts and are application specific. For example, in the movie domain, slots like `movie_name`, `theater_name`, `time`, `date`, `ticket_price`, `num_tickets`, etc. are necessary.

Finally, a slot is called *informable* if the value for this slot can be used to constrain the conversation, such as `phone_number`; a slot is called *requestable* if the speaker can ask for its value, such

Table 4.1: An example movie-booking dialogue, adapted from Li et al. (2016d).

System	Hello! How may I assist you?
User	Can I get tickets for Zoolander 2 tomorrow?
System	Sure! Which city?
User	I want to watch it in Seattle.
System	How many tickets do you need?
User	Two, please. And I'd like to watch at Regal Meridian 16.
System	9:25 pm is available at Regal Meridian 16. Does it work for you?
User	Well, yes.
System	Okay, I was able to book 3 tickets for Zoolander 2 tomorrow at Regal Meridian 16 theater in Seattle at 9:25 pm.
User	Thank you.
System	Thank you. Good bye!

as `ticket_price`. Note that a slot can be both informable and requestable, an example being `movie_name`.

Dialogue Acts

The interaction between a dialogue agent and a user, as shown in the previous example, mirrors the interaction between an RL agent and the environment (Fig. 2.5), where a user utterance is the observation, and the system utterance is the action selected by the dialogue agent. The dialogue acts theory gives a formal foundation for this intuition (Core and Allen, 1997; Traum, 1999).

In this framework, the utterances of a user or agent are considered actions that can change the (mental) state of both the user and the system, thus the state of the conversation. These actions can be used to suggest, inform, request certain information, among others. A simple example dialogue act is `greet`, which corresponds to natural language sentences like "Hello! How may I assist you?". It allows the system to greet the user and start a conversation. Some dialogue acts may have slots or slot-value pairs as arguments. For example, the following question in the movie-booking example above:

"How many tickets do you need?"

is to request information about a certain slot:

`request(num_tickets)`,

while the following sentence

"I want to watch it in Seattle."

is to inform the city name:

`inform(city=''seattle'')`.

In general, dialogue acts are domain specific. Therefore, the set of dialogue acts in a movie domain, for instance, will be different from that in the restaurant domain (Schatzmann and Young, 2009).

Dialogue as Optimal Decision Making

Equipped with dialogue acts, we are ready to model multi-turn conversations between a dialogue agent and a user as an RL problem. Here, the dialogue system is the RL agent, and the user is the environment. At every turn of the dialogue,

- the agent keeps track of the dialogue state, based on information revealed so far in the conversation, and then takes an action; the action may be a response to the user in the form of dialogue acts, or an internal operation such as a database lookup or an API call;
- the user responds with the next utterance, which will be used by the agent to update its internal dialogue state in the next turn;
- an immediate reward is computed to measure the quality and/or cost for this turn of conversation.

This process is precisely the agent-environment interaction discussed in Sec. 2.3. We now discuss how a reward function is determined.

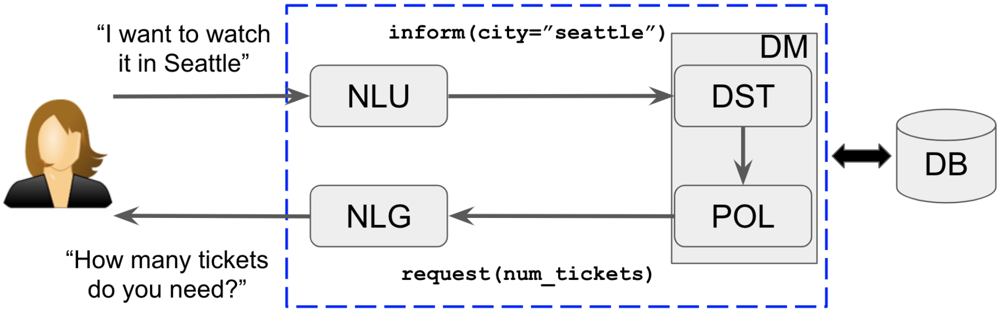

"I want to watch it in Seattle"
inform(city="seattle")
NLU
DM
DST
POL
DB
NLG
request(num_tickets)
"How many tickets do you need?"

Figure 4.1: An architecture for multi-turn task-oriented dialogues. It consists of the following modules: NLU (Natural Language Understanding), DM (Dialogue Manager), and NLG (Natural Language Generation). DM contains two sub-modules, DST (Dialogue State Tracker) and POL (Dialogue Policy). The dialogue system, indicated by the dashed rectangle, may have access to an external database (DB).

An appropriate reward function should capture desired features of a dialogue system. In task-oriented dialogues, we would like the system to succeed in helping the user in as few turns as possible. Therefore, it is natural to give a high reward (say $+20$) at the end of the conversation if the task is successfully solved, or a low reward (say -20) otherwise. Furthermore, we may give a small penalty (say, -1 reward) to every intermediate turn of the conversation, so that the agent is encouraged to make the dialogue as short as possible. The above is of course just a simplistic illustration of how to set a reward function for task-oriented dialogues, but in practice more sophisticated reward functions may be used, such as those that measure diversity and coherence of the conversation. Further discussion of the reward function can be found in Sections 4.4.6, 4.2.1 and 5.4.

To build a system, the pipeline architecture depicted in Fig. 4.1 is often used in practice. It consists of the following modules.

- Natural Language Understanding (NLU): This module takes the user's raw utterance as input and converts it to the semantic form of dialogue acts.
- Dialogue Manager (DM): This module is the central controller of the dialogue system. It often has a Dialogue State Tracking (DST) sub-module that is responsible for keeping track of the current dialogue state. The other sub-module, the policy, relies on the internal state provided by DST to select an action. Note that an action can be a response to the user, or some operation on backend databases (e.g., looking up certain information).
- Natural Language Generation (NLG): If the policy chooses to respond to the user, this module will convert this action, often a dialogue act, into a natural language form.

Dialogue Manager

There is a huge literature on building (spoken) dialogue managers. A comprehensive survey is out of the scope of the this chapter. Interested readers are referred to some of the earlier examples (Cole, 1999; Larsson and Traum, 2000; Rich et al., 2001; Allen et al., 2001; Bos et al., 2003; Bohus and Rudnicky, 2009), as well as excellent surveys like McTear (2002), Paek and Pieraccini (2008), and Young et al. (2013) for more information. Here, we review a small subset of traditional approaches from the decision-theoretic view we take in this paper.

Levin et al. (2000) viewed conversation as a decision making problem. Walker (2000) and Singh et al. (2002) are two early applications of reinforcement learning to manage dialogue systems. While promising, these approaches assumed that the dialogue state can only take finitely many possible values, and is fully observable (that is, the DST is perfect). Both assumptions are often violated in real-world applications, given ambiguity in user utterance and unavoidable errors in NLU.

To handle uncertainty inherent in dialogue systems, Roy et al. (2000) and Williams and Young (2007) proposed to use Partially Observable Markov Decision Process (POMDP) as a principled mathematical framework for modeling and optimizing dialogue systems. The idea is to take user utterances as observations to maintain a posterior distribution of the unobserved dialogue state; the distribution is sometimes referred to as the "belief state." Since exact optimization in POMDPs is computationally intractable, authors have studied approximation techniques (Roy et al., 2000; Williams and Young, 2007; Young et al., 2010; Li et al., 2009; Gašić and Young, 2014) and alternative representations such as the information states framework (Larsson and Traum, 2000; Daubigney et al., 2012). Still, compared to the neural approaches covered in later sections, these methods often require more domain knowledge to engineer features and design states.

Another important limitation of traditional approaches is that each module in Fig. 4.1 is often optimized separately. Consequently, when the system does not perform well, it can be challenging to solve the "credit assignment" problem, namely, to identify which component in the system causes undesired system response and needs to be improved. Indeed, as argued by McTear (2002), "[t]he key to a successful dialogue system is the integration of these components into a working system." The recent marriage of differentiable neural models and reinforcement learning allows a dialogue system to be optimized in an end-to-end fashion, potentially leading to higher conversation quality; see Sec. 4.6 for further discussions and recent works on this topic.

4.2 Evaluation and User Simulation

Evaluation has been an important research topic for dialogue systems. Different approaches have been used, including corpus-based approaches, user simulation, lab user study, actual user study, etc. We will discuss pros and cons of these various methods, and in practice trade-offs are made to find the best option or a combination of them.

4.2.1 Evaluation Metrics

While individual components in a dialogue system can often be optimized against more well-defined metrics such as accuracy, precision, recall, F1 and BLEU scores, evaluating a whole dialogue system requires a more holistic view and is more challenging (Walker et al., 1997, 1998, 2000; Paek, 2001; Hartikainen et al., 2004). In the reinforcement-learning framework, it implies that the reward function has to take multiple aspects of dialogue quality into consideration. In practice, the reward function is often a weighted linear combination of a subset of the following metrics.

The first class of metrics measures *task completion success*. The most common choice is perhaps *task success rate*—the fraction of dialogues that successfully solve the user's problem (buying the right movie tickets, finding proper restaurants, etc.). Effectively, the reward corresponding to this metric is 0 for every turn, except for the last turn where it is $+1$ for a successful dialogue and -1 otherwise. Many examples are found in the literature (Walker et al., 1997; Williams, 2006; Peng et al., 2017). Other variants have also been used, such as those to measure partial success (Singh et al., 2002; Young et al., 2016).

The second class measures cost incurred in a dialogue, such as time elapsed. A simple yet useful example is the number of turns, which reflects the intuition that a more succinct dialogue is preferred with everything else being equal. The reward is simply -1 per turn, although more complicated choices exist (Walker et al., 1997).

In addition, other aspects of dialogue quality may also be encoded into the reward function, although this is a relatively under-investigated direction. In the context of chatbots (Chapter 5), coherence, diversity and personal styles have been used to result in more human-like dialogues (Li et al., 2016a,b). They can be useful for task-oriented dialogues as well. In Sec. 4.4.6, we will review a few recent works that aim to learn reward functions automatically from data.

4.2.2 Simulation-Based Evaluation

Typically, an RL algorithm needs to interact with a user to learn (Sec. 2.3). But running RL on either recruited users or actual users can be expensive. A natural way to get around this challenge is to build a *simulated* user, with which an RL algorithm can interact at virtually no cost. Essentially, a simulated user tries to mimic what a real user does in a conversation: it keeps track of the dialogue state, and converses with an RL dialogue system.

Substantial research has gone into building realistic user simulators (Schatzmann et al., 2005a; Georgila et al., 2006; Pietquin and Dutoit, 2006; Pietquin and Hastie, 2013). There are many differ-

```
{
  request_slots:
  {
    ticket: UNK
    theater: UNK
    start_time: UNK
  },
  inform_slots:
  {
    number_of_people: 3
    date: tomorrow
    movie_name: batman vs. superman
  }
}
```

Figure 4.2: An example user goal in the movie-ticket-booking domain

ent dimensions to categorize a user simulator, such as deterministic vs. stochastic, content-based vs. collaboration-based, static vs. non-static user goals during the conversations, among others. Here, we highlight two dimensions, and refer interested users to Schatzmann et al. (2006) for further details on creating and evaluating user simulators :

- Along the *granularity* dimension, the user simulator can operate either at the dialogue-act level (also known as intention level), or at the utterance level (Jung et al., 2009).
- Along the *methodology* dimension, the user simulator can be implemented using a rule-based approach, or a model-based approach with the model learned from a real conversational corpus.

Agenda-Based Simulation. As an example, we describe a popular hidden agenda-based user simulator developed by Schatzmann and Young (2009), as instantiated in Li et al. (2016d) and Ultes et al. (2017c). Each dialogue simulation starts with a randomly generated user goal that is unknown to the dialogue manager. In general the user goal consists of two parts: the `inform`-slots contain a number of slot-value pairs that serve as constraints the user wants to impose on the dialogue; the `request`-slots are slots whose values are initially unknown to the user and will be filled out during the conversation. Fig. 4.2 shows an example user goal in a movie domain, in which the user is trying to buy 3 tickets for tomorrow for the movie `batman vs. superman`.

Furthermore, to make the user goal more realistic, domain-specific constraints are added, so that certain slots are required to appear in the user goal. For instance, it makes sense to require a user to know the number of tickets she wants in the movie domain.

During the course of a dialogue, the simulated user maintains a stack data structure known as *user agenda*. Each entry in the agenda corresponds to a pending intention the user aims to achieve, and their priorities are implicitly determined by the first-in-last-out operations of the agenda stack. In other words, the agenda provides a convenient way of encoding the history of conversation and the "state-of-mind" of the user. Simulation of a user boils down to how to maintain the agenda after each turn of the dialogue, when more information is revealed. Machine learning or expert-defined rules can be used to set parameters in the stack-update process.

Model-based Simulation. Another approach to building user simulators is entirely based on data (Eckert et al., 1997; Levin et al., 2000; Chandramohan et al., 2011). Here, we describe a recent example due to El Asri et al. (2016). Similar to the agenda-based approach, the simulator also starts an episode with a randomly generated user goal and constraints. These are fixed during a conversation.

In each turn, the user model takes as input a sequence of contexts collected so far in the conversation, and outputs the next action. Specifically, the context at a turn of conversation consists of:

- the most recent machine action,
- inconsistency between machine information and user goal,

- constraint status, and
- request status.

With these contexts, an LSTM or other sequence-to-sequence models are used to output the next user utterance. The model can be learned from human-human dialogue corpora. In practice, it often works well by combining both rule-based and model-based techniques to create user simulators.

Further Remarks on User Simulation. While there has been much work on user simulation, building a human-like simulator remains challenging. In fact, even user simulator evaluation itself continues to be an ongoing research topic (Williams, 2008; Ai and Litman, 2008; Pietquin and Hastie, 2013). In practice, it is often observed that dialogue policies that are overfitted to a particular user simulator may not work well when serving another user simulator or real humans (Schatzmann et al., 2005b; Dhingra et al., 2017). The gap between a user simulator and humans is the major limitation of user simulation-based dialogue policy optimization.

Some user simulators are publicly available for research purposes. Other than the aforementioned agenda-based simulators by Li et al. (2016d); Ultes et al. (2017c), a large corpus with an evaluation environment, called AirDialogue (in the flight booking domain), was recently made available (Wei et al., 2018). At the IEEE workshop on Spoken Language Technology in 2018, Microsoft organized a dialogue challenge[1] of building end-to-end task-oriented dialogue systems by providing an experiment platform with built-in user simulators in several domains (Li et al., 2018).

4.2.3 Human-based Evaluation

Due to the discrepancy between simulated users and human users, it is often necessary to test a dialogue system on human users to reliably evaluate its quality. There are roughly two types of human users.

The first is human subjects recruited in a lab study, possibly through crowd-sourcing platforms. Typically, the participants are asked to test-use a dialogue system to solve a given task (depending on the domain of the dialogues), so that a collection of dialogues are obtained. Metrics of interest such as task-completion rate and average turns per dialogue can be measured, as done with a simulated user. In other cases, a fraction of these subjects are asked to test-use a baseline dialogue system, so that the two can be compared against various metrics.

Many published studies involving human subjects are of the first type (Walker, 2000; Singh et al., 2002; Ai et al., 2007; Rieser and Lemon, 2011; Gašić et al., 2013; Wen et al., 2015; Young et al., 2016; Peng et al., 2017; Lipton et al., 2018). While this approach has benefits over simulation-based evaluation, it is rather expensive and time-consuming to get a large number of subjects that can participate for a long time. Consequently, it has the following limitations:

- The small number of subjects prevents detection of statistically significant yet numerically small differences in metrics, often leading to inconclusive results.
- Only a very small number of dialogue systems may be compared.
- It is often impractical to run an RL agent that learns by interacting with these users, except in relatively simple dialogue applications.

The other type of humans for dialogue system evaluation is *actual* users (e.g., Black et al. (2011)). They are similar to the first type of users, except that they come with their actual tasks to be solved by conversing with the system. Consequently, metrics evaluated on them are even more reliable than those computed on recruited human subjects with artificially generated tasks. Furthermore, the number of actual users can be much larger, thus resulting in greater flexibility in evaluation. In this process, many online and offline evaluation techniques such as A/B-testing and counterfactual estimation can be used (Hofmann et al., 2016). The major downside of experimenting with actual users is the risk of negative user experience and disruption of normal services.

---

[1] `https://github.com/xiul-msr/e2e_dialog_challenge`

4.2.4 Other Evaluation Techniques

Recently, researchers have started to investigate a different approach to evaluation that is inspired by the self-play technique in RL (Tesauro, 1995; Mnih et al., 2015). This technique is typically used in a two-player game (such as the game of Go), where both players are controlled by the same RL agent, possibly initialized differently. By playing the agent against itself, a large amount of trajectories can be generated at relatively low cost, from which the RL agent can learn a good policy.

Self-play must be adapted to be used for dialogue management, as the two parties involved in a conversation often play asymmetric roles (unlike in games such as Go). Shah et al. (2018) described such a dialogue self-play procedure, which can generate conversations between a simulated user and the system agent. Promising results have been observed in negotiation dialogues (Lewis et al., 2017) and task-oriented dialogues (Liu and Lane, 2017; Shah et al., 2018; Wei et al., 2018). It provides an interesting solution to avoid the evaluation cost of involving human users as well as overfitting to untruthful simulated users.

In practice, it is reasonable to have a hybrid approach to evaluation. One possibility is to start with simulated users, then validate or fine-tune the dialogue policy on human users (cf., Shah et al. (2018)). Furthermore, there are more systematic approaches to using both sources of users for policy learning (see Sec. 4.4.5).

4.3 Natural Language Understanding and Dialogue State Tracking

NLU and DST are two closely related components essential to a dialogue system. They can have a significant impact on the overall system's performance (see, e.g., Li et al. (2017e)). This section reviews some of the classic and state-of-the-art approaches.

4.3.1 Natural Language Understanding

The NLU module takes user utterance as input, and performs three tasks: domain detection, intent determination, and slot tagging. An example output for the three tasks is given in Fig. 4.3. Typically, a pipeline approach is taken, so that the three tasks are solved one after another. Accuracy and F1 score are two of the most common metrics used to evaluate a model's prediction quality. NLU is a pre-processing step for later modules in the dialogue system, whose quality has a significant impact on the system's overall quality (Li et al., 2017d).

Among them, the first two tasks are often framed as a classification problem, which infers the domain or intent (from a predefined set of candidates) based on the current user utterance (Schapire and Singer, 2000; Yaman et al., 2008; Sarikaya et al., 2014). Neural approaches to multi-class classification have been used in the recent literature and outperformed traditional statistical methods. Ravuri and Stolcke (2015; 2016) studied the use of standard recurrent neural networks, and found them to be more effective. For short sentences where information has to be inferred from the context, Lee and Dernoncourt (2016) proposed to use recurrent and convolutional neural networks that also consider texts prior to the current utterance. Better results were shown on several benchmarks.

The more challenging task of slot tagging is often treated as sequence classification, where the classifier predicts semantic class labels for subsequences of the input utterance (Wang et al., 2005; Mesnil et al., 2013). Fig. 4.3 shows an ATIS (Airline Travel Information System) utterance example in the Inside-Outside-Beginning (IOB) format (Ramshaw and Marcus, 1995), where for each word the model predicts a semantic tag.

Yao et al. (2013) and Mesnil et al. (2015) applied recurrent neural networks to slot tagging, where inputs are one-hot encoding of the words in the utterance, and obtained higher accuracy than statistical baselines such as conditional random fields and support vector machines. Moreover, it is also shown that a-prior word information can be effectively incorporated into basic recurrent models to yield further accuracy gains.

As an example, this section describes the use of bidirectional LSTM (Graves and Schmidhuber, 2005), or bLSTM in short, in NLU tasks, following Hakkani-Tür et al. (2016) who also discussed other models for the same tasks. The model, as shown in Fig. 4.4, uses two sets of LSTM cells applied to the input sequence (the forward) and the *reversed* input sequence (the backward). The concatenated hidden layers of the forward and backward LSTMs are used as input to another neural

W	find	recent	comedies	by	james	cameron
	↓	↓	↓	↓	↓	↓
S	O	B-date	B-genre	O	B-dir	I-dir
D	movies					
I	find_movie					

Figure 4.3: An example output of NLU, where the utterance (W) is used to predict domain (I), intent (I), and the slot tagging (S). The IOB representation is used. Figure credit: Hakkani-Tür et al. (2016).

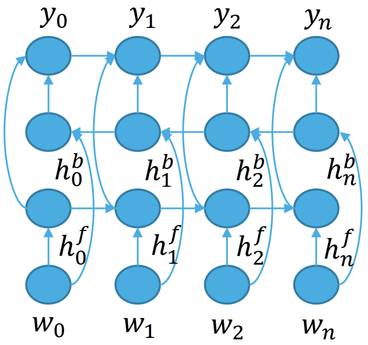


Figure 4.4: A bLSTM model for joint optimization in NLU. Picture credit: Hakkani-Tür et al. (2016).

network to compute the output sequence. Mathematically, upon the tth input token, $\mathbf{w}_t$, operations of the forward part of bLSTM are defined by the following set of equations:

$$\begin{aligned}
\mathbf{i}_t &= g(\mathbf{W}_{wi}\mathbf{w}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1}) \\
\mathbf{f}_t &= g(\mathbf{W}_{wf}\mathbf{w}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1}) \\
\mathbf{o}_t &= g(\mathbf{W}_{wo}\mathbf{w}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1}) \\
\hat{\mathbf{c}}_t &= \tanh(\mathbf{W}_{wc}\mathbf{w}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1}) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)\,,
\end{aligned}$$

where $\mathbf{h}_{t-1}$ is the hidden layer, $\mathbf{W}_\star$ the trainable parameters, and $g(\cdot)$ the sigmoid function. As in standard LSTMs, $\mathbf{i}_t$, $\mathbf{f}_t$ and $\mathbf{o}_t$ are the input, forget, and output gates, respectively. The backward part is similar, with the input reversed.

To predict the slot tags as shown in Fig. 4.3, the input $\mathbf{w}_t$ is often a one-hot vector of a word embedding vector. The output upon input $\mathbf{w}_t$ is predicted according to the following distribution p_t:

$$p_t = \text{softmax}(\mathbf{W}_{hy}^{(f)}\mathbf{h}_t^{(f)} + \mathbf{W}_{hy}^{(b)}\mathbf{h}_t^{(b)})\,,$$

where the superscripts, (f) and (b), denote forward and backward parts of the bLSTM, respectively. For tasks like domain and intent classification, the output is predicted at the end of the input sequence, and simpler architectures may be used (Ravuri and Stolcke, 2015, 2016).

In many situations, the present utterance alone can be ambiguous or lack all necessary information. Contexts that include information from previous utterances are expected to help improve model accuracy. Hori et al. (2015) treated conversation history as a long sequence of words, with alternating roles (words from user, vs. words from system), and proposed a variant to LSTM with role-dependent layers. Chen et al. (2016b) built on memory networks that learn which part of contextual information should be attended to, when making slot-tagging predictions. Both models achieved higher accuracy than context-free models.

Although the three NLU tasks are often studied separately, there are benefits to jointly solving them (similar to multi-task learning), and over multiple domains, so that it may require fewer labeled

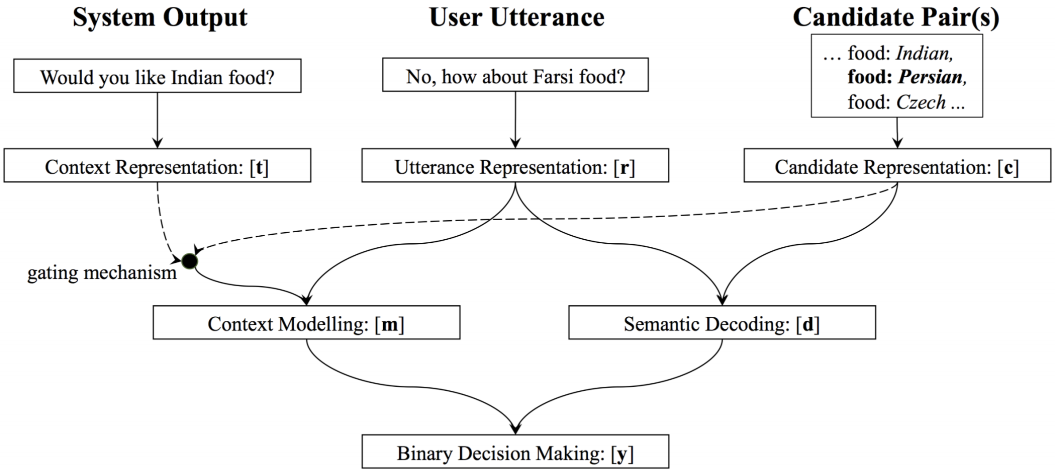

System Output
User Utterance
Candidate Pair(s)
Would you like Indian food?
No, how about Farsi food?
… food: Indian,
food: Persian,
food: Czech …
Context Representation: [t]
Utterance Representation: [r]
Candidate Representation: [c]
gating mechanism
Context Modelling: [m]
Semantic Decoding: [d]
Binary Decision Making: [y]

Figure 4.5: Neural Belief Tracker. Figure credit: Mrkšić et al. (2017).

data when creating NLU models for a new domain (Hakkani-Tür et al., 2016; Liu and Lane, 2016). Another line of interesting work that can lead to substantial reduction of labeling cost in new domains is *zero-shot learning*, where slots from different domains are represented in a shared latent semantic space through embedding of the slots' (text) descriptions (Bapna et al., 2017; Lee and Jha, 2019). Interested readers are referred to recent tutorials, such as Chen and Gao (2017) and Chen et al. (2017e), for more details.

4.3.2 Dialogue State Tracking

In slot-filling problems, a dialogue state contains all information about what the user is looking for at the current turn of the conversation. This state is what the dialogue policy takes as input for deciding what action to take next (Fig. 4.1).

For example, in the restaurant domain, where a user tries to make a reservation, the dialogue state may consists of the following components (Henderson, 2015):

- The goal constraint for every informable slot, in the form of a value assignment to that slot. The value can be "`don't care`" (if the user has no preference) or "`none`" (if the user has not yet specified the value).
- The subset of requested slots that the user has asked the system to inform.
- The current dialogue search method, taking values `by constraint`, `by alternative` and `finished`. It encodes how the user is trying to interact with the dialogue system.

Many alternatives have also been used in the literature, such as a compact, binary representation recently proposed by Kotti et al. (2018), and the StateNet tracker of Ren et al. (2018b) that is more scalable with the domain size (number of slots and number of slot values).

In the past, DST can either be created by experts, or obtained from data by statistical learning algorithms like conditional random fields (Henderson, 2015). More recently, neural approaches have started to gain popularity, with applications of deep neural networks (Henderson et al., 2013) and recurrent networks (Mrkšić et al., 2015) as some of the early examples.

A more recent DST model is the Neural Belief Tracker proposed by Mrkšić et al. (2017), shown in Fig. 4.5. The model takes three items as input. The first two are the last system and user utterances, each of which is first mapped to an internal, vector representation. The authors studied two models for representation learning, based on multi-layer perceptrons and convolutional neural networks, both of which take advantage of pre-trained collections of word vectors and output an embedding for the input utterance. The third input is any slot-value pair that is being tracked by DST. Then, the three embeddings may interact among themselves for context modeling, to provide further contextual information from the flow of conversation, and semantic decoding, to decide if the user explicitly expressed an intent matching the input slot-value pair. Finally, the context modeling and semantic decoding vectors go through a softmax layer to produce a final prediction. The same process is repeated for all possible candidate slot-value pairs.

A different representation of dialogue states, called *belief spans*, is explored by Lei et al. (2018) in the *Sequicity* framework. A belief span consists of two fields: one for informable slots and the other for requestable slots. Each field collects values that have been found for respective slots in the conversation so far. One of the main benefits of belief spans and Sequicity is that it facilitates the use of neural sequence-to-sequence models to learn dialogue systems, which take the belief spans as input and output system responses. This greatly simplifies system design and optimization, compared to more traditional, pipeline approaches (c.f., Sec. 4.6).

Dialogue State Tracking Challenge (DSTC) is a series of challenges that provide common testbeds and evaluation measures for dialogue state tracking. Starting from Williams et al. (2013), it has successfully attracted many research teams to focus on a wide range of technical problems in DST (Williams et al., 2014; Henderson et al., 2014b,a; Kim et al., 2016a,b; Hori et al., 2017). Corpora used by DSTC over the years have covered human-computer and human-human conversations, different domains such as restaurant and tourist, cross-language learning. More information may be found in the DSTC website.[2]

4.4 Dialogue Policy Learning

In this section, we will focus on dialogue policy optimization based on reinforcement learning.

4.4.1 Deep RL for Policy Optimization

The dialogue policy may be optimized by many standard reinforcement learning algorithms. There are two ways to use RL: online and batch. The online approach requires the learner to interact with users to improve its policy; the batch approach assumes a fixed set of transitions, and optimizes the policy based on the data only, without interacting with users (see, e.g., Li et al. (2009); Pietquin et al. (2011)). In this chapter, we discuss the online setting which often has batch learning as an internal step. Many covered topics can be useful in the batch setting. Here, we use the DQN as an example, following Lipton et al. (2018), to illustrate the basic work flow.

Model: Architecture, Training and Inference. The DQN's input is an encoding of the current dialogue state. One option is to encode it as a feature vector, consisting of: (1) one-hot representations of the dialogue act and slot corresponding to the last user action; (2) the same one-hot representations of the dialogue act and slot corresponding to the last system action; (3) a bag of slots corresponding to all previously filled slots in the conversation so far; (4) the current turn count; and (5) the number of results from the knowledge base that match the already filled-in constraints for informed slots. Denote this input vector by $\mathbf{s}$.

DQN outputs a real-valued vector, whose entries correspond to all possible (dialogue-act, slot) pairs that can be chosen by the dialogue system. Available prior knowledge can be used to reduce the number of outputs, if some (dialogue-act, slot) pairs do not make sense for a system, such as `request(price)`. Denote this output vector by $\mathbf{q}$.

The model may have $L \geq 1$ hidden layers, parameterized by matrices $\{\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_L\}$, so that

$$\begin{aligned}
\mathbf{h}_0 &= \mathbf{s} \\
\mathbf{h}_l &= g(\mathbf{W}_l \mathbf{h}_{l-1}) \,, \qquad l = 1, 2, \ldots, L-1 \\
\mathbf{q} &= \mathbf{W}_L \mathbf{h}_{L-1} \,,
\end{aligned}$$

where $g(\cdot)$ is an activation function such as ReLU or sigmoid. Note that the last layer does not need an activation function, and the output $\mathbf{q}$ is to approximate $Q(\mathbf{s}, \cdot)$, the Q-values in state $\mathbf{s}$.

To learn parameters in the network, one can use an off-the-shelf reinforcement-learning algorithm (e.g., Eqn. 2.4 or 2.5 with experience replay); see Sec. 2.3 for the exact update rules and improved algorithms. Once these parameters are learned, the network induces a *greedy* action-selection policy

---

[2] `https://www.microsoft.com/en-us/research/event/dialog-state-tracking-challenge`

as follows: for a current dialogue state $\mathbf{s}$, use a forward pass on the network to compute $\mathbf{q}$, the Q-values for all actions. One can pick the action, which is a (dialogue act, slot) pair, that corresponds to the entry in $\mathbf{q}$ with the largest value. Due to the need for exploration, the above greedy action selection may not be desired; see Sec. 4.4.2 for a discussion on this subject.

Warm-start Policy. Learning a good policy from scratch often requires many data, but the process can be significantly sped up by restricting the policy search using expert-generated dialogues (Henderson et al., 2008) or teacher advice (Chen et al., 2017d), or by initializing the policy to be a reasonable one before switching to online interaction with (simulated) users.

One approach is to use imitation learning (also known as behavioral cloning) to mimic an expert-provided policy. A popular option is to use supervised learning to directly learn the expert's action in a state; see Su et al. (2016b); Dhingra et al. (2017); Williams et al. (2017); Liu and Lane (2017) for a few recent examples. Li et al. (2014) turned imitation learning into an induced reinforcement learning problem, and then applied an off-the-shelf RL algorithm to learn the expert's policy.

Finally, Lipton et al. (2018) proposed a simple yet effective alternative known as Replay Buffer Spiking (RBS) that is particularly suited to DQN. The idea is to pre-fill the experience replay buffer of DQN with a small number of dialogues generated by running a naïve yet occasionally successful, rule-based agent. This technique is shown to be essential for DQN in simulated studies.

Other Approaches. In the above example, a standard multi-layer perceptron is used in the DQN to approximate the Q-function. It may be replaced by other models, such as a Bayesian version described in the next subsection for efficient exploration, and recurrent networks (Zhao and Eskénazi, 2016; Williams et al., 2017) that can more easily capture information from conversational histories than expert-designed dialogue states. In another recent example, Chen et al. (2018) used graph neural networks to model the Q-function, with nodes in the graph corresponding to slots of the domain. The nodes may share some of the parameters across multiple slots, therefore increasing learning speed.

Furthermore, one may replace the above value function-based methods by others like policy gradient (Sec. 2.3.2), as done by Fatemi et al. (2016); Dhingra et al. (2017); Strub et al. (2017); Williams et al. (2017); Liu et al. (2018a).

4.4.2 Efficient Exploration and Domain Extension

Without a teacher, an RL agent learns from data collected by interacting with an initially unknown environment. In general, the agent has to try new actions in novel states, in order to discover potentially better policies. Hence, it has to strike a good trade-off between exploitation (choosing good actions to maximize reward, based on information collected thus far) and exploration (choosing novel actions to discover potentially better alternatives), leading to the need for efficient exploration (Sutton and Barto, 2018). In the context of dialogue policy learning, the implication is that the policy learner actively tries new ways to converse with a user, in the hope of discovering a better policy in the long run (e.g., Daubigney et al., 2011).

While exploration in finite-state RL is relatively well-understood (Strehl et al., 2009; Jaksch et al., 2010; Osband and Roy, 2017; Dann et al., 2017), exploration when parametric models like neural networks are used is an active research topic (Bellemare et al., 2016; Osband et al., 2016; Houthooft et al., 2016; Jiang et al., 2017). Here, a general-purpose exploration strategy is described, which is particularly suited for dialogue systems that may evolve over time.

After a task-oriented dialogue system is deployed to serve users, there may be a need over time to add more intents and/or slots to make the system more versatile. This problem, referred to as *domain extension* (Gašic et al., 2014), makes exploration even more challenging: the agent needs to explicitly quantify the uncertainty in its parameters for intents/slots, so as to explore new ones more aggressively while avoiding exploring those that have already been learned. Lipton et al. (2018) approached the problem using a Bayesian-by-Backprop variant of DQN.

Their model, called BBQ, is identical to DQN, except that it maintains a posterior *distribution* q over the network weights $\mathbf{w} = (w_1, w_2, \ldots, w_d)$. For computational convenience, q is a multivariate Gaussian distribution with diagonal covariance, parameterized by $\theta = \{(\mu_i, \rho_i)\}_{i=1}^d$, where weight w_i has a Gaussian posterior distribution, $\mathcal{N}(\mu_i, \sigma_i^2)$ and $\sigma_i = \log(1 + \exp(\rho_i))$. The posterior information leads to a natural exploration strategy, inspired by Thompson Sampling (Thompson, 1933; Chapelle and Li, 2012; Russo et al., 2018). When selecting actions, the agent simply draws a random weight $\tilde{\mathbf{w}} \sim q$, and then selects the action with the highest value output by the network. Experiments show that BBQ explores more efficiently than state-of-the-art baselines for dialogue domain extension.

The BBQ model is updated as follows. Given observed transitions $\mathcal{T} = \{(s, a, r, s')\}$, one uses the target network (see Sec. 2.3) to compute the target values for each (s, a) in $\mathcal{T}$, resulting in the set $\mathcal{D} = \{(x, y)\}$, where $x = (s, a)$ and y may be computed as in DQN. Then, parameter θ is updated to represent the posterior distribution of weights. Since the exact posterior is not Gaussian any more, and thus not representable by BBQ, it is approximated as follows: θ is chosen by minimizing the *variational free energy* (Hinton and Van Camp, 1993), the KL-divergence between the variational approximation $q(\mathbf{w}|\theta)$ and the posterior $p(\mathbf{w}|\mathcal{D})$:

$$\begin{aligned}\theta^* &= \operatorname{argmin}_\theta \mathrm{KL}[q(\mathbf{w}|\theta)||p(\mathbf{w}|\mathcal{D})] \\ &= \operatorname{argmin}_\theta \left\{ \mathrm{KL}[q(\mathbf{w}|\theta)||p(\mathbf{w})] - \mathbf{E}_{q(\mathbf{w}|\theta)}[\log p(\mathcal{D}|\mathbf{w})] \right\}.\end{aligned}$$

In other words, the new parameter θ is chosen so that the new Gaussian distribution is closest to the posterior measured by KL-divergence.

4.4.3 Composite-task Dialogues

In many real-world problems, a task may consist of a set of subtasks that need to be solved collectively. Similarly, dialogues can often be decomposed into a sequence of related sub-dialogues, each of which focuses on a subtopic (Litman and Allen, 1987). Consider for example a travel planning dialogue system, which needs to book flights, hotels and car rental in a collective way so as to satisfy certain cross-subtask constraints known as *slot constraints* (Peng et al., 2017). Slot constraints are application specific. In a travel planning problem, one natural constraint is that the outbound flight's arrival time should be earlier than the hotel check-in time.

Complex tasks with slot constraints are referred to as *composite tasks* by Peng et al. (2017). Optimizing the dialogue policy for a composite task is challenging for two reasons. First, the policy has to handle many slots, as each subtask often corresponds to a domain with its own set of slots, and the set of slots of a composite-task consists of slots from all subtasks. Furthermore, thanks to slot constraints, these subtasks cannot be solved independently. Therefore, the state space considered by a composite-task is much larger. Second, a composite-task dialogue often requires many more turns to complete. Typical reward functions give a success-or-not reward only at the end of the whole dialogue. As a result, this reward signal is very sparse and considerably delayed, making policy optimization much harder.

Cuayáhuitl et al. (2010) proposed to use hierarchical reinforcement learning to optimize a composite task's dialogue policy, with tabular versions of the MAXQ (Dietterich, 2000) and Hierarchical Abstract Machine (Parr and Russell, 1998) approaches. While promising, their solutions assume finite states, so do not apply directly to larger-scale conversational problems.

More recently, Peng et al. (2017) tackled the composite-task dialogue policy learning problem under the more general *options* framework (Sutton et al., 1999b), where the task hierarchy has two levels. As illustrated in Fig. 4.6, a top-level policy π_g selects which subtask g to solve, and a low-level policy $\pi_{a,g}$ solves the subtask specified by π_g. Assuming predefined subtasks, they extend the DQN model that results in substantially faster learning speed and superior policies. A similar approach is taken by Budzianowski et al. (2017), who used Gaussian process RL instead of deep RL for policy learning.

A major assumption in options/subgoal-based hierarchical reinforcement learning is the need for reasonable options and subgoals. Tang et al. (2018) considered the problem of discovering subgoals from dialogue demonstrations. Inspired by a sequence segmentation approach that is successfully applied to machine translation (Wang et al., 2017a), the authors developed the Subgoal Discovery Network (SDN), which learns to identify "bottleneck" states in successful dialogues. It is shown that the hierarchical DQN optimized with subgoals discovered by SDN is competitive to expert-designed subgoals.

Finally, another interesting attempt is made by Casanueva et al. (2018) based on Feudal Reinforcement Learning (FRL) (Dayan and Hinton, 1993). In contrast to the above methods that decompose a

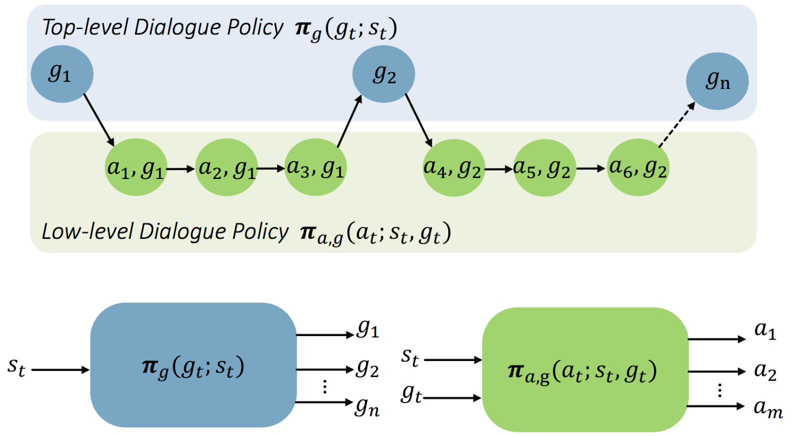


Figure 4.6: A two-level hierarchical dialogue policy. Figure credit: Peng et al. (2017).

Table 4.2: An example of multi-domain dialogue, adapted from Cuayáhuitl et al. (2016). The first column specifies which domain is triggered in the system, based on user utterances received so far.

Domain	Agent	Utterance
`meta`	system	"Hi! How can I help you?"
	user	"I'm looking for a hotel in Seattle on January 2nd for 2 nights."
`hotel`	system	"A hotel for 2 nights in Seattle on January 2nd?"
	user	"Yes."
	system	"I found Hilton Seattle."
`meta`	system	"Anything else I can help with?"
	user	"I'm looking for cheap Japanese food in the downtown."
`restaurant`	system	"Did you say cheap Japanese food?"
	user	"Yes."
	system	"I found the following results."
	...	

task into *temporally* separated subtasks, FRL decomposes a complex decision *spatially*. In each turn of a dialogue, the feudal policy first decides between information-gathering actions and information-providing actions, then chooses a primitive action that falls in the corresponding high-level category.

4.4.4 Multi-domain Dialogues

A multi-domain dialogue can converse with a user to have a conversation that may involve more than one domain (Komatani et al., 2006; Hakkani-Tür et al., 2012; Wang et al., 2014). Table 4.2 shows an example, where the dialogue covers both the `hotel` and `restaurant` domains, in addition to a special `meta` domain for sub-dialogues that contain domain-independent system and user responses.

Different from composite tasks, sub-dialogues corresponding to different domains in a conversation are separate tasks, without cross-task slot constraints. Similar to composite-task systems, a multi-domain dialogue system needs to keep track of a much larger dialogue state space that has slots from all domains, so directly applying RL can be inefficient. It thus raises the need to learn re-usable policies whose parameters can be shared across multiple domains as long as they are related.

Gašić et al. (2015) proposed to use a Bayesian Committee Machine (BCM) for efficient multi-domain policy learning. During training time, a number of policies are trained on different, potentially small, datasets. The authors used Gaussian processes RL algorithms to optimize those policies, although they can be replaced by deep learning alternatives. During test time, in each turn of a dialogue, these policies recommend an action, and all recommendations are aggregated into a final action to be taken by the BCM policy.

Cuayáhuitl et al. (2016) developed another related technique known as NDQN—Network of DQNs, where each DQN is trained for a specialized skill to converse in a particular sub-dialogue. A meta-

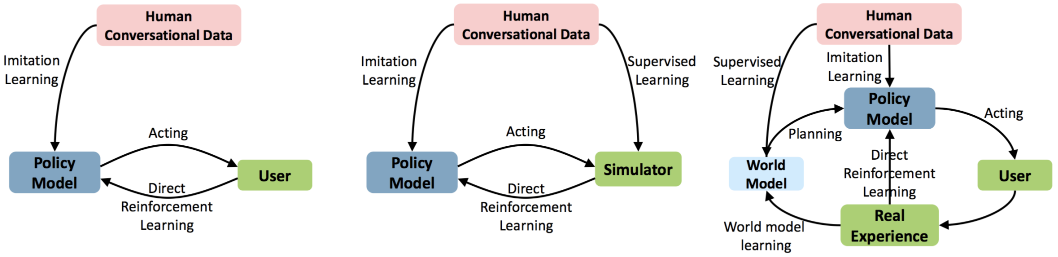


Figure 4.7: Three strategies for optimizing dialogue policies based on reinforcement learning. Figure credit: Peng et al. (2018).

policy controls how to switch between these DQNs, and can also be optimized using (deep) reinforcement learning.

More recently, Papangelis et al. (2018a) studied another approach in which policies optimized for difference domains can be shared, through a set of features that describe a domain. It is shown to be able to handle unseen domains, and thus reduce the need for domain knowledge to design the ontology.

4.4.5 Integration of Planning and Learning

As mentioned in Sec. 4.2, optimizing the policy of a task-oriented dialogue against humans is costly, since it requires many interactions between the dialogue system and humans (left panel of Fig. 4.7). Simulated users provide an inexpensive alternative to RL-based policy optimization (middle panel of Fig. 4.7), but may not be a sufficiently truthful approximation of human users.

Here, we are concerned with the use of a user model to generate more data to improve sample complexity in optimizing a dialogue system. Inspired by the Dyna-Q framework (Sutton, 1990), Peng et al. (2018) proposed Deep Dyna-Q (DDQ) to handle large-scale problems with deep learning models, as shown by the right panel of Fig. 4.7. Intuitively, DDQ allows interactions with both human users and simulated users. Training of DDQ consists of three parts:

- *direct reinforcement learning*: the dialogue system interacts with a real user, collects real dialogues and improves the policy by either imitation learning or reinforcement learning;
- *world model learning*: the world model (user simulator) is refined using real dialogues collected by direct reinforcement learning;
- *planning*: the dialogue policy is improved against simulated users by reinforcement learning.

Human-in-the-loop experiments show that DDQ is able to efficiently improve the dialogue policy by interacting with real users, which is important for deploying dialogue systems in practice.

One challenge with DDQ is to balance samples from real users (direct reinforcement learning) and simulated users (planning). Peng et al. (2018) used a heuristics that reduces planning steps in later stage of DDQ when more real user interactions are available. In contrast, Su et al. (2018b) proposed the Discriminative Deep Dyna-Q (D3Q) that is inspired by generative adversarial networks. Specifically, it incorporates a discriminator which is trained to differentiate experiences of simulated users from those of real users. During the planning step, a simulated experience is used for policy training only when it appears to be a real-user experience according to the discriminator.

4.4.6 Reward Function Learning

The dialogue policy is often optimized to maximize long-term reward when interacting with users. The reward function is therefore critical to creating high-quality dialogue systems. One possibility is to have users provide feedback during or at the end of a conversation to rate the quality, but feedback like this is intrusive and costly. Often, easier-to-measure quantities such as time-elapsed are used to compute a reward function. Unfortunately, in practice, designing an appropriate reward function is not always obvious, and substantial domain knowledge is needed (Sec. 4.1). This inspires the use of machine learning to find a good reward function from data (Walker et al., 2000; Rieser and Lemon, 2008; Rieser et al., 2010; El Asri et al., 2012) which can better correlate with user satisfaction (Rieser and Lemon, 2011), or is more consistent with expert demonstrations (Li et al., 2014).

Su et al. (2015) proposed to rate dialogue success with two neural network models, a recurrent and a convolutional network. Their approach is found to result in competitive dialogue policies, when compared to a baseline that uses prior knowledge of user goals. However, these models assume the availability of labeled data in the form of (dialogue, success-or-not) pairs, in which the success-or-not feedback provided by users can be expensive to obtain. To reduce the labeling cost, Su et al. (2016a; 2018a) investigated an active learning approach based on Gaussian processes, which aims to learn the reward function and policy at the same time while interacting with human users.

Ultes et al. (2017a) argued that dialogue success only measures one aspect of the dialogue policy's quality. Focusing on information-seeking tasks, the authors proposed a new reward estimator based on *interaction quality* that balances multiple aspects of the dialogue policy. Later on, Ultes et al. (2017b) used multi-objective RL to automatically learn how to linearly combine multiple metrics of interest in the definition of reward function.

Finally, inspired by adversarial training in deep learning, Liu and Lane (2018) proposed to view the reward function as a discriminator that distinguishes dialogues generated by humans from those by the dialogue policy. Therefore, there are two learning processes in their approach: the reward function as a discriminator, and the dialogue policy optimized to maximize the reward function. The authors showed that such an adversarially learned reward function can lead to better dialogue policies than with hand-designed reward functions.

4.5 Natural Language Generation

Natural Language Generation (NLG) is responsible for converting a communication goal, selected by the dialogue manager, into a natural language form. It is an important component that affects naturalness of a dialogue system, and thus the user experience.

There exist many approaches to language generation. The most common in practice is perhaps template- or rule-based ones, where domain experts design a set of templates or rules, and hand-craft heuristics to select a proper candidate to generate sentences. Even though machine learning can be used to train certain parts of these systems (Langkilde and Knight, 1998; Stent et al., 2004; Walker et al., 2007), the cost to write and maintain templates and rules leads to challenges in adapting to new domains or different user populations. Furthermore, the quality of these NLG systems is limited by the quality of hand-crafted templates and rules.

These challenges motivate the study of more data-driven approaches, known as *corpus-based* methods, that aim to optimize a generation module from corpora (Oh and Rudnicky, 2002; Angeli et al., 2010; Kondadadi et al., 2013; Mairesse and Young, 2014). Most such methods are based on supervised learning, while Rieser and Lemon (2010) takes a decision-theoretic view and uses reinforcement learning to make a trade-off between sentence length and information revealed.[3]

In recent years, there is a growing interest in neural approaches to language generation. An elegant model, known as Semantically Controlled LSTM (SC-LSTM) (Wen et al., 2015), is a variant of LSTM (Hochreiter and Schmidhuber, 1997), with an extra component that gives a semantic control on the language generation results. As shown in Fig. 4.8, a basic SC-LSTM cell has two parts: a typical LSTM cell (upper part in the figure) and a sentence planning cell (lower part) for semantic control.

---

[3] Some authors (Stone et al., 2003; Koller and Stone, 2007) have taken a similar, decision-theoretic point of view for NLG. Their formulate NLG as a planning problem, as opposed to data-driven or corpus-based methods being discussed here.

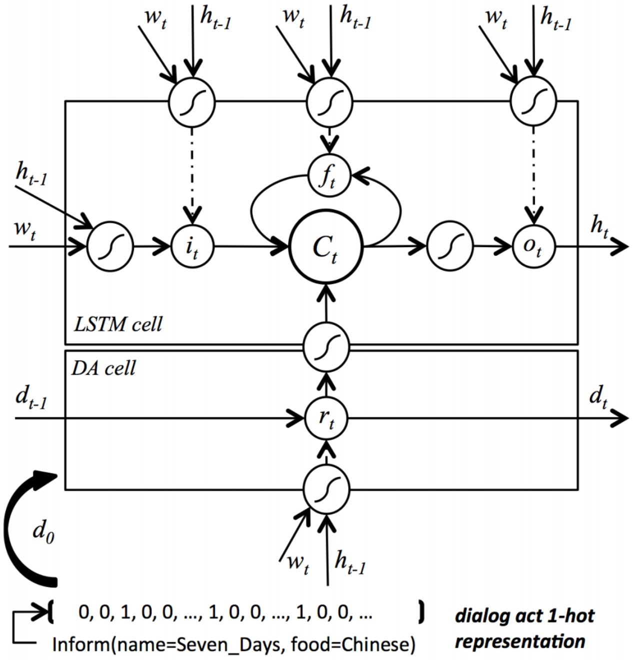


Figure 4.8: A Semantic Controlled LSTM (SC-LSTM) Cell. Picture credit: Wen et al. (2015).

More precisely, the operations upon receiving the tth input token, denoted $\mathbf{w}_t$, are defined by the following set of equations:

$$\begin{aligned}
\mathbf{i}_t &= g(\mathbf{W}_{wi}\mathbf{w}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1}) \\
\mathbf{f}_t &= g(\mathbf{W}_{wf}\mathbf{w}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1}) \\
\mathbf{o}_t &= g(\mathbf{W}_{wo}\mathbf{w}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1}) \\
\mathbf{r}_t &= g(\mathbf{W}_{wr}\mathbf{w}_t + \alpha\mathbf{W}_{hr}\mathbf{h}_{t-1}) \\
\mathbf{d}_t &= \mathbf{r}_t \odot \mathbf{d}_{t-1} \\
\hat{\mathbf{c}}_t &= \tanh(\mathbf{W}_{wc}\mathbf{w}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1}) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \hat{\mathbf{c}}_t + \tanh(\mathbf{W}_{dc}\mathbf{d}_t) \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)\,,
\end{aligned}$$

where $\mathbf{h}_{t-1}$ is the hidden layer, $\mathbf{W}_\star$ the trainable parameters, and $g(\cdot)$ the sigmoid function. As in a standard LSTM cell, $\mathbf{i}_t$, $\mathbf{f}_t$ and $\mathbf{o}_t$ are the input, forget, and output gates, respectively. The extra component introduced to SC-LSTM is the *reading gate* $\mathbf{r}_t$, which is used to compute a sequence of dialogue acts $\{\mathbf{d}_t\}$ starting from the original dialogue act $\mathbf{d}_0$. This sequence is to ensure that the generated utterance represents the intended meaning, and the reading gate is to control what information to be retained for future steps. It is in this sense that the gate $\mathbf{r}_t$ plays the role of *sentence planning* (Wen et al., 2015). Finally, given the hidden layer $\mathbf{h}_t$, the output distribution is given by a softmax function:

$$w_{t+1} \sim \mathrm{softmax}(\mathbf{W}_{ho}\mathbf{h}_t)\,.$$

Wen et al. (2015) proposed several improvements to the basic SC-LSTM architecture. One was to make the model deeper by stacking multiple LSTM cells on top of the structure in Fig. 4.8. Another was utterance reranking: they trained another instance of SC-LSTM on the reversed input sequence, similar to bidirectional recurrent networks, and then combined both instances to finalize reranking.

The basic approach outlined above may be extended in several ways. For example, Wen et al. (2016) investigated the use of multi-domain learning to reduce the amount of data to train a neural language generator, and Su et al. (2018c) proposed a hierarchical approach that leverages linguistic patterns to further improve generation results. Language generation remains an active research area. The next chapter will cover more recent works for chitchat conversations, in which many techniques can also be useful in task-oriented dialogue systems.

4.6 End-to-end Learning

Traditionally, components in most dialogue systems are optimized separately. This modularized approach provides the flexibility that allows each module to be created in a relatively independent way. However, it often leads to a more complex system design, and improvements in individual modules do not necessarily translate into improvement of the whole dialogue system. Lemon (2011) argued for, and empirically demonstrated, the benefit of jointly optimizing dialogue management and natural language generation, within a reinforcement-learning framework. More recently, with the increasing popularity of neural models, there have been growing interests in jointly optimizing multiple components, or even end-to-end learning of a dialogue system.

One benefit of neural models is that they are often differentiable and can be optimized by gradient-based methods like back-propagation (Goodfellow et al., 2016). In addition to language understanding, state tracking and policy learning that have been covered in previous sections, speech recognition & synthesis (for spoken dialogue systems) may be learned by neural models and back-propagation to achieve state-of-the-art performance (Hinton et al., 2012; van den Oord et al., 2016; Wen et al., 2015). In the extreme, if all components in a task-oriented dialogue system (Fig. 4.1) are differentiable, the whole system becomes a larger differentiable system that can be optimized by back-propagation against metrics that quantify overall quality of the whole system. This is an advantage compared to traditional approaches that optimize individual components separately. There are two general classes of approaches to building an end-to-end dialogue system:

Supervised Learning. The first is based on supervised learning, where desired system responses are first collected and then used to train multiple components of a dialogue system in order to maximize prediction accuracy (Bordes et al., 2017; Wen et al., 2017; Yang et al., 2017b; Eric et al., 2017; Madotto et al., 2018; Wu et al., 2018).

Wen et al. (2017) introduced a modular neural dialogue system, where most modules are represented by a neural network. However, their approach relies on non-differentiable knowledge-base lookup operators, so training of the components is done separately in a supervised manner. This challenge is addressed by Dhingra et al. (2017) who proposed "soft" knowledge-base lookups; see Sec. 3.5 for more details.

Bordes et al. (2017) treated dialogue system learning as the problem of learning a mapping from dialogue histories to system responses. They show memory networks and supervised embedding models outperform standard baselines on a number of simulated dialogue tasks. A similar approach was taken by Madotto et al. (2018) in their Mem2Seq model. This model uses mechanisms from pointer networks (Vinyals et al., 2015a) so as to incorporate external information from knowledge bases.

Finally, Eric et al. (2017) proposed an end-to-end trainable Key-Value Retrieval Network, which is equipped with an attention-based key-value retrieval mechanism over entries of a KB, and can learn to extract relevant information from the KB.

Reinforcement Learning. While supervised learning can produce promising results, they require training data that may be expensive to obtain. Furthermore, this approach does not allow a dialogue system to explore different policies that can potentially be better than expert policies that produce responses for supervised training. This inspire another line of work that uses reinforcement learning to optimize end-to-end dialogue systems (Zhao and Eskénazi, 2016; Williams and Zweig, 2016; Dhingra et al., 2017; Li et al., 2017d; Braunschweiler and Papangelis, 2018; Strub et al., 2017; Liu and Lane, 2017; Liu et al., 2018a).

Zhao and Eskénazi (2016) proposed a model that takes user utterance as input and outputs a semantic system action. Their model is a recurrent variant of DQN based on LSTM, which learns to compress a user utterance sequence to infer an internal state of the dialogue. Compared to classic approaches, this method is able to jointly optimize the policy as well as language understanding and state tracking beyond standard supervised learning.

Another approach, taken by Williams et al. (2017), is to use LSTM to avoid the tedious step of state tracking engineering, and jointly optimize state tracker and the policy. Their model, called Hybrid Code Networks (HCN), also makes it easy for engineers to incorporate business rules and other prior knowledge via software and action templates. They show that HCN can be trained end-to-end, demonstrating much faster learning than several end-to-end techniques.

Strub et al. (2017) applied policy gradient to optimize a visually grounded task-oriented dialogue in the GuessWhat?! game in an end-to-end fashion. In the game, both the user and the dialogue system have access to an image. The user chooses an object in the image without revealing it, and the dialogue system is to locate this object by asking the user a sequence of yes-no questions.

Finally, it is possible to combine supervised and reinforcement learning in an end-to-end trainable system. Liu et al. (2018a) proposed such a hybrid approach. First, they used supervised learning on human-human dialogues to pre-train the policy. Second, they used an imitation learning algorithm, known as DAgger (Ross et al., 2011), to fine tune the policy with human teachers who can suggest correct dialogue actions. In the last step, reinforcement learning was used to continue policy learning with online user feedback.

4.7 Further Remarks

In this chapter, we have surveyed recent neural approaches to task-oriented dialogue systems, focusing on slot-filling problems. This is a new area with many exciting research opportunities. While it is out of the scope of the paper to give a full coverage of more general dialogue problems and all research directions, we briefly describe a small sample of them to conclude this chapter.

Beyond Slot-filling Dialogues. Task-oriented dialogues in practice can be much more diverse and complex than slot-filling ones. Information-seeking or navigation dialogues are another popular example that has been mentioned in different contexts (e.g., Dhingra et al. (2017), Papangelis et al. (2018b), and Sec. 3.5). Another direction is to enrich the dialogue context. Rather than text-only or speech-only ones, our daily dialogues are often *multimodal*, and involve both verbal and nonverbal inputs like vision (Bohus et al., 2014; DeVault et al., 2014; de Vries et al., 2017; Zhang et al., 2018a). Challenges such as how to combine information from multiple modalities to make decisions arise naturally.

So far, we have looked at dialogues that involve two parties—the user and the dialogue agent, and the latter is to assist the former. In general, the task can be more complex such as mixed-initiative dialogues (Horvitz, 1999) and negotiations (Barlier et al., 2015; Lewis et al., 2017). More generally, there may be multiple parties involved in a conversation, where turn taking becomes more challenging (Bohus and Horvitz, 2009, 2011). In such scenarios, it is helpful to take a game-theoretic view, more general than the MDP view as in single-agent decision making.

Weaker Learning Signals. In the literature, a dialogue system can be optimized by supervised, imitation, or reinforcement learning. Some require expert labels/demonstrations, while some require a reward signal from a (simulated) user. There are other *weaker* form of learning signals that facilitate dialogue management at scale. A promising direction is to consider preferential input: instead of having an absolute judgment (either in the form of label or reward) of the policy quality, one only requires a preferential input that indicates which one of two dialogues is better. Such comparable feedback is often easier and cheaper to obtain, and can be more reliable than absolute feedback.

Related Areas. Evaluation remains a major research challenge. Although user simulation can be useful (Sec. 4.2.2), a more appealing and robust solution is to use real human-human conversation corpora directly for evaluation. Unfortunately, this problem, known as off-policy evaluation in the RL literature, is challenging with numerous current research efforts (Precup et al., 2000; Jiang and Li, 2016; Thomas and Brunskill, 2016; Liu et al., 2018c). Such off-policy techniques can find important use in evaluating and optimizing dialogue systems.

Another related line of research is deep reinforcement learning applied to text games (Narasimhan et al., 2015; Côté et al., 2018), which are in many ways similar to a conversation, except that the scenarios are predefined by the game designer. Recent advances for solving text games, such as handling natural-language actions (Narasimhan et al., 2015; He et al., 2016; Côté et al., 2018) and interpretable policies (Chen et al., 2017c) may be useful for task-oriented dialogues as well.

Chapter 5

Fully Data-Driven Conversation Models and Social Bots

Researchers have recently begun to explore fully data-driven and end-to-end (E2E) approaches to conversational response generation, e.g., within the sequence-to-sequence (seq2seq) framework (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014). These models are trained entirely from data without resorting to any expert knowledge, which means they do not rely on the four traditional components of dialogue systems noted in Chapter 4. Such end-to-end models have been particularly successful with social bot (chitchat) scenarios, as social bots rarely require interaction with the user's environment, and the lack of external dependencies such as API calls simplifies end-to-end training. By contrast, task-completion scenarios typically require such APIs in the form of, e.g., knowledge base access. The other reason this framework has been successful with chitchat is that it easily scales to large free-form and open-domain datasets, which means the user can typically chat on any topic of her liking. While social bots are of significant importance in facilitating smooth interaction between humans and their devices, more recent work also focuses on scenarios going beyond chitchat, e.g., recommendation.

5.1 End-to-End Conversation Models

Most of the earliest end-to-end (E2E) conversation models are inspired by statistical machine translation (SMT) (Koehn et al., 2003; Och and Ney, 2004), including neural machine translation (Kalchbrenner and Blunsom, 2013; Cho et al., 2014a; Bahdanau et al., 2015). The casting of the conversational response generation task (i.e., predict a response T_i based on the previous dialogue turn T_{i-1}) as an SMT problem is a relatively natural one, as one can treat turn T_{i-1} as the "foreign sentence" and turn T_i as its "translation". This means one can apply any off-the-shelf SMT algorithm to a conversational dataset to build a response generation system. This was the idea originally proposed in one of the first works on fully data-driven conversational AI (Ritter et al., 2011), which applied a phrase-based translation approach (Koehn et al., 2003) to dialogue datasets extracted from Twitter (Serban et al., 2015). A different E2E approach was proposed in (Jafarpour et al., 2010), but it relied on IR-based methods rather than machine translation.

While these two papers constituted a paradigm shift, they had several limitations. The most significant one is their representation of the data as (query, response) pairs, which hinders their ability to generate responses that are contextually appropriate. This is a serious limitation as dialogue turns in chitchat are often short (e.g., a few word utterance such as "really?"), in which case conversational models critically need longer contexts to produce plausible responses. This limitation motivated the work of Sordoni et al. (2015b), which proposed an RNN-based approach to conversational response generation (similar to Fig. 2.2) to exploit longer context. Together with the contemporaneous works (Shang et al., 2015; Vinyals and Le, 2015), these papers presented the first neural approaches to fully E2E conversation modeling. While these three papers have some distinct properties, they are all based on RNN architectures, which nowadays are often modeled with a Long Short-Term Memory (LSTM) model (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014).

5.1.1 The LSTM Model

We give an overview of LSTM-based response generation. LSTM is arguably the most popular seq2seq model, although alternative models like GRU (Cho et al., 2014b) are often as effective. LSTM is an extension of the RNN model in Fig. 2.2, and is often more effective at exploiting long-term context.

An LSTM-based response generation system is usually modeled as follows (Vinyals and Le, 2015; Li et al., 2016a): Given a dialogue history represented as a sequence of words $S = \{s_1, s_2, ..., s_{N_s}\}$ (S here stands for source), the LSTM associates each time step k with input, memory, and output gates, denoted respectively as $\mathbf{i}_k$, $\mathbf{f}_k$ and $\mathbf{o}_k$. N_s is the number of words in the source S.[1] Then, the hidden state $\mathbf{h}_k$ of the LSTM for each time step k is computed as follows:

$$\mathbf{i}_k = \sigma(\mathbf{W}_i[\mathbf{h}_{k-1}; \mathbf{e}_k]) \tag{5.1}$$

$$\mathbf{f}_k = \sigma(\mathbf{W}_f[\mathbf{h}_{k-1}; \mathbf{e}_k]) \tag{5.2}$$

$$\mathbf{o}_k = \sigma(\mathbf{W}_o[\mathbf{h}_{k-1}; \mathbf{e}_k]) \tag{5.3}$$

$$\mathbf{l}_k = \tanh(\mathbf{W}_l[\mathbf{h}_{k-1}; \mathbf{e}_k]) \tag{5.4}$$

$$\mathbf{c}_k = \mathbf{f}_k \circ \mathbf{c}_{k-1} + \mathbf{i}_k \circ \mathbf{l}_k \tag{5.5}$$

$$\mathbf{h}_k^s = \mathbf{o}_k \circ \tanh(\mathbf{c}_k) \tag{5.6}$$

where matrices $\mathbf{W}_i$, $\mathbf{W}_f$, $\mathbf{W}_o$, $\mathbf{W}_l$ belong to $\mathbb{R}^{d \times 2d}$, $\circ$ denotes the element-wise product. As it is a response generation task, each conversational context S is paired with a sequence of output words to predict: $T = \{t_1, t_2, ..., t_{N_t}\}$. Here, N_t is the length of the response and t represents a word token that is associated with a d-dimensional word embedding e_t (distinct from the source).

The LSTM model defines the probability of the next token to predict using the softmax function. Specifically, let $f(\mathbf{h}_{k-1}, \mathbf{e}_{y_k})$ be the softmax activation function of $\mathbf{h}_{k-1}$ and $\mathbf{e}_{y_k}$, where $\mathbf{h}_{k-1}$ is the hidden vector at time $k-1$. Then, the probability of outputing token T is given by

$$\begin{aligned} p(T|S) &= \prod_{k=1}^{N_t} p(t_k|s_1, s_2, ..., s_t, t_1, t_2, ..., t_{k-1}) \\ &= \prod_{k=1}^{N_t} \frac{\exp(f(h_{k-1}, e_{y_k}))}{\sum_{y'} \exp(f(h_{k-1}, e_{y'}))} . \end{aligned}$$

5.1.2 The HRED Model

While the LSTM model has been shown to be effective in encoding textual contexts up to 500 words (Khandelwal et al., 2018), dialogue histories can often be long and there is sometimes a need to exploit longer-term context. Hierarchical models were designed to address this limitation by capturing longer context (Yao et al., 2015; Serban et al., 2016, 2017; Xing et al., 2018). One popular approach is the Hierarchical Recurrent Encoder-Decoder (HRED) model, originally proposed in (Sordoni et al., 2015a) for query suggestion and applied to response generation in (Serban et al., 2016).

The HRED architecture is depicted in Fig. 5.1, where it is compared to the standard RNN architecture. HRED models dialogue using a two-level hierarchy that combines two RNNs: one at a word level and one at the dialogue turn level. This architecture models the fact that dialogue history consists of a sequence of turns, each consisting of a sequence of tokens. This model introduces a temporal structure that makes the hidden state of the current dialogue turn directly dependent on the hidden state of the previous dialogue turn, effectively allowing information to flow over longer time spans, and helping reduce the vanishing gradient problem (Hochreiter, 1991), a problem that limits RNN's (including LSTM's) ability to model very long word sequences. Note that, in this particular work, RNN hidden states are implemented using GRU (Cho et al., 2014b) instead of LSTM.

---

[1] The notation distinguishes $\mathbf{e}$ and $\mathbf{h}$ where $\mathbf{e}_k$ is the embedding vector for an individual word at time step k, and $\mathbf{h}_k$ is the vector computed by the LSTM model at time k by combining $\mathbf{e}_k$ and $\mathbf{h}_{k-1}$. $\mathbf{c}_k$ is the cell state vector at time k, and σ represents the sigmoid function.

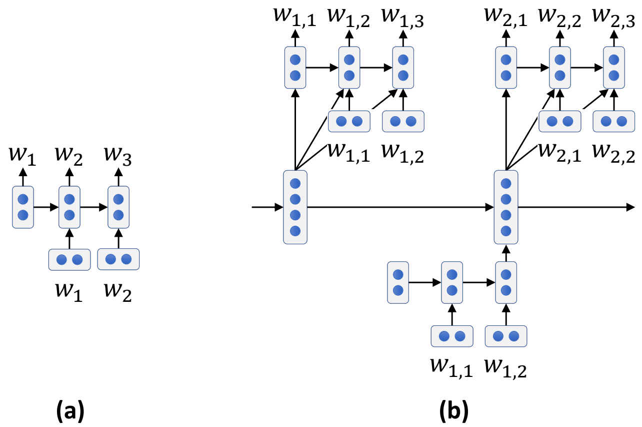


Figure 5.1: (a) Recurrent architecture used by models such as RNN, GRU, LSTM, etc. (2) Two-level hierarchy representative of HRED. Note: To simplify the notation, the figure represents utterances of length 3.

5.1.3 Attention Models

The seq2seq framework has been tremendously successful in text generation tasks such as machine translation, but its encoding of the entire source sequence into a fixed-size vector has certain limitations, especially when dealing with long source sequences. Attention-based models (Bahdanau et al., 2015; Vaswani et al., 2017) alleviate this limitation by allowing the model to search and condition on parts of a source sentence that are relevant to predicting the next target word, thus moving away from a framework that represents the entire source sequence merely as a single fixed-size vector. While attention models and variants (Bahdanau et al., 2015; Luong et al., 2015, etc.) have contributed to significant progress in the state-of-the-art in translation (Wu et al., 2016) and are very commonly used in neural machine translation nowadays, attention models have been somewhat less effective in E2E dialogue modeling. This can probably be explained by the fact that attention models effectively attempt to "jointly translate and align" (Bahdanau et al., 2015), which is a desirable goal in machine translation as each information piece in the source sequence (foreign sentence) typically needs to be conveyed in the target (translation) exactly once, but this is less true in dialogue data. Indeed, in dialogue entire spans of the source may not map to anything in the target and vice-versa.[2] Some specific attention models for dialogue have been shown to be useful (Yao et al., 2015; Mei et al., 2017; Shao et al., 2017), e.g., to avoid word repetitions (which are discussed further in Sec. 5.2).

5.1.4 Pointer-Network Models

Multiple model extensions (Gu et al., 2016; He et al., 2017a) of the seq2seq framework improve the model's ability to "copy and paste" words between the conversational context and the response. Compared to other tasks such as translation, this ability is particularly important in dialogue, as the response often repeats spans of the input (e.g., "good morning" in response to "good morning") or uses rare words such as proper nouns, which the model would have difficulty generating with a standard RNN. Originally inspired by the Pointer Network model (Vinyals et al., 2015a)—which produces an output sequence consisting of elements from the input sequence—these models hypothesize target words that are either drawn from a fixed-size vocabulary (akin to a seq2seq model) or selected from the source sequence (akin to a pointer network) using an attention mechanism. An instance of this model is CopyNet (Gu et al., 2016), which was shown to significantly improve over RNNs thanks to its ability to repeat proper nouns and other words of the input.

---

[2] Ritter et al. (2011) also found that alignment produced by an off-the-shelf word aligner (Och and Ney, 2003) produced alignments of poor quality, and an extension of their work with attention models (Ritter 2018, pc) yield attention scores that did not correspond to meaningful alignments.

5.2 Challenges and Remedies

The response generation task faces challenges that are rather specific to conversation modeling. Much of the recent research is aimed at addressing the following issues.

5.2.1 Response Blandness

Utterances generated by neural response generation systems are often bland and deflective. While this problem has been noted in other tasks such as image captioning (Mao et al., 2015), the problem is particularly acute in E2E response generation, as commonly used models such as seq2seq tend to generate uninformative responses such as "I don't know" or "I'm OK". Li et al. (2016a) suggested that this is due to their training objective, which optimizes the likelihood of the training data according to $p(T|S)$, where S is the source (dialogue history) and T is the target response. The objective $p(T|S)$ is asymmetrical in T and S, which causes the trained systems to prefer responses T that unconditionally enjoy high probability, i.e., irrespectively of the context S. For example, such systems often respond "I don't know" if S is a question, as the response "I don't know" is plausible for almost all questions. Li et al. (2016a) suggested replacing the conditional probability $p(T|S)$ with mutual information $\frac{p(T,S)}{p(T)p(S)}$ as an objective, since the latter formulation is symmetrical in S and T, thus giving no incentive for the learner to bias responses T to be particularly bland and deflective, unless such a bias emerges from the training data itself. While this argument may be true in general, optimizing the mutual information objective (also known as Maximum Mutual Information or MMI (Huang et al., 2001)) can be challenging, so Li et al. (2016a) used that objective at inference time. More specifically, given a conversation history S, the goal at inference time is to find the best T according to:[3]

$$\begin{aligned}\hat{T} &= \text{argmax}_T \left\{ \log \frac{p(S,T)}{p(S)p(T)} \right\} \\ &= \text{argmax}_T \left\{ \log p(T|S) - \log p(T) \right\}\end{aligned} \tag{5.7}$$

A hyperparameter λ was introduced to control how much to penalize generic responses, with either formulations:[4]

$$\begin{aligned}\hat{T} &= \text{argmax}_T \left\{ \log p(T|S) - \lambda \log p(T) \right\} \\ &= \text{argmax}_T \big\{ (1-\lambda) \log p(T|S) \\ &\qquad + \lambda \log p(S|T) - \lambda \log p(S) \big\} \\ &= \text{argmax}_T \left\{ (1-\lambda) \log p(T|S) + \lambda \log p(S|T) \right\}.\end{aligned} \tag{5.8}$$

Thus, this weighted MMI objective function can be viewed as representing a tradeoff between sources given targets (i.e., $p(S|T)$) and targets given sources (i.e., $p(T|S)$), which is also a tradeoff between response appropriateness and lack of blandness. Note, however, that despite this tradeoff, Li et al. (2016a) have not entirely solved the blandness problem, as this objective is only used at inference and not training time. This approach first generates N-best lists according to $p(T|S)$ and rescores them with MMI. Since such N-best lists tend to be overall relatively bland due to the $p(T|S)$ inference criterion (beam search), MMI rescoring often mitigates rather than completely eliminates the blandness problem.

More recently, researchers (Li et al., 2017c; Xu et al., 2017; Zhang et al., 2018e) have used adversarial training and Generative Adversarial Networks (GAN) (Goodfellow et al., 2014), which often have the effect of reducing blandness. Intuitively, the effect of GAN on blandness can be understood as follows: adversarial training puts a Generator and Discriminator against each other (hence the term "adversarial") using a minimax objective, and the objective for each of them is to make their counterpart the least effective. The Generator is the response generation system to be deployed, while the goal of the Discriminator is to be able to identify whether a given response is generated by a human (i.e., from the training data) or is the output of the Generator. Then, if the Generator

---

[3] Recall that $\log \frac{p(S,T)}{p(S)p(T)} = \log \frac{p(T|S)}{p(T)} = \log p(T|S) - \log p(T)$

[4] The second formulation is derived from:
$\log p(T) = \log p(T|S) + \log p(S) - \log p(S|T)$.

always responds "I don't know" or with other deflective responses, the Discriminator would have little problem distinguishing them from human responses in most of the cases, as most humans do not respond with "I don't know" all the time. Therefore, in order to fool the Discriminator, the Generator progressively steers away from such predictable responses. More formally, the optimality of GAN is achieved when the hypothesis distribution matches the oracle distribution, thus encouraging the generated responses to spread out to reflect the true diversity of real responses. To promote more diversity, Zhang et al. (2018e) explicitly optimize a variational lower bound on pairwise mutual information between query and response to encourage generating more informative responses during training time.

Serban et al. (2017) presented a latent Variable Hierarchical Recurrent Encoder-Decoder (VHRED) model that also aims to generate less bland and more specific responses. It extends the HRED model described previously in this chapter, by adding a high-dimensional stochastic latent variable to the target. This additional latent variable is meant to address the challenge associated with the *shallow generation* process. As noted in (Serban et al., 2017), this process is problematic from an inference standpoint because the generation model is forced to produce a high-level structure—i.e., an entire response—on a word-by-word basis. This generation process is made easier in the VHRED model, as the model exploits a high-dimensional latent variable that determines high-level aspects of the response (topic, names, verb, etc.), so that the other parts of the model can focus on lower-level aspects of generation, e.g., ensuring fluency. The VHRED model incidentally helps reducing blandness as suggested by sample outputs of (Serban et al., 2017). Indeed, as the content of the response is conditioned on the latent variable, the generated response is only bland and devoid of semantic content if the latent variable determines that the response should be as such. More recently, Zhang et al. (2018b) presented a model that also introduces an additional variable (modeled using a Gaussian kernel layer), which is added to control the level of specificity of the response, going from bland to very specific.

While most response generation systems surveyed earlier in this chapter are generation-based (i.e., generating new sentences word-by-word), a more conservative solution to mitigating blandness is to replace generation-based models with retrieval-based models for response generation (Jafarpour et al., 2010; Lu and Li, 2014; Inaba and Takahashi, 2016; Al-Rfou et al., 2016; Yan et al., 2016), in which the pool of possible responses is constructed in advance (e.g., pre-existing human responses). These approaches come at the cost of reduced flexibility: In generation, the set of possible responses grows exponentially in the number of words, but the set of responses of a retrieval system is fixed, and as such retrieval systems often do not have any appropriate responses for many conversational inputs. Despite this limitation, retrieval systems have been widely used in popular commercial systems, and we survey them in Chapter 6.

5.2.2 Speaker Consistency

It has been shown that the popular seq2seq approach often produces conversations that are incoherent (Li et al., 2016b), where the system may for instance contradict what it had just said in the previous turn (or sometimes even in the same turn). While some of this effect can be attributed to the limitation of the learning algorithms, Li et al. (2016b) suggested that the main cause of this inconsistency is probably due to the training data itself. Indeed, conversational datasets (see Sec. 5.5) feature multiple speakers, which often have different or conflicting personas and backgrounds. For example, to the question "how old are you?", a seq2seq model may give valid responses such as "23", "27", or "40", all of which are represented in the training data.

This sets apart the response generation task from more traditional NLP tasks: While models for other tasks such as machine translation are trained on data that is mostly one-to-one semantically, conversational data is often one-to-many or many-to-many as the above example implies.[5] As one-to-many training instances are akin to noise to any learning algorithm, one needs more expressive models that exploits a richer input to better account for such diverse responses.

To do this, Li et al. (2016b) proposed a persona-based response generation system, which is an extension of the LSTM model of Sec. 5.1.1 that uses speaker embeddings in addition to word embeddings. Intuitively, these two types of embeddings work similarly: while word embeddings form

---

[5] Conversational data is also many-to-one, for example with multiple semantically-unrelated inputs that map to "I don't know."

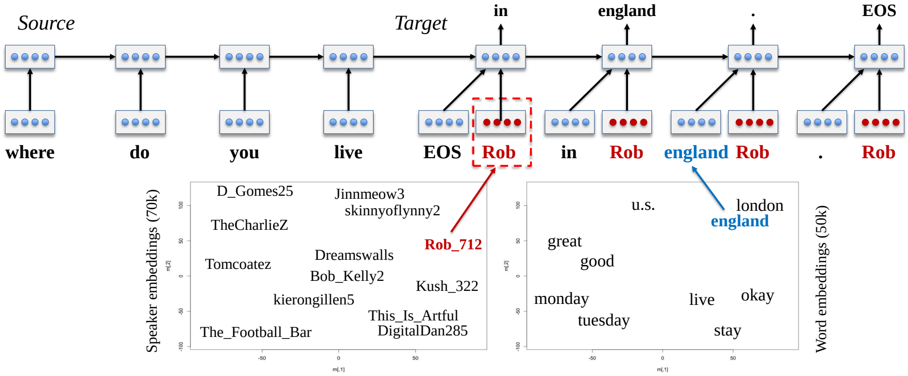


Figure 5.2: Persona-based response generation system. Figure credit: Li et al. (2016b)

a latent space in which spacial proximity (i.e., low Euclidean distance) means two words are semantically or functionally close, speaker embeddings also constitute a latent space in which two nearby speakers tend to converse in the same way, e.g., having similar speaking styles (e.g., British English) or often talking about the same topic (e.g., sports).

Like word embeddings, speaker embedding parameters are learned jointly with all other parameters of the model from their one-hot representations. At inference time, one just needs to specify the one-hot encoding of the desired speaker to produce a response that reflects her speaking style. The global architecture of the model is displayed in Fig. 5.2, which shows that each target hidden state is conditioned not only on the previous hidden state and the current word embedding (e.g., “England”), but also on the speaker embedding (e.g., of “Rob”). This model not only helps generate more personalized responses, but also alleviates the one-to-many modeling problem mentioned earlier.

Other approaches also utilized personalized information. For example, Al-Rfou et al. (2016) presented a persona-based response generation model, but geared for retrieval using an extremely large dataset consisting of 2.1 billion responses. Their retrieval model is implemented as a binary classifier (i.e., good response or not) using a deep neural network. The distinctive feature of their model is a multi-loss objective, which augments a single-loss model $p(R|I, A, C)$ of the response R, input I, speaker (“author”) A, and context C, by adding auxiliary losses that, e.g., model the probability of the response given the author $p(R|A)$. This multi-loss model was shown to be quite helpful (Al-Rfou et al., 2016), as the multiple losses help cope with the fact that certain traits of the author are often correlated with the context or input, which makes it difficult to learn good speaker embedding representation. By adding a loss for $p(R|A)$, the model is able to learn a more distinctive speaker embedding representation for the author.

More recently, Luan et al. (2017) presented an extension of the speaker embedding model of Li et al. (2016b), which combines a seq2seq model trained on conversational datasets with an autoencoder trained on non-conversational data, where the seq2seq and autoencoder are combined in a multi-task learning setup (Caruana, 1998). The tying of the decoder parameters of both seq2seq and autoencoder enables Luan et al. (2017) to train a response generation system for a given persona without actually requiring any conversational data available for that persona. This is an advantage of their approach, as conversational data for a given user or persona might not always be available. In (Bhatia et al., 2017), the idea of (Li et al., 2016b) is extended to a social-graph embedding model.

While (Serban et al., 2017) is not a persona-based response generation model *per se*, their work shares some similarities with speaker embedding models such as (Li et al., 2016b). Indeed, both Li et al. (2016b) and Serban et al. (2017) introduced a continuous high-dimensional variable in the target side of the model in order to bias the response towards information encoded in a vector. In the case of (Serban et al., 2017), that variable is latent, and is trained by maximizing a variational lower-bound on the log-likelihood. In the case of (Li et al., 2016b), the variable (i.e., the speaker embedding) is technically also latent, although it is a direct function of the one-hot representation of speaker. (Li et al., 2016b) might be a good fit when utterance-level information (e.g., speaker ID or topic) is available. On the other hand, the strength of (Serban et al., 2017) is that it learns a latent variable that best “explains” the data, and may learn a representation that is more optimal than the one based strictly on speaker or topic information.

5.2.3 Word Repetitions

Word or content repetition is a common problem with neural generation tasks other than machine translation, as has been noted with tasks such as response generation, image captioning, visual story generation, and general language modeling (Shao et al., 2017; Huang et al., 2018; Holtzman et al., 2018). While machine translation is a relatively one-to-one task where each piece of information in the source (e.g., a name) is usually conveyed exactly once in the target, other tasks such as dialogue or story generation are much less constrained, and a given word or phrase in the source can map to zero or multiple words or phrases in the target. This effectively makes the response generation task much more challenging, as generating a given word or phrase doesn’t completely preclude the need of generating the same word or phrase again. While the attention model (Bahdanau et al., 2015) helps prevent repetition errors in machine translation as that task is relatively one-to-one,[6] the attention models originally designed for machine translation (Bahdanau et al., 2015; Luong et al., 2015) often do not help reduce word repetitions in dialogue.

In light of the above limitations, Shao et al. (2017) proposed a new model that adds self-attention to the decoder, aiming at improving the generation of longer and coherent responses while incidentally mitigating the word repetition problem. Target-side attention helps the model more easily keep track of what information has been generated in the output so far,[7] so that the model can more easily discriminate against unwanted word or phrase repetitions.

5.2.4 Further Challenges

The above issues are significant problems that have only been partially solved and that require further investigation. However, a much bigger challenge faced by these E2E systems is response appropriateness. As explained in Chapter 1, one of the most distinctive characteristics of earlier E2E systems, when compared to traditional dialogue systems, is their lack of *grounding*. When asked “what is the weather forecast for tomorrow?”, E2E systems are likely to produce responses such as “sunny” and “rainy”, without a principled basis for selecting one response or the other, as the context or input might not even specify a geographical location. Ghazvininejad et al. (2018) argued that seq2seq and similar models are usually quite good at producing responses that have plausible overall *structure*, but often struggle when it comes to generating names and facts that connect to the real world, due to the lack of grounding. In other words, responses are often pragmatically correct (e.g., a question would usually be followed by an answer, and an apology by a downplay), but the semantic content of the response is often inappropriate. Hence, recent research in E2E dialogue has increasingly focused on designing *grounded* neural conversation models, which we will survey next.

5.3 Grounded Conversation Models

Unlike task-oriented dialogue systems, most E2E conversation models are not grounded in the real world, which prevents these systems from effectively conversing about anything that relates to the user’s environment. This limitation is also inherited from machine translation, which neither models nor needs are grounded. Recent approaches to neural response generation address this problem by grounding systems in the persona of the speaker or addressee (Li et al., 2016b; Al-Rfou et al., 2016), textual knowledge sources such as Foursquare (Ghazvininejad et al., 2018), the user’s or agent’s visual environment (Das et al., 2017a; Mostafazadeh et al., 2017), and affect or emotion of the user (Huber et al., 2018; Winata et al., 2017; Xu et al., 2018). At a high level, most of these works have in common the idea of augmenting their context encoder to not only represent the conversation history, but also some additional input drawn from the user’s environment, such as an image (Das et al., 2017a; Mostafazadeh et al., 2017) or textual information (Ghazvininejad et al., 2018).

---

[6] Ding et al. (2017) indeed found that word repetition errors, usually few in machine translation, are often caused by incorrect attention.

[7] A seq2seq model can also keep track of what information has been generated so far. However, this becomes more difficult as contexts and responses become longer, as a seq2seq hidden state is a fixed-size vector.

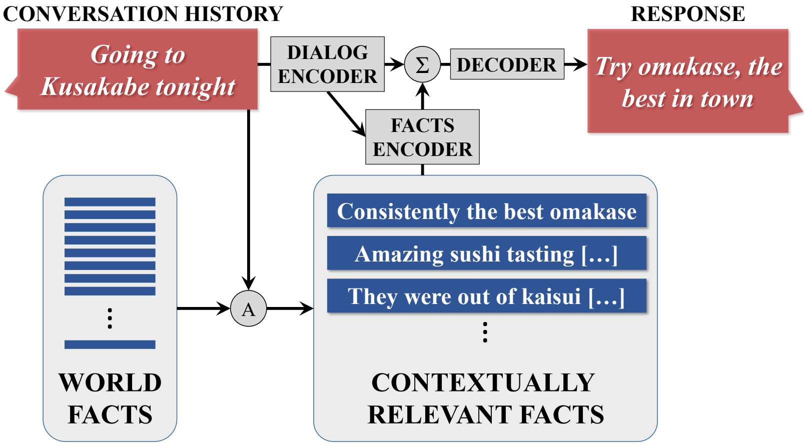

CONVERSATION HISTORY
Going to Kusakabe tonight
DIALOG ENCODER
Σ
DECODER
RESPONSE
Try omakase, the best in town
FACTS ENCODER
A
WORLD FACTS
Consistently the best omakase
Amazing sushi tasting […]
They were out of kaisui […]
CONTEXTUALLY RELEVANT FACTS

Figure 5.3: A neural conversation model grounded in "facts" relevant to the current conversation. Figure credit: Ghazvininejad et al. (2018)

As an illustrative example of such grounded models, we give a brief overview of Ghazvininejad et al. (2018), whose underlying model is depicted in Fig. 5.3. The model mainly consists of two encoders and one decoder. The decoder and the dialogue encoder are similar to those of standard seq2seq models. The additional encoder is called the *facts encoder*, which infuses into the model factual information or so-called *facts* relevant to the conversation history, e.g., restaurant reviews (e.g., "amazing sushi tasting") that pertain to a restaurant that happened to be mentioned in the conversation history (e.g., "Kusakabe"). While the model in this work was trained and evaluated with Foursquare reviews, this approach makes no specific assumption that the grounding consists of reviews, or that trigger words are restaurants (in fact, some of the trigger words are, e.g., hotels and museums). To find facts that are relevant to the conversation, their system uses an IR system to retrieve text from a very large collection of facts or *world facts* (e.g., all Foursquare reviews of several large cities) using search words extracted from the conversation context. While the dialogue encoder of this model is a standard LSTM, the facts encoder is an instance of the Memory Network of Chen et al. (2016b), which uses an associative memory for modeling the facts relevant to a particular problem, which in this case is a restaurant mentioned in a conversation.

There are two main benefits to this approach and other similar work on grounded conversation modeling. First, the approach splits the input of the E2E system into two parts: the input from the user and the input from her environment. This separation is crucial because it addresses the limitation of earlier E2E (e.g., seq2seq) models which always respond deterministically to the same query (e.g.to "what's the weather forecast for tomorrow?"). By splitting input into two sources (user and environment), the system can effectively generate different responses to the same user input depending on what has changed in the real world, without having to retrain the entire system. Second, this approach is much more sample efficient compared to a standard seq2seq approach. For an ungrounded system to produce a response like the one in Fig. 5.3, the system would require that every entity any user might conceivably talk about (e.g., "Kusakabe" restaurant) be seen in the *conversational* training data, which is an unrealistic and impractical assumption. While the amount of language modeling data (i.e., non-conversational data) is abundant and can be used to train grounded conversation systems (e.g., using Wikipedia, Foursquare), the amount of available conversational data is typically much more limited. Grounded conversational models don't have that limitation, and, e.g., the system of Ghazvininejad et al. (2018) can converse about venues that are not even mentioned in the conversational training data.

5.4 Beyond Supervised Learning

There is often a sharp disconnect between conversational training data (human-to-human) and envisioned online scenarios (human-computer). This makes it difficult to optimize conversation models towards specific objectives, e.g., maximizing engagement by reducing blandness. Another limitation of the supervised learning setup is their tendency to optimize for an immediate reward (i.e., one

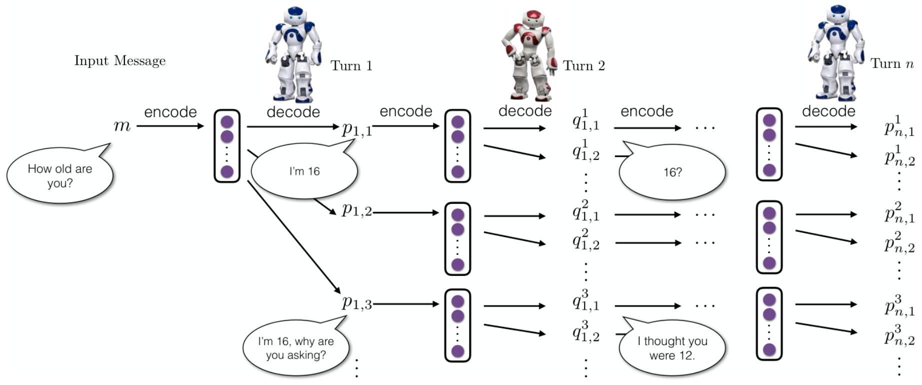


Figure 5.4: Deep reinforcement learning for response generation, pitching the system to optimize against a user simulator (both systems are E2E generation systems.) Figure credit: Li et al. (2017b)

response at a time) rather than a long-term reward. This also partially explains why their responses are often bland and thus fail to promote long-term user engagement. To address these limitations, some researchers have explored reinforcement learning (RL) for E2E systems (Li et al., 2016c) which could be augmented with human-in-the-loop architectures (Li et al., 2017a,b). Unlike RL for task-oriented dialogue, a main challenge that E2E systems are facing is the lack of well-defined metrics for success (i.e., reward functions), in part because they have to deal with informal genres such as chitchat, where the user goal is not explicitly specified.

Li et al. (2016c) constitutes the first attempt to use RL in a fully E2E approach to conversational response generation. Instead of training the system on human-to-human conversations as in the supervised setup of (Sordoni et al., 2015b; Vinyals and Le, 2015), the system of Li et al. (2016c) is trained by conversing with a user simulator which mimics human users' behaviors.

As depicted in Fig. 5.4, human users have to be replaced with a user simulator because it is prohibitively expensive to train an RL system using thousands or tens of thousands of turns of real user dialogues. In this work, a standard seq2seq model is used as a user simulator. The system is trained using policy gradient (Sec. 2.3). The objective is to maximize the expected total reward over the dialogues generated by the user simulator and the agent to be learned. Formally, the objective is

$$J(\theta) = \mathbb{E}[R(T_1, T_2, \ldots, T_N)] \tag{5.9}$$

where $R(.)$ is the reward function, and T_i's are dialogue turns. The above objective can be optimized using gradient descent, by factoring the log probability of the conversation and the aggregated reward, which is independent of the model parameters:

$$\begin{aligned} \nabla J(\theta) &= \nabla \log p(T_1, T_2, \ldots, T_N) R(T_1, T_2, ..., T_N) \\ &\simeq \nabla \log \prod_i p(T_i | T_{i-1}) R(T_1, T_2, ..., T_N) \end{aligned} \tag{5.10}$$

where $p(T_i|T_{i-1})$ is parameterized the same way as the standard seq2seq model of Sec. 5.1.1, except that the model here is optimized using RL. The above gradient is often approximated using sampling, and Li et al. (2016c) used a single sampled conversation for each parameter update. While the above policy gradient setup is relatively common in RL, the main challenge in learning dialogue models is how to devise an effective reward function. Li et al. (2016c) used a combination of three reward functions that are designed to mitigate the problems of the supervised seq2seq model, which was used in their work as initialization parameters. The three reward functions are:

- $-p(\text{Dull Response}|T_i)$: Li et al. (2016c) created a short list of dull responses such as "I don't know" selected from the training data. This reward function penalizes those turns T_i that are likely to lead to any of these dull responses. This is called the *ease of answering* reward, as it promotes conversational turns that are not too difficult to respond to, so as to keep the user engaged in the conversation. For example, the reward function gives a very low reward to turns whose response is "I don't know", as this evasive response indicates that the previous turn was difficult to respond to, which may ultimately terminate the conversation.

- $-\log \text{Sigmoid} \cos(T_{i-1}, T_i)$: This *information flow* reward function ensures that consecutive turns T_{i-1} and T_i are not very similar to each other (e.g., "how are you?" followed by "how are you?"), as Li et al. (2016c) assumed that conversations with little new information are often not engaging and therefore more likely to be terminated.
- $\log p(T_{i-1}|T_i) + \log p(T_i|T_{i-1})$: This *meaningfulness* reward function was mostly introduced to counterbalance the aforementioned two rewards. For example, the two other reward functions prefer the type of conversations that constantly introduce new information and change topics so frequently that users find them hard to follow. To avoid this, the meaningfulness reward encourages consecutive turns in a dialogue session to be related to each other.

5.5 Data

Serban et al. (2015) presented a comprehensive survey of existing datasets that are useful beyond the E2E and social bot research. What distinguishes E2E conversation modeling from other NLP and dialogue tasks is that data is available in very large quantities, thanks in part to social media (e.g., Twitter and Reddit). On the other hand, most of this social media data is neither redistributable nor available through language resource organizations (such as the Linguistic Data Consortium), which means there are still no established public datasets (either with Twitter or Reddit) for training and testing response generation systems. Although these social media companies offer API access to enable researchers to download social media posts in relatively small quantities and then to reconstruct conversations from them, the strict legal terms of the service specified by these companies inevitably affect the reproducibility of the research. Most notably, Twitter makes certain tweets (e.g., retracted tweets or tweets from suspended user) unavailable through the API and requires that any such previously downloaded tweets be deleted. This makes it difficult to establish any standard training or test datasets, as these datasets deplete over time.[8] Consequently, in most of the papers cited in this chapter, their authors have created their own (subsets of) conversational data for training and testing, and then evaluated their systems against baselines and competing systems on these fixed datasets. Dodge et al. (2016) used an existing dataset to define standard training and test sets, but it is relatively small. Some of the most notable E2E and chitchat datasets include:

- **Twitter**: Used since the first data-driven response generation systems (Ritter et al., 2011), Twitter data offers a wealth of conversational data that is practically unbounded, as Twitter produces new data each day that is more than most system developers can handle.[9] While the data itself is made accessible through the Twitter API as individual tweets, its metadata easily enables the construction of conversation histories, e.g., between two users. This dataset forms the basis of the DSTC Task 2 competition in 2017 (Hori and Hori, 2017).
- **Reddit**: Reddit is a social media source that is also practically unbounded, and represents about 3.2 billion dialogue turns as of July 2018. It was for example used in Al-Rfou et al. (2016) to build a large-scale response retrieval system. Reddit data is organized by topics (i.e."subreddits"), and its responses don't have a character limit as opposed to Twitter.
- **OpenSubtitles**: This dataset consists of subtitles made available on the opensubtitles.org website. It offers captions of many commercial movies, and contains about 8 billion words as of 2011 in multiple languages (Tiedemann, 2012).
- **Ubuntu**: The Ubuntu dataset (Lowe et al., 2015) has also been used extensively for E2E conversation modeling. It differs from other datasets such as Twitter in that it is less focused on chitchat but more goal-oriented, as it contains many dialogues that are specific to the Ubuntu operating system.
- **Persona-Chat** dataset: This crowdsourced dataset (Zhang et al., 2018c) was developed to meet the need for conversational data where dialogues exhibit distinct user personas. In collecting Persona-Chat, every crowdworker was asked to impersonate a given character

---

[8] Anecdotally, the authors of Li et al. (2016a, pc) found that a Twitter dataset from 2013 had lost about 25% of its tweets by 2015 due to retracted tweets and Twitter account suspensions.

[9] For example, the latest official statistics from Twitter, dating back from 2014, states that Twitter users post on average more than 500 million tweets per day: `https://blog.twitter.com/official/en_us/a/2014/the-2014-yearontwitter.html`

described using five facts. Then that worker took part in dialogues while trying to stay in character. The resulting dataset contains about 160k utterances.

5.6 Evaluation

Evaluation is a long-standing research topic for generation tasks such as machine translation and summarization. E2E dialogue is no different. While it is common to evaluate response generation systems using human raters (Ritter et al., 2011; Sordoni et al., 2015b; Shang et al., 2015, etc.), this type of evaluation is often expensive and researchers often have to resort to automatic metrics for quantifying day-to-day progress and for performing automatic system optimization. E2E dialogue research mostly borrowed those metrics from machine translation and summarization, using string and n-gram matching metrics like BLEU (Papineni et al., 2002) and ROUGE (Lin, 2004). Proposed more recently, METEOR (Banerjee and Lavie, 2005) aims to improve BLEU by identifying synonyms and paraphrases between the system output and the human reference, and has also been used to evaluate dialogue. deltaBLEU (Galley et al., 2015) is an extension of BLEU that exploits numerical ratings associated with conversational responses.

There has been significant debate as to whether such automatic metrics are actually appropriate for evaluating conversational response generation systems. For example, Liu et al. (2016) argued that they are not appropriate by showing that most of these machine translation metrics correlate poorly with human judgments. However, their correlation analysis was performed at the sentence level, but decent sentence-level correlation has long been known to be difficult to achieve even for machine translation (Callison-Burch et al., 2009; Graham et al., 2015), the task for which the underlying metrics (e.g., BLEU and METEOR) were specifically intended.[10] In particular, BLEU (Papineni et al., 2002) was designed from the outset to be used as a corpus-level rather than sentence-level metric, since assessments based on n-gram matches are brittle when computed on a single sentence. Indeed, the empirical study of Koehn (2004) suggested that BLEU is not reliable on test sets consisting of fewer than 600 sentences. Koehn (2004)'s study was on translation, a task that is arguably simpler than response generation, so the need to move beyond sentence-level correlation is probably even more critical in dialogue. When measured at a corpus- or system-level, correlations are typically much higher than that at sentence-level (Przybocki et al., 2009), e.g., with Spearman's ρ above 0.95 for the best metrics on WMT translation tasks (Graham and Baldwin, 2014).[11] In the case of dialogue, Galley et al. (2015) showed that the correlation of string-based metrics (BLEU and deltaBLEU) significantly increases with the units of measurement bigger than a sentence. Specifically, their Spearman's ρ coefficient goes up from 0.1 (essentially no correlation) at sentence-level to nearly 0.5 when measuring correlation on corpora of 100 responses each.

Recently, Lowe et al. (2017) proposed a machine-learned metric for E2E dialogue evaluation. They presented a variant of the VHRED model (Serban et al., 2017) that takes context, user input, gold and system responses as input, and produces a qualitative score between 1 and 5. As VHRED is effective for modeling conversations, Lowe et al. (2017) was able to achieve an impressive Spearman's ρ correlation of 0.42 at the sentence level. On the other hand, the fact that this metric is trainable leads to other potential problems such as overfitting and "gaming of the metric" (Albrecht and Hwa, 2007),[12] which might explain why previously proposed machine-learned evaluation metrics (Corston-Oliver

---

[10] For example, in the official report of the WMT shared task, Callison-Burch et al. (2009, Section 6.2) computed the percentage of times popular metrics are consistent with human ranking at the sentence level, but the results did not bode well for sentence-level studies: "Many metrics failed to reach [a random] baseline (including most metrics in the out-of-English direction). This indicates that sentence-level evaluation of machine translation quality is very difficult."

[11] In one of the largest scale system-level correlation studies to date, Graham and Baldwin (2014) found that BLEU is relatively competitive against most translation metrics proposed more recently, as they show there "is currently insufficient evidence for a high proportion of metrics to conclude that they outperform BLEU". Such a large scale study remains to be done for dialogue.

[12] In discussing the potential pitfalls of machine-learned evaluation metrics, Albrecht and Hwa (2007) argued for example that it would be "prudent to defend against the potential of a system gaming a subset of the features." In the case of deep learning, this gaming would be reminiscent of making non-random perturbations to an input to drastically change the network's predictions, as it was done, e.g., with images in (Szegedy et al., 2013) to show how easily deep learning models can be fooled. However, preventing such a gaming is difficult if the machine-learned metric is to become a standard evaluation, and this would presumably require model parameters to be publicly available.

et al., 2001; Kulesza and Shieber, 2004; Lita et al., 2005; Albrecht and Hwa, 2007; Giménez and Màrquez, 2008; Pado et al., 2009; Stanojević and Sima'an, 2014, etc.) are not commonly used in official machine translation benchmarks. The problem of "gameable metrics" is potentially serious, for example in the frequent cases where automatic evaluation metrics are used directly as training objectives (Och, 2003; Ranzato et al., 2015) as unintended "gaming" may occur unbeknownst to the system developer. If a generation system is optimized directly on a trainable metric, then the system and the metric become akin to an adversarial pair in GANs (Goodfellow et al., 2014), where the only goal of the generation system (Generator) is to fool the metric (Discriminator). Arguably, such attempts become easier with trainable metrics as they typically incorporate thousands or millions of parameters, compared to a relatively parameterless metric like BLEU that is known to be fairly robust to such exploitation and was shown to be the best metric for direct optimization (Cer et al., 2010) among other established string-based metrics. To prevent machine-learned metrics from being gamed, one would need to iteratively train the Generator and Discriminator as in GANs, but most trainable metrics in the literature do not exploit this iterative process. Adversarial setups proposed for dialogue and related tasks (Kannan and Vinyals, 2016; Li et al., 2017c; Holtzman et al., 2018) offer solutions to this problem, but it is also well-known that such setups suffer from instability (Salimans et al., 2016) due to the nature of GANs' minimax formulation. This fragility is potentially troublesome as the outcome of an automatic evaluation should ideally be stable (Cer et al., 2010) and reproducible over time, e.g., to track progress of E2E dialogue research over the years. All of this suggests that automatic evaluation for E2E dialogue is far from a solved problem.

5.7 Open Benchmarks

Open benchmarks have been the key to achieving progress in many AI tasks such as speech recognition, information retrieval, and machine translation. Although end-to-end conversational AI is a relatively nascent research problem, some open benchmarks have already been developed:

- **Dialog System Technology Challenges (DSTC)**: In 2017, DSTC proposed for the first time an "End-to-End Conversation Modeling" track,[13] which requires systems to be fully data-driven using Twitter data. Two of the tasks in the subsequent challenge (DSTC7) focus on grounded conversation scenarios. One is focused on audio-visual scene-aware dialogue and the other on response generation grounded in external knowledge (e.g., Foursquare and Wikipedia), with conversations extracted from Reddit.[14]
- **ConvAI Competition**: This is a NIPS competition that has been featured so far at two conferences. It offers prizes in the form of Amazon Mechanical Turk funding. The competition aims at "training and evaluating models for non-goal-oriented dialogue systems", and in 2018 uses the Persona-Chat dataset (Zhang et al., 2018c), among other datasets.
- **NTCIR STC**: This benchmark focuses on conversation "via short texts". The first benchmark focused on retrieval-based methods, and in 2017 was expanded to evaluate generation-based approaches.
- **Alexa Prize**: In 2017, Amazon organized an open competition on building "social bots" that can converse with humans on a range of current events and topics. The competition enables participants to test their systems with real users (Alexa users), and offers a form of indirect supervision as users are asked to rate each of their conversations with each of the Alexa Prize systems. The inaugural prize featured 15 academic teams (Ram et al., 2018).[15]

---

[13] `http://workshop.colips.org/dstc6/call.html`

[14] `http://workshop.colips.org/dstc7/`

[15] These 15 systems are described in the online proceeding: `https://developer.amazon.com/alexaprize/proceedings`

Chapter 6

Conversational AI in Industry

This chapter pictures the landscape of conversational systems in industry, including task-oriented systems (e.g., personal assistants), QA systems, and chatbots.

6.1 Question Answering Systems

Search engine companies, including Google, Microsoft and Baidu, have incorporated multi-turn QA capabilities into their search engines to make user experience more conversational, which is particularly appealing for mobile devices. Since relatively little is publicly known about the internals of these systems (e.g., Google and Baidu), this section presents a few example commercial QA systems whose architectures have been at least partially described in public source, including Bing QA, Satori QA and customer support agents.

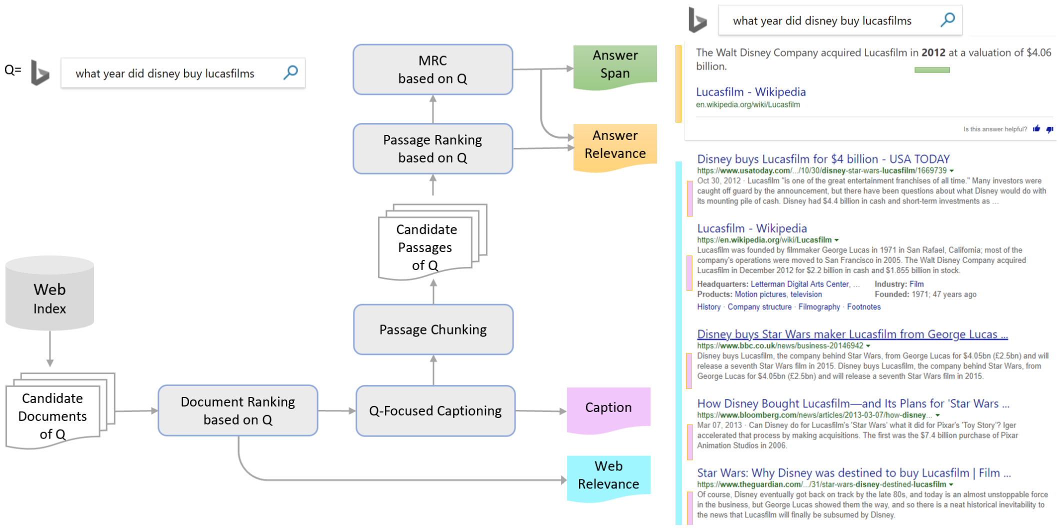
6.1.1 Bing QA

Bing QA is an example of the Web-scale text-QA agents. It is an extension of the Microsoft Bing Web search engine. Instead of returning *ten blue links*, Bing QA generates a direct answer to a user query by reading the passages retrieved by the Bing Web search engine using MRC models, as illustrated in Fig. 6.1.

The Web QA task that Bing QA is dealing with is far more challenging than most of the academic MRC tasks described in Chapter 3. For example, Web QA and SQuAD differs in:

- **Scale and quality of the text collection.** SQuAD assumes the answer is a text span in a passage which is a *clean* text section from a Wikipedia page. Web QA needs to identify an answer from billions of Web documents which consist of trillions of *noisy* passages that often contain contradictory, wrong, obsolete information due to the dynamic nature of Web content.
- **Runtime latency.** In an academic setting, an MRC model might take seconds to read and re-read documents to generate an answer, while in the Web QA setting the MRC part (e.g., in Bing QA) is required to add no more than 10 mini seconds to the entire serving stack.
- **User experience.** While SQuAD MRC models provide a text span as an answer, Web QA needs to provide different user experiences depending on different devices where the answer is shown, e.g., a voice answer in a mobile device or a rich answer in a Search Engine Result Page (SERP). Fig. 6.1 (Right) shows an example of the SERP for the question “what year did Disney buy lucasfilms?”, where Bing QA presents not only the answer as a highlighted text span, but also various supporting evidence and related Web search results (i.e., captions of retrieved documents, passages, audios and videos) that are consistent with the answer.

As a result, a commercial Web QA agent such as Bing QA often incorporates a MRC module as a post-web component on top of its Web search engine stack. An overview of the Bing QA agent is



what year did disney buy lucasfilms

The Walt Disney Company acquired Lucasfilm in **2012** at a valuation of $4.06 billion.

Lucasfilm - Wikipedia
en.wikipedia.org/wiki/Lucasfilm

Is this answer helpful?

Disney buys Lucasfilm for $4 billion - USA TODAY
https://www.usatoday.com/.../10/30/disney-star-wars-lucasfilm/1669739
Oct 30, 2012 · Lucasfilm "is one of the great entertainment franchises of all time." Many investors were caught off guard by the announcement, but there have been questions about what Disney would do with its mounting pile of cash. Disney had $4.4 billion in cash and short-term investments as ...

Lucasfilm - Wikipedia
https://en.wikipedia.org/wiki/Lucasfilm
Lucasfilm was founded by filmmaker George Lucas in 1971 in San Rafael, California; most of the company's operations were moved to San Francisco in 2005. The Walt Disney Company acquired Lucasfilm in December 2012 for $2.2 billion in cash and $1.855 billion in stock.
Headquarters: Letterman Digital Arts Center, ...
Industry: Film
Products: Motion pictures, television
Founded: 1971; 47 years ago
History · Company structure · Filmography · Footnotes

Disney buys Star Wars maker Lucasfilm from George Lucas ...
https://www.bbc.co.uk/news/business-20146942
Disney buys Lucasfilm, the company behind Star Wars, from George Lucas for $4.05bn (£2.5bn) and will release a seventh Star Wars film in 2015. Disney buys Lucasfilm, the company behind Star Wars, from George Lucas for $4.05bn (£2.5bn) and will release a seventh Star Wars film in 2015.

How Disney Bought Lucasfilm—and Its Plans for 'Star Wars ...
https://www.bloomberg.com/news/articles/2013-03-07/how-disney...
Mar 07, 2013 · Can Disney do for Lucasfilm's 'Star Wars' what it did for Pixar's 'Toy Story'? Iger accelerated that process by making acquisitions. The first was the $7.4 billion purchase of Pixar Animation Studios in 2006.

Star Wars: Why Disney was destined to buy Lucasfilm | Film ...
https://www.theguardian.com/.../31/star-wars-disney-destined-lucasfilm
Of course, Disney eventually got back on track by the late 80s, and today is an almost unstoppable force in the business, but George Lucas showed them the way, and so there is a neat historical inevitability to the news that Lucasfilm will finally be subsumed by Disney.

Figure 6.1: (Left) An overview of the Bing QA architecture. (Right) An example of a search engine result page of the question "what year did disney buy lucasfilms?". Example graciously provided by Rangan Majumder.

illustrated in Fig. 6.1 (Left). Given the question "what year did Disney buy lucasfilms?", a set of candidate documents are retrieved from Web Index via a fast, primary ranker. Then in the Document Ranking module, a sophisticated document ranker based on boosted trees (Wu et al., 2010) is used to assign relevance scores for these documents. The top-ranked relevant documents are presented in a SERP, with their captions generated from a Query-Focused Captioning module, as shown in Fig. 6.1 (Right). The Passage Chunking module segments the top documents into a set of candidate passages, which are further ranked by the Passage Ranking module based on another passage-level boosted trees ranker (Wu et al., 2010). Finally, the MRC module identifies the answer span "2012" from the top-ranked passages.

Although turning Bing QA into a conversational QA agent of Sec. 3.8 requires the integration of additional components such as dialogue manager, which is a nontrivial ongoing engineering effort, Bing QA can already deal with conversational queries (e.g., follow up questions) using a Conversational Query Understanding (CQU) module (Ren et al., 2018a). As the example in Fig. 6.2, CQU reformulates a conversational query into a search engine friendly query in two steps: (1) determine whether a query depends upon the context in the same search session (i.e., previous queries and answers), and (2) if so, rewrite that query to include the necessary context e.g., replace "its" with "California" in Q2 and add "Stanford" in Q5 in Fig. 6.2.

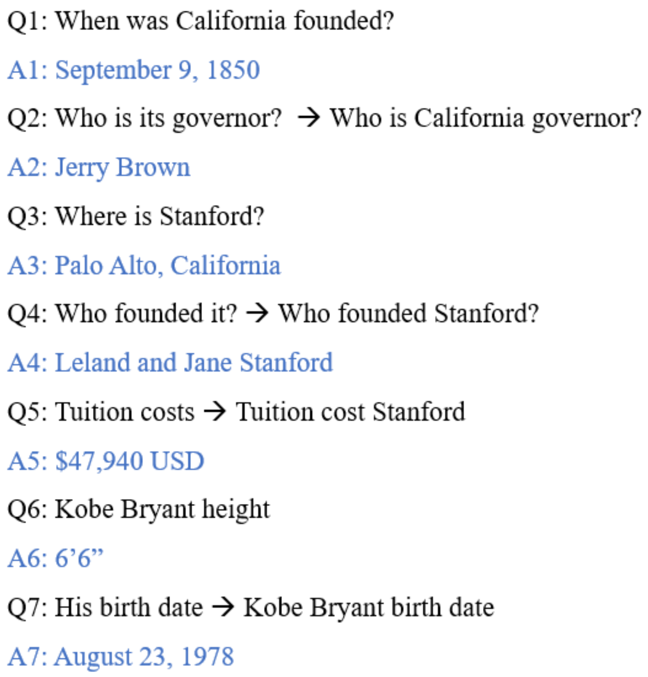
6.1.2 Satori QA

Satori QA is an example of the KB-QA agents, as described in Sec. 3.1–3.5. Satori is Microsoft's knowledge graph, which is seeded by Freebase, and now is several orders of magnitude larger than Freebase. Satori QA is a hybrid system that uses both neural approaches and symbolic approaches. It generates answers to factual questions.

Similar to Web QA, Satori QA has to deal with the issues regarding scalability, noisy content, speed, etc. One commonly used design strategy of improving system's robustness and runtime efficiency is to decompose a complex question into a sequence of simpler questions, which can be answered more easily by a Web-scale KB-QA system, and compute the final answer by recomposing the sequence of answers, as exemplified in Fig. 6.3 (Talmor and Berant, 2018).

6.1.3 Customer Support Agents

Several IT companies, including Microsoft and Salesforce, have developed a variety of customer support agents. These agents are multi-turn conversational KB-QA agents, as described in 3.5.



Q1: When was California founded?

A1: September 9, 1850

Q2: Who is its governor? → Who is California governor?

A2: Jerry Brown

Q3: Where is Stanford?

A3: Palo Alto, California

Q4: Who founded it? → Who founded Stanford?

A4: Leland and Jane Stanford

Q5: Tuition costs → Tuition cost Stanford

A5: $47,940 USD

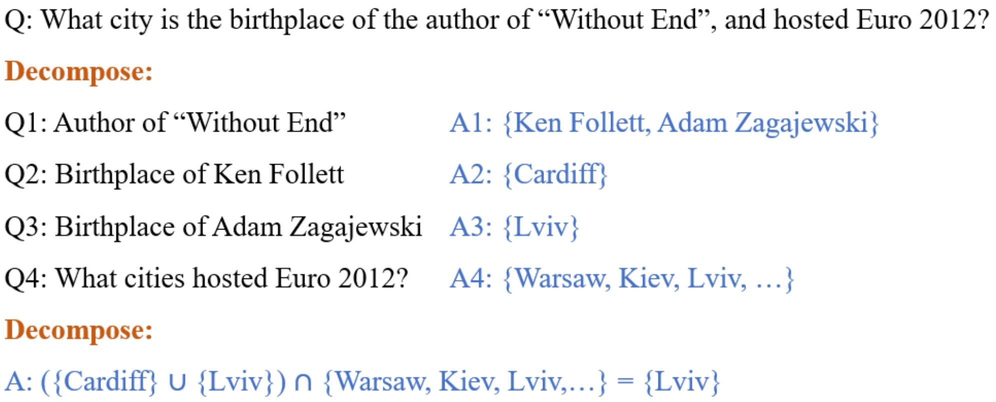
Q6: Kobe Bryant height

A6: 6’6”

Q7: His birth date → Kobe Bryant birth date

A7: August 23, 1978

Figure 6.2: An example query session, where some queries are rewritten to include context information via the CQU module as indicated by the arrows. Examples adapted from Ren et al. (2018a).



Q: What city is the birthplace of the author of “Without End”, and hosted Euro 2012?

Decompose:

Q1: Author of “Without End”	A1: {Ken Follett, Adam Zagajewski}
Q2: Birthplace of Ken Follett	A2: {Cardiff}
Q3: Birthplace of Adam Zagajewski	A3: {Lviv}
Q4: What cities hosted Euro 2012?	A4: {Warsaw, Kiev, Lviv, …}

Decompose:

A: $(\{\text{Cardiff}\} \cup \{\text{Lviv}\}) \cap \{\text{Warsaw, Kiev, Lviv,}\ldots\} = \{\text{Lviv}\}$

Figure 6.3: Given a complex question Q, we decompose it to a sequence of simple questions $Q_1, Q_2, \ldots$, use a Web-scale KB-QA agent to generate for each Q_i an answer A_i, from which we compute the final answer A. Figure credit: Talmor and Berant (2018).

Given a user’s description of a problem e.g., “cannot update the personal information of my account”, the agent needs to recommend a pre-compiled solution or ask a human agent to help. The dialogue often consists of multiple turns as the agent asks the user to clarify the problem while navigating the knowledge base to find the solution. These agents often take both text and voice as input.

6.2 Task-oriented Dialogue Systems (Virtual Assistants)

Commercial task-oriented dialogue systems nowadays often reside in smart phones, smart speakers and personal computers. They can perform a range of tasks or services for a user, and are sometimes referred to as virtual assistants or intelligent personal assistants. Some of the example services are providing weather information, setting alarms, and calling center support. In the US, the most widely used systems include Apple's Siri, Google Assistant, Amazon Alexa, and Microsoft Cortana, among others. Users can interact with them naturally through voice, text or images. To activate a virtual assistant using voice, a wake word might be used, such as "OK Google."

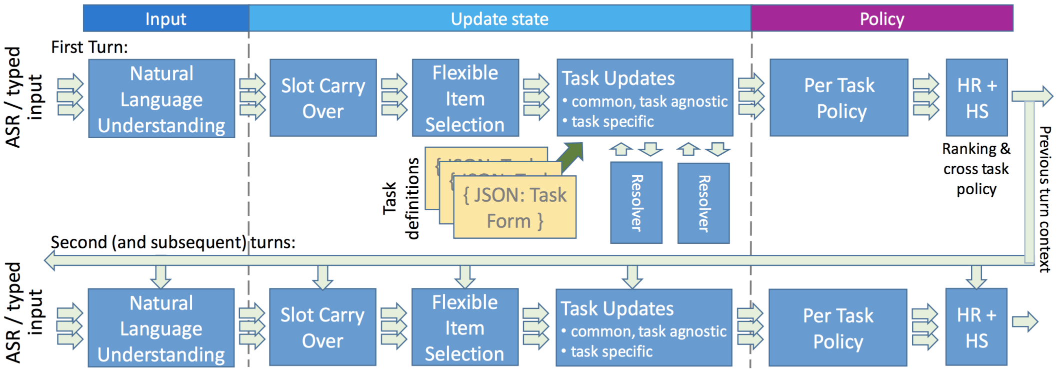


Figure 6.4: Architecture of Task Completion Platform. Figure credit: Crook et al. (2016).

There are also a number of fast-growing tools available to facilitate the development of virtual assistants, including Amazon's Alexa Skills Kit[1], IBM's Watson Assistant[2], and similar offerings from Microsoft and Google, among others. A comprehensive survey is outside of the scope of this section, and not all information of such tools is publicly available. Here, we will give a high-level description of a sample of them:

- The Task Completion Platform (TCP) of Microsoft (Crook et al., 2016) is a platform for creating multi-domain dialogue systems. As shown in Fig. 6.4, TCP follows a similar structure as in Fig. 4.1, containing language understanding, state tracking, and a policy. A useful feature of TCP is a task configuration language, TaskForm, which allows the definitions of individual tasks to be decoupled from the platform's overarching dialogue policy. TCP is used to power many of the multi-turn dialogues supported by the Cortana personal assistant.

- Another tool from Microsoft is **LUIS**, a cloud-based API service for natural language understanding[3]. It provides a suite of pre-built domains and intentions, as well as a convenient interface for a non-expert to use machine learning to obtain an NLU model by providing training examples. Once a developer creates and publishes a LUIS app, the app can be used as a NLU blackbox module by a client dialogue system: the client sends a text utterance to the app, which will return language understanding results in the JSON format, as illustrated in Fig. 6.5.

- While LUIS focuses on language understanding, the **Azure Bot Service**[4] allows developers to build, test, deploy, and manage dialogue systems in one place. It can take advantages of a suite of intelligent services, including LUIS, image captioning, speech-to-text capabilities, among others.

- **DialogFlow** is Google's development suite for creating dialogue systems on websites, mobile and IoT devices.[5] Similar to the above tools, it provides mechanisms to facilitate development of various modules of a dialogue system, including language understanding and carrying information over multiple turns. Furthermore, it can deploy a dialogue system as an action that users can invoke through Google Assistant.

---

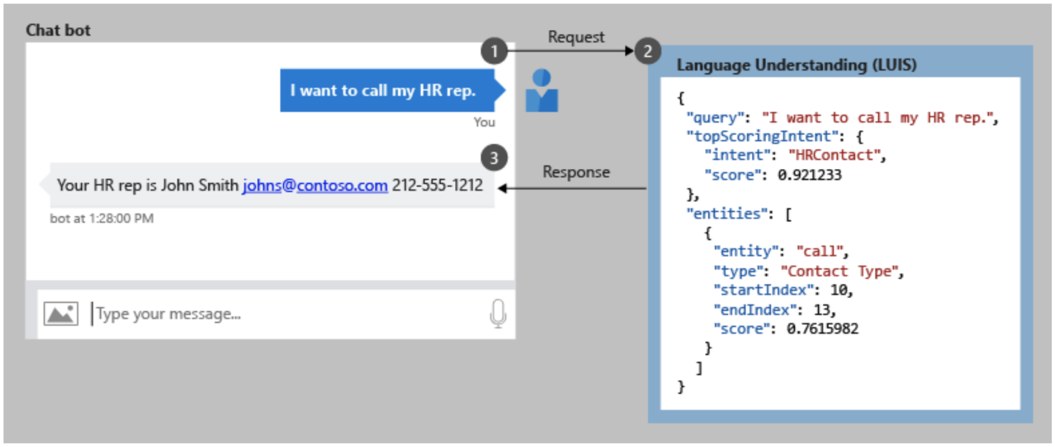
[1] `https://developer.amazon.com/alexa-skills-kit`

[2] `https://ibm.biz/wcsdialog`

[3] `https://www.luis.ai`

[4] `https://docs.microsoft.com/en-us/azure/bot-service/?view=azure-bot-service-3.0`

[5] `https://dialogflow.com`



I want to call my HR rep.

Your HR rep is John Smith johns@contoso.com 212-555-1212

```
{
 "query": "I want to call my HR rep.",
 "topScoringIntent": {
   "intent": "HRContact",
   "score": 0.921233
 },
 "entities": [
   {
    "entity": "call",
    "type": "Contact Type",
    "startIndex": 10,
    "endIndex": 13,
    "score": 0.7615982
   }
  ]
}
```

Figure 6.5: Use of LUIS by a client dialogue system. Figure credit: `https://docs.microsoft.com/en-us/azure/cognitive-services/LUIS` .

6.3 Chatbots

There have been publicly-available conversational systems going back many decades (Weizenbaum, 1966; Colby, 1975). Those precursors of today's chatbot systems relied heavily on hand-crafted rules, and are very different from the data-driven conversational AI systems discussed in Chapter 5. Nowadays publicly available and commercial chatbot systems are often a combination of statistical methods and hand-crafted components, where statistical methods provide robustness to conversational systems (e.g., via intent classifiers) while rule-based components are often still used in practice, e.g., to handle common chitchat queries (e.g., "tell me a joke"). Examples include personal assistants like Amazon's Alexa, Google Assistant, Facebook M, and Microsoft's Cortana, which in addition to personal assistant skills are able to handle chitchat user inputs. Other commercial systems such as XiaoIce,[6] Replika, (Fedorenko et al., 2017) Zo,[7] and Ruuh[8] focus almost entirely on chitchat. Since relatively little is publicly known about the internals of main commercial systems (Alexa, Google Assistant, etc.), the rest of this section focuses on commercial systems whose architecture have been at least partially described in some public source.

One of the earliest such systems is **XiaoIce**, which was initially released in 2014. XiaoIce is designed as an AI companion with an emotional connection to satisfy the human need for communication, affection, and social belonging (Zhou et al., 2018). The overall architecture of XiaoIce is shown in Fig. 6.6. It consists of three layers.

- **User experience layer:** It connects XiaoIce to popular chat platforms (e.g., WeChat, QQ), and deals with conversations in two communication modes. The full-duplex module handles voice-stream-based conversations where both a user and XiaoIce can talk to each other simultaneously. The other module deals with message-based conversations where a user and XiaoIce have to take turns to talk.
- **Conversation engine layer:** In each dialogue turn, the dialogue state is first updated using the state tracker, and either Core Chat (and a topic) or a dialogue skill is selected by the dialogue policy to generate a response. A unique component of XiaoIce is the empathetic computing module, designed to understand not only the content of the user input (e.g., topic) but also the empathy aspects (e.g., emotion, intent, opinion on topic, and the user's background and general interests), to ensure the generation of an empathetic response that fits XiaoIce's persona. Another central module, Core Chat, combines neural generation techniques (Sec. 5.1) and retrieval-based methods (Zhou et al., 2018). As Fig. 6.7 show, XiaoIce is capable of generating socially attractive responses (e.g., having a sense of humor, comforting, etc.), and can determine whether to drive the conversation when, e.g.,

---

[6] `https://www.msxiaobing.com`
[7] `https://www.zo.ai`
[8] `https://www.facebook.com/Ruuh`

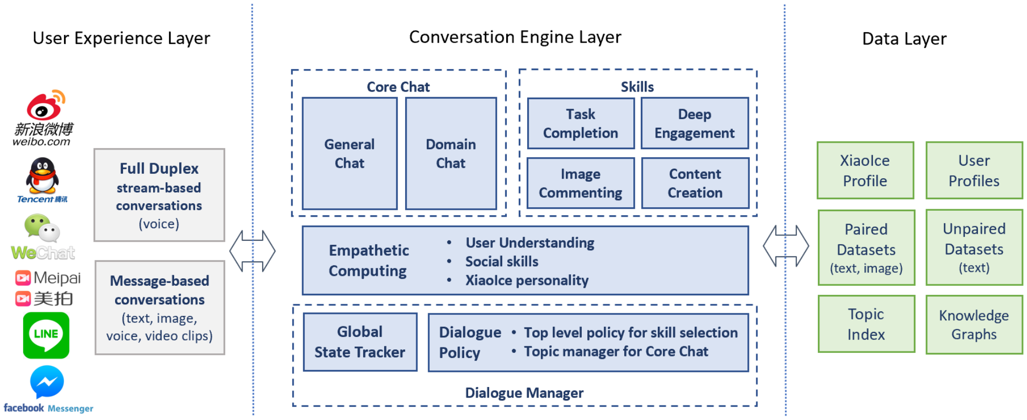
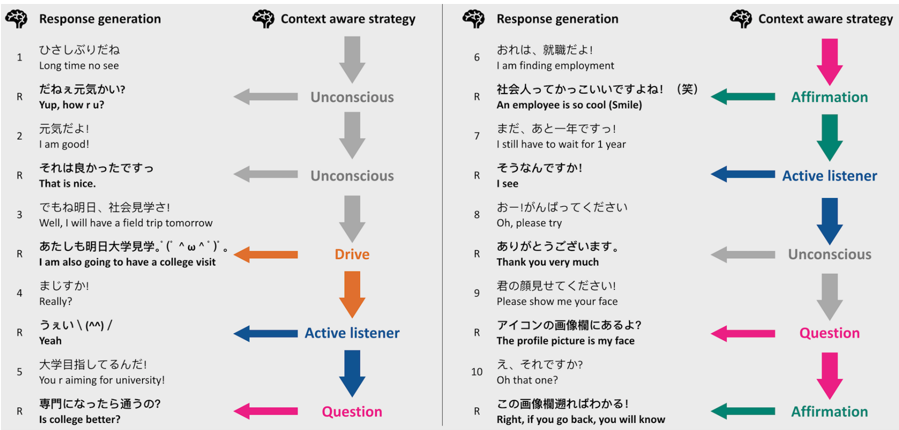


Figure 6.6: XiaoIce system architecture. Figure credit: Zhou et al. (2018)



1 ひさしぶりだね
Long time no see
R **だねぇ元気かい?**
Yup, how r u?
2 元気だよ!
I am good!
R **それは良かったですっ**
That is nice.
3 でもね明日、社会見学さ!
Well, I will have a field trip tomorrow
R **あたしも明日大学見学。゜(゜^ω^゜)゜。**
I am also going to have a college visit
4 まじすか!
Really?
R **うぇい\(^^)/**
Yeah
5 大学目指してるんだ!
You r aiming for university!
R **専門になったら通うの?**
Is college better?
6 おれは、就職だよ!
I am finding employment
R **社会人ってかっこいいですよね!（笑）**
An employee is so cool (Smile)
7 まだ、あと一年ですっ!
I still have to wait for 1 year
R **そうなんですか!**
I see
8 おー!がんばってください
Oh, please try
R **ありがとうございます。**
Thank you very much
9 君の顔見せてください!
Please show me your face
R **アイコンの画像欄にあるよ?**
The profile picture is my face
10 え、それですか?
Oh that one?
R **この画像欄遡ればわかる!**
Right, if you go back, you will know

Figure 6.7: Conversation between a user and XiaoIce. The empathy model provides a context-aware strategy that can drive the conversation when needed.

the conversation is somewhat stalled, or whether to perform active listening when the user herself is engaged.[9]

- **Data layer:** It consists of a set of databases that store the collected human conversational data (in text pairs or text-image pairs), non-conversational data and knowledge graphs used for Core Chat and skills, and the profiles of XiaoIce and all the registered users for empathetic computing.

The **Replika** system (Fedorenko et al., 2017) for chitchat combines neural generation and retrieval-based methods, and is able to condition responses on images as in (Mostafazadeh et al., 2017). The neural generation component of Replika is persona-based (Li et al., 2016b), as it is trained to mimic specific characters. While Replika is a company, the Replika system has been open-sourced[10] and can thus be used as a benchmark for future research.

Alexa Prize systems (Ram et al., 2018) are social chatbots that are exposed to real users, and as such anyone with an Alexa device is able to interact with these social bots and give them ratings. This interaction is triggered with the "Alexa, let's chat" command, which then triggers a free-form conversation about any topic selected by either the user or the system. These systems featured not only fully data-driven approaches, but also more engineered and modularized approaches. For example, the winning system of the 2017 competition (Sounding Board[11]) contained a chitchat

---

[9] `https://www.leiphone.com/news/201807/rgyKfVsEUdK1BpXf.html`
[10] `https://github.com/lukalabs/cakechat`
[11] `https://sounding-board.github.io`

component as well as individual "miniskills" enabling the system to handle distinct tasks (e.g., QA) and topics (e.g., news, sports). Due to the diversity of systems in the Alexa prize, it would be impractical to overview these systems in this survey, and instead we refer the interested reader to the Alexa Prize online proceedings (Ram et al., 2018).

Chapter 7

Conclusions and Research Trends

Conversational AI is a rapidly growing field. This paper surveys neural approaches that were recently developed. Some of them have already been widely used in commercial systems.

- Dialogue systems for question answering, task completion, chitchat and recommendation etc. can be conceptualized using a unified mathematical framework of optimal decision process. The neural approaches to AI, developed in the last few years, leverage the recent breakthrough in RL and DL to significantly improve the performance of dialogue agents across a wide range of tasks and domains.
- A number of commercial dialogue systems allow users to easily access various services and information via conversation. Most of these systems use hybrid approaches that combine the strength of symbolic methods and neural models.
- There are two types of QA agents. KB-QA agents allow users to query large-scale knowledge bases via conversation without composing complicated SQL-like queries. Text-QA agents, equipped with neural MRC models, are becoming more popular than traditional search engines (e.g., Bing and Google) for the query types to which users expect a concise direct answer.
- Traditional task-oriented systems use handcrafted dialogue manager modules, or shallow machine-learning models to optimize the modules separately. Recently, researchers have begun to explore DL and RL to optimize the system in a more holistic way with less domain knowledge, and to automate the optimization of systems in a changing environment such that they can efficiently adapt to different tasks, domains and user behaviors.
- Chatbots are important in facilitating smooth and natural interaction between humans and their electronic devices. More recent work focuses on scenarios beyond chitchat, e.g., recommendation. Most state-of-the-art chatbots use fully data-driven and end-to-end generation of conversational responses within the framework of neural machine translation.

We have discussed some of the main challenges in conversational AI, common to Question Answering agents, task-oriented dialogue bots and chatbots.

- **Towards a unified modeling framework for dialogues:** Chapter 1 presents a unified view where an open-domain dialogue is formulated as an optimal decision process. Although the view provides a useful design principle, it remains to be proved the effectiveness of having a unified modeling framework for system development. Microsoft XiaoIce, initially designed as a chitchat system based on a retrieval engine, has gradually incorporated many ML components and skills, including QA, task completion and recommendation, using a unified modeling framework based on empathic computing and RL, aiming to maximize user engagement in the long run, measured by expected conversation-turn per session. We plan to present the design and development of XiaoIce in a future publication. McCann et al. (2018) presented a platform effort of developing a unified model to handle various tasks including QA, dialogue and chitchat.

- **Towards fully end-to-end dialogue systems:** Recent work combines the benefit of task-oriented dialogue with more end-to-end capabilities. The grounded models discussed in Sec. 5.3 represent a step towards more goal-oriented conversations, as the ability to interact with the user's environment is a key requirement for most goal-oriented dialogue systems. Grounded conversation modeling discussed in this paper is still preliminary, and future challenges include enabling API calls in fully data-driven pipelines.
- **Dealing with heterogeneous data:** Conversational data is often heterogeneous. For example, chitchat data is plentiful but not directly relevant to goal-oriented systems, and goal-oriented conversational datasets are typically very small. Future research will need to address the challenge of capitalizing on both, for example in a multi-task setup similar to Luan et al. (2017). Another research direction is the work of Zhao et al. (2017), which brought synergies between chitchat and task-oriented data using a "data augmentation" technique. Their resulting system is not only able to handle chitchat, but also more robust to goal-oriented dialogues. Another challenge is to better exploit non-conversational data (e.g., Wikipedia) as part of the training of conversational systems (Ghazvininejad et al., 2018).
- **Incorporating EQ (or empathy) into dialogue:** This is useful for both chatbots and QA bots. For example, XiaoIce incorporates an EQ module so as to deliver a more understandable response or recommendation (as in 3.1 of (Shum et al., 2018)). Fung et al. (2016) embedded an empathy module into a dialogue agent to recognize users' emotion using multimodality, and generate emotion-aware responses.
- **Scalable training for task-oriented dialogues:** It is important to fast update a dialogue agent to handle a changing environment. For example, Lipton et al. (2018) proposed an efficient exploration method to tackle a domain extension setting, where new slots can be gradually introduced. Chen et al. (2016b) proposed a zero-shot learning for unseen intents so that a dialogue agent trained on one domain can detect unseen intents in a new domain without manually labeled data and without retraining.
- **Commonsense knowledge** is crucial for any dialogue agents. This is challenging because common sense knowledge is often not explicitly stored in existing knowledge base. Some new datasets are developed to foster the research on common sense reasoning, such as Reading Comprehension with Commonsense Reasoning Dataset (ReCoRD) (Zhang et al., 2018d), Winograd Schema Challenge (WSC) (Morgenstern and Ortiz, 2015) and Choice Of Plausible Alternatives (COPA) (Roemmele et al., 2011).
- **Model interpretability:** In some cases, a dialogue agent is required not only to give a recommendation or an answer, but also provide explanations. This is very important in e.g., business scenarios, where a user cannot make a business decision without justification. Shen et al. (2018); Xiong et al. (2017); Das et al. (2017b) combine the interpretability of symbolic approaches and the robustness of neural approaches and develop an inference algorithm on KB that not only improves the accuracy in answering questions but also provides explanations why the answer is generated, i.e., the paths in the KB that leads to the answer node.

Bibliography

Agichtein, E., Carmel, D., Pelleg, D., Pinter, Y., and Harman, D. (2015). Overview of the TREC 2015 LiveQA track. In *TREC*.

Ai, H. and Litman, D. J. (2008). Assessing dialog system user simulation evaluation measures using human judges. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 622–629.

Ai, H., Raux, A., Bohus, D., Eskenazi, M., and Litman, D. (2007). Comparing spoken dialog corpora collected with recruited subjects versus real users. In *Proceedings of the 8th SIGdial Workshop on Discourse and Dialogue*, pages 124–131.

Al-Rfou, R., Pickett, M., Snaider, J., Sung, Y., Strope, B., and Kurzweil, R. (2016). Conversational contextual cues: The case of personalization and history for response ranking. *CoRR*, abs/1606.00372.

Albrecht, J. and Hwa, R. (2007). A re-examination of machine learning approaches for sentence-level mt evaluation. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 880–887, Prague, Czech Republic.

Allen, J. F., Byron, D. K., Dzikovska, M. O., Ferguson, G., Galescu, L., and Stent, A. (2001). Toward conversational human-computer interaction. *AI Magazine*, 22(4):27–38.

Angeli, G., Liang, P., and Klein, D. (2010). A simple domain-independent probabilistic approach to generation. *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 502–512.

Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., and Ives, Z. (2007). DBpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer.

Bahdanau, D., Cho, K., and Bengio, Y. (2015). Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Banerjee, S. and Lavie, A. (2005). METEOR: An automatic metric for mt evaluation with improved correlation with human judgments. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72.

Bao, J., Duan, N., Zhou, M., and Zhao, T. (2014). Knowledge-based question answering as machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 967–976.

Bapna, A., Tür, G., Hakkani-Tür, D., and Heck, L. P. (2017). Towards zero-shot frame semantic parsing for domain scaling. In *Proceedings of the 18th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2476–2480.

Barlier, M., Pérolat, J., Laroche, R., and Pietquin, O. (2015). Human-machine dialogue as a stochastic game. In *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 2–11.

Baxter, J. and Bartlett, P. (2001). Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350.

Baxter, J., Bartlett, P., and Weaver, L. (2001). Experiments with infinite-horizon, policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:351–381.

Bell, J. (1999). Pragmatic reasoning: Inferring contexts. In *International and Interdisciplinary Conference on Modeling and Using Context*, pages 42–53. Springer.

Bellemare, M. G., Srinivasan, S., Ostrovski, G., Schaul, T., Saxton, D., and Munos, R. (2016). Unifying count-based exploration and intrinsic motivation. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1471–1479.

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

Bertsekas, D. P. and Tsitsiklis, J. N. (1996). *Neuro-Dynamic Programming*. Athena Scientific.

Bhatia, P., Gavaldà, M., and Einolghozati, A. (2017). soc2seq: Social embedding meets conversation model. *CoRR*, abs/1702.05512.

Bird, S., Klein, E., and Loper, E. (2009). *Natural Language Processing with Python*. O'Reilly.

Black, A. W., Burger, S., Conkie, A., Hastie, H. W., Keizer, S., Lemon, O., Merigaud, N., Parent, G., Schubiner, G., Thomson, B., Williams, J. D., Yu, K., Young, S. J., and Eskénazi, M. (2011). Spoken dialog challenge 2010: Comparison of live and control test results. In *Proceedings of the 12th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 2–7.

Bohus, D. and Horvitz, E. (2009). Models for multiparty engagement in open-world dialog. In *Proceedings of the 10th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 225–234.

Bohus, D. and Horvitz, E. (2011). Multiparty turn taking in situated dialog: Study, lessons, and directions. In *Proceedings of the 12nnual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 98–109.

Bohus, D. and Rudnicky, A. I. (2009). The RavenClaw dialog management framework: Architecture and systems. *Computer Speech & Language*, 23(3):332–361.

Bohus, D., Saw, C. W., and Horvitz, E. (2014). Directions robot: In-the-wild experiences and lessons learned. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 637–644.

Bollacker, K., Evans, C., Paritosh, P., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Bordes, A., Boureau, Y.-L., and Weston, J. (2017). Learning end-to-end goal-oriented dialog. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., and Yakhnenko, O. (2013). Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Bos, J., Klein, E., Lemon, O., and Oka, T. (2003). DIPPER: Description and formalisation of an information-state update dialogue system architecture. In *Proceedings of the 4th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 115–124.

Braunschweiler, N. and Papangelis, A. (2018). Comparison of an end-to-end trainable dialogue system with a modular statistical dialogue system. In *Proceedings of the 19th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 576–580.

Budzianowski, P., Ultes, S., Su, P.-H., Mrkšić, N., Wen, T.-H., nigo Casanueva, I., Rojas-Barahona, L. M., and Gašić, M. (2017). Sub-domain modelling for dialogue management with hierarchical reinforcement learning. In *Proceedings of the 18h Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 86–92.

Callison-Burch, C., Koehn, P., Monz, C., and Schroeder, J. (2009). Findings of the 2009 Workshop on Statistical Machine Translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation*, pages 1–28, Athens, Greece.

Caruana, R. (1998). Multitask learning. In *Learning to learn*, pages 95–133. Springer.

Casanueva, I., Budzianowski, P., Su, P.-H., Ultes, S., Rojas-Barahona, L. M., Tseng, B.-H., and Gašić, M. (2018). Feudal reinforcement learning for dialogue management in large domains. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 714–719.

Cer, D., Manning, C. D., and Jurafsky, D. (2010). The best lexical metric for phrase-based statistical MT system optimization. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 555–563, Stroudsburg, PA, USA.

Chandramohan, S., Geist, M., Lefèvre, F., and Pietquin, O. (2011). User simulation in dialogue systems using inverse reinforcement learning. In *Proceedings of the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1025–1028.

Chapelle, O. and Li, L. (2012). An empirical evaluation of Thompson sampling. In *Advances in Neural Information Processing Systems 24 (NIPS)*, pages 2249–2257.

Chen, D., Bolton, J., and Manning, C. D. (2016a). A thorough examination of the CNN/Daily mail reading comprehension task. *arXiv preprint arXiv:1606.02858*.

Chen, D., Fisch, A., Weston, J., and Bordes, A. (2017a). Reading Wikipedia to answer open-domain questions. *arXiv 1704.00051*.

Chen, H., Liu, X., Yin, D., and Tang, J. (2017b). A survey on dialogue systems: Recent advances and new frontiers. *arXiv preprint arXiv:1711.01731*.

Chen, J., Wang, C., Xiao, L., He, J., Li, L., and Deng, L. (2017c). Q-LDA: Uncovering latent patterns in text-based sequential decision processes. In *Advances in Neural Information Processing Systems 30*, pages 4984–4993.

Chen, L., Tan, B., Long, S., and Yu, K. (2018). Structured dialogue policy with graph neural networks. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING)*, pages 1257–1268.

Chen, L., Zhou, X., Chang, C., Yang, R., and Yu, K. (2017d). Agent-aware dropout DQN for safe and efficient on-line dialogue policy learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2454–2464.

Chen, Y.-N., Celikyilmaz, A., and Hakkani-Tür, D. (2017e). Deep learning for dialogue systems. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Tutorial Abstracts)*, pages 8–14.

Chen, Y.-N. and Gao, J. (2017). Open-domain neural dialogue systems. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Tutorial Abstracts)*, pages 6–10.

Chen, Y.-N., Hakkani-Tür, D., Tur, G., Gao, J., and Deng, L. (2016b). End-to-end memory networks with knowledge carryover for multi-turn spoken language understanding. In *Proceedings of The 17th Annual Meeting of the International Speech Communication Association*, pages 3245–3249.

Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014a). On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Doha, Qatar.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014b). Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar.

Choi, E., He, H., Iyyer, M., Yatskar, M., Yih, W.-t., Choi, Y., Liang, P., and Zettlemoyer, L. (2018). QuAC: Question answering in context. *arXiv preprint arXiv:1808.07036*.

Clark, P., Cowhey, I., Etzioni, O., Khot, T., Sabharwal, A., Schoenick, C., and Tafjord, O. (2018). Think you have solved question answering? try ARC, the AI2 reasoning challenge. *arXiv preprint arXiv:1803.05457*.

Colby, K. M. (1975). *Artificial Paranoia: A Computer Simulation of Paranoid Processes*. Elsevier Science Inc., New York, NY, USA.

Cole, R. A. (1999). Tools for research and education in speech science. In *Proceedings of International Conference of Phonetic Sciences*, pages 1277–1280.

Core, M. G. and Allen, J. F. (1997). Coding dialogs with the DAMSL annotation scheme. In *Proceedings of AAAI Fall Symposium on Communicative Action in Humans and Machines*, pages 28–35.

Corston-Oliver, S., Gamon, M., and Brockett, C. (2001). A machine learning approach to the automatic evaluation of machine translation. In *Proceedings of 39th Annual Meeting of the Association for Computational Linguistics*, pages 148–155, Toulouse, France.

Côté, M.-A., Ákos Kádár, Yuan, X., Kybartas, B., Barnes, T., Fine, E., Moore, J., Hausknecht, M., El Asri, L., Adada, M., Tay, W., and Trischler, A. (2018). TextWorld: A learning environment for text-based games. arXiv:1806.11532.

Crook, P. A., Marin, A., Agarwal, V., Aggarwal, K., Anastasakos, T., Bikkula, R., Boies, D., Celikyilmaz, A., Chandramohan, S., Feizollahi, Z., Holenstein, R., Jeong, M., Khan, O. Z., Kim, Y.-B., Krawczyk, E., Liu, X., Panic, D., Radostev, V., Ramesh, N., Robichaud, J.-P., Rochette, A., Stromberg, L., and Sarikaya, R. (2016). Task Completion Platform: A self-serve multi-domain goal oriented dialogue platform. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL): Demonstrations Session*, pages 47–51.

Cuayáhuitl, H., Renals, S., Lemon, O., and Shimodaira, H. (2010). Evaluation of a hierarchical reinforcement learning spoken dialogue system. *Computer Speech and Language*, 24(2):395–429.

Cuayáhuitl, H., Yu, S., Williamson, A., and Carse, J. (2016). Deep reinforcement learning for multi-domain dialogue systems. *arXiv preprint arXiv:1611.08675*.

Dai, B., Shaw, A., He, N., Li, L., and Song, L. (2018a). Boosting the actor with dual critic. In *Proceedings of the Sixth International Conference on Learning Representations (ICLR)*.

Dai, B., Shaw, A., Li, L., Xiao, L., He, N., Liu, Z., Chen, J., and Song, L. (2018b). SBEED: Convergent reinforcement learning with nonlinear function approximation. In *Proceedings of the Thirty-Fifth International Conference on Machine Learning (ICML-18)*, pages 1133–1142.

Dang, H. T., Kelly, D., and Lin, J. J. (2007). Overview of the TREC 2007 question answering track. In *TREC*, volume 7, page 63.

Dann, C., Lattimore, T., and Brunskill, E. (2017). Unifying PAC and regret: Uniform PAC bounds for episodic reinforcement learning. In *Advances in Neural Information Processing Systems 30 (NIPS)*, pages 5717–5727.

Das, A., Kottur, S., Gupta, K., Singh, A., Yadav, D., Moura, J. M., Parikh, D., and Batra, D. (2017a). Visual Dialog. In *CVPR*.

Das, R., Dhuliawala, S., Zaheer, M., Vilnis, L., Durugkar, I., Krishnamurthy, A., Smola, A., and McCallum, A. (2017b). Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. *arXiv preprint arXiv:1711.05851*.

Daubigney, L., Gašić, M., Chandramohan, S., Geist, M., Pietquin, O., and Young, S. J. (2011). Uncertainty management for on-line optimisation of a POMDP-based large-scale spoken dialogue system. In *Proceedings of the 12th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1301–1304.

Daubigney, L., Geist, M., Chandramohan, S., and Pietquin, O. (2012). A comprehensive reinforcement learning framework for dialogue management optimization. *IEEE Journal of Selected Topics in Signal Processing*, 6(8):891–902.

Dayan, P. and Hinton, G. E. (1993). Feudal reinforcement learning. In *Advances in Neural Information Processing Systems 5 (NIPS)*, pages 271–278.

de Vries, H., Strub, F., Chandar, S., Pietquin, O., Larochelle, H., and Courville, A. C. (2017). GuessWhat?! visual object discovery through multi-modal dialogue. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4466–4475.

DeVault, D., Artstein, R., Benn, G., Dey, T., Fast, E., Gainer, A., Georgila, K., Gratch, J., Hartholt, A., Lhommet, M., Lucas, G. M., Marsella, S., Morbini, F., Nazarian, A., Scherer, S., Stratou, G., Suri, A., Traum, D. R., Wood, R., Xu, Y., Rizzo, A. A., and Morency, L.-P. (2014). SimSensei kiosk: A virtual human interviewer for healthcare decision support. In *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 1061–1068.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2018). BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dhingra, B., Li, L., Li, X., Gao, J., Chen, Y.-N., Ahmed, F., and Deng, L. (2017). Towards end-to-end reinforcement learning of dialogue agents for information access. In *ACL (1)*, pages 484–495.

Dhingra, B., Liu, H., Yang, Z., Cohen, W. W., and Salakhutdinov, R. (2016). Gated-attention readers for text comprehension. *arXiv preprint arXiv:1606.01549*.

Dietterich, T. G. (2000). Hierarchical reinforcement learning with the MAXQ value function decomposition. *Journal of Artificial Intelligence Research*, 13:227–303.

Ding, Y., Liu, Y., Luan, H., and Sun, M. (2017). Visualizing and understanding neural machine translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1150–1159, Vancouver, Canada.

Dodge, J., Gane, A., Zhang, X., Bordes, A., Chopra, S., Miller, A., Szlam, A., and Weston, J. (2016). Evaluating prerequisite qualities for learning end-to-end dialog systems. In *ICLR*.

Dunn, M., Sagun, L., Higgins, M., Guney, U., Cirik, V., and Cho, K. (2017). SearchQA: A new Q&A dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Eckert, W., Levin, E., and Pieraccini, R. (1997). User modeling for spoken dialogue system evaluation. In *Proceedings of the 1997 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 80–87.

El Asri, L., He, J., and Suleman, K. (2016). A sequence-to-sequence model for user simulation in spoken dialogue systems. In *Proceedings of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1151–1155.

El Asri, L., Laroche, R., and Pietquin, O. (2012). Reward function learning for dialogue management. In *Proceedings of the Sixth Starting AI Researchers' Symposium (STAIRS)*, pages 95–106.

Engel, Y., Mannor, S., and Meir, R. (2005). Reinforcement learning with Gaussian processes. In *Proceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 201–208.

Eric, M., Krishnan, L., Charette, F., and Manning, C. D. (2017). Key-value retrieval networks for task-oriented dialogue. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 37–49.

Ernst, D., Geurts, P., and Wehenkel, L. (2005). Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6:503–556.

Fang, H., Gupta, S., Iandola, F., Srivastava, R. K., Deng, L., Dollár, P., Gao, J., He, X., Mitchell, M., Platt, J. C., Zitnick, L., and Zweig, G. (2015). From captions to visual concepts and back. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1473–1482.

Fatemi, M., El Asri, L., Schulz, H., He, J., and Suleman, K. (2016). Policy networks with two-stage training for dialogue systems. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 101–110.

Fedorenko, D. G., Smetanin, N., and Rodichev, A. (2017). Avoiding echo-responses in a retrieval-based conversation system. *CoRR*, abs/1712.05626.

Fung, P., Bertero, D., Wan, Y., Dey, A., Chan, R. H. Y., Siddique, F. B., Yang, Y., Wu, C., and Lin, R. (2016). Towards empathetic human-robot interactions. *CoRR*, abs/1605.04072.

Galley, M., Brockett, C., Sordoni, A., Ji, Y., Auli, M., Quirk, C., Mitchell, M., Gao, J., and Dolan, B. (2015). deltaBLEU: A discriminative metric for generation tasks with intrinsically diverse targets. In *ACL-IJCNLP*, page 445–450.

Gao, J. (2017). An introduction to deep learning for natural language processing. In *International Summer School on Deep Learning, Bilbao*.

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

Gao, J., Galley, M., and Li, L. (2018b). Neural approaches to conversational AI. *Proc. of ACL 2018, Tutorial Abstracts*, pages 2–7.

Gao, J., He, X., Yih, W.-t., and Deng, L. (2014a). Learning continuous phrase representations for translation modeling. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 699–709.

Gao, J., Pantel, P., Gamon, M., He, X., and Deng, L. (2014b). Modeling interestingness with deep neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2–13.

Gardner, M., Talukdar, P., Krishnamurthy, J., and Mitchell, T. (2014). Incorporating vector space similarity in random walk inference over knowledge bases. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 397–406.

Gašić, M., Breslin, C., Henderson, M., Kim, D., Szummer, M., Thomson, B., Tsiakoulis, P., and Young, S. J. (2013). On-line policy optimisation of Bayesian spoken dialogue systems via human interaction. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8367–8371.

Gašic, M., Kim, D., Tsiakoulis, P., Breslin, C., Henderson, M., Szummer, M., Thomson, B., and Young, S. (2014). Incremental on-line adaptation of POMDP-based dialogue managers to extended domains. In *Proceedings of the 15th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 140–144.

Gašić, M., Mrkšić, N., hao Su, P., Vandyke, D., Wen, T.-H., and Young, S. J. (2015). Policy committee for adaptation in multi-domain spoken dialogue systems. In *Proceedings of the 2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 806–812.

Gašić, M. and Young, S. J. (2014). Gaussian processes for POMDP-based dialogue manager optimization. *IEEE Trans. Audio, Speech & Language Processing*, 22(1):28–40.

Georgila, K., Henderson, J., and Lemon, O. (2006). User simulation for spoken dialogue systems: Learning and evaluation. In *Proceedings of the 9th International Conference on Spoken Language Processing (INTERSPEECH)*, pages 1065–1068.

Ghazvininejad, M., Brockett, C., Chang, M.-W., Dolan, B., Gao, J., Yih, W.-t., and Galley, M. (2018). A knowledge-grounded neural conversation model. In *AAAI*, pages 5110–5117.

Giménez, J. and Màrquez, L. (2008). A smorgasbord of features for automatic MT evaluation. In *Proceedings of the Third Workshop on Statistical Machine Translation*, pages 195–198.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In *NIPS*, pages 2672–2680.

Graham, Y. and Baldwin, T. (2014). Testing for significance of increased correlation with human judgment. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 172–176, Doha, Qatar.

Graham, Y., Baldwin, T., and Mathur, N. (2015). Accurate evaluation of segment-level machine translation metrics. In *NAACL-HLT*.

Graves, A. and Schmidhuber, J. (2005). Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks*, 18:602–610.

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

Gu, S., Lillicrap, T., Ghahramani, Z., Turner, R. E., and Levine, S. (2017). Q-Prop: Sample-efficient policy gradient with an off-policy critic. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*.

Guu, K., Miller, J., and Liang, P. (2015). Traversing knowledge graphs in vector space. *arXiv preprint arXiv:1506.01094*.

Hakkani-Tür, D., Tür, G., Celikyilmaz, A., Chen, Y.-N., Gao, J., Deng, L., and Wang, Y.-Y. (2016). Multi-domain joint semantic frame parsing using Bi-directional RNN-LSTM. In *Proceedings of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 715–719.

Hakkani-Tür, D., Tur, G., Heck, L., Fidler, A., and Celikyilmaz, A. (2012). A discriminative classification-based approach to information state updates for a multi-domain dialog system. In *Proceedings of the 13th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 330–333.

Hartikainen, M., Salonen, E.-P., and Turunen, M. (2004). Subjective evaluation of spoken dialogue systems using SERVQUAL method. In *Proceedings of the 8th International Conference on Spoken Language Processing (INTERSPEECH)*, pages 2273–2276.

Hausknecht, M. and Stone, P. (2015). Deep recurrent Q-learning for partially observable MDPs. In *Proceedings of the AAAI Fall Symposium on Sequential Decision Making for Intelligent Agents*, pages 29–37.

He, J., Chen, J., He, X., Gao, J., Li, L., Deng, L., and Ostendorf, M. (2016). Deep reinforcement learning with a natural language action space. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, page 1621–1630.

He, S., Liu, C., Liu, K., and Zhao, J. (2017a). Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. In *ACL*, volume 1, pages 199–208.

He, W., Liu, K., Lyu, Y., Zhao, S., Xiao, X., Liu, Y., Wang, Y., Wu, H., She, Q., Liu, X., Wu, T., and Wang, H. (2017b). DuReader: a chinese machine reading comprehension dataset from real-world applications. *arXiv preprint arXiv:1711.05073*.

Henderson, J., Lemon, O., and Georgila, K. (2008). Hybrid reinforcement/supervised learning of dialogue policies from fixed data sets. *Computational Linguistics*, 34(4):487–511.

Henderson, M. (2015). Machine learning for dialog state tracking: A review. In *Proceedings of The First International Workshop on Machine Learning in Spoken Language Processing*.

Henderson, M., Thomson, B., and Williams, J. D. (2014a). The 3rd dialog state tracking challenge. In *Proceedings of the 2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 324–329.

Henderson, M., Thomson, B., and Williams, J. D. (2014b). The second dialog state tracking challenge. In *Proceedings of the 15th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 263–272.

Henderson, M., Thomson, B., and Young, S. J. (2013). Deep neural network approach for the dialog state tracking challenge. In *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 467–471.

Hermann, K. M., Kocisky, T., Grefenstette, E., Espeholt, L., Kay, W., Suleyman, M., and Blunsom, P. (2015). Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701.

Hewlett, D., Lacoste, A., Jones, L., Polosukhin, I., Fandrianto, A., Han, J., Kelcey, M., and Berthelot, D. (2016). WikiReading: A novel large-scale language understanding task over wikipedia. *arXiv preprint arXiv:1608.03542*.

Hill, F., Bordes, A., Chopra, S., and Weston, J. (2015). The Goldilocks principle: Reading children's books with explicit memory representations. *arXiv preprint arXiv:1511.02301*.

Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., and Kingsbury, B. (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.

Hinton, G. E. and Van Camp, D. (1993). Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the sixth annual conference on Computational learning theory*, pages 5–13. ACM.

Hochreiter, S. (1991). Untersuchungen zu dynamischen neuronalen Netzen. Diploma thesis, Institut für Informatik, Lehrstuhl Prof. Brauer, Technische Universität München.

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

Hofmann, K., Li, L., and Radlinski, F. (2016). Online evaluation for information retrieval. *Foundations and Trends in Information Retrieval*, 10(1):1–117.

Holtzman, A., Buys, J., Forbes, M., Bosselut, A., Golub, D., and Choi, Y. (2018). Learning to write with cooperative discriminators. In *ACL*, pages 1638–1649, Melbourne, Australia.

Hori, C. and Hori, T. (2017). End-to-end conversation modeling track in DSTC6. *CoRR*, abs/1706.07440.

Hori, C., Hori, T., Watanabe, S., and Hershey, J. R. (2015). Context sensitive spoken language understanding using role dependent LSTM layers. Technical Report TR2015-134, Mitsubishi Electric Research Laboratories.

Hori, C., Perez, J., Yoshino, K., and Kim, S. (2017). The sixth dialog state tracking challenge. `http://workshop.colips.org/dstc6`.

Horvitz, E. (1999). Principles of mixed-initiative user interfaces. In *Proceeding of the Conference on Human Factors in Computing Systems (CHI)*, pages 159–166.

Houthooft, R., Chen, X., Duan, Y., Schulman, J., Turck, F. D., and Abbeel, P. (2016). VIME: Variational information maximizing exploration. In *Advances in Neural Information Processing Systems 29 (NIPS)*, pages 1109–1117.

Hu, M., Peng, Y., and Qiu, X. (2017). Mnemonic reader for machine comprehension. *arXiv preprint arXiv:1705.02798*.

Huang, H.-Y., Zhu, C., Shen, Y., and Chen, W. (2017). FusionNet: Fusing via fully-aware attention with application to machine comprehension. *arXiv preprint arXiv:1711.07341*.

Huang, P.-S., He, X., Gao, J., Deng, L., Acero, A., and Heck, L. (2013). Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338.

Huang, Q., Gan, Z., Celikyilmaz, A., Wu, D. O., Wang, J., and He, X. (2018). Hierarchically structured reinforcement learning for topically coherent visual story generation. *CoRR*, abs/1805.08191.

Huang, X., Acero, A., and Hon, H.-W. (2001). *Spoken language processing: A guide to theory, algorithm, and system development*. Prentice Hall.

Huber, B., McDuff, D., Brockett, C., Galley, M., and Dolan, B. (2018). Emotional dialogue generation using image-grounded language models. In *CHI*, pages 277:1–277:12.

Inaba, M. and Takahashi, K. (2016). Neural utterance ranking model for conversational dialogue systems. In *SIGDIAL*, pages 393–403.

Iyyer, M., Yih, W.-t., and Chang, M.-W. (2017). Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1821–1831.

Jafarpour, S., Burges, C. J., and Ritter, A. (2010). Filter, rank, and transfer the knowledge: Learning to chat. *Advances in Ranking*, 10:2329–9290.

Jaksch, T., Ortner, R., and Auer, P. (2010). Near-optimal regret bounds for reinforcement learning. *Journal of Machine Learning Research*, 11:1563–1600.

Jia, R. and Liang, P. (2017). Adversarial examples for evaluating reading comprehension systems. *arXiv preprint arXiv:1707.07328*.

Jiang, N., Krishnamurthy, A., Agarwal, A., Langford, J., and Schapire, R. E. (2017). Contextual decision processes with low Bellman rank are PAC-learnable. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 1704–1713.

Jiang, N. and Li, L. (2016). Doubly robust off-policy evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 652–661.

Joshi, M., Choi, E., Weld, D. S., and Zettlemoyer, L. (2017). TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551*.

Jung, S., Lee, C., Kim, K., Jeong, M., and Lee, G. G. (2009). Data-driven user simulation for automated evaluation of spoken dialog systems. *Computer Speech and Language*, 23:479–509.

Jurafsky, D. and Martin, J. H. (2009). *Speech & language processing*. Prentice Hall.

Jurafsky, D. and Martin, J. H. (2018). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice-Hall. Draft of August 12th, 2018. Website: `https://web.stanford.edu/~jurafsky/slp3`.

Kaelbling, L. P., Littman, M. L., and Moore, A. P. (1996). Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285.

Kakade, S. (2001). A natural policy gradient. In *Advances in Neural Information Processing Systems 13 (NIPS)*, pages 1531–1538.

Kalchbrenner, N. and Blunsom, P. (2013). Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA.

Kannan, A. and Vinyals, O. (2016). Adversarial evaluation of dialogue models. In *NIPS Workshop on Adversarial Training*.

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

Kim, S., D'Haro, L. F., Banchs, R. E., Williams, J. D., and Henderson, M. (2016a). The fourth dialog state tracking challenge. In *Proceedings of the 7th International Workshop on Spoken Dialogue Systems (IWSDS)*, pages 435–449.

Kim, S., D'Haro, L. F., Banchs, R. E., Williams, J. D., Henderson, M., and Yoshino, K. (2016b). The fifth dialog state tracking challenge. In *Proceedings of the 2016 IEEE Spoken Language Technology Workshop (SLT-16)*, pages 511–517.

Kočisky, T., Schwarz, J., Blunsom, P., Dyer, C., Hermann, K. M., Melis, G., and Grefenstette, E. (2017). The NarrativeQA reading comprehension challenge. *arXiv preprint arXiv:1712.07040*.

Koehn, P. (2004). Statistical significance tests for machine translation evaluation. In *Proceedings of EMNLP 2004*, pages 388–395, Barcelona, Spain.

Koehn, P., Och, F. J., and Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, NAACL '03, pages 48–54.

Koller, A. and Stone, M. (2007). Sentence generation as a planning problem. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL-07)*, pages 336–343.

Komatani, K., Kanda, N., Nakano, M., Nakadai, K., Tsujino, H., Ogata, T., and Okuno, H. G. (2006). Multi-domain spoken dialogue system with extensibility and robustness against speech recognition errors. In *Proceedings of the SIGDIAL 2006 Workshop*, pages 9–17.

Konda, V. R. and Tsitsiklis, J. N. (1999). Actor-critic algorithms. In *Advances in Neural Information Processing Systems 12 (NIPS)*, pages 1008–1014.

Kondadadi, R., Howald, B., and Schilder, F. (2013). A statistical NLG framework for aggregated planning and realization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1406–1415.

Kotti, M., Diakoloukas, V., Papangelis, A., Lagoudakis, M., and Stylianou, Y. (2018). A case study on the importance of belief state representation for dialogue policy management. In *Proceedings of the 19th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 986–990.

Kulesza, A. and Shieber, S. M. (2004). A learning approach to improving sentence-level MT evaluation. In *Proceedings of the 10th International Conference on Theoretical and Methodological Issues in Machine Translation*, Baltimore, MD.

Kumar, A., Irsoy, O., Ondruska, P., Iyyer, M., Bradbury, J., Gulrajani, I., Zhong, V., Paulus, R., and Socher, R. (2016). Ask me anything: Dynamic memory networks for natural language processing. In *International Conference on Machine Learning*, pages 1378–1387.

Lai, G., Xie, Q., Liu, H., Yang, Y., and Hovy, E. (2017). RACE: Large-scale reading comprehension dataset from examinations. *arXiv preprint arXiv:1704.04683*.

Langkilde, I. and Knight, K. (1998). Generation that exploits corpus-based statistical knowledge. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics (COLING-ACL)*, pages 704–710.

Lao, N. and Cohen, W. W. (2010). Relational retrieval using a combination of path-constrained random walks. *Machine learning*, 81(1):53–67.

Lao, N., Mitchell, T., and Cohen, W. W. (2011). Random walk inference and learning in a large scale knowledge base. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 529–539.

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

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

Lee, S. and Jha, R. (2019). Zero-shot adaptive transfer for conversational language understanding. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19), to appear*.

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

Lemon, O. (2011). Learning what to say and how to say it: Joint optimisation of spoken dialogue management and natural language generation. *Computer Speech & Language*, 25(2):210–221.

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

Lewis, M., Yarats, D., Dauphin, Y., Parikh, D., and Batra, D. (2017). Deal or no deal? End-to-end learning of negotiation dialogues. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing (EMNLP-17)*, pages 2443–2453.

Li, J., Galley, M., Brockett, C., Gao, J., and Dolan, B. (2016a). A diversity-promoting objective function for neural conversation models. In *NAACL-HLT*, pages 110–119.

Li, J., Galley, M., Brockett, C., Gao, J., and Dolan, B. (2016b). A persona-based neural conversation model. In *ACL*, page 994–1003.

Li, J., Miller, A. H., Chopra, S., Ranzato, M., and Weston, J. (2017a). Dialogue learning with human-in-the-loop. In *ICLR*.

Li, J., Miller, A. H., Chopra, S., Ranzato, M., and Weston, J. (2017b). Learning through dialogue interactions by asking questions. In *ICLR*.

Li, J., Monroe, W., Ritter, A., Jurafsky, D., Galley, M., and Gao, J. (2016c). Deep reinforcement learning for dialogue generation. In *EMNLP*, pages 1192–1202.

Li, J., Monroe, W., Shi, T., Jean, S., Ritter, A., and Jurafsky, D. (2017c). Adversarial learning for neural dialogue generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2157–2169, Copenhagen, Denmark.

Li, L., He, H., and Williams, J. D. (2014). Temporal supervised learning for inferring a dialog policy from example conversations. In *Proceedings of the 2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 312–317.

Li, L., Williams, J. D., and Balakrishnan, S. (2009). Reinforcement learning for spoken dialog management using least-squares policy iteration and fast feature selection. In *Proceedings of the 10th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2475–2478.

Li, X., Chen, Y.-N., Li, L., Gao, J., and Celikyilmaz, A. (2017d). End-to-end task-completion neural dialogue systems. In *Proceedings of the 8th International Joint Conference on Natural Language Processing (IJCNLP)*, pages 733–743.

Li, X., Chen, Y.-N., Li, L., Gao, J., and Celikyilmaz, A. (2017e). Investigation of language understanding impact for reinforcement learning based dialogue systems. CoRR abs/1703.07055.

Li, X., Li, L., Gao, J., He, X., Chen, J., Deng, L., and He, J. (2015). Recurrent reinforcement learning: A hybrid approach. arXiv:1509.03044.

Li, X., Lipton, Z. C., Dhingra, B., Li, L., Gao, J., and Chen, Y.-N. (2016d). A user simulator for task-completion dialogues. CoRR abs/1612.05688.

Li, X., Panda, S., Liu, J., and Gao, J. (2018). Microsoft dialogue challenge: Building end-to-end task-completion dialogue systems. *arXiv preprint arXiv:1807.11125*.

Li, Y. (2018). Deep reinforcement learning. arXiv 1810.06339.

Liang, C., Berant, J., Le, Q., Forbus, K. D., and Lao, N. (2016). Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. *arXiv preprint arXiv:1611.00020*.

Lin, C.-Y. (2004). ROUGE: A package for automatic evaluation of summaries. In *ACL workshop*, pages 74–81.

Lin, L.-J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. *Machine Learning*, 8(3–4):293–321.

Lipton, Z. C., Gao, J., Li, L., Li, X., Ahmed, F., and Deng, L. (2018). BBQ-networks: Efficient exploration in deep reinforcement learning for task-oriented dialogue systems. In *AAAI*, pages 5237–5244.

Lita, L. V., Rogati, M., and Lavie, A. (2005). BLANC: Learning evaluation metrics for MT. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 740–747.

Litman, D. J. and Allen, J. F. (1987). A plan recognition model for subdialogues in conversations. *Cognitive Science*, 11(163–200).

Liu, B. and Lane, I. (2016). Attention-based recurrent neural network models for joint intent detection and slot filling. In *Proceedings of the 17th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 685–689.

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

Liu, B. and Lane, I. (2018). Adversarial learning of task-oriented neural dialog models. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 350–359.

Liu, B., Tür, G., Hakkani-Tür, D., Shah, P., and Heck, L. P. (2018a). Dialogue learning with human teaching and feedback in end-to-end trainable task-oriented dialogue systems. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 2060–2069.

Liu, C.-W., Lowe, R., Serban, I., Noseworthy, M., Charlin, L., and Pineau, J. (2016). How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. In *EMNLP*, pages 2122–2132.

Liu, H., Feng, Y., Mao, Y., Zhou, D., Peng, J., and Liu, Q. (2018b). Action-depedent control variates for policy optimization via Stein's identity. In *Proceedings of the 6th International Conference on Learning Representations (ICLR)*.

Liu, Q., Li, L., Tang, Z., and Zhou, D. (2018c). Breaking the curse of horizon: Infinite-horizon off-policy estimation. In *Advances in Neural Information Processing Systems 31 (NIPS-18)*, pages 5361–5371.

Liu, R., Wei, W., Mao, W., and Chikina, M. (2017). Phase conductor on multi-layered attentions for machine comprehension. *arXiv preprint arXiv:1710.10504*.

Liu, X., Shen, Y., Duh, K., and Gao, J. (2018d). Stochastic answer networks for machine reading comprehension. In *ACL*, pages 1694–1704.

Lowe, R., Noseworthy, M., Serban, I. V., Angelard-Gontier, N., Bengio, Y., and Pineau, J. (2017). Towards an automatic turing test: Learning to evaluate dialogue responses. In *ACL*, page 1116–1126.

Lowe, R., Pow, N., Serban, I., and Pineau, J. (2015). The Ubuntu Dialogue Corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *SIGDIAL*, pages 285–294.

Lu, Z. and Li, H. (2014). A deep architecture for matching short texts. In *Advances in Neural Information Processing Systems 27*, pages 1368–1375. Curran Associates, Inc.

Luan, Y., Brockett, C., Dolan, B., Gao, J., and Galley, M. (2017). Multi-task learning for speaker-role adaptation in neural conversation models. In *IJCNLP*, pages 605–614.

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

Madotto, A., Wu, C.-S., and Fung, P. (2018). Mem2Seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 1468–1478.

Mairesse, F. and Young, S. (2014). Stochastic language generation in dialogue using factored language models. *Computational Linguistics*, 40(4):763–799.

Manning, C., Surdeanu, M., Bauer, J., Finkel, J., Bethard, S., and McClosky, D. (2014). The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Mao, J., Xu, W., Yang, Y., Wang, J., Huang, Z., and Yuille, A. (2015). Deep captioning with multimodal recurrent neural networks (m-RNN). In *ICLR*.

McCann, B., Bradbury, J., Xiong, C., and Socher, R. (2017). Learned in translation: Contextualized word vectors. In *Advances in Neural Information Processing Systems*, pages 6297–6308.

McCann, B., Keskar, N. S., Xiong, C., and Socher, R. (2018). The natural language decathlon: Multitask learning as question answering. *arXiv preprint arXiv:1806.08730*.

McTear, M. F. (2002). Spoken dialogue technology: Enabling the conversational user interface. *ACM Computing Surveys*, 34(1):90–169.

Mei, H., Bansal, M., and Walter, M. R. (2017). Coherent dialogue with attention-based language models. In *AAAI*, pages 3252–3258.

Melamud, O., Goldberger, J., and Dagan, I. (2016). context2vec: Learning generic context embedding with bidirectional lstm. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 51–61.

Mesnil, G., Dauphin, Y., Yao, K., Bengio, Y., Deng, L., Hakkani-Tür, D. Z., He, X., Heck, L. P., Tür, G., Yu, D., and Zweig, G. (2015). Using recurrent neural networks for slot filling in spoken language understanding. *IEEE/ACM Transactions on Audio, Speech & Language Processing*, 23(3):530–539.

Mesnil, G., He, X., Deng, L., and Bengio, Y. (2013). Investigation of recurrent-neural-network architectures and learning methods for spoken language understanding. In *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 3771–3775.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Minsky, M. and Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. MIT Press.

Mitchell, T. (1997). *Machine Learning*. McGraw-Hill, New York.

Mnih, V., Adrià, Badia, P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 1928–1937.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518:529–533.

Morgenstern, L. and Ortiz, C. L. (2015). The Winograd Schema Challenge: Evaluating progress in commonsense reasoning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pages 4024–4025.

Mostafazadeh, N., Brockett, C., Dolan, B., Galley, M., Gao, J., Spithourakis, G., and Vanderwende, L. (2017). Image-grounded conversations: Multimodal context for natural question and response generation. In *IJCNLP*, pages 462–472.

Mou, L., Lu, Z., Li, H., and Jin, Z. (2016). Coupling distributed and symbolic execution for natural language queries. *arXiv preprint arXiv:1612.02741*.

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

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

Munos, R. and Szepesvári, C. (2008). Finite-time bounds for sampling-based fitted value iteration. *Journal of Machine Learning Research*, 9:815–857.

Narasimhan, K., Kulkarni, T. D., and Barzilay, R. (2015). Language understanding for text-based games using deep reinforcement learning. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1–11.

Neelakantan, A., Roth, B., and McCallum, A. (2015). Compositional vector space models for knowledge base completion. *arXiv preprint arXiv:1504.06662*.

Nguyen, D. Q. (2017). An overview of embedding models of entities and relationships for knowledge base completion. *arXiv preprint arXiv:1703.08098*.

Nguyen, T., Rosenberg, M., Song, X., Gao, J., Tiwary, S., Majumder, R., and Deng, L. (2016). MS MARCO: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, pages 160–167, Sapporo, Japan.

Och, F. J. and Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

Och, F. J. and Ney, H. (2004). The alignment template approach to statistical machine translation. *Comput. Linguist.*, 30(4):417–449.

Oh, A. H. and Rudnicky, A. I. (2002). Stochastic natural language generation for spoken dialog systems. *Computer Speech & Language*, 16(3–4):387–407.

Osband, I., Blundell, C., Pritzel, A., and Roy, B. V. (2016). Deep exploration via bootstrapped DQN. In *Advances in Neural Information Processing Systems 29 (NIPS-16)*, pages 4026–4034.

Osband, I. and Roy, B. V. (2017). Why is posterior sampling better than optimism for reinforcement learning? In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, pages 2701–2710.

Pado, S., Cer, D., Galley, M., Jurafsky, D., and Manning, C. D. (2009). Measuring machine translation quality as semantic equivalence: A metric based on entailment features. *Machine Translation*, pages 181–193.

Paek, T. (2001). Empirical methods for evaluating dialog systems. In *Proceedings of the 2nd Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 1–9.

Paek, T. and Pieraccini, R. (2008). Automating spoken dialogue management design using machine learning: An industry perspective. *Speech Communication*, 50(8–9):716–729.

Papangelis, A., Kotti, M., and Stylianou, Y. (2018a). Towards scalable information-seeking multi-domain dialogue. In *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6064–6068.

Papangelis, A., Papadakos, P., Stylianou, Y., and Tzitzikas, Y. (2018b). Spoken dialogue for information navigation. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 229–234.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation. In *ACL*, pages 311–318.

Parr, R. and Russell, S. J. (1998). Reinforcement learning with hierarchies of machines. In *Advances of Neural Information Processing Systems 10 (NIPS)*, pages 1043–1049.

Pasupat, P. and Liang, P. (2015). Compositional semantic parsing on semi-structured tables. *arXiv preprint arXiv:1508.00305*.

Peng, B., Li, X., Gao, J., Liu, J., and Wong, K.-F. (2018). Integrating planning for task-completion dialogue policy learning. CoRR abs/1801.06176.

Peng, B., Li, X., Li, L., Gao, J., Celikyilmaz, A., Lee, S., and Wong, K.-F. (2017). Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning. In *EMNLP*, pages 2231–2240.

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

Peters, J., Vijayakumar, S., and Schaal, S. (2005). Natural actor-critic. In *Proceedings of the 16th European Conference on Machine Learning (ECML)*, pages 280–291.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Pietquin, O. and Dutoit, T. (2006). A probabilistic framework for dialog simulation and optimal strategy learning. *IEEE Transactions on Audio, Speech & Language Processing*, 14(2):589–599.

Pietquin, O., Geist, M., Chandramohan, S., and Frezza-Buet, H. (2011). Sample-efficient batch reinforcement learning for dialogue management optimization. *ACM Transactions on Speech and Language Processing*, 7(3):7:1–7:21.

Pietquin, O. and Hastie, H. (2013). A survey on metrics for the evaluation of user simulations. *The Knowledge Engineering Review*, 28(1):59–73.

Precup, D., Sutton, R. S., and Singh, S. P. (2000). Eligibility traces for off-policy policy evaluation. In *Proceedings of the 17th International Conference on Machine Learning (ICML)*, pages 759–766.

Przybocki, M., Peterson, K., Bronsart, S., and Sanders, G. (2009). The NIST 2008 metrics for machine translation challenge—overview, methodology, metrics, and results. *Machine Translation*, 23(2):71–103.

Puterman, M. L. (1994). *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, New York.

Rajpurkar, P., Jia, R., and Liang, P. (2018). Know what you don't know: Unanswerable questions for SQuAD. *arXiv 1806.03822*.

Rajpurkar, P., Zhang, J., Lopyrev, K., and Liang, P. (2016). SQuAD: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

Ram, A., Prasad, R., Khatri, C., Venkatesh, A., Gabriel, R., Liu, Q., Nunn, J., Hedayatnia, B., Cheng, M., Nagar, A., King, E., Bland, K., Wartick, A., Pan, Y., Song, H., Jayadevan, S., Hwang, G., and Pettigrue, A. (2018). Conversational AI: the science behind the alexa prize. *CoRR*, abs/1801.03604.

Ramshaw, L. A. and Marcus, M. (1995). Text chunking using transformation based learning. In *Third Workshop on Very Large Corpora (VLC at ACL)*, pages 82–94.

Ranzato, M., Chopra, S., Auli, M., and Zaremba, W. (2015). Sequence level training with recurrent neural networks. *arXiv 1511.06732*.

Ravuri, S. V. and Stolcke, A. (2015). Recurrent neural network and LSTM models for lexical utterance classification. In *Proceedings of the 16th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 135–139.

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

Reddy, S., Chen, D., and Manning, C. D. (2018). CoQA: A conversational question answering challenge. *arXiv preprint arXiv:1808.07042*.

Ren, G., Ni, X., Malik, M., and Ke, Q. (2018a). Conversational query understanding using sequence to sequence modeling. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web*, pages 1715–1724. International World Wide Web Conferences Steering Committee.

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

Rich, C., Sidner, C. L., and Lesh, N. (2001). COLLAGEN: Applying collaborative discourse theory to human-computer interaction. *AI Magazine*, 22(4):15–25.

Richardson, M., Burges, C. J., and Renshaw, E. (2013). MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203.

Richardson, S. D., Dolan, W. B., and Vanderwende, L. (1998). MindNet: acquiring and structuring semantic information from text. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th International Conference on Computational Linguistics-Volume 2*, pages 1098–1102.

Rieser, V. and Lemon, O. (2008). Learning effective multimodal dialogue strategies from wizard-of-oz data: Bootstrapping and evaluation. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 638–646.

Rieser, V. and Lemon, O. (2010). Natural language generation as planning under uncertainty for spoken dialogue systems. In *Empirical Methods in Natural Language Generation: Data-oriented Methods and Empirical Evaluation*, volume 5790 of *Lecture Notes in Computer Science*, pages 105–120. Springer.

Rieser, V. and Lemon, O. (2011). Learning and evaluation of dialogue strategies for new applications: Empirical methods for optimization from small data sets. *Computational Linguistics*, 37(1):153–196.

Rieser, V., Lemon, O., and Liu, X. (2010). Optimising information presentation for spoken dialogue systems. In *Proceedings of the Forty-Eighth Annual Meeting of the Association for Computational Linguistics (ACL-10)*, pages 1009–1018.

Ritter, A., Cherry, C., and Dolan, W. (2011). Data-driven response generation in social media. In *EMNLP*, pages 583–593.

Roemmele, M., Bejan, C., and S. Gordon, A. (2011). Choice of plausible alternatives: An evaluation of commonsense causal reasoning. In *AAAI Spring Symposium - Technical Report*.

Rosenblatt, F. (1957). The perceptron: A perceiving and recognizing automaton. Report 85-460-1, Project PARA, Cornell Aeronautical Laboratory, Ithaca, New York.

Rosenblatt, F. (1962). *Principles of Neurodynamics*. Spartan Books, New York.

Ross, S., Gordon, G. J., and Bagnell, D. (2011). A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 627–635.

Roy, N., Pineau, J., and Thrun, S. (2000). Spoken dialogue management using probabilistic reasoning. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 93–100.

Russo, D. J., Van Roy, B., Kazerouni, A., Osband, I., and Wen, Z. (2018). A tutorial on Thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–96.

Saha, A., Pahuja, V., Khapra, M. M., Sankaranarayanan, K., and Chandar, S. (2018). Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. *arXiv preprint arXiv:1801.10314*.

Salimans, T., Goodfellow, I. J., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved techniques for training GANs. *CoRR*, abs/1606.03498.

Sarikaya, R., Hinton, G. E., and Deoras, A. (2014). Application of deep belief networks for natural language understanding. *IEEE/ACM Transactions on Audio, Speech & Language Processing*, 22(4):778–784.

Schapire, R. E. and Singer, Y. (2000). BoosTexter: A boosting-based system for text categorization. *Machine Learning*, 39(2/3):135–168.

Schatzmann, J., Georgila, K., and Young, S. (2005a). Quantitative evaluation of user simulation techniques for spoken dialogue systems. In *Proceedings of the 6th SIGdial Workshop on Discourse and Dialogue*, pages 45–54.

Schatzmann, J., Stuttle, M. N., Weilhammer, K., and Young, S. (2005b). Effects of the user model on simulation-based learning of dialogue strategies. In *Proceedings of the IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, pages 220–225.

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

Schatzmann, J. and Young, S. (2009). The hidden agenda user simulation model. *IEEE Transactions on Audio, Speech, and Language Processing*, 17(4):733–747.

Schulman, J., Levine, S., Abbeel, P., Jordan, M. I., and Moritz, P. (2015a). Trust region policy optimization. In *Proceedings of the Thirty-Second International Conference on Machine Learning (ICML)*, pages 1889–1897.

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2015b). High-dimensional continuous control using generalized advantage estimation. arXiv:1506.02438.

See, A., Liu, P. J., and Manning, C. D. (2017). Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Seo, M., Kembhavi, A., Farhadi, A., and Hajishirzi, H. (2016). Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Serban, I. V., Lowe, R., Charlin, L., and Pineau, J. (2015). A survey of available corpora for building data-driven dialogue systems. *arXiv preprint arXiv:1512.05742*.

Serban, I. V., Lowe, R., Henderson, P., Charlin, L., and Pineau, J. (2018). A survey of available corpora for building data-driven dialogue systems: The journal version. *Dialogue & Discourse*, 9(1):1–49.

Serban, I. V., Sordoni, A., Bengio, Y., Courville, A. C., and Pineau, J. (2016). Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, pages 3776–3783.

Serban, I. V., Sordoni, A., Lowe, R., Charlin, L., Pineau, J., Courville, A., and Bengio, Y. (2017). A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*, pages 3295–3301.

Shah, P., Hakkani-Tür, D. Z., Liu, B., and Tür, G. (2018). Bootstrapping a neural conversational agent with dialogue self-play, crowdsourcing and on-line reinforcement learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HTL)*, pages 41–51.

Shang, L., Lu, Z., and Li, H. (2015). Neural responding machine for short-text conversation. In *ACL-IJCNLP*, pages 1577–1586.

Shao, C. C., Liu, T., Lai, Y., Tseng, Y., and Tsai, S. (2018). DRCD: a chinese machine reading comprehension dataset. *arXiv preprint arXiv:1806.00920*.

Shao, Y., Gouws, S., Britz, D., Goldie, A., Strope, B., and Kurzweil, R. (2017). Generating high-quality and informative conversation responses with sequence-to-sequence models. In *EMNLP*, pages 2210–2219.

Shen, Y., Chen, J., Huang, P., Guo, Y., and Gao, J. (2018). M-walk: Learning to walk in graph with monte carlo tree search. *CoRR*, abs/1802.04394.

Shen, Y., He, X., Gao, J., Deng, L., and Mesnil, G. (2014). A latent semantic model with convolutional-pooling structure for information retrieval. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*, pages 101–110. ACM.

Shen, Y., Huang, P., Chang, M., and Gao, J. (2016). Implicit ReasoNet: Modeling large-scale structured relationships with shared memory. *CoRR*, abs/1611.04642.

Shen, Y., Huang, P.-S., Chang, M.-W., and Gao, J. (2017a). Traversing knowledge graph in vector space without symbolic space guidance. *arXiv preprint arXiv:1611.04642*.

Shen, Y., Huang, P.-S., Gao, J., and Chen, W. (2017b). ReasoNet: Learning to stop reading in machine comprehension. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1047–1055. ACM.

Shen, Y., Liu, X., Duh, K., and Gao, J. (2017c). An empirical analysis of multiple-turn reasoning strategies in reading comprehension tasks. *arXiv preprint arXiv:1711.03230*.

Shum, H., He, X., and Li, D. (2018). From Eliza to XiaoIce: Challenges and opportunities with social chatbots. *CoRR*, abs/1801.01957.

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

Socher, R., Chen, D., Manning, C. D., and Ng, A. (2013). Reasoning with neural tensor networks for knowledge base completion. In *Advances in neural information processing systems*, pages 926–934.

Sordoni, A., Bachman, P., Trischler, A., and Bengio, Y. (2016). Iterative alternating neural attention for machine reading. *arXiv preprint arXiv:1606.02245*.

Sordoni, A., Bengio, Y., Vahabi, H., Lioma, C., Grue Simonsen, J., and Nie, J.-Y. (2015a). A hierarchical recurrent encoder-decoder for generative context-aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, CIKM '15, pages 553–562. ACM.

Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J.-Y., Gao, J., and Dolan, B. (2015b). A neural network approach to context-sensitive generation of conversational responses. In *NAACL-HLT*, pages 196—-205.

Stanojević, M. and Sima'an, K. (2014). Fitting sentence level translation evaluation with many dense features. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 202–206, Doha, Qatar.

Stent, A., Prasad, R., and Walker, M. A. (2004). Trainable sentence planning for complex information presentations in spoken dialog systems. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 79–86.

Stone, M., Doran, C., Webber, B. L., Bleam, T., and Palmer, M. (2003). Microplanning with communicative intentions: The SPUD system. *Computational Intelligence*, 19(4):311–381.

Strehl, A. L., Li, L., and Littman, M. L. (2009). Reinforcement learning in finite MDPs: PAC analysis. *Journal of Machine Learning Research*, 10:2413–2444.

Strub, F., de Vries, H., Mary, J., Piot, B., Courville, A. C., and Pietquin, O. (2017). End-to-end optimization of goal-driven and visually grounded dialogue systems. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2765–2771.

Su, P.-H., Gasic, M., Mrksic, N., Rojas-Barahona, L. M., Ultes, S., Vandyke, D., Wen, T.-H., and Young, S. J. (2016a). On-line active reward learning for policy optimisation in spoken dialogue systems. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL)*, volume 1, pages 2431–2441.

Su, P.-H., Gašić, M., Mrkšić, N., Rojas-Barahona, L., Ultes, S., Vandyke, D., Wen, T.-H., and Young, S. (2016b). Continuously learning neural dialogue management. arXiv preprint: 1606.02689.

Su, P.-H., Gašić, M., and Young, S. (2018a). Reward estimation for dialogue policy optimisation. *Computer Speech & Language*, 51:24–43.

Su, P.-H., Vandyke, D., Gašić, M., Kim, D., Mrkšić, N., Wen, T.-H., and Young, S. (2015). Learning from real users: Rating dialogue success with neural networks for reinforcement learning in spoken dialogue systems. In *Proceedings of the 16th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2007–2011.

Su, S.-Y., Li, X., Gao, J., Liu, J., and Chen, Y.-N. (2018b). Discriminative deep Dyna-Q: Robust planning for dialogue policy learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3813–3823.

Su, S.-Y., Lo, K.-L., Yeh, Y. T., and Chen, Y.-N. (2018c). Natural language generation by hierarchical decoding with linguistic patterns. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 61–66.

Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). YAGO: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

Suhr, A., Iyer, S., and Artzi, Y. (2018). Learning to map context-dependent sentences to executable formal queries. *arXiv preprint arXiv:1804.06868*.

Sutskever, I., Vinyals, O., and Le, Q. (2014). Sequence to sequence learning with neural networks. In *NIPS*, pages 3104–3112.

Sutton, R. S. (1988). Learning to predict by the methods of temporal differences. *Machine Learning*, 3(1):9–44.

Sutton, R. S. (1990). Integrated architectures for learning, planning, and reacting based on approximating dynamic programming. In *Proceedings of the seventh international conference on machine learning*, pages 216–224.

Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT Press, 2nd edition edition.

Sutton, R. S., McAllester, D., Singh, S. P., and Mansour, Y. (1999a). Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems 12 (NIPS)*, pages 1057–1063.

Sutton, R. S., Precup, D., and Singh, S. P. (1999b). Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1–2):181–211. An earlier version appeared as Technical Report 98-74, Department of Computer Science, University of Massachusetts, Amherst, MA 01003. April, 1998.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. (2013). Intriguing properties of neural networks. *CoRR*, abs/1312.6199.

Szepesvári, C. (2010). *Algorithms for Reinforcement Learning*. Morgan & Claypool.

Talmor, A. and Berant, J. (2018). The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Tang, D., Li, X., Gao, J., Wang, C., Li, L., and Jebara, T. (2018). Subgoal discovery for hierarchical dialogue policy learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2298—2309.

Tesauro, G. (1995). Temporal difference learning and TD-Gammon. *Communications of the ACM*, 38(3):58–68.

Thomas, P. S. and Brunskill, E. (2016). Data-efficient off-policy policy evaluation for reinforcement learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML)*, pages 2139–2148.

Thompson, W. R. (1933). On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3–4):285–294.

Tiedemann, J. (2012). Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Toutanova, K., Lin, V., Yih, W.-t., Poon, H., and Quirk, C. (2016). Compositional learning of embeddings for relation paths in knowledge base and text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1434–1444.

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

Trischler, A., Wang, T., Yuan, X., Harris, J., Sordoni, A., Bachman, P., and Suleman, K. (2016). NewsQA: A machine comprehension dataset. *arXiv preprint arXiv:1611.09830*.

Tromp, J. and Farnebäck, G. (2006). Combinatorics of Go. In *Proceedings of the Fifth International Conference on Computers and Games*, number 4630 in Lecture Notes in Computer Science, pages 84–99.

Tur, G. and De Mori, R. (2011). *Spoken language understanding: Systems for extracting semantic information from speech*. John Wiley & Sons.

Ultes, S., Budzianowski, P., nigo Casanueva, I., Mrkšić, N., Rojas-Barahona, L. M., Su, P.-H., Wen, T.-H., Gašić, M., and Young, S. J. (2017a). Domain-independent user satisfaction reward estimation for dialogue policy learning. In *Proceedings of the 18th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 1721–1725.

Ultes, S., Budzianowski, P., nigo Casanueva, I., Mrkšić, N., Rojas-Barahona, L. M., Su, P.-H., Wen, T.-H., Gašić, M., and Young, S. J. (2017b). Reward-balancing for statistical spoken dialogue systems using multi-objective reinforcement learning. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 65–70.

Ultes, S., Rojas-Barahona, L. M., Su, P.-H., Vandyke, D., Kim, D., nigo Casanueva, I., Budzianowski, P., Mrkšić, N., Wen, T.-H., Gašić, M., and Young, S. J. (2017c). PyDial: A multi-domain statistical dialogue system toolkit. In *Proceedings of the Fifty-fifth Annual Meeting of the Association for Computational Linguistics (ACL), System Demonstrations*, pages 73–78.

van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. (2016). WaveNet: A generative model for raw audio. arXiv:1609.03499.

van Hasselt, H., Guez, A., and Silver, D. (2016). Deep reinforcement learning with double Q-learning. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (AAAI-16)*, pages 2094–2100.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Vinyals, O., Fortunato, M., and Jaitly, N. (2015a). Pointer networks. In *Advances in Neural Information Processing Systems 28*, pages 2692–2700. Curran Associates, Inc.

Vinyals, O. and Le, Q. (2015). A neural conversational model. In *ICML Deep Learning Workshop*.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. (2015b). Show and tell: A neural image caption generator. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3156–3164.

Walker, M., Kamm, C., and Litman, D. (2000). Towards developing general models of usability with PARADISE. *Natural Language Engineering*, 6(3–4):363–377.

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

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

Walker, M. A., Litman, D. J., Kamm, C. A., and Abella, A. (1998). Evaluating spoken dialogue agents with PARADISE: Two case studies. *Computer Speech & Language*, 12(4):317–347.

Walker, M. A., Stent, A., Mairesse, F., and Prasad, R. (2007). Individual and domain adaptation in sentence planning for dialogue. *Journal of Artificial Intelligence Research*, 30:413–456.

Wang, C., Wang, Y., Huang, P.-S., Mohamed, A., Zhou, D., and Deng, L. (2017a). Sequence modeling via segmentations. In *Proceedings of the 34th International Conference on Machine Learning*, pages 3674–3683.

Wang, W., Yang, N., Wei, F., Chang, B., and Zhou, M. (2017b). Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 189–198.

Wang, Y.-Y., Deng, L., and Acero, A. (2005). Spoken language understanding: An introduction to the statistical framework. *IEEE Signal Processing Magazine*, 22(5):16–31.

Wang, Z., Chen, H., Wang, G., Tian, H., Wu, H., and Wang, H. (2014). Policy learning for domain selection in an extensible multi-domain spoken dialogue system. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 57–67.

Wang, Z., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M., and de Freitas, N. (2016). Dueling network architectures for deep reinforcement learning. In *Proceedings of the Third International Conference on Machine Learning (ICML-16)*, pages 1995–2003.

Watkins, C. J. (1989). *Learning from Delayed Rewards*. PhD thesis, King's College, University of Cambridge, UK.

Wei, W., Le, Q. V., Dai, A. M., and Li, L.-J. (2018). AirDialogue: An environment for goal-oriented dialogue research. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3844–3854.

Weissenborn, D., Wiese, G., and Seiffe, L. (2017). FastQA: A simple and efficient neural architecture for question answering. *arXiv preprint arXiv:1703.04816*.

Weizenbaum, J. (1966). ELIZA: a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9(1):36–45.

Welbl, J., Stenetorp, P., and Riedel, S. (2017). Constructing datasets for multi-hop reading comprehension across documents. *arXiv preprint arXiv:1710.06481*.

Wen, T.-H., Gašić, M., Mrkšić, N., Rojas-Barahona, L. M., Su, P.-H., Vandyke, D., and Young, S. J. (2016). Multi-domain neural network language generation for spoken dialogue systems. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (HLT-NAACL)*, pages 120–129.

Wen, T.-H., Gašić, M., Mrkšić, N., Su, P.-H., Vandyke, D., and Young, S. J. (2015). Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1711–1721.

Wen, T.-H., Vandyke, D., Mrkšić, N., Gašić, M., Rojas-Barahona, L. M., Su, P.-H., Ultes, S., and Young, S. J. (2017). A network-based end-to-end trainable task-oriented dialogue system. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, pages 438–449. arXiv reprint arXiv:1604.04562.

Wiering, M. and van Otterlo, M. (2012). *Reinforcement Learning: State of the Art*. Springer.

Williams, J. D. (2006). *Partially Observable Markov Decision Processes for Spoken Dialogue Management*. PhD thesis, Cambridge University, Cambridge, UK.

Williams, J. D. (2008). Evaluating user simulations with the Cramér-von Mises divergence. *Speech Communication*, 50(10):829–846.

Williams, J. D., Asadi, K., and Zweig, G. (2017). Hybrid code networks: Practical and efficient end-to-end dialog control with supervised and reinforcement learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (ACL)*, volume 1, pages 665–677.

Williams, J. D., Henderson, M., Raux, A., Thomson, B., Black, A. W., and Ramachandran, D. (2014). The dialog state tracking challenge series. *AI Magazine*, 35(4):121–124.

Williams, J. D., Raux, A., Ramachandran, D., and Black, A. W. (2013). The dialog state tracking challenge. In *Proceedings of the 14th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 404–413.

Williams, J. D. and Young, S. J. (2007). Partially observable Markov decision processes for spoken dialog systems. *Computer Speech and Language*, 21(2):393–422.

Williams, J. D. and Zweig, G. (2016). End-to-end LSTM-based dialog control optimized with supervised and reinforcement learning.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256.

Winata, G. I., Kampman, O., Yang, Y., Dey, A., and Fung, P. (2017). Nora the empathetic psychologist. In *Proc. Interspeech*, pages 3437–3438.

Wu, C.-S., Madotto, A., Winata, G. I., and Fung, P. (2018). End-to-end dynamic query memory network for entity-value independent task-oriented dialog. In *Proceedings of the 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6154–6158.

Wu, J., Li, M., and Lee, C.-H. (2015). A probabilistic framework for representing dialog systems and entropy-based dialog management through dynamic stochastic state evolution. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 23(11):2026–2035.

Wu, Q., Burges, C. J., Svore, K. M., and Gao, J. (2010). Adapting boosting for information retrieval measures. *Information Retrieval*, 13(3):254–270.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., and Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Xing, C., Wu, W., Wu, Y., Zhou, M., Huang, Y., and Ma, W. (2018). Hierarchical recurrent attention network for response generation. In *AAAI*, pages 5610–5617.

Xiong, C., Zhong, V., and Socher, R. (2016). Dynamic coattention networks for question answering. *arXiv preprint arXiv:1611.01604*.

Xiong, W., Hoang, T., and Wang, W. Y. (2017). DeepPath: A reinforcement learning method for knowledge graph reasoning. *arXiv preprint arXiv:1707.06690*.

Xu, P., Madotto, A., Wu, C.-S., Park, J. H., and Fung, P. (2018). Emo2vec: Learning generalized emotion representation by multi-task training. *arXiv preprint arXiv:1809.04505*.

Xu, Z., Liu, B., Wang, B., Sun, C., Wang, X., Wang, Z., and Qi, C. (2017). Neural response generation via GAN with an approximate embedding layer. In *EMNLP*, pages 617–626.

Yaman, S., Deng, L., Yu, D., Wang, Y.-Y., and Acero, A. (2008). An integrative and discriminative technique for spoken utterance classification. *IEEE Transactions on Audio, Speech & Language Processing*, 16(6):1207–1214.

Yan, R., Song, Y., and Wu, H. (2016). Learning to respond with deep neural networks for retrieval-based human-computer conversation system. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR, pages 55–64, New York, NY, USA. ACM.

Yang, B., Yih, W.-t., He, X., Gao, J., and Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. In *ICLR*.

Yang, F., Yang, Z., and Cohen, W. W. (2017a). Differentiable learning of logical rules for knowledge base completion. *CoRR, abs/1702.08367*.

Yang, X., Chen, Y.-N., Hakkani-Tür, D. Z., Crook, P., Li, X., Gao, J., and Deng, L. (2017b). End-to-end joint learning of natural language understanding and dialogue manager. In *Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5690–5694.

Yao, K., Zweig, G., Hwang, M.-Y., Shi, Y., and Yu, D. (2013). Recurrent neural networks for language understanding. In *Proceedings of the 14th Annual Conference of the International Speech Communication Association (INTERSPEECH)*, pages 2524–2528.

Yao, K., Zweig, G., and Peng, B. (2015). Attention with intention for a neural network conversation model. In *NIPS workshop on Machine Learning for Spoken Language Understanding and Interaction*.

Yao, X. and Van Durme, B. (2014). Information extraction over structured data: Question answering with Freebase. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 956–966.

Yih, W.-t., Chang, M.-W., He, X., and Gao, J. (2015a). Semantic parsing via staged query graph generation: Question answering with knowledge base. In *ACL*, pages 1321–1331.

Yih, W.-t., He, X., and Gao, J. (2015b). Deep learning and continuous representations for natural language processing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Tutorial*.

Yih, W.-t., He, X., and Gao, J. (2016). Deep learning and continuous representations for natural language processing. In *IJCAI: Tutorial*.

Young, S., Breslin, C., Gašić, M., Henderson, M., Kim, D., Szummer, M., Thomson, B., Tsiakoulis, P., and Hancock, E. T. (2016). Evaluation of statistical POMDP-based dialogue systems in noisy environments. In *Situated Dialog in Speech-Based Human-Computer Interaction*, Signals and Communication Technology, pages 3–14. Springer.

Young, S., Gašić, M., Thomson, B., and Williams, J. D. (2013). POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179.

Young, S. J., Gašić, M., Keizer, S., Mairesse, F., Schatzmann, J., Thomson, B., and Yu, K. (2010). The Hidden Information State model: A practical framework for POMDP-based spoken dialogue management. *Computer Speech & Language*, 24(2):150–174.

Yu, A. W., Dohan, D., Luong, M.-T., Zhao, R., Chen, K., Norouzi, M., and Le, Q. V. (2018). QANet: Combining local convolution with global self-attention for reading comprehension. *arXiv preprint arXiv:1804.09541*.

Zhang, J., Zhao, T., and Yu, Z. (2018a). Multimodal hierarchical reinforcement learning policy for task-oriented visual dialog. In *Proceedings of the 19th Annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*, pages 140–150.

Zhang, R., Guo, J., Fan, Y., Lan, Y., Xu, J., and Cheng, X. (2018b). Learning to control the specificity in neural response generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1108–1117, Melbourne, Australia.

Zhang, S., Dinan, E., Urbanek, J., Szlam, A., Kiela, D., and Weston, J. (2018c). Personalizing dialogue agents: I have a dog, do you have pets too? In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2204–2213, Melbourne, Australia.

Zhang, S., Liu, X., Liu, J., Gao, J., Duh, K., and Van Durme, B. (2018d). ReCoRD: Bridging the gap between human and machine commonsense reading comprehension. *arXiv preprint arXiv:1810.12885*.

Zhang, Y., Galley, M., Gao, J., Gan, Z., Li, X., Brockett, C., and Dolan, B. (2018e). Generating informative and diverse conversational responses via adversarial information maximization. In *NeurIPS*, pages 1815–1825.

Zhao, T. and Eskénazi, M. (2016). Towards end-to-end learning for dialog state tracking and management using deep reinforcement learning. In *Proceedings of the 17th Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, pages 1–10.

Zhao, T., Lu, A., Lee, K., and Eskenazi, M. (2017). Generative encoder-decoder models for task-oriented spoken dialog systems with chatting capability. In *ACL*, pages 27–36.

Zhou, L., Gao, J., Li, D., and Shum, H.-Y. (2018). The design and implementation of XiaoIce, an empathetic social chatbot. *arXiv preprint arXiv:1812.08989*.