University of California
Santa Barbara

Knowledge-Grounded Natural Language Processing

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Computer Science

by

Zhiyu Chen

June 2022

The Dissertation of Zhiyu Chen is approved.

Professor Tao Yang

Professor Xifeng Yan, Co-Chair

Professor William Yang Wang, Co-Chair

June 2022

Knowledge-Grounded Natural Language Processing

To my family for the unconditional love.

To my friends for the endless accompany and support.

To myself for the dedication to explore and change the world.

Acknowledgements

My deepest gratitude goes to my advisors William Yang Wang and Xifeng Yan. They brought me to the Ph.D. journey and wonderful life at UCSB. They guided me through the research field of natural language processing and helped me make my own way with endless support. They also shared their insights about careers and life, helping to shape up my future. Besides, I also want to thank my committee member Tao Yang for his insightful feedback and suggestions about my research.

I would like to thank all my my collaborators. It is my truly fortunate to meet and work with many excellent people. Here I want to thank Yu Su, Wenhu Chen, Hanwen Zha, Honglei Liu, Bing Liu, Seungwhan Moon, Hu Xu, Chinnadhurai Sankar, Paul Crook, Hao Zhou, Charese Smiley, Sameena Shah, Harini Eavani, Yinyin Liu, Matt Beane, Ting-Hao Huang, Bryan Routledge, Xian Qian, Shengqi Yang, Yu Deng, Ruchi Mahindru, Daniela Russo, Shu Tao, Shiyang Li, Xinlu Zhang, Xiyou Zhou.

I would like to thank my colleagues and friends, for all the helpful discussions, career advice, and emotional support. Here I want to thank Chengcheng Wan, Xiaojie Wu, Xinyi Zhang, Chong Liu, Yanju Chen, Michael Saxon, Dan Qiao, Ming Yin, Wanrong Zhu, Yujie Lu, Xin Jiang, Wenda Xu, Hong Wang, Jing Qian, Jiangyue Cai, and many more that I cannot list them all here for the wonderful time we spent together.

At last, I want to thank my family for the unconditional love and endless support as always.

Thesis Statement

An artificial intelligent assistant uses the natural language interface as its core component to conduct natural language understanding and natural language generation grounded on various knowledge forms. In this thesis, I aim to push forward in both directions, with the ultimate goal of building the assistant that can perform understanding, reasoning, and realization to converse like human beings.

Curriculum Vitæ
Zhiyu Chen

Education

University of California, Santa Barbara, USA *Sept. 2017 - June. 2022*
Ph.D. in Computer Science
Advisor: Prof. William Wang and Prof. Xifeng Yan
Major GPA: 3.98/4.0

Shanghai Jiao Tong University, China *Sept. 2013 - June 2017*
B.S. in Computer Science
Major GPA: 89/100

Publications

[11] **KETOD: Knowledge-Enriched Task-Oriented Dialogue**
Zhiyu Chen, Bing Liu, Seungwhan Moon, Chinnadhurai Sankar, Paul Crook, and William Yang Wang
Findings of NAACL 2022

[10] **Inductive Relation Prediction by BERT**
Hanwen Zha, Zhiyu Chen, and Xifeng Yan
AAAI 2022

[9] **FinQA: A Dataset of Numerical Reasoning over Financial Data**
Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, and William Yang Wang
EMNLP 2021

[8] **NUANCED: Natural Utterance Annotation for Nuanced Conversation with Estimated Distributions**
Zhiyu Chen, Honglei Liu, Hu Xu, Seungwhan Moon, Hao Zhou, and Bing Liu
Findings of EMNLP 2021

[7] **Logic2Text: High-Fidelity Natural Language Generation from Logical Forms**
Zhiyu Chen, Wenhu Chen, Hanwen Zha, Xiyou Zhou, Yunkai Zhang, Sairam Sundaresan, and William Yang Wang
Findings of EMNLP 2020

[6] **HybridQA: A Dataset of Multi-Hop Question Answering over Tabular and Textual Data**
Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang

Wang
Findings of EMNLP 2020

[5] Few-Shot NLG with Pre-Trained Language Model
Zhiyu Chen, Harini Eavani, Wenhu Chen, Yinyin Liu, and William Yang Wang
ACL 2020

[4] Logical Natural Language Generation From Open-Domain ables
Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang
ACL 2020

[3] Global Textual Relation Embedding for Relational Understanding
Zhiyu Chen, Hanwen Zha, Honglei Liu, Wenhu Chen, Xifeng Yan, and Yu Su
ACL 2019

[2] How Large A Vocabulary Does Text Classification Need? A Variational Approach on Vocabulary Selection
Wenhu Chen, Yu Su, Yilin Shen, Zhiyu Chen, Xifeng Yan, and William Yang Wang
NAACL 2019

[1] IMAP: An Iterative Method for Aligning Protein-Protein Interaction Networks
Xuezhi Cao, Zhiyu Chen, Xinyi Zhang, and Yong Yu
BIBM 2017

Research and Internship Experiences

Facebook *June 2021 - Sept. 2021*
Research Intern Menlo Park, CA, USA
- Proposed knowledge-enriched task-oriented dialogue

Facebook *June 2020 - Sept. 2020*
Research Intern Menlo Park, CA, USA
- Proposed user-centric dialogue system

WeWork *June 2019 - Sept. 2019*
Research Intern Palo Alto, CA, USA
- Designed a text summarization & sentence compression system for low-resource training data

IBM Thomas J. Watson Research Center *June 2018 - Sept. 2018*
Research Intern White Plains, NY, USA
- Designed a knowledge base QA system for IT support

Intel Asia-Pacific Research & Development Ltd. *Aug. 2016 - Dec. 2016*
Software Engineer Intern Shanghai, China
- Worked on image encoding and multi-thread programming

Abstract

Knowledge-Grounded Natural Language Processing

by

Zhiyu Chen

As the primary means of human communication, natural language bears the functionality to bridge the abstract knowledge form and its realizations to all kinds of human usages. In an artificial intelligence assistant, the natural language interface is grounded on various knowledge sources and designated to perform conveyance, navigation, and reasoning among those sources. Based on the mechanism to interact with the knowledge groundings, the natural language processing tasks can be primarily categorized into 1) natural language understanding (NLU): understanding the semantics of the natural language from the user to perform navigation or reasoning among the knowledge sources, and 2) natural language generation (NLG): generating natural language responses to the user grounded on the knowledge sources. In this thesis, I will elaborate on my work on both NLU and NLG divisions in order to extend the edge of current research toward building an advanced, human-like AI assistant.

For the NLU division, first, current large pre-trained language models can achieve very strong performances on many NLP tasks but still struggle with tasks requiring complex reasoning abilities. In this thesis, I will discuss my work on complex numerical reasoning over structured and unstructured knowledge. Second, current dialogue assistants follow a fixed ontology to represent user preferences, which makes them essentially agent-centric and lack the power to represent fine-grained, realistic user preferences. In this thesis, I will discuss our first step to build the user-centric dialogue system by proposing a new formulation for user preference representations.

For the NLG division, first, current models can obtain strong performances with large-scale training data, but this is not realistic for real-world applications. In this thesis, I will discuss my work on few-shot natural language generation from tabular data. Second, NLG models often suffer from factual correctness when generating descriptions with logical inferences. This thesis proposes to study high-fidelity natural language generation with logical inferences. At last, I will discuss my work for NLG in the important real application domain of dialogue systems. Current research on dialogue systems typically studies each type of dialogue individually. However, the ultimate goal of conversational AI is to build a unified assistant capable of conversing with all kinds of dialogues and switching among them seamlessly. Towards this goal, this thesis studies to enrich task-oriented dialogue with knowledge-grounded chitchat.

In the above work, we define the novel tasks and propose new datasets, as well as design approaches to tackle the new challenges. Finally, we will make the conclusion and discuss the future plans and directions.

Contents

Curriculum Vitae **vii**

Abstract **ix**

1 Introduction **1**
1.1 Knowledge-Grounded Natural Language Processing 1
1.2 Natural Language Understanding 2
1.3 Natural Language Generation 4
1.4 Summary 6

Natural Language Understanding **6**

2 Question Answering with Numerical Reasoning **7**
2.1 Introduction 7
2.2 Related Work 10
2.3 Task Definition 11
2.4 The FinQA Dataset 13
2.5 Baseline Systems 17
2.6 Experiments 21

3 Representation Learning for Fine-Grained User Preference **26**
3.1 Introduction 26
3.2 Related Work 28
3.3 The nuanced Dataset 29
3.4 Experiments 33

Natural Language Generation **36**

4 Few-Shot Natural Language Generation **38**
4.1 Introduction 38
4.2 Related Work 41
4.3 Method 42

4.4 Experiments 44

5 High-Fidelity Natural Language Generation from Logical Forms 49
5.1 Introduction 49
5.2 Related Work 52
5.3 Dataset Construction 54
5.4 Dataset Statistics and Analysis 59
5.5 Experiments 60

6 Knowledge-Enriched Task-Oriented Dialogue 67
6.1 Introduction 67
6.2 Related Work 70
6.3 The KETOD Dataset 71
6.4 Approaches 75
6.5 Experiments 79

7 Conclusion and Future Work 86
7.1 Natural Language Understanding 86
7.2 Natural Language Generation 88
7.3 Summary 89

Appendix 90

A FinQA: A Dataset of Numerical Reasoning over Financial Data 91
A.1 Operation Definitions 91
A.2 Experiment Details 92
A.3 Case Studies 92
A.4 Annotation Interface 93

B NUANCED: Natural Utterance Annotation for Nuanced Conversation with Estimated Distributions 96
B.1 Model Implementation and Training Details 96
B.2 Analysis on Slots 98
B.3 Case Studies 98

C Few-Shot NLG with Pre-Trained Language Model 100
C.1 Implementation Details 100
C.2 ROUGE-4 Results 100
C.3 Human Evaluation Details 102

D Logic2Text: High-Fidelity Natural Language Generation from Logical Forms 104
D.1 Logic Type Definitions & Logical Form Annotation 104
D.2 Function Definitions 108

D.3 Model Implementation Details 109
D.4 Human Evaluation Details 112
D.5 Generation Examples 113

E KETOD: Knowledge-Enriched Task-Oriented Dialogue 115
E.1 Dataset Construction 115
E.2 Model and Training Details 116
E.3 Evaluation Details 117

Bibliography 119

Chapter 1

Introduction

1.1 Knowledge-Grounded Natural Language Processing

As the primary means of human communication, natural language is the innate bearer of the functionality to bridge the abstract knowledge form and its realizations to all kinds of human usages. In our daily life, we understand the meaning of the utterances from others and use our own knowledge to finish the task or respond. Meanwhile, we use language to express knowledge from our cognition so as to be understood by others. In artificial intelligence (AI) assistant, analogous to the way humans ground language on knowledge, a natural language interface typically serves as the core component to be grounded on various knowledge sources and designated to perform conveyance, navigation, and reasoning among those sources. Popular AI assistants include, for example, the Amazon Alexa, the Apple Siri, as well as some specialized assistants in specific domains.

To build such assistants, based on the mechanism to interact with the knowledge groundings, the natural language processing tasks can be primarily categorized into 1)

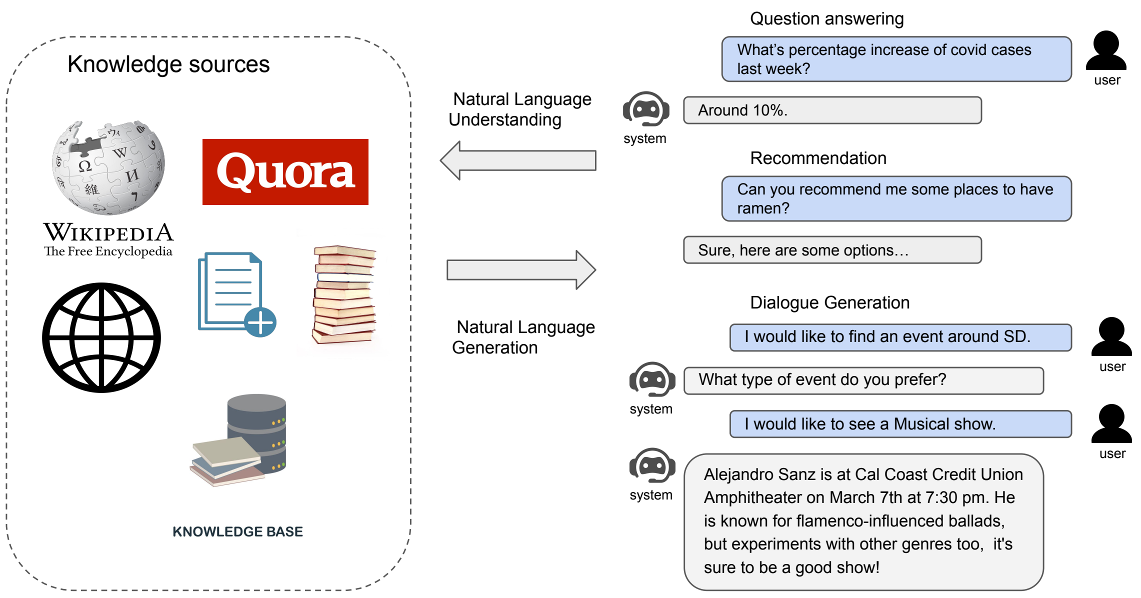

Knowledge sources
WIKIPEDIA The Free Encyclopedia
Quora
KNOWLEDGE BASE
Natural Language Understanding
Natural Language Generation
Question answering
What's percentage increase of covid cases last week?
user
system
Around 10%.
Recommendation
Can you recommend me some places to have ramen?
Sure, here are some options…
Dialogue Generation
I would like to find an event around SD.
What type of event do you prefer?
I would like to see a Musical show.
Alejandro Sanz is at Cal Coast Credit Union Amphitheater on March 7th at 7:30 pm. He is known for flamenco-influenced ballads, but experiments with other genres too, it's sure to be a good show!

Figure 1.1: The natural language interface in AI assistants.

natural language understanding (NLU): understanding the semantic meaning of the natural language from the user to perform navigation or reasoning among the knowledge sources, and 2) natural language generation (NLG): generating natural language responses to the user grounded on the knowledge sources. Figure 1.1 illustrates these two branches of tasks.

In the next two sections, I will dive into these two divisions and summarize my own work and contributions.

1.2 Natural Language Understanding

In natural language understanding (NLU) tasks, the system aims to understand the semantic meaning of the user utterances and take the actions correspondingly. The major building blocks for NLU in modern AI assistants correspond to two core tasks, Question Answering (QA), where the system provides answers to user queries, and dialogue understanding, where the system understands user intentions and makes responses or takes actions in an interactive way.

Question Answering In QA research, there have been decades of efforts on studying different types of questions, modalities, knowledge forms, and domains [1, 2, 3]. For example, text-based reading comprehensions [4, 5], knowledge base question answering [6, 7], open-domain question answering [8, 9], numerical question answering [10, 11], and question answering for specialized domains [12], etc. Based on the challenge types and difficulty, these tasks can be divided into two categories: 1) QA with a focus on semantic matching and understanding, where the models mostly need to learn the language patterns and navigate to the knowledge evidence, and 2) QA with a focus on reasoning, including logical, mathematical, or commonsense reasoning over the knowledge evidence, that humans typically need to think slowly and deliberately to solve [13]. Recently, with the surge of deep neural networks, followed by the era of large pre-trained language models (LM) [14, 15], researchers have achieved strong performance on the first type QA tasks, for example, many pre-training based models have beaten the human performance on the SQuAD dataset [16]. However, for the second type of tasks, language models still struggle a lot even with billions of parameters [17]. Therefore, we anticipate that such tasks requiring complex reasoning will become the next research focus.

In this dissertation, we focus on studying the QA task requiring complex numerical reasoning, and we propose FinQA [18], a new large-scale dataset over financial reports consisting of both tabular and textual knowledge forms. To answer FinQA questions, the model needs to understand the question, the input financial report, and derive a calculation program to get the answer. In this work, we further introduce new models and conduct comprehensive experiments on our dataset, discussing limitations and future directions.

Dialogue Understanding During the conversation with the users, the AI assistant needs to understand user intents and make corresponding actions and responses. The task-oriented dialogue system is the current research focus in industrial applications, where the goal is to complete certain user tasks such as ticket booking, restaurant recommendations, etc. Existing work formulates the dialogue understanding as the dialogue state tracking task to learn the semantic meaning of the user utterances and map to a set of pre-defined slot-value pairs [19, 20, 21, 22, 23, 24]. The pre-defined slot-value ontology provides a consistent and easy way to do database queries to access the knowledge. However, such formulation essentially makes the dialogue system agent-centric. The agent-centric systems require the users to unnaturally adapt to and even have a learning curve on the system ontology, which is largely unknown to the users (such as the sample instructions for most smart speakers).

In this dissertation, we take the first step to build user-centric dialogue systems. We propose a new dialogue state formulation to capture fine-grained user preferences. We construct a new dataset, named NUANCED, and demonstrate the effectiveness of such a formulation.

1.3 Natural Language Generation

Opposite to the NLU tasks, where the models aim to map natural language to meaning representations, in NLG tasks, we target generating natural language from the meaning representations. The NLG module takes charge of realizing the knowledge-groundings and system actions into responses to the users in the AI assistants. Common usages include generating reports from structured data, such as game records, weather records, and response generation in dialogue systems.

NLG from Structured Data Natural language generation from structured data or knowledge (NLG) has been studied for many years. Early traditional NLG systems follow the pipeline paradigm that explicitly divides generation into content selection, planning, and surface realization [25]. Recently, with the success of deep neural networks, data-driven, neural-based approaches have been used [26, 27, 28] and achieved remarkable performance on a number of benchmarks.

In this dissertation, we investigate NLG from structured data in two directions. First, current deep neural network models can achieve good performance only with sufficient training data. This prohibits most models from being adapted to real-world applications. To tackle this, we propose a new model for NLG in the few-shot scenario and obtain significant improvements. Second, current work on NLG mostly focuses on generating surface descriptions. When encountered with higher-level logical inferences, the models suffer from hallucinations. We investigate how to generate more complex, logically entailed descriptions while preserving fidelity.

Dialogue Response Generation Response generation in dialogue systems is another important application of NLG. There are two types of most commonly studied dialogue systems, task-oriented dialogue (TOD) and knowledge-grounded chit-chat. In TOD systems, the response generation module takes the retrieved database results and actions, typically in structured data form, as the input and generates the responses to the user. In chit-chat dialogue, the systems take relevant world knowledge to generate chit-chat grounded on such knowledge. TOD systems focus on the completion of the user tasks; as a result, the generated system responses are concise and templated. And chit-chat systems aim to generate natural and engaging conversations. Existing studies mostly focus on one specific type of dialogue, either task-oriented dialogue or knowledge-grounded chit-chat. However, the ultimate goal of conversational AI assistants is a human-like, unified system capable of conversing with the users naturally and seamlessly among all kinds of dialogues.

In this dissertation, we propose to generate the TOD responses enriched with knowledge-grounded chit-chat so as to conduct more social, natural, and engaging conversations. We propose a new dataset, design new models, and demonstrate the effectiveness of our approach.

1.4 Summary

In this section, I briefly summarize my contributions during my Ph.D. studies. I have been working on building the natural language interface of AI assistants in both NLU and NLG divisions.

For natural language understanding (NLU), in [18], I studied how to answer complex numerical reasoning questions over textual and tabular data. In [29], I propose a new formulation for dialogue state tracking to better capture fine-grained user preferences.

For natural language generation (NLG), in [30], I studied NLG in few-shot scenario. In [31], I studied high-fidelity generation with logical inferences. In [32], I propose to enrich task-oriented dialogue with knowledge-grounded chit-chat.

Chapter 2

Question Answering with Numerical Reasoning

2.1 Introduction

Financial analysis is a critical means of assessing business performance, and the consequences of poor analysis can involve costs of billions of dollars [33, 34]. To facilitate high quality, timely decision making, professionals — such as analysts or investors — perform complex quantitative analysis to select information from financial reports. Such analysis demands advanced expertise in reasoning among heterogeneous (structured and unstructured) data sources and performing complex numerical reasoning, for example, comparing financial ratios of profitability or growth. These challenges are compounded by an exponentially expanding collection of company financial documents [35, 36] such that it is genuinely unclear whether dedicated human effort can produce fiscal analysis of sufficient quality for current decision making. This poses an interesting question: can we automate such deep analysis of financial data?

A few NLP studies in Question Answering (QA) explored the numerical reasoning capabilities needed to answer questions correctly. For example, the DROP dataset [10] focused on Wikipedia-based questions that require numerical reasoning, *e.g.,* "Where did Charles travel to first, Castile or Barcelona?" needs a comparison between the times of two events. However, most prior work only targeted the general domain, where the questions involve much less calculation (mostly one-step calculation) than that of the financial domain. Financial QA is more challenging than classic QA [16, 37] because it requires the system to spot relevant information across heterogeneous sources, such as tables and unstructured texts, and then create a numerical reasoning path to connect all the information. It also takes substantial knowledge to ask meaningful financial questions. It is not clear how well the large language models, which performed well for general-domain QA, can be adapted to answer realistic, complex financial questions.

This chapter introduces **FinQA**, a **expert-annotated** dataset that contains 8,281 financial QA pairs, along with their numerical reasoning processes. Eleven finance professionals collectively constructed **FinQA** based on the earnings reports of S&P 500 companies [38]. The questions in **FinQA**, such as "Considering the weighted average fair value of options, what was the change of shares vested from 2005 to 2006?" (Figure 2.1) and "What was the net change in tax positions in 2014?", require information from both tables and unstructured texts to answer. The reasoning processes answering these questions are made of many common calculations in financial analysis, such as addition, comparison, and table aggregation. To the best of our knowledge, **FinQA** is the first dataset of its kind to tackle complicated QA tasks based on the real-world financial documents.

We propose a retriever-generator QA framework to first retrieve supporting facts from financial reports, then to generate executable reasoning programs to answer the questions. Equipped with pre-trained language models, such as BERT [39] and RoBERTa [40], our proposed approach outperforms all other baselines and achieves an execution accuracy of

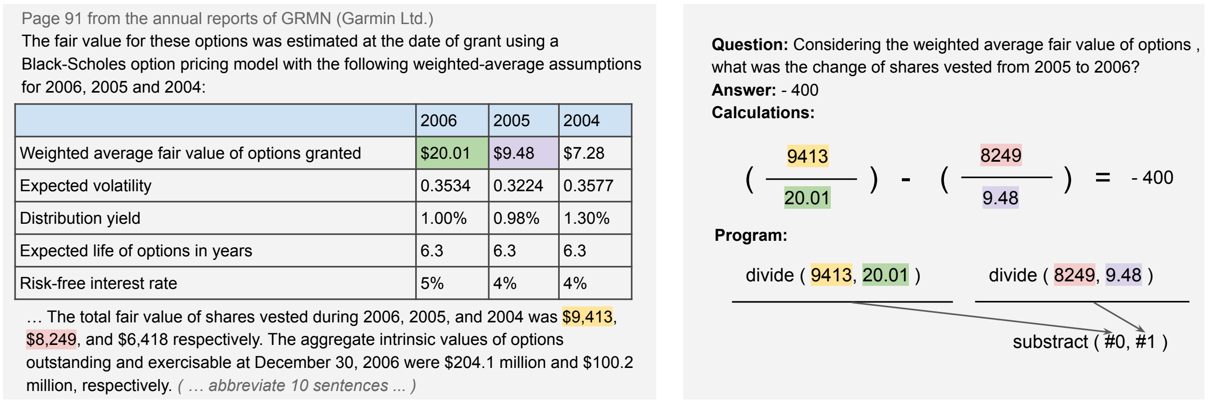
65.05%. Although our system outperforms the non-expert crowd (50.68%), the significant accuracy gap between the model and human experts (91.16%) motivates the need for future research.

The main contribution of this work is three-fold:

- We propose the task of QA over financial data to assist financial analysis. The task emphasizes an important phenomenon for the NLP community to study and analyze how the current pre-trained models perform on complex and specialized domains.

- We construct a new large-scale dataset, FinQA, with 8,281 examples written by financial experts, with fully annotated numerical reasoning programs.

- We experiment on various baselines and find that the models are still far behind expert performance, strongly motivating future research.



Page 91 from the annual reports of GRMN (Garmin Ltd.)

The fair value for these options was estimated at the date of grant using a Black-Scholes option pricing model with the following weighted-average assumptions for 2006, 2005 and 2004:

	2006	2005	2004
Weighted average fair value of options granted	$20.01	$9.48	$7.28
Expected volatility	0.3534	0.3224	0.3577
Distribution yield	1.00%	0.98%	1.30%
Expected life of options in years	6.3	6.3	6.3
Risk-free interest rate	5%	4%	4%

… The total fair value of shares vested during 2006, 2005, and 2004 was $9,413, $8,249, and $6,418 respectively. The aggregate intrinsic values of options outstanding and exercisable at December 30, 2006 were $204.1 million and $100.2 million, respectively. *(… abbreviate 10 sentences ...)*

Question: Considering the weighted average fair value of options , what was the change of shares vested from 2005 to 2006?

Answer: - 400

Calculations:

$$\left(\frac{9413}{20.01}\right) - \left(\frac{8249}{9.48}\right) = -400$$

Program:

divide (9413, 20.01) divide (8249, 9.48)

substract (#0, #1)

Figure 2.1: An example from FinQA: The system needs to learn how to calculate the number of shares, then select relevant numbers from both the table and the text to generate the reasoning program to get the answer.

2.2 Related Work

Questions Answering. There have been several QA datasets involving numerical understandings and calculations. The major source is from structured tables or knowledge bases, owning the nature to succinctly organize numerical information. Popular datasets include ComplexWebQuestions [41], WikiTableQuestions [42], Spider [43], TabFact [44], etc. For reading comprehension, the dataset most related to ours is the DROP dataset [10], which applies simple calculations over texts. The top methods on DROP typically use specific prediction heads for each kind of calculation. HybridQA [45] targets QA over both the table and the text, but not with the focus of numerical reasoning. All these existing datasets are built upon the general domain (mostly based on Wikipedia). In contrast, our dataset focus on the finance domain, which demonstrates much more complex nature in numerical reasoning questions, combining both the structured tables and unstructured texts. Another kind of QA datasets related to ours is the math word problem datasets, like MaWPS [46], MathQA [11]. The task is to generate the solution programs given a short input math problem. Existing models include [47, 48, 49], etc.

Financial NLP. Financial NLP has become one of the major application domains attracting growing attentions. Previous works in finance domain include risk management to detect fraud [50, 51, 52], sentiment analysis to assist market prediction [53, 54, 55], opinionated Question Answering [56], such as the FiQA[1] dataset built from forums and social media. Recent works attempt to develop pre-trained models specialized for finance domain [57, 58], and the downstream tasks are mostly sentiment classifications. To the best of our knowledge, there is no previous work and dataset on building QA systems of numerical reasoning on financial reports.

[1]https://sites.google.com/view/fiqa/home

2.3 Task Definition

Problem Formulation. Presented with a financial report consisting of textual contents E and structured table T, given a question Q, the task is to generate the reasoning program $G = \{w_0, w_1, ...w_n\}$, where w_i is the program tokens defined by domain specific language (DSL), then it is executed to get the answer A:

$$P(A|T, E, Q) = \sum P(G_i|T, E, Q) \tag{2.1}$$

Where $\{G_i\}$ is all the correct programs to evaluate to the answer. For financial tables, there is typically a description header (blue header in Figure 2.1), which often gives the timing information; and each row has its name on the left. Some of the financial tables may demonstrate more complicated layouts, *e.g.*, nested structures. As a first step for this direction, in this chapter we only focus on the regular layout cases for simplicity.

Domain Specific Language. In this work, we use DSL consisting of mathematical operations and table operations as executable programs. The program consists of a sequence of operations:

$$\text{op}_1[\text{args}_{\mathbf{1}}], \text{op}_2[\text{args}_{\mathbf{2}}]..., \text{op}_n[\text{args}_{\boldsymbol{n}}] \tag{2.2}$$

Each operation takes a list of arguments $\boldsymbol{args_n}$. On consulting with financial experts, as most of the accounting and financial valuation theory primarily include linear algebra, we include 10 common types of operations in our dataset. There are 6 mathematical operations: `add`, `subtract`, `multiply`, `divide`, `greater`, `exp`, and 4 table aggregation operations `table-max`, `table-min`, `table-sum`, `table-average`, that apply aggregation operations on table rows. The mathematical operations take arguments of either numbers from the given reports, or a numerical result from a previous step; The table operations take arguments of table row names. We use the special token $\#n$ to denote the result from the nth step. For example, in Figure 2.1, the program consists of 3 steps; The first and the second division steps take arguments from the table and the text, respectively, then the third step subtracts the results from the two previous steps. Refer to A for more details of the operations and the grammars.

Evaluations. Previous studies on QA with numerical reasoning only evaluate the execution accuracy, i.e., the final results from the generated programs, such as DROP [10] and MathQA [11]. However, the applications for the finance domain generally pose much higher requirements of explainability and transparency. Therefore, we also provide the gold programs for our dataset. Besides execution accuracy, we also propose to evaluate the accuracy of the generated programs. Specifically, we replace all the arguments in a program with symbols, and then we evaluate if two symbolic programs are *mathematically equivalent*. For example, the following two programs are equivalent programs:

$$\text{add}(a_1, a_2), \text{add}(a_3, a_4), \text{subtract}(\#0, \#1)$$
$$\text{add}(a_4, a_3), \text{add}(a_1, a_2), \text{subtract}(\#1, \#0)$$

Note that execution accuracy tends to overestimate the performance because sometimes the model just hit the correct answer by chance; While program accuracy tends to produce false negatives since some questions may have multiple correct programs.

2.4 The FinQA Dataset

2.4.1 Data Preparation

Data Source. We develop FinQA based on the publicly available earnings reports of S&P 500 companies from 1999 to 2019, collected in the FinTabNet dataset [38]. An earnings report is a set of pages in a PDF file that outlines the financials of a company, which usually contains tables and texts. The FinTabNet dataset has annotated the tables in each report.

Data Filtering. Realistic earnings reports contain many tables not suitable for numerical reasoning tasks. Equipped with the table annotations in FinTabNet, we filter the data as follows: First, we extract the pages in earnings reports with at most one table. Second, we exclude the tables with over 20 rows, over 2 description headers, or with other complex nested structures. We also exclude the tables with tedious contents, such as catalogs, which is common in FinTabNet. As stated in §2.3, these over-complicated tables are out of the scope of this work. Finally, for the tables with 2 description headers, we merge them into a single header to simplify the representations. As a result, a total of 12,719 pages were selected for further annotation.

2.4.2 Annotation Procedure

Recruiting Expert Annotators. We post job ads on UpWork[2] and hire eleven US-based experts with professional finance backgrounds (CPAs, MBAs, etc.) Each hire is interviewed using four example report pages and asked to compose example Q&A pairs. After hiring, each annotator first goes through a training session to learn the task and the annotation interface (A). When the workers fully master the annotation process, we

[2]UpWork (www.upwork.com) is a platform where requesters can recruit skilled freelancers.

launch the official batches for them to work on.

An annotator can compose up to two questions for each given report page or skip if it is hard to compose any meaningful question. We pay around $2.0 for each question, which leads to an average hourly wage of $35.0. The whole data collection took around eight weeks.

We do not use popular micro-task platforms, such as Amazon Mechanical Turk (MTurk), because our preliminary studies show that many MTurk workers can not perform this task effectively. Our experiment with MTurk workers in § 2.4.3 further echo this observation. As most existing QA datasets were constructed by MTurk workers [37, 10, 45], it requires substantial domain-specific knowledge to compose meaningful questions that are hard for computers to answer.

Annotation Task Design. For each page selected in §2.4.1, the annotators are asked to *(i)* write a meaningful financial question, *(ii)* compose a reasoning program to answer the question, and *(iii)* to annotate the supporting fact. Each page is assigned to one or two experts for annotation. We detail each part as follows. **(I) Financial question:** For a given page of earnings reports, the annotators are asked first to compose a question that is "meaningful for financial analysis or learning insights of the company financial reports" and require numerical calculations to answer. We encourage the experts to write questions that require the information from both the text and the table to answer. **(II) Reasoning program:** After providing the question, the annotators are then asked to elaborate the operation steps to answer the question. Specifically, they compose a maximum of 5 steps of operation, where each operation has four slots: "operation", "argument1", "argument2", and "result". The "operation" is one of the ten predefined operations described in §2.3. An "argument" is a number or a table's row name, either from the report or a previous step's result. For operations that only use one argument, such as table aggregation, workers can leave argument2 blank. The annotation interface (see A) automatically validates the inputs to ensure correctness. **(III) Supporting fact:** We also ask the annotators to mark all the sentences in the text and the table rows that contain the information needed to answer the question.

2.4.3 Data Quality Assessment

External experts answer FinQA questions with a high accuracy and a high inter-annotator agreement. To validate the quality of the annotations, as well as to set up human expert performance upper bound, we hire another two financial professionals on UpWork. We randomly sample 200 examples from our dataset, and ask the professionals to answer the questions as well as write the operation steps, following the same procedure as in the dataset construction. The payment is $2.0 per question. For execution accuracy, they reach 92.25% and 90.06%, respectively (mean = 91.16%). For program accuracy, they reach 89.44% and 85.53% (mean = 87.49%). The agreements between the two annotators are 92.65% for execution accuracy, and 86.76% for program accuracy.

Non-expert crowd workers answer FinQA questions with a low accuracy. We also test how well non-expert MTurk workers can answer FinQA questions. We distribute the samples to MTurk[3] and take the similar process to distribute each example to two workers. We end up with an average execution accuracy of 50.68% and a program accuracy of 48.17%, which is far below the expert performance; the agreement rate is only around 60%. These results echo our preliminary study's observations for MTurk workers in §2.4.2.

[3]Three built-in worker qualifications are used: HIT Approval Rate ($\geq$95%), Number of Approved HITs ($\geq$ 3000), and Locale (US Only) Qualification. We do not select any profession constraints. We pay $2.0 for each question.

Examples (Q&A pairs with program, fact)	8,281
Report pages	2,789
Vocabulary	22.3k
Avg. # sentences in input text	24.32
Avg. # tokens in input text	628.11
Avg. # rows in input table	6.36
Avg. # tokens in input table	59.42
Avg. # tokens in all inputs (text & table)	687.53
Max. # tokens in all inputs (text & table)	2,679
Avg. question length	16.63

Table 2.1: Statistics of FinQA.

2.4.4 Data Analysis

FinQA contains 8,281 examples. The data is released as training (6,251), validation (883), and test (1,147) following an 75%/10%/15% split. The three sets do not have overlapping input reports. We quantitatively analyze some key properties of FinQA. Table 2.1 shows the general statistics.

Statistics of Supporting Facts. In **FinQA**, 23.42% of the questions only require the information in the text to answer; 62.43% of the questions only require the information in the table to answer; and 14.15% need both the text and table to answer. Meanwhile, 46.30% of the examples have one sentence or one table row as the fact; 42.63% has two pieces of facts; and 11.07% has more than two pieces of facts. For the examples with more than one piece of fact, we also calculate the maximum distances between all the same example's facts. 55.48% has a maximum distance of 3 or less sentences[4]; 24.35% has a maximum distance of 4-6 sentences; and 20.17% has over 6 sentences.

Statistics of Reasoning Programs. In the programs, the most frequent operations, `add`, `subtract`, `multiply`, and `divide`, have the distributions of 14.98%, 28.20%, 5.82%,

[4]For tables, we consider one row as one "sentence".

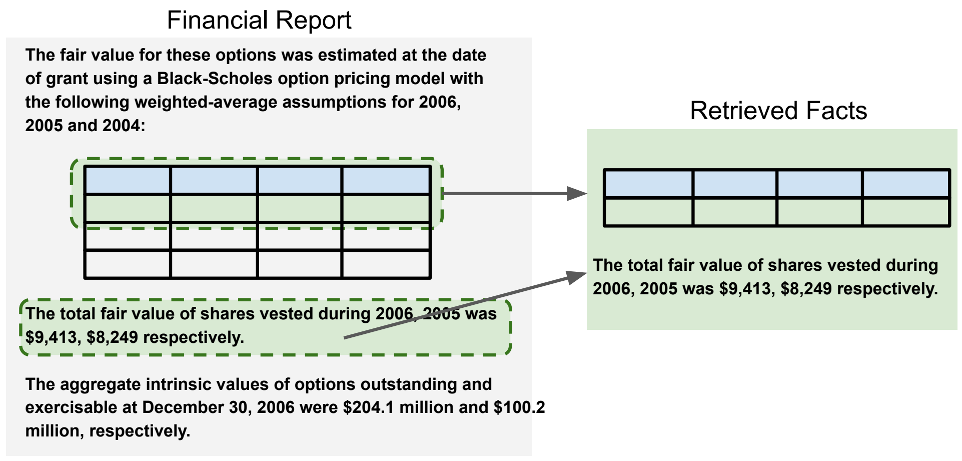

Financial Report
The fair value for these options was estimated at the date of grant using a Black-Scholes option pricing model with the following weighted-average assumptions for 2006, 2005 and 2004:
The total fair value of shares vested during 2006, 2005 was $9,413, $8,249 respectively.
The aggregate intrinsic values of options outstanding and exercisable at December 30, 2006 were $204.1 million and $100.2 million, respectively.
Retrieved Facts
The total fair value of shares vested during 2006, 2005 was $9,413, $8,249 respectively.

Figure 2.2: The retriever retrieves supporting facts (text sentences or table rows) from the input financial report.

and 45.29%, respectively. The operation `division` has the highest frequency, as calculating ratios is common in financial analysis. In **FinQA**, 59.10% of the programs have 1 step, 32.71% have 2 steps, and the rest 8.19% have 3 or more steps.

2.5 Baseline Systems

In this section, we first describe our main baseline framework **FinQANet** in §2.5.1, and then we introduce other baselines in §2.5.2.

2.5.1 The FinQANet Framework

As a preliminary attempt on FinQA, we propose **FinQANet**, with a retriever to first retrieve the supporting facts from the input financial report, then a generator to generate the program to get the answer.

Retriever The full page of the financial report can go beyond 2,000 tokens, which cannot be coped with the current popular QA models [39]. Therefore we first retrieve the supporting facts from the input report. For the tables, we use templates to turn each

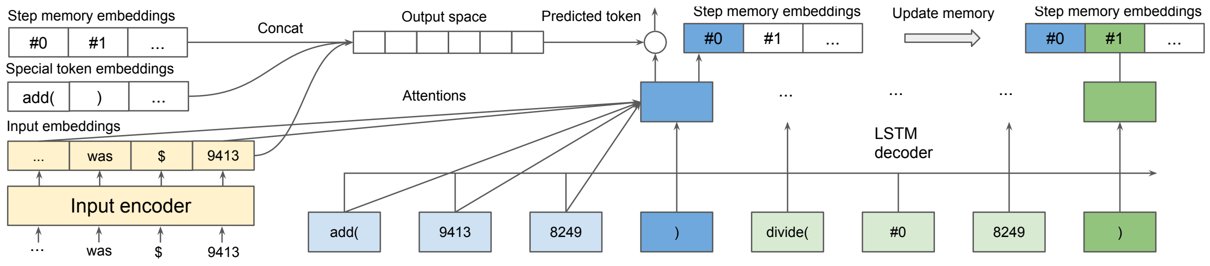


Figure 2.3: The program generator. The retriever results and the question are first encoded using pre-trained LMs. At each decoding step, the model can generate from the numbers or table row names from the input, the special tokens in the DSL, or the step memory tokens. At the end of the generation of each operation step, we update the step memory token embeddings.

row into sentences. For example, the last row of the table in Figure 2.1 is represented as 'the risk-free interest rate of 2006 is 5%; ...'. We concatenate each supporting fact with the question and train a classifier using pre-trained LMs like BERT [39]. Then we take the top n retrieved facts, reordered as they appear in the input report. This set of retriever results will serve as the input to the second phase. Figure 2.2 illustrates the retrieving procedure. Another common strategy is sliding window [59]. We take the sliding window of a fixed size with a stride to go through the report, then the windows containing all the supporting facts are marked as positive. However, we observe in the experiments that the length of the input to the program generator in the second phase greatly influences the performance. The performance of using sliding window falls far behind the previous method.

Program Generator Given the retrieved supporting facts from the retriever, the program generator aims to generate the executable program to answer the question. Figure 2.3 gives an overview of the program generator. The generated tokens come from 3 sources: 1) The input passage (retriever output) and the question tokens $\{e_i\}$, like the numbers or the table row names. 2) The special tokens $\{s_i\}$ from the DSL, like the function names, predefined constants, etc. 3) The step memory tokens $\{m_i\}$ to denote the results from previous steps, like #0, #1 , etc. We first use pre-trained LMs to encode $\{e_i\}$, denote the output embeddings as $\{h_i^e\}$. The embeddings of the special tokens and the step memory tokens are randomly initialized and denoted as $\{h_i^s\}$ and $\{h_i^m\}$ respectively. Denote all the token embeddings $H = [h_i^e; h_i^s; h_i^m]$.

An LSTM is used for decoding. At each decoding step T, the program token embeddings H are fed as the input; The decoder output h_T is used to calculate the attention vector att_p and att_h over the input and the decoding history. Then a context vector c_T combines all the contextual information:

$$c_T = W_c[att_p; att_h; h_T] \tag{2.3}$$

Meanwhile, another attention vector $att_p^{'}$ over the input is applied to all the token embeddings:

$$H_T^{'} = W_h[H; H \circ att_p^{'}] \tag{2.4}$$

Different from other program tokens, the step memory tokens $\{m_i\}$ imply the reasoning path of the program. To make use of such structure information, at each decoding step indicating the end of one $operation[args]$ unit, i.e., the step to generate the ending parentheses in our DSL, we compute another context vector a_T:

$$a_T = W_a[att_p; att_h; h_T] \tag{2.5}$$

Then the step memory token embedding corresponding to the current step is updated as a_T.

The final prediction is calculated with:

$$w_T = softmax(H_T^{'} \cdot c_T) \tag{2.6}$$

During inference time, based on the grammar of the DSL, we use masks at each decoding step to ensure the structural correctness of the generated programs. In the retriever phase, we take the top n retrieved results as the input to the program generator. Therefore, for the training of the program generator, we use the retriever result on the training set (combined with the gold facts if there is any wrong prediction) as the input.

2.5.2 Other Baselines

TF-IDF + Single Op. We use TF-IDF to retrieve the top 2 sentences from the input report. Since the most common case in our dataset is one-step program and the most common operation is division, we take the first number from each sentence and apply the division operation.

Retriever + Direct Generation. To demonstrate the necessity of generating the reasoning programs, we keep the architecture the same as our model, but directly generating the final results.

Retriever + Seq2seq. We use a Seq2seq architecture for the generator, similar to the Seq2seq baseline in the MathQA dataset [11]. A bi-LSTM is used for encoding the input, and then an LSTM is used for decoding with attention.

Retriever + NeRd. The Neural Symbolic Reader(NeRd) [49] is also a pointer-generator based model for program generation, with the state of the art results on the MathQA dataset [11]. Different from ours, it directly learns the program with nested format as a sequence, i.e., without the step memory tokens. This way the model is able to learn the program structures as patterns from very large-scale data (˜40k for MathQA), but may fail on learning the reasoning paths. We keep the retriever part the same and compare with the generator part to demonstrate the usefulness of structure learning.

Pre-Trained Longformer. There are also works on modeling very long documents with thousands of characters, with the attention mechanism that scales linearly with sequence length, like the Longformer [60]. To demonstrate the necessity of breaking up into the pipeline of retriever and program generator, we remove the retriever and directly use the pre-trained Longformer as the input encoder in the program generator, and encode the whole report. The table rows are linearized similar as in §2.5.1.

2.6 Experiments

2.6.1 Experiment Setups.

For the retriever, we use BERT-base as the classifier (other pre-trained models perform similarly). Since most of the examples in our dataset have 1 or 2 facts, and we find that longer inputs lower the performance of the program generator, we take the top 3 ranked facts as the retriever results. For the program generator, we experiment on using BERT [39], RoBERTa [40], and FinBert [58] as the encoder, to test the performances of popular large pre-trained models. For all models, we use the Adam optimizer [61]. Check A for more details of training and parameter settings.

2.6.2 QA Model Performance

Table 2.2 presents the results for all the baseline systems. We evaluate the execution accuracy (exe acc) and program accuracy (prog acc) as explained in §2.3. For the BERT-based retriever, we have 89.66% recall for the top 3 retrieved facts and 93.63% recall for the top 5. Using TF-IDF results in 82.91% recall for the top 5 facts. We use the

Baselines	**Exe Acc**	**Prog Acc**
TF-IDF + Single Op	1.01	0.90
Retriever + Direct Generation	0.30	-
Pre-Trained Longformer (base)	21.90	20.48
Retriever + Seq2seq	19.71	18.38
Retriever + NeRd (BERT-base)	48.57	46.76
FinQANet (FinBert)	50.10	47.52
FinQANet (BERT-base)	50.00	48.00
FinQANet (BERT-large)	53.52	51.62
FinQANet (RoBERTa-base)	56.10	54.38
FinQANet (RoBERTa-large)	**61.24**	**58.86**
FinQANet-Gold (RoBERTa-large)	70.00	68.76
Human Expert Performance	91.16	87.49
General Crowd Performance	50.68	48.17

Table 2.2: The execution accuracy (Exe Acc) and program accuracy (Prog Acc) for all the models. Although our best system (61.24%) outperforms the non-expert crowd (50.68%), the significant accuracy gap between the model and human experts (91.16%) motivates the need for future research.

same retriever results for all retriever-generator based models. Directly generating the execution results gives near-zero scores, which indicates the necessity of generating the reasoning programs. If without using the retriever-generator pipeline, but directly applying an end-to-end pre-trained Longformer model, the performance falls far behind. Because longer inputs have more numbers which put more confusions on the program generator and thus make it harder to learn. Generally, the program generators using pre-trained models perform much better than the Seq2seq baseline, as there is language modeling knowledge that can also be used for the finance domain. And larger pre-trained models give better performance, as they tend to see more financial corpus during their pre-training. FinBert [58] is a pre-trained model for the finance domain; its main downstream tasks are sentiment analysis. The performance of using FinBert is no better

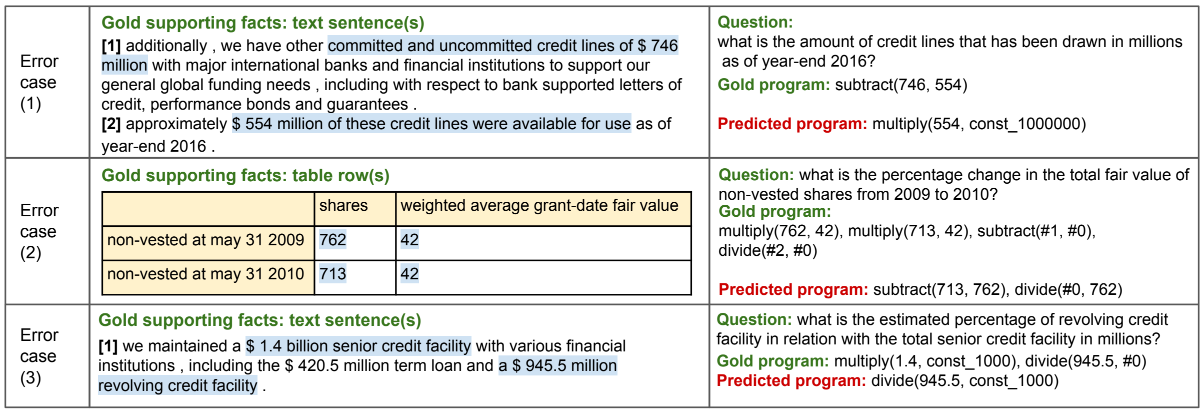
Methods	**Exe Acc**	**Prog Acc**
full results	**61.24**	**58.86**
Necessity of table and text		
table-only inference	45.81	43.62
text-only inference	15.80	15.33
Performances on table and text		
table-only questions	67.38	64.48
text-only questions	54.86	53.70
table-text questions	43.80	41.61
Performances regarding program steps		
1 step programs	67.61	65.28
2 step programs	59.08	56.37
¿2 step programs	22.78	21.52
Programs with constants	43.88	39.80

Table 2.3: Performance breakdown of FinQANet (RoBERTa-large). The model benefits from using both table and text, as inferences on individual source yield much lower performance. FinQANet is better at answering table-only questions, and the questions that require more steps to solve are indeed more challenging to the model.

than BERT-large, mostly because its pre-training corpus is limited (˜30M words from news articles). Comparing FinQANet with the retriever + NeRd baseline [49], it shows the improvements from learning the logical structure of the programs. We also run the program generator using the gold retriever result, shown as FinQANet-Gold. Another interesting observation is the comparisons with human performances. While there is still a large gap from the human expert upper bound, the best performing model already surpasses the general crowd performance.

2.6.3 Performance Breakdown

We conduct a set of performance breakdowns using the FinQANet (RoBERTa-large) model. Table 2.3 shows all the results.



Error case (1)

Gold supporting facts: text sentence(s)

[1] additionally , we have other committed and uncommitted credit lines of $ 746 million with major international banks and financial institutions to support our general global funding needs , including with respect to bank supported letters of credit, performance bonds and guarantees .
[2] approximately $ 554 million of these credit lines were available for use as of year-end 2016 .

Question: what is the amount of credit lines that has been drawn in millions as of year-end 2016?
Gold program: subtract(746, 554)
Predicted program: multiply(554, const_1000000)

Error case (2)

Gold supporting facts: table row(s)

	shares	weighted average grant-date fair value
non-vested at may 31 2009	762	42
non-vested at may 31 2010	713	42

Question: what is the percentage change in the total fair value of non-vested shares from 2009 to 2010?
Gold program: multiply(762, 42), multiply(713, 42), subtract(#1, #0), divide(#2, #0)
Predicted program: subtract(713, 762), divide(#0, 762)

Error case (3)

Gold supporting facts: text sentence(s)

[1] we maintained a $ 1.4 billion senior credit facility with various financial institutions , including the $ 420.5 million term loan and a $ 945.5 million revolving credit facility .

Question: what is the estimated percentage of revolving credit facility in relation with the total senior credit facility in millions?
Gold program: multiply(1.4, const_1000), divide(945.5, #0)
Predicted program: divide(945.5, const_1000)

Figure 2.4: Error cases. In these examples, the retriever results all correctly cover the gold facts; thus we only present the gold facts, gold program, and the predicted program to study the errors of the program generator. We give more error cases in A, including the cases for the retriever errors. **Example 1**: The financial knowledge to calculate the 'credit lines that has been drawn'. **Example 2**: Complex reasoning of 4 steps. **Example 3**: Number unit conversion between 'billion' and 'million'.

Necessity of using both table and text. We run inferences taking facts only from a single source from the retriever. Inferences on individual source (table-only: 45.81%, text-only: 15.80%) are both far behind the full results (61.24%).

The model performs the best on the table-only questions. The model performs the best on table-only questions (67.38%). Tables tend to have more unified structures and might be easier for the model to learn. Table 2.3 also shows that the questions involving both tables and texts are the most challenging ones for the model (43.80%).

Questions that need more than two steps to answer are challenging. The model has a low accuracy (22.78%) on the questions that need three or more steps. Meanwhile, not surprisingly, the questions that require only one step are the easiest.

Constants in programs. Many programs in **FinQA** contain constants as arguments. A constant is often used to convert an English number word to another. For example, we need first to use the constant "1,000" to convert "1.5 billion" to "1,500 million" so that it can be added with “50 million”. A constant is also used to explicate the implicit numbers hidden in the language. For example, to calculate “the average for the year 2012, 2013, and 2014”, the program needs to use the constant “3” as the denominator, which is not mentioned explicitly in the text. As shown in Table 2.3, the programs with constants yield great challenges for our model, as the performance (43.88%) is much lower than that of the whole set (61.24%).

2.6.4 Error Analysis

We sample 50 error cases from the results of the FinQANet (RoBERTa-large) model and analyze them manually. 15% of the errors are caused by the retriever, *e.g.*, missing facts. Half of the rest are due to the lack of financial knowledge, such as the meaning of some terminology. And the rest half are primarily numerical reasoning errors, including complex programs with multiple steps, numerical unit conversions, or resolving the ordering and matching of the numbers and the years. Many error cases involve both the numerical reasoning problems and misunderstandings of financial knowledge. We show three representative error cases in Figure 2.4.

Chapter 3

Representation Learning for Fine-Grained User Preference

3.1 Introduction

Conversational artificial intelligence is one of the long-standing research problems in natural language processing, such as task-oriented dialogue [62, 63, 64], conversational recommendation [65, 66] and chi-chat [67, 68] etc. However, most existing systems are *agent-centric*. Such systems require the users to *unnaturally* adapt to and even have a learning curve on the system ontology, which is largely unknown to the users (such as the sample instructions for most smart speakers). Figure 3.1 shows a dialogue snippet commonly found in traditional datasets: the user is *expected* to closely follow the system ontology with the exact ontology terms, or at most with minor variations like synonyms.

In the real-world use cases, such formulation may easily results in information loss, or breaks a conversation if the user speaks anything out of the system ontology; In this work, we argue that a smart agent can ideally be more *user-centric*, by allowing users to speak freely without restrictions. The system is expected to uncover the connection

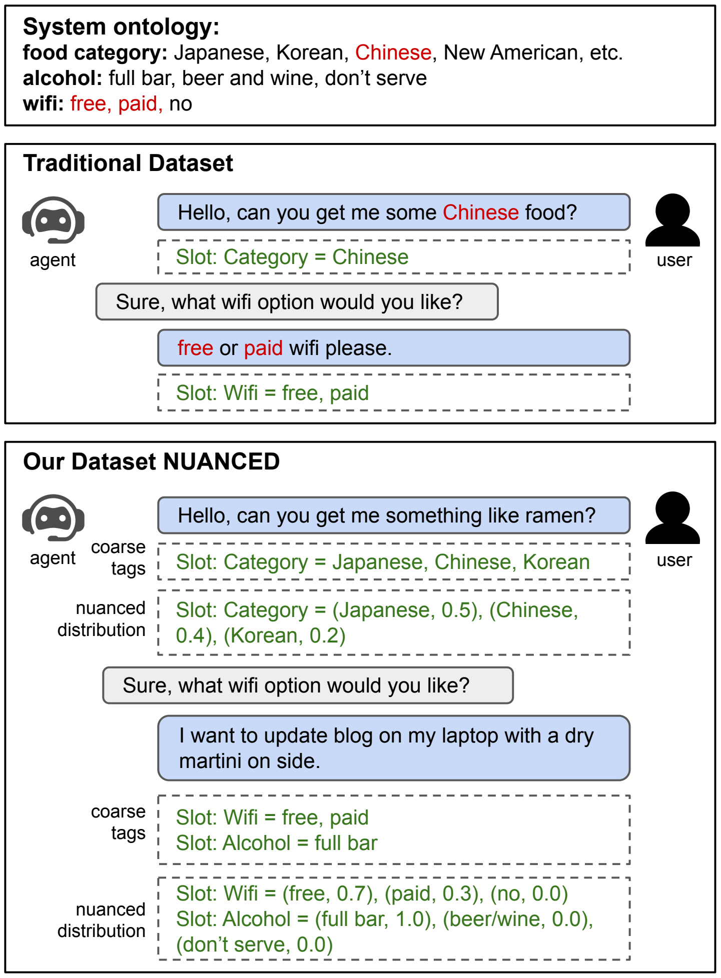

System ontology:
food category: Japanese, Korean, Chinese, New American, etc.
alcohol: full bar, beer and wine, don't serve
wifi: free, paid, no
Traditional Dataset
agent
user
Hello, can you get me some Chinese food?
Slot: Category = Chinese
Sure, what wifi option would you like?
free or paid wifi please.
Slot: Wifi = free, paid
Our Dataset NUANCED
agent
user
Hello, can you get me something like ramen?
coarse tags
Slot: Category = Japanese, Chinese, Korean
nuanced distribution
Slot: Category = (Japanese, 0.5), (Chinese, 0.4), (Korean, 0.2)
Sure, what wifi option would you like?
I want to update blog on my laptop with a dry martini on side.
coarse tags
Slot: Wifi = free, paid
Slot: Alcohol = full bar
nuanced distribution
Slot: Wifi = (free, 0.7), (paid, 0.3), (no, 0.0)
Slot: Alcohol = (full bar, 1.0), (beer/wine, 0.0), (don't serve, 0.0)

Figure 3.1: Examples of traditional dataset and NUANCED: In NUANCED, we model the user preferences as distributions over the ontology to allow mapping of entities unknown to multiple values and slots for efficient conversation.

between the freestyle user utterance and one or more slots and values by the system ontology.

To build a *user-centric* dialogue system, we propose to model the mapping from the free form user utterances to the system ontology as probability distributions to capture fine-grained user preferences. To learn the distributions, we construct a new dataset, named NUANCED (**N**atural **U**tterance **A**nnotation for **N**uanced **C**onversation with **E**stimated **D**istributions). NUANCED targets conversational recommendation because such type of dialogue system encourages more modeling of soft matching and implicit reasoning for user preference. We employ professional linguists to annotate the dataset, and end up with 5.1k dialogues and 26k turns of high-quality user utterances. Our dataset captures a wide range of phenomena naturally occurring in realistic user utterances, including specified factoid knowledge, commonsense knowledge and users' own situations. We conduct comprehensive experiments and analyses to demonstrate the challenges. We hope NUANCED can serve as a valuable resource to bridge the gap between current researches and real-world applications.

3.2 Related Work

Task-oriented dialogue systems are typically divided into several sub modules, including user intent detection [69, 70], dialogue state tracking [19, 71], dialogue policy learning [72, 73], and response generation [74, 75]. More recent approaches begin to build unified models that bring the pipeline together [76, 64]. *Conversational recommendation* focus on combining the recommendation system with online conversation to capture user preference [77, 65, 66]. Previous works mostly focus on learning the agent side policy to ask the right questions and make accurate recommendations, such as [78, 79, 80, 81]. Chit-Chat [67, 68] is the most free form dialogue but almost with no knowledge grounding or state tracking. Both existing task-oriented, conversational recommendation systems have a pre-defined system ontology as a representation connected to the back-end database. The ontology defines all entity attributes as slots and the option values for each slot. In existing datasets, such as the DSTC challenges [82], Multi-WOZ [63], MGConvRex [78], etc, the utterances from the users mostly closely follow the system ontology. While in task-oriented dialogue systems, parsing the user utterances into dialogue states is more on hard matching, in conversational recommendation systems soft matching is more encouraged since the user preferences are more salient and diverse in this type of conversations.

3.3 The nuanced Dataset

3.3.1 User Preference Modeling

Given a system ontology, denote the set of all slots as $\{S_i\}$, with the option values for each slot as $\{V_i^j\}$. Denote the current user utterance as T and dialogue context (of past turns) as C. We model the user preference as a distribution over each slot-value, namely *preference distribution*:

$$P_i^j = P(V_i^j | T, C). \tag{3.1}$$

Note that we expect the representation to be general, expandable, and to hold the fewest assumptions, i.e., there is no assumption on the dependency among slot-values, nor the completeness of the value set. Therefore we model the distribution as a Bernoulli distribution over each slot-value. Intuitively, P_i^j represents the probability that the user chooses an item with attributes V_i^j, under the observed condition of the dialogue up to the current turn. Note that the preference distributions may differ among individuals which causes variances, In this work, we aim to aggregate estimated distributions from large-scale data collected from multiple workers as “commonsense” distributions. We leave modeling user-specific distributions to future work.

3.3.2 Dataset Construction

We first simulate the dialogue flow with the preference distributions, then we ask the annotators to compose utterances that imply the distribution.

Dialogue Simulator We follow the approach from the MGConvRex dataset [78] to build the user visiting histories from real-world data. For each user with its visiting history as a list of restaurants with slot-values, we sample a subset of the history and aggregate to get a value distribution for each slot. For example, in the list of restaurants of a user's visiting history, we sampled two restaurants, restaurant 1 and restaurant 2. Restaurant 1 has the slot-values of Alcohol = full_bar, Restaurant 2 has the slot-value of Alcohol = beer_and_wine. Then the aggregated distributions is Alcohol = (full_bar, 0.5), (beer_and_wine, 0.5), (no_serve, 0.0). As generally, for the same user, the attributes of its visited restaurants tends to follow certain trends. Therefore the aggregated distributions created this way can be more natural. Using the sampled distribution as the ground truth distribution, we simulate the dialogue skeletons of the following scenarios: 1) Straight dialogue flow: the system asks each slot, followed by the user response filled with preference distributions; 2) User updating preference: the user updates the preference distributions in a previous turn; 3) System yes/no questions: the system can choose to ask confirmation questions; For each turn, we randomly select 1 to 3 slots, corresponding to the cases that the user utterances naturally imply multiple slot-values. The system turns are composed using templates.

User Utterances Composition After simulating the dialogue skeletons, we employ professional linguists to do the composition to ensure high quality. We provide two composing strategies: **Implicit Reasoning**: do not mention the slot-value terms explicitly. This is the focus of this work because we expect that users are unaware of the system ontology and to depict their requests naturally. **Explicitly Mention**: use the slot-value terms (or synonyms), as a backup option when the first one is not applicable. We also emphasize the following aspects: 1) Read the whole dialogue first to have an overall "story" in mind before composing each utterance to ensure consistency; 2) Try to compose utterances as diverse as possible; 3) Reject any cases with invalid or unnatural preference distributions. We provide learning sessions to linguists to ensure they all master the tasks.

3.3.3 Dataset Statistics and Analysis

With an average of 5.39 user turns per dialogue, we have 5,100 dialogues consisting of 25,757 user turns. The user utterances have an average length of 19.43 tokens. 84.7% of the utterances are composed using *implicit reasoning*; 6.5% of the utterances explicitly mention the ontology terms, and the rest use mixed strategies. The train / valid / test split is 3,600 / 500 / 1,000 in the number of dialogues, and 18,182 / 2,529 / 5,046 in the number of user turns. To evaluate the quality of our dataset, we randomly sample 500 examples and ask the linguistics whether a preference distribution is reasonable based on the corresponding utterance. We end up with a turn-level correctness rate of 90.2%.

Reasoning types	**Example user utterances**	**Example preference distributions**
Type I Factoid Knowledge (37.3%)	I really want a G&T or a Riesling, but I could also have a tonic water.	Slot: Alcohol = (full_bar, 0.7), (beer_and_wine, 0.2), (don't_serve, 0.1)
Type II Commonsense knowledge or User Situations (43.8%)	five to ten dollars, I don't want a place with people wearing ties, you know?	Slot: Price = (cheap, 0.6), (affordable, 0.4), (moderately_priced, 0.0), (expensive, 0.0) Slot: Attire = (casual, 1.0), (dressy, 0.0), (formal, 0.0)
Type III Mixed Type I & II (19.0%)	I want to update blog on my laptop, with a dry martini on side.	Slot: Wifi = (free, 0.7), (paid, 0.3), (no, 0.0) Slot: Alcohol = (full_bar, 1.0), (beer_and_wine, 0.0), (don't_serve, 0.0)

Table 3.1: Examples of reasoning types. Type I utterance: G&T is only served in a full bar, while Riesling is a kind of wine and tonic water does not require alcohol options. Type II utterance, 'place without people wearing ties' indicates casual attire, and 'five to ten dollars' indicates a price range of cheap or affordable. Type III utterance, we need both kinds of reasonings.

Among the utterances involving implicit reasoning, we summarize 3 basic reasoning types. The examples are shown in Table 3.1. **Type I (Factoid Knowledge)** is largely agreed on by people and is relatively stable. **Type II (Commonsense Knowledge or User Situations)** may not be formally defined. For example, a food item less than $10 is considered cheap. In many cases, such knowledge needs to be inferred from a situation described by users. **Type III (Mix of Type I and II)** may appear in a single utterance.

NUANCED-reduced We also provide a *reduced* variant called NUANCED-reduced, by discretizing the distributions for preference into binary numbers. For all slot-values with a positive preference distribution[1] we label them as 1.0, otherwise 0.0. This reduced variant does not have continuous probabilities to tell the nuanced differences but it still needs to map free form utterances to binary labels. We conduct human evaluation by asking the annotators to decide which representation can better capture more fine-grained user preferences. As Table 3.2 shows, NUANCED can better capture the nuanced information. Note that in real applications, which version of the data to use may depend on requirements of the system, i.e., level of granularity for state representation.

NUANCED win	**NUANCED-reduced win**	**Tied**
54.7%	16.7%	28.6%

Table 3.2: Human evaluation results of comparing two versions.

[1]In practice we set a threshold of 10%, because in the utterance composition stage a preference distribution lower than 10% is generally considered ignorable.

3.4 Experiments

In this section, we conduct experiments on both versions of the datasets in §3.4.1 and §3.4.2, respectively.

3.4.1 NUANCED-reduced

Baselines

Exact match & Random guess We follow the preceding system query to make slot prediction; we then use an exact match to predict the slot-values; if no match is found, we apply a random guess.

BERT [39], The input is the concatenation of the slot name, current turn system question and user utterance, and the dialogue context of past turns. We add two types of prediction heads on the `[CLS]` token of BERT, one for slot prediction (whether the input slot is updated or not), and the other for the value prediction of each slot. The loss is a combination of cross-entropy loss for slot prediction and mean squared error (MSE) loss for value prediction. During inference, we set up a threshold to decide positive or negative predictions.

Transformer [83] We use the similar architecture as the BERT baseline but train the weights from scratch.

Train-ConvRex As MGConvRex dataset [78] has similar domain and ontology, we compare the BERT model trained on MGConvRex[2] with that tested on NUANCED-reduced. We use this baseline to demonstrate the open challenges caused by users' free-form speaking.

We refer the readers to B for more details. For all baselines, we evaluate on the turn level slot prediction accuracy and joint accuracy.

[2]We contacted the first author to obtain the dataset.

Results for NUANCED-reduced As shown in Table 3.3, the BERT model achieves the best performance as the external knowledge obtained from pre-training helps draw a better relevance between unrecognized entities from the user and entities from the agent. Train-ConvRex limits such mapping to system ontology, indicating that existing dialogue datasets may limit what an agent can understand from users. Lastly, by comparing with BERT without dialogue context (or past turns), we notice that context may help in learning better values but yields more noise for slot prediction.

Baselines	**Slot Accuracy (%)**	**Joint Accuracy (%)**
Exact match & Random guess	48.83	4.84
Train-ConvRex	38.70	4.02
Transformer	74.14	21.52
BERT	88.21	36.56
BERT w/o context	88.78	34.99

Table 3.3: Results on NUANCED-reduced. *Slot Accuracy*: percentage of turns that all slots are correct; *Joint Accuracy*: percentage of turns that all slots and values are correct.

3.4.2 NUANCED

Baselines

Exact match & Random guess Similar to NUANCED-reduced, we assign a probability of 1.0 for matched values or random value otherwise.

BERT, Transformer Similar to NUANCED-reduced, we use MSE loss between the ground truth and the predicted distribution.

Train-reduced-X We train the model on NUANCED-reduced and test on NUANCED to see how data with binary states can infer states in the continuous space. We define a fixed number of X as the continuous number for all positive predictions. We experiment with X = 0.5 and 1.0.

We keep the same evaluation for slot prediction. For value predictions, we evaluate the *soft* average mean absolute error (MAE) between the ground truth distribution and the predictions.

Results for NUANCED

As in Table 3.4, BERT reaches the best performance. One interesting observation is that using the same model BERT, the slot prediction accuracy increases (from 88.21% to 89.62%) compared with training on the reduced version. NUANCED helps to reduce the noise of sparse entities in context (past turns). This is probably because numbers in continuous space can draw more relevance among different entities. As we can see, Train-reduced-X has a much larger error. This indicates that simply adapting the results from the reduced state labels suffers from information loss, i.e., the nuanced differences in continuous distributions.

Baselines	**Slot Accuracy (%)**	**Correct slots mean MAE (1e-2)**
Exact match & Random guess	48.83	46.84
Train-reduced-1.0	88.21	40.72
Train-reduced-0.5	88.21	21.62
Transformer	78.42	16.78
BERT	89.62	14.20
BERT w/o context	88.08	14.49

Table 3.4: Evaluation results on NUANCED. *Correct slots mean MAE* (lower the better): mean absolute error of predicted distribution for all correctly predicted updated slots;

Analysis on Slots We study how the models perform on different kinds of turns, shown in Table 3.5. Generally speaking, the turns with more slots are relatively harder to learn. The turns that update the preference in previous turns have the highest error, the preference distribution needs to be jointly inferred from the previous mention and the current turn. We also study the performance on each slot in B, and provide some case studies in B.

Type of turn	**all**	**1 slot**	**2 slots**	**3 slots**	**updating preferences**
Slot Accuracy(%)	89.62	96.67	78.91	67.65	90.61
Mean MAE(1e-2)	14.12	14.06	13.55	14.20	15.63

Table 3.5: Performance for different kinds of slots: *all*: all kinds of turns; *n slots*: turns that the user utterance jointly implies n slots; *updating preferences*: turns that the user utterance updates the preference in previous turns.

Human Evaluation We further conduct a human evaluation on baseline models. We first evaluate the model outputs of Transformer, BERT, and BERT w/o context, through pairwise comparison between the model predictions and the gold labels. The results on 200 samples are shown in Table 3.6. There is a large gap between the best-performing baseline and the gold reference, which indicates significant room for improvement for future research. Further, we study the breakdown of predictions of BERT on 3 different types of reasoning. As shown in Table 3.7, the type 1 utterances, that involve factoid knowledge, are relatively harder to learn. This is close to our intuition because factoid knowledge is huge (and keeps increasing) and the limited utterances in the dataset may not cover all of the knowledge.

Methods	Model output win(%)	Tied(%)	Gold win(%)
Transformer	10	9.5	80.5
Bert	23.6	20.9	55.4
Bert w/o context	19.5	9.6	70.9

Table 3.6: Human evaluation results for the model predictions.

Methods	Model output win(%)	Tied(%)	Gold win(%)
Type I	22.5	19.9	57.6
Type II	27.4	24.1	48.5
Type III	21.1	11.2	67.7

Table 3.7: Human evaluation results for different reasoning types. Type I: factoid knowledge; Type II: commonsense knowledge or user situations; Type III: Mixed Type I & II.

Chapter 4

Few-Shot Natural Language Generation

4.1 Introduction

Natural language generation (NLG) from structured data or knowledge [84] is an important research problem for various NLP applications. Some examples are task-oriented dialog, question answering [85, 86, 87, 88, 89] and interdisciplinary applications such as medicine [90, 91] and health-care [90, 92]. There is great potential to use automatic NLG systems in a wide range of real-life applications. Recently, deep neural network based NLG systems have been developed, such as those seen in the E2E challenge [93], WeatherGov [94], as well as more complex ones such as WikiBio [26] and RotoWire [95]. Compared to traditional slot-filling pipeline approaches, such neural-based systems greatly reduce feature engineering efforts and improve text diversity as well as fluency.

Although they achieve good performance on benchmarks such as E2E challenge [93] and WikiBio [96], their performance depends on large training datasets, e.g., 500k table-text training pairs for WikiBio [96] in a single domain. Such data-hungry nature makes neural-based NLG systems difficult to be widely adopted in real-world applications as they have significant manual data curation overhead. This leads us to formulate an interesting research question:

> 1. *Can we significantly reduce human annotation effort to achieve reasonable performance using neural NLG models?*
> 2. *Can we make the best of generative pre-training, as prior knowledge, to generate text from structured data?*

Motivated by this, we propose the new task of *few-shot natural language generation*: given only a handful of labeled instances (e.g., 50 - 200 training instances), the system is required to produce satisfactory text outputs (e.g., BLEU $\geq$ 20). To the best of our knowledge, such a problem in NLG community still remains under-explored. Herein, we propose a simple yet very effective approach that can generalize across different domains.

In general, to describe information in a table, we need two skills to compose coherent and faithful sentences. One skill is to select and copy factual content from the table - this can be learned quickly by reading a handful of tables. The other is to compose grammatically correct sentences that bring those facts together - this skill is not restricted to any domain. One can think of a latent “switch” that helps us alternate between these two skills to produce factually correct and coherent sentences. To do this, we use the pre-trained language model [97, 98] as the innate language skill, which provides strong prior knowledge on how to compose fluent and coherent sentences. The ability to switch and select/copy from tables can be learned successfully using only a few training instances, freeing the neural NLG model from data-intensive training. Previous best performing methods based on large training data, such as [26], which does not apply such switch mechanism but trains a strong domain-specific language model, perform very poorly under few-shot setting.

Since we are operating under a highly data-restricted few-shot regime, we strive for simplicity of model architecture. This simplicity also implies better generalizability and reproducibility for real-world applications. We crawl multi-domain table-to-text data from Wikipedia as our training/test instances. With just 200 training instances, our method can achieve very reasonable performance.

In a nutshell, our contributions are summarized as the following:

- We propose the new research problem of few-shot NLG, which has great potential to benefit a wide range of real-world applications.
- To study different algorithms for our proposed problem, we create a multi-domain table-to-text dataset.
- Our proposed algorithm can make use of the external resources as prior knowledge to significantly decrease human annotation effort and improve the baseline performance by an average of over 8.0 BLEU on various domains.

Input Table

Attribute (R)	Value (V)
Name	Walter Extra
Nationality	German
Occupation	Aircraft designer and manufacturer
...	...

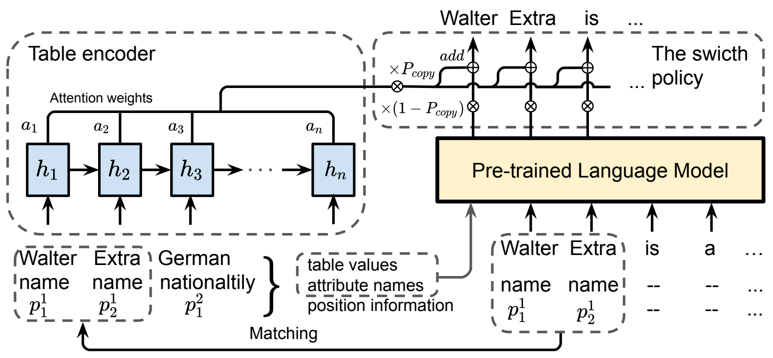


Figure 4.1: Overview of our approach: Under the base framework with switch policy, the pre-trained language model serves as the generator. We follow the same encoder as in [26]. The architecture is simple in terms of both implementation and parameter space that needs to be learned from scratch, which should not be large given the few-shot learning setting.

4.2 Related Work

4.2.1 NLG from Structured Data

As it is a core objective in many NLP applications, natural language generation from structured data/knowledge (NLG) has been studied for many years. Early traditional NLG systems follow the pipeline paradigm that explicitly divides generation into content selection, macro/micro planning and surface realization [25]. Such a pipeline paradigm largely relies on templates and hand-engineered features. Many works have been proposed to tackle the individual modules, such as [94, 99, 100]. Later works [101, 102] investigated modeling context selection and surface realization in an unified framework.

Most recently, with the success of deep neural networks, data-driven, neural based approaches have been used, including the end-to-end methods that jointly model context selection and surface realization [26, 27, 28]. Such data-driven approaches achieve good performance on several benchmarks like E2E challenge [93], WebNLG challenge [103] and WikiBio [96]. However, they rely on massive amount of training data. ElSahar et al. [104] propose zero-shot learning for question generation from knowledge graphs, but their work applies on the transfer learning setting for unseen knowledge base types, based on seen ones and their textual contexts, which still requires large in-domain training dataset. This is different from our few-shot learning setting. Ma et al. [105] propose low-resource table-to-text generation with 1,000 paired examples and large-scale target-side examples. In contrast, in our setting, only tens to hundreds of paired training examples are required, meanwhile without the need for any target examples. This is especially important for real-world use cases where such large target-side gold references are mostly hard to obtain. Therefore, our task is more challenging and closer to real-world settings.

4.2.2 Large Scale Pre-Trained Models

Many of the current best-performing methods for various NLP tasks adopt a combination of pre-training followed by supervised fine-tuning, using task-specific data. Different levels of pre-training include word embeddings [106, 107, 108], sentence embeddings [109, 110], and most recently, language modeling based pre-training like BERT [111] and GPT-2 [98]. Such models are pre-trained on large-scale open-domain corpora, and provide down-streaming tasks with rich prior knowledge while boosting their performance. In this chapter, we adopt the idea of employing a pre-trained language model to endow in-domain NLG models with language modeling ability, which cannot be well learned from few shot training instances.

4.3 Method

4.3.1 Problem Formulation

We are provided with semi-structured data: a table of attribute-value pairs $\{R_i : V_i\}_{i=1}^{n}$. Both R_i and V_i can be either a string/number, a phrase or a sentence. Each value is represented as a sequence of words $V_i = \{v_j\}_{j=1}^{m}$. For each word v_j, we have its corresponding attribute name R_i and position information of the word in the value sequence. The target is to generate a natural language description based on the semi-structured data, provided with only a handful of training instances.

4.3.2 Base Framework with Switch Policy

We start with the field-gated dual attention model proposed in [26], which achieves state-of-the-art performance (BLEU) on WikiBio dataset. Their method uses an LSTM decoder with dual attention weights. We first apply a switch policy that decouples the framework into table content selection/copying and language model based generation. Inspired by the pointer generator [112], at each time step, we maintain a soft switch p_{copy} to choose between generating from softmax over vocabulary or copying from input table values with the attention weights as the probability distribution.

$$p_{copy} = \text{sigmoid}(W_c c_t + W_s s_t + W_x x_t + b)$$

Where $c_t = \sum_i a_t^i h_i$, $\{h_i\}$ is the encoder hidden states, x_t, s_t, a_t is the decoder input, state and attention weights respectively at time step t. W_c, W_s, W_x and b are trainable parameters.

The pointer generator learns to alternate between copying and generating based on large training data and shows its advantage of copying out-of-vocabulary words from input. In our task, the training data is very limited, and many of the table values are not OOV. We need to explicitly "teach" the model where to copy and where to generate. Therefore, to provide the model accurate guidance of the behavior of the switch, we match the target text with input table values to get the positions of where to copy. At these positions, we maximize the copy probability p_{copy} via an additional loss term. Our loss function:

$$L = L_c + \lambda \sum_{\substack{w_j \in m \\ m \in \{V_i\}}} (1 - p_{copy}^j)$$

Where L_c is the original loss between model outputs and target texts. w_j is the target token at position j, $\{V_i\}$ is the input table value list defined in Section 4.3.1, and m means a matched phrase. λ is hyperparameter as the weight for this copy loss term. We also concatenate the decoder input with its matched attribute name and position information in the input table as x_t to calculate p_{copy} .

4.3.3 Pre-Trained LM as Generator

We use a pre-trained language model as the generator, serving as the "innate language skill". Due to the vocabulary limitation of few training instances, we leave the pre-trained word embedding fixed while fine-tuning other parameters of the pre-trained language model, so that it can generalize with tokens unseen during training.

Figure 4.1 shows our model architecture. We use the pre-trained language model GPT-2[1] proposed in [98], which is a 12-layer transformer. The final hidden state of the transformer is used to calculate attention weights and the copy switch p_{copy}. We first feed the embedded attribute-value list serving as the context for generation. In this architecture, the generator is fine-tuned from pre-trained parameters while the encoder and attention part is learned from scratch, the initial geometry of the two sides are different. Therefore we need to apply larger weight to the copy loss p_{copy}, to give the model a stronger signal to "teach" it to copy facts from the input table.

4.4 Experiments

4.4.1 Datasets and Experiment Setup

The original WikiBio dataset [96] contains 700k English Wikipedia articles of well-known humans, with the Wiki infobox serving as input structured data and the first sentence of the article serving as target text. To demonstrate generalizability, we collect datasets from two new domains: *Books* and *Songs* by crawling Wikipedia pages. After filtering and cleanup, we end up with 23,651 instances for *Books* domain and 39,450 instances for *Songs* domain[2]. Together with the *Humans* domain of the original WikiBio

[1]https://github.com/openai/gpt-2

[2]Note that the target text sometimes contains information not in the infobox. This is out of the scope of the few-shot generation in this work. Therefore we further filter the datasets and remove the ones with rare words out of infobox. Check [113] for a related study of this issue on the WikiBio dataset

dataset, for all three domains we conduct experiments by varying the training dataset size to 50, 100, 200 and 500. The rest of data is used for validation (1,000) and testing. The weight λ of the copy loss term is set to 0.7. Other parameter settings can be found in C. To deal with vocabulary limitation of few-shot training, for all models we adopt the Byte Pair Encoding (BPE) [114] and subword vocabulary in [98].

We compare the proposed method with other approaches investigated in Section 4.3, serving as the baselines - **Base-original:** the original model in [26]; **Base:** uses the same architecture, but in addition applies the pre-trained word embedding and fix it during training; **Base + switch:** adds the switch policy; **Base + switch + LM-scratch:** makes the architecture same as our method, except training the model from scratch instead of using pre-trained weights for generator. **Template:** template-based non-neural approach, manually crafted for each domain.

4.4.2 Results and Analysis

Following previous work [26], we first conduct automatic evaluations using BLEU-4, shown in Table 4.1. The ROUGE-4 (F-measure) results follow the same trend with BLEU-4 results, which we show in C.

As we can see, the original model **Base-original** [26], which obtains the state-of-the-art result on WikiBio full set, performs very poorly under few-shot setting. It generates all tokens from softmax over vocabulary, which results in severe overfitting with limited training data, and the results are far behind the template-based baseline. With the switch policy, **Base+switch** first brings an improvement of an average of over 10.0 BLEU points. This indicates that the content selection ability is easier to be learned with a handful of training instances. However, it forms very limited, not fluent sentences. With the augmentation of the pre-trained language model, our model

Base+switch+LM brings one more significant improvement of an average over 8.0 BLEU points. We provide sample outputs of these methods using 200 training instances in Table 4.2.

Domain	Humans					Books					Songs				
# of training instances	-	50	100	200	500	-	50	100	200	500	-	50	100	200	500
Template	16.3	-	-	-	-	25.6	-	-	-	-	30.1	-	-	-	-
Base-original	-	2.2	3.7	4.9	5.1	-	5.8	6.1	7.4	6.7	-	9.2	10.7	11.1	11.3
Base	-	2.9	5.1	6.1	8.3	-	7.3	6.8	7.8	8.8	-	10.4	12.0	11.6	13.1
Base + switch	-	15.6	17.8	21.3	26.2	-	24.7	26.9	30.5	33.2	-	29.7	30.6	32.5	34.9
Base + switch + LM-scratch	-	6.6	11.5	15.3	18.6	-	7.1	9.2	14.9	21.8	-	11.6	16.2	20.6	23.7
Base + switch + LM (Ours)	-	**25.7**	**29.5**	**36.1**	**41.7**	-	**34.3**	**36.2**	**37.9**	**40.3**	-	**36.1**	**37.2**	**39.4**	**42.2**

Table 4.1: BLEU-4 results on three domains. Base-original: the original method in [26]; Base: applies pre-trained word embedding; Base+switch: adds the switch policy; Base+switch+LM-scratch: makes the same architecture as our method, but trains the model from scratch without pre-trained weights for the generator. Template: manually crafted templates

Attribute	Value	Attribute	Value
name	andri ibo	fullname	andri ibo
birth date	3 april 1990	birth place	sentani , jayapura , indonesia
height	173 cm	currentclub	persipura jayapura
position	defender	...	

Gold Reference: andri ibo (born april 3 , 1990) is an indonesian footballer who currently plays for persipura jayapura in the indonesia super league .

Generated texts of different methods

Base: vasco emanuel freitas (born december 20 , 1992 in kong kong) is a hong kussian football player and currently plays for hong kong first division league side tsw pegasus .

Base+switch: andri ibo andri ibo (3 april 1990) is a international cricketer .

Base+switch+LM (Ours): andri ibo (born 3 april 1990) is an indonesian football defender , who currently plays for persipura jayapura .

Table 4.2: A sample input table and generated summaries from the test set of *Humans* domain, using 200 training instances

Table 4.3 shows the effect of the copy switch loss p_{copy} introduced in Section 4.3.2, giving the model a stronger signal to learn to copy from input table.

Ma et al. [105] propose the Pivot model, for low-resource NLG with 1,000 paired examples and large-scale target-side examples. We compare our method with the Pivot model in table 4.4. Note that here we train and evaluate the models on the original

# of training instances	50	100	200	500
Base + switch + LM	**25.7**	**29.5**	**36.1**	**41.7**
- w/o copy loss p_{copy}	21.4	25.5	31.3	38.0

Table 4.3: Ablation study: Effect of the copy loss term on *Humans* domain, measured by BLEU-4. The loss term brings an average improvement of over 4.0 BLEU points.

WikiBio dataset used in their work, in order to maintain the size of the target side examples for their settings.

# of paired training instances	50	100	200	500	1000
Pivot	7.0	10.2	16.8	20.3	27.3
Ours	17.2	23.8	25.4	28.6	31.2

Table 4.4: Comparison with the Pivot model [105]. Compared to their method using additional large-scale target side examples, our method requires no additional target side data, while achieving better performance.

Human Evaluation

We also conduct human evaluation studies using Amazon Mechanical Turk, based on two aspects: *Factual correctness* and *Language naturalness*. We evaluate 500 samples. Each evaluation unit is assigned to 3 workers to eliminate human variance. The first study attempts to evaluate how well the generated text correctly conveys information in the table, by counting the number of facts in the text supported by the table, and contradicting with or missing from the table. The 2nd and 3rd columns of Table 4.5 show the average number of supporting and contradicting facts for our method, comparing to the strongest baseline and the gold reference. The second study evaluates whether the generated text is grammatically correct and fluent, regardless of factual correctness. We conduct pairwise comparison among all methods, and calculate the average times each method is chosen to be better than another, shown in the 4th column of Table 4.5. Our method brings a significant improvement over the strongest baseline ($p < 0.01$ in Tukey's HSD test for all measures). The copy loss term further alleviates producing incorrect facts. The language naturalness result of our method without the copy loss is slightly better, because this evaluation does not consider factual correctness; thus the generated texts with more wrong facts can still get high score. See C for more details of our evaluation procedure.

	# Supp.	# Cont.	Lan. Score
Gold Reference	4.25	0.84	1.85
Base + switch	2.57	2.17	0.93
Base + switch + LM (ours)	**3.64**	**1.12**	1.59
- w/o copy loss p_{copy}	3.54	1.30	**1.63**

Table 4.5: Human evaluation results: Average number of supporting facts (column 2, the larger the better), contradicting facts (column 3, the smaller the better), and language naturalness score (column 4, the larger the better).

Chapter 5

High-Fidelity Natural Language Generation from Logical Forms

5.1 Introduction

Natural language generation (NLG) from structured data has been an important research problem in many applications. Recent data-driven methods have achieved good performances on various NLG tasks [26, 115, 116]. However most studies focus on surface descriptions of simple record sequences, for example, attribute-value pairs of fixed or very limited schema, like E2E [93] and WikiBio [96]. In real-world cases for multi-row tables, it is often more desirable and plausible to provide descriptions involving higher-level logical inference across data records. For example, in Figure 5.1, instead of plain restatements, human readers would be more favorable to abstract descriptions that can summarize or conclude information over the table records. To produce such logical-level generations of high fidelity, it is not yet appropriate to provide only the table as the input in a real-world NLG system, based on the following reasons:

1) *Low Fidelity.* Given only the table, it is challenging for existing neural models to

produce such logically correct generations involving reasoning and symbolic calculations, e.g., `max`, `min`, `counting`, `averaging`, etc.

2) *Uncontrollable content selection.* Given a table, the space of logically entailed descriptions is exponentially large, due to vast number of combinations of different operations and arguments from the table, e.g., `count`, `comparison`, `superlative`, etc. It is hard and uncontrollable for neural models to decide a valid, favorable choice of logical selections solely based on the table, due to the difficulty of imposing high-level semantic constraints in the compositional generation process.

To combat with the above problems, we argue that it is necessary to leverage intermediate meaning representations to achieve faithful and controllable logical generations. To this end, we formulate the task of logical-level NLG as a **logical form to text** problem. Specifically, besides the table information, the generation module is provided with a logical form representing the semantics of the target text (see Figure 5.1 for an example). By separating logical reasoning and language realization, the correctness of the intermediate logical form is guaranteed, and the challenge for the realization module is fully shifted to semantic understanding.

To facilitate research in this direction, we propose a new dataset named Logic2Text, consisting of 5.6k open-domain tables, 10.8k manually annotated (logical form, description) pairs. Our dataset is of high quality in terms of (1) natural and interesting descriptions; (2) accurate logical forms with 100% execution correctness. In our dataset, the coarse logic types are 7 common ones to describe multi-row tables: `count`, `superlative`, `comparative`, `aggregation`, `majority`, `unique`, and `ordinal`. We employ a Python-like program to serve as our logical forms, which can be easily converted to other types of logical forms. Figure 5.1 shows two examples of our dataset. Compared with previous surface-level NLG datasets, one major distinction of our dataset is the free schema of the logical forms, which can be represented as diversified graph structures. The new dataset poses great challenges on the model's ability to understand the structural semantics in graph representation.

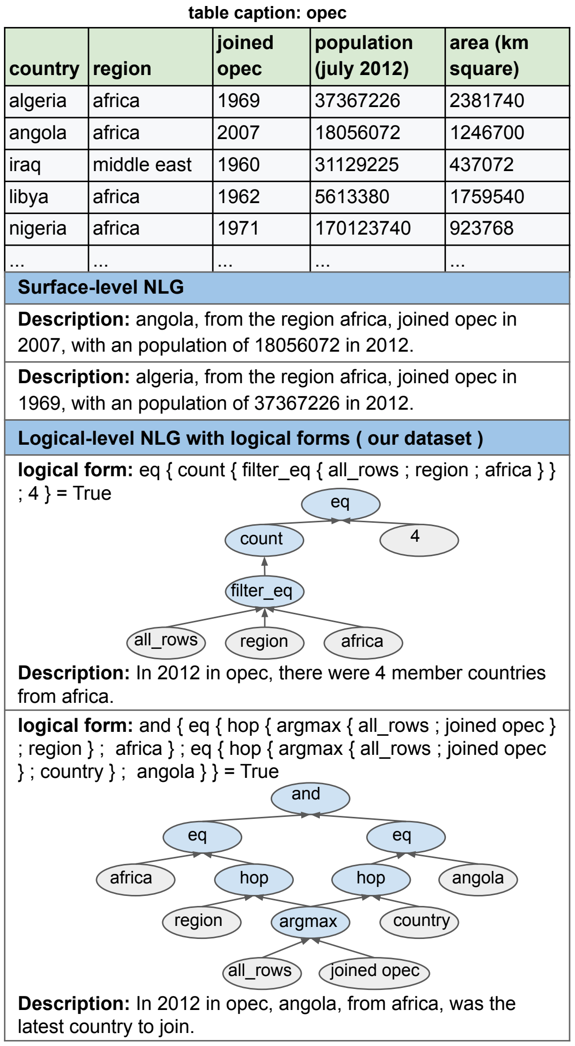


table caption: opec

country	region	joined opec	population (july 2012)	area (km square)
algeria	africa	1969	37367226	2381740
angola	africa	2007	18056072	1246700
iraq	middle east	1960	31129225	437072
libya	africa	1962	5613380	1759540
nigeria	africa	1971	170123740	923768
...	...	...	...	...

Surface-level NLG

Description: angola, from the region africa, joined opec in 2007, with an population of 18056072 in 2012.

Description: algeria, from the region africa, joined opec in 1969, with an population of 37367226 in 2012.

Logical-level NLG with logical forms (our dataset)

logical form: eq { count { filter_eq { all_rows ; region ; africa } } ; 4 } = True

Description: In 2012 in opec, there were 4 member countries from africa.

logical form: and { eq { hop { argmax { all_rows ; joined opec } ; region } ; africa } ; eq { hop { argmax { all_rows ; joined opec } ; country } ; angola } } = True

Description: In 2012 in opec, angola, from africa, was the latest country to join.

Figure 5.1: Examples of surface-level NLG compared with NLG with logical forms of our dataset. Here are two examples with logic type `count` and `superlative`. The function nodes are in blue, and the text nodes in grey.

We employ an array of popular generation models as the baseline approaches. The experiments are conducted in (1) *Fully-supervised setting.* We train the models using the full dataset to analyze their performances. (2) *Few-shot setting.* We simulate the low-resource scenario in real-world use cases. Experimental results show that the logical forms are critical to acquiring high-fidelity generations. The pre-trained language model outperforms other baselines (pointer-generator, graph2seq, transformer, etc.), but still makes factual and logical errors.

In summary, our contributions are the following:

- We propose a new large-scale dataset, Logic2Text, with descriptions of common logic types accompanied by the underlying logical forms. The logical forms present diversified graph structures, which raises more challenges on semantic understandings.
- We surveyed several popular generation models as the baselines under fully-supervised and few-shot settings, as well as analyze their pros and cons.

Our dataset can also be used in the reverse way (text to logical form) to facilitate tasks related to semantic parsing. [117] propose the task of fact verification against tables, however the performance is greatly limited due to the lack of the ground truth logical forms. This can be one direct application of our dataset. In this work, we focus on NLG.

5.2 Related Work

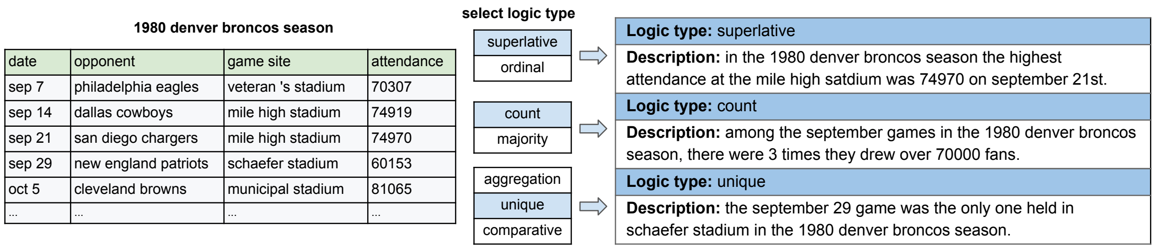
NLG from structured data or knowledge has been studied for many years. There are various applications, such as the automatic generations of weather reports [94], sport reports [95], clinical and health reports [92, 118], response generation in task-oriented dialogue systems [75, 63, 119], etc.

Traditional methods typically employ a pipeline-based approach including content selection, planning and surface realization [25, 120]. Recent data-driven methods tend to conflate the pipeline modules into one end-to-end neural networks, such as [26, 95, 27, 121]. Most recently, large-scale pre-trained models [98, 122, 123] have achieved new state-of-the-arts on various generation tasks. Chen et al. [116] demonstrate that a simple pre-training based method can achieve very reasonable performance on the WikiBio dataset [96] under few-shot setting. More recent works begin to focus on fidelity preserving of the generation, such as [113, 124]. Their work obtains good performances on surface-level NLG. In contrast, our work focus on the fidelity of logical-level generations.

There are a few popular NLG datasets mostly on surface-level generation. Such as WeatherGov [94], E2E [93], WikiBio [96], and ToTTo [125]. RotoWire [95] is a more challenging dataset on generating basketball game reports from multi-row tables. But the reports are still limited to superficial restatements of table records, with very few involving logical inference. [126] investigate generation of interesting trivia from superlative wikipedia tables. [127] propose the task of generating arbitrary sentences with logical inference from the table. Their task mainly works for probing purpose, i.e., to test the ability of neural models to produce any logically correct descriptions solely based on the table. However, such a task formulation is not yet appropriate for building a real-world NLG system due to low-fidelity, as we discussed in the introduction. The best-performing model in [127] only obtains a factual correctness rate over 20% based on human evaluation, which is clearly far from an acceptable level in real-world systems.

Another line of works related to ours is the text generation from syntactic or semantic sentence structure, such as generation from CCG grammar [128], UCG grammar [129], AMR [130]. There are many early works attempting algorithmic approaches on such kinds of logical formulations [131, 132, 133, 131], etc. Later proposed datasets include the Groningen Meaning Bank [134], the AMR bank [135], the DeepBank [136], etc. In contrast, our work focus on the logical formulations executed on database style tables, and common symbolic operations on tables, such as count, superlative, comparison. As nowadays much of the production data is stored in table based DB, we believe such a dataset should help building systems with table based data.

5.3 Dataset Construction



date	opponent	game site	attendance
sep 7	philadelphia eagles	veteran 's stadium	70307
sep 14	dallas cowboys	mile high stadium	74919
sep 21	san diego chargers	mile high stadium	74970
sep 29	new england patriots	schaefer stadium	60153
oct 5	cleveland browns	municipal stadium	81065
...	...	...	...

Logic type: superlative
Description: in the 1980 denver broncos season the highest attendance at the mile high satdium was 74970 on september 21st.

Logic type: count
Description: among the september games in the 1980 denver broncos season, there were 3 times they drew over 70000 fans.

Logic type: unique
Description: the september 29 game was the only one held in schaefer stadium in the 1980 denver broncos season.

Figure 5.2: description composition: the workers are asked to select three logic types and compose a statement based on the selected logic type, that describe interesting facts in the table.

The table source of LOGIC2TEXT is from WikiTables[1] [137], a collection of open-domain tables crawled from Wikipedia. We follow [117] to filter out over-complicated tables and take a subset of tables with less than 20 rows and 10 columns.

In this dataset, we start from 7 types of most commonly used logics [117] to describe multi-row tables: `count`, `superlative`, `comparative`, `aggregation`, `majority`, `unique`, and `ordinal`. For example, for logic type `count`, the definition is: counting some rows in the table based on the values in one column, with the scope of all table rows or a subset. Refer to D for the definitions of all logic types. Each description involves exactly one type of logic. This matches the observation that humans generally do not describe their interested information in tables with over-complicated logics. For logical forms, we use a python-like program, and the function set is an extension of [117]. Refer to D for definitions of all functions.

Our dataset is constructed in 3 stages: §5.3.1 Description composition and verification, §5.3.2 Logical form annotation and derivation, §5.3.3 Logical form execution and

[1]`http://websail-fe.cs.northwestern.edu/wikiTables/about/`

verification. We adopt the workflow of composing descriptions first and then deriving the logical forms, because under such an order, the annotators can compose natural descriptions based on the interesting facts in the table, which is hard to be achieved by automatic enumeration of logical forms followed by template re-writing. For all crowd-sourcing tasks we hire Amazon Mechanical Turkers[2] (AMT) under three requirements: (1) from English native countries (“US”,“CA”,“GB”, “AU”); (2) Approval rate higher than 95% for all HITs; (3) More than 500 approved HITs. We follow the human subject research protocols[3] to pay the workers. We maintain strict high criterions for approval and review at least 10 random samples for each worker to decide whether to approve or reject all his/her HITs.

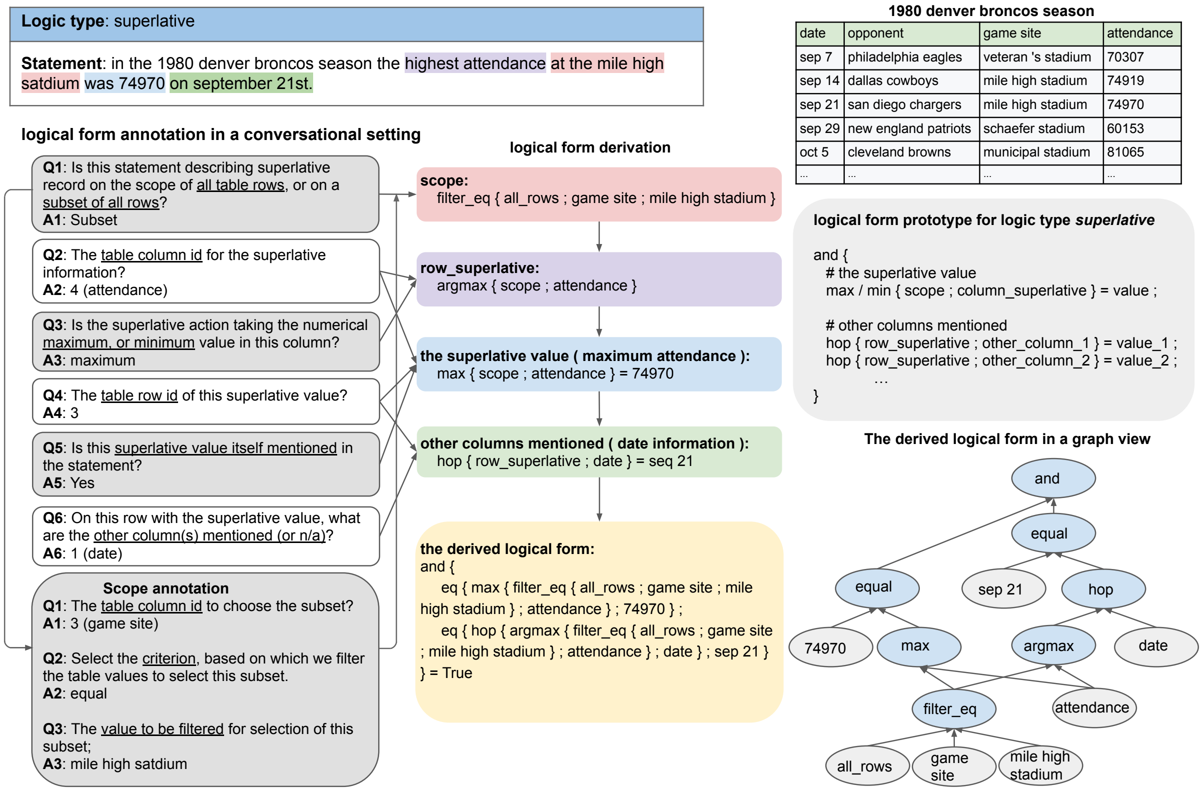
5.3.1 Description Composition & Verification

In this first stage, the human workers are asked to compose statements of a *certain logic type*, that describe *interesting* facts in the table. It’s possible that some logic types cannot be applied to certain tables. Therefore we design the following working procedure: For each table, the 7 logic types are randomly put into three groups (with sizes 2, 2, and 3). The worker is asked to choose one logic type from each group and compose a description based on the chosen logic type. They must follow the requirements (1) try to choose diversified logic types, (2) avoid template-like language and try to compose natural and interesting descriptions, (3) include the information in table captions, so as to compose comprehensive descriptions without unspecified pronouns. An example of the workflow is shown in Figure 5.2. We provide the workers detailed explanations for each logic type by their corresponding definitions, accompanied by examples. After collecting the descriptions, we add a verification stage to filter out descriptions of low

[2]`https://www.mturk.com/`
[3]`https://en.wikipedia.org/wiki/Minimum_wage_in_the_United_States`

quality. We redistribute the collected descriptions grouped by each logic type, then ask three questions: Is this description (1) of the correct logic type presented? (2) factually correct? (3) grammatically correct and fluent? We filter out the description if any question receives a negative response.

5.3.2 Logical Form Annotation & Derivation



Logic type: superlative

Statement: in the 1980 denver broncos season the highest attendance at the mile high satdium was 74970 on september 21st.

logical form annotation in a conversational setting

Q1: Is this statement describing superlative record on the scope of all table rows, or on a subset of all rows?
A1: Subset

Q2: The table column id for the superlative information?
A2: 4 (attendance)

Q3: Is the superlative action taking the numerical maximum, or minimum value in this column?
A3: maximum

Q4: The table row id of this superlative value?
A4: 3

Q5: Is this superlative value itself mentioned in the statement?
A5: Yes

Q6: On this row with the superlative value, what are the other column(s) mentioned (or n/a)?
A6: 1 (date)

Scope annotation

Q1: The table column id to choose the subset?
A1: 3 (game site)

Q2: Select the criterion, based on which we filter the table values to select this subset.
A2: equal

Q3: The value to be filtered for selection of this subset;
A3: mile high satdium

logical form derivation

scope:
filter_eq { all_rows ; game site ; mile high stadium }

row_superlative:
argmax { scope ; attendance }

the superlative value (maximum attendance):
max { scope ; attendance } = 74970

other columns mentioned (date information):
hop { row_superlative ; date } = seq 21

the derived logical form:
and {
eq { max { filter_eq { all_rows ; game site ; mile high stadium } ; attendance } ; 74970 } ;
eq { hop { argmax { filter_eq { all_rows ; game site ; mile high stadium } ; attendance } ; date } ; sep 21 }
} = True

1980 denver broncos season

date	opponent	game site	attendance
sep 7	philadelphia eagles	veteran 's stadium	70307
sep 14	dallas cowboys	mile high stadium	74919
sep 21	san diego chargers	mile high stadium	74970
sep 29	new england patriots	schaefer stadium	60153
oct 5	cleveland browns	municipal stadium	81065
...	...	...	...

logical form prototype for logic type *superlative*

```
and {
  # the superlative value
  max / min { scope ; column_superlative } = value ;

  # other columns mentioned
  hop { row_superlative ; other_column_1 } = value_1 ;
  hop { row_superlative ; other_column_2 } = value_2 ;
      ...
}
```

The derived logical form in a graph view

Figure 5.3: logical form annotation & derivation: Note that in this example the questions are all in concise forms. In the AMT interface shown to the workers, we write instructions in a more casual and detailed manner, accompanied by several examples.

As the core step of our dataset construction pipeline, we design a workflow to obtain the semantic information via conversations with human workers, then use the information to derive the logical forms. The questions in the conversation are specifically designed for each logic type. Here we go through the example of logic type `superlative` given in Figure 5.3 to illustrate our annotation process.

The logical form structure prototype is shown in the right grey part, consisting the description of the superlative value, and other mentioned columns on the row with the superlative value. Then we ask the follow-up questions to derive the complete logical form based on the prototype, shown on the left part of Figure 5.3: Q1. What is the scope of the superlative operation? If the scope is a subset of all table rows, we perform another round of conversation to annotate the scope. Q2. What is the table column of the superlative operation? Q3. What is the specific type of the superlative operation: maximum or minimum. Q4. What is the table row with the superlative value. Q5. Is the superlative value itself mentioned in the description or not? Q6. What are the other columns mentioned in the description? After collecting the answers of the above questions, we can derive the logical form, as shown in the middle part of Figure 5.3.

We provide the workers with detailed explanations of the prototype for each logical types, as well as several examples. Note that the prototype covers most, but not all of the logical descriptions due to their diverse nature. Thus we also provide the option to skip the example if it cannot be formulated by the given question set. Check D for the annotation process of other logic types.

5.3.3 Logical Form Execution & Verification

After the collection of logical forms, we use the Stanford CoreNLP toolkits[4] to tokenize all text content (all table information, the descriptions, and the texts in the logical forms). To remove incorrect logical forms, we execute the logical forms and perform another round of semantic verification.

Logical Form Execution The functionality in our logical form is based on the ones

[4]`https://stanfordnlp.github.io/CoreNLP/index.html`

used in [117]. We extend the function set to deal with semi-structured table cells (dates, mixed numbers and strings, etc.). We execute all logical forms against the corresponding table, and only keeps the ones that evaluate to `True`. This guarantees that the logical forms in our dataset achieve 100% execution correctness.

Semantic Verification Note that execution correctness does not guarantee semantic correctness. Therefore we perform another round of semantic verification. Since AMT workers do not have experts knowledge to understand the logical forms, we convert the logical form into natural language interpretation based on the operations of each function. We then ask the workers to verify whether the interpretation correctly matches the meaning of the description, with neither insufficient nor redundant information. Then we remove the examples receiving negative responses.

Expert Evaluation To demonstrate the quality of our dataset, we employ two computer science graduate students to conduct evaluations. We randomly sample 200 examples for each logic type to verify the semantic correctness. Each example is examined by both students, and the decision is made after discussion. The result shows that each logic type reaches a correct rate no less than 90%.

Tables	5,554
Examples	10,753
Vocabulary	14.0k
Avg. description length	16.77
Avg. # nodes in logical form	9.00
Avg. # function nodes in logical form	3.27
Avg. length of the linearized logical form	24.35

Table 5.1: General statistics of Logic2Text.

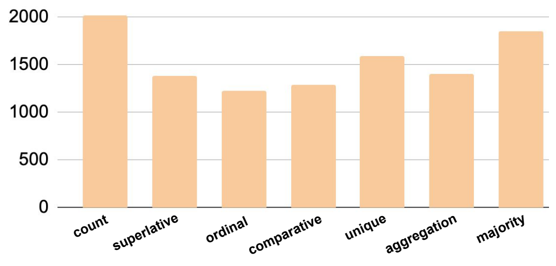


Figure 5.4: Distribution of logic types.

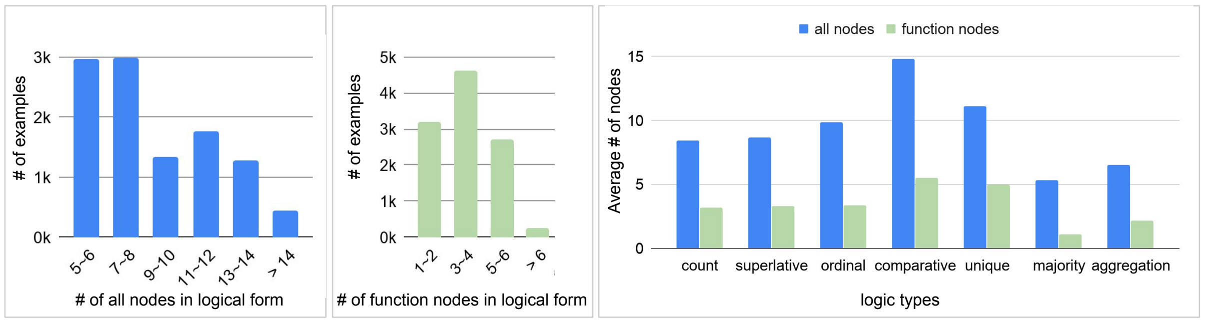


Figure 5.5: The distribution of our dataset regarding the number of all nodes (*Left*) and function nodes (*Mid*) in the logical form. *Right:* average number of all nodes and function nodes in the logical forms for each logic type.

5.4 Dataset Statistics and Analysis

We follow a rough ratio of 8:1:1 to split our dataset into 8,566 for training, 1,095 for development, and 1,092 for testing. The train, dev, and test sets have no overlap tables. We show the statistics of the dataset in Table 5.1 and the distributions of 7 logic types in Figure 5.4. Each table has 1-3 descriptions with different logic types. Since the logical forms present graph structure nature, we analyze the complexity of the logical forms based on the number of nodes in the graph, regarding the number of function nodes (`count`, `max`, etc.) and the number of all nodes (both function nodes and text nodes), respectively. As shown in Figure 5.5, the logical forms in LOGIC2TEXT have a minimum of 5 nodes and maximum over 14 nodes. For different logic types, `comparative` has the most number of nodes, because it involves the selection and operation for two table rows. `superlative`, `ordinal`, and `unqiue` primarily focus on one table row, sometimes with

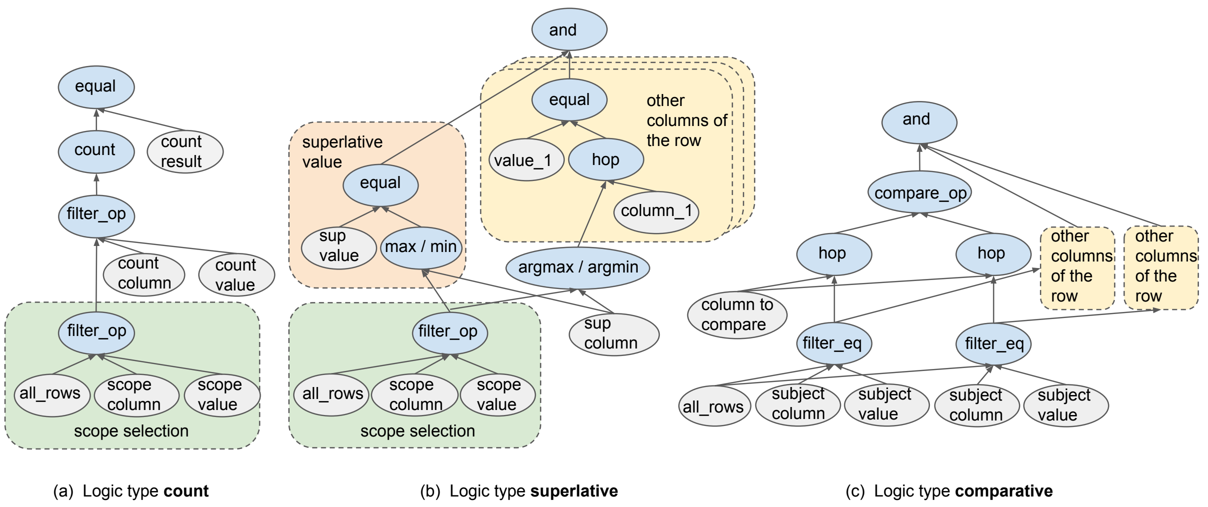


Figure 5.6: Overview of logical form structures for logic type `count`, `superlative`, and `comparative`. (a) `count`: the structure in the green shadow is optional, representing the scope of counting. It can be all table rows (a single text node) or a subset of rows from a filter operation. (b) `superlative`: the structure in the orange shadow is optional, depending on the presence of the max/minimum value in the description. The structure in the yellow shadow appears 0 or more times.

the scope being a subset of all table rows, which makes the logical forms more complex. `count`, `majority`, and `aggregation` are summarization based logic types on multiple table rows. They are the three relatively simpler ones in terms of logical form structures. Figure 5.6 gives the logical form structures for 3 example logic types.

5.5 Experiments

In this section we first describe the baseline models of our dataset in §5.5.1; Then we conduct experiments in fully-supervised setting §5.5.2; We demonstrate the importance of the logical form in §5.5.3 and perform ablation studies in §5.5.4; At last we carry out experiments under few-shot setting §5.5.5.

5.5.1 Baseline Models

Apart from the logical forms serving as the primary input to the generation model, the table information is also crucial to provide context information. Following human's order to comprehend the table and produce descriptions, the input C is formulated as the sequence of table captions, table headers, table content, and the logical form. The goal is to generate a sequence w that maximize $P(w \mid C)$:

$$w = argmax \prod P(w_t \mid w_{0:t-1}, C) \tag{5.1}$$

We employ the following models as our baselines for Logic2Text:

Template We manually craft generation templates for each logic type based on the logical form.

Seq2seq+att We employ the seq2seq with an attention model from [138]. The input sequence is formulated as the concatenation of the table caption, table headers, the linearized table content, and the linearized logical form.

Pointer generator [112] adds the copy mechanism upon the seq2seq with an attention model, allowing the decoder to copy tokens from the input directly. Such a mechanism is known to be critical for fidelity-preserving generation with abundant entities, numbers, etc.

Graph2seq+copy There is a line of research for graph neural network based encoders, such as [139, 140], etc. We employ one representative model, Graph2seq [140], to encode the logical forms. The table caption and headers are first fed into a seq2seq, followed by the graph encoder for the logical form. We also add the copy mechanism to allow copying from the input.

Transformer+copy The popular Transformer model [83] has shown remarkable progress in many tasks including NLG. In addition to the original Transformer structure, we add the copy mechanism where the last hidden layer is used to calculate the attention score and the copy switch. We also add segment embeddings for different input components, similar as [39].

GPT-2 Generally, with Transformer based structures, recent large-scale pre-trained models have achieved new SOTA results in a wide range of NLP tasks. A typical workflow is to use the pre-trained model as initialization, then fine-tune the model on task-specific data. In this work, we employ the generative pre-training model, GPT-2 [98], as one of our baselines.

For all neural models we use Byte-Pair Encoding (BPE) [114] and the subword vocabulary used in [98]. Refer to D for more implementation details.

5.5.2 Fully-Supervised Setting

For automatic evaluations, we employ BLEU-4[5] (B-4), ROUGE-1, 2, 4, and L (F measure)[6], noted as R-1, R-2, R-4, and R-L. The results for all baselines are presented in Table 5.2.

For models without pre-training, the copy mechanism brings a significant improvement, comparing pointer-generator and seq2seq. This is because the descriptions in our dataset involve much factual information from the table and the logical form, e.g., entity names, and numbers. However, the pre-trained language model GPT-2 can mostly accurately produce these factual terms even without a copy mechanism, demonstrating the powerful prior knowledge obtained from large-scale pre-training.

Compared to the pointer generator, which takes linearized logical form as input, Graph2seq+copy directly models the graph structure and gets a slight improvement. The Transformer+copy model obtains better performance than the Graph2seq+copy model,

[5]Standard script NIST mteval-v13a.pl
[6]rouge-1.5.5.

Models	B-4	R-1	R-2	R-4	R-L
Template	17.57	50.56	24.20	6.61	37.81
Seq2seq+att	12.46	36.22	15.91	4.49	31.03
Pointer generator	24.03	56.23	30.51	10.78	46.85
Graph2seq+copy	25.38	58.15	32.79	12.25	49.47
Transformer+copy	26.42	58.77	33.05	12.83	49.01
GPT-2	**31.44**	**64.16**	**39.48**	**17.46**	**53.99**

Table 5.2: Automatic evaluation results for all baseline models under fully-supervised setting.

as the Transformer architecture is indeed a graph neural network with self-attention as aggregation function over the neighbors and regards the input as a fully-connected graph. Recent works [141, 142, 143] have shown that Transformer-based structure can capture hierarchical syntactic structures and graph representations. The GPT-2 model obtains the best performance among all with a significantly larger improvement. As a pre-trained language model with the Transformer structure, it combines the strength of both structural modeling and language modeling prior. Some example generations are provided in D.

Human Evaluation

Automatic scores are not sufficient for precise evaluation of factual and logical correctness. Therefore we conduct human evaluations through (1) crowdsourcing on Amazon Mechanical Turkers (AMT), and (2) human expert evaluations.

For human evaluations on AMT, we randomly sample 500 examples from each of the top best-performing methods (GPT-2 and Transformer+copy), and the gold references. The evaluations are conducted on two axes: *factual correctness* and *language fluency*. For factual correctness, we ask the workers to verify whether the description is factually supported by the table; For language fluency, we conduct pairwise comparisons between different methods. For both evaluations, we distribute each task to 3 workers to eliminate human variance. The evaluation results of language fluency and factual correctness are shown in Table 5.4 and the first row of Table 5.3, respectively. For more details of the evaluation, check D. To conduct a precise evaluation of semantic correctness, i.e.,

	Gold	GPT-2	Transformer+copy
% factually correct	98.1	82.4	65.1
% semantically correct	92.0	73.0	43.0

Table 5.3: Human evaluation results of factual correctness (first row) and semantic correctness (second row).

	% win	% loss	% tie
GPT-2 vs Gold	35.6	43.3	21.1
GPT-2 vs Transformer+copy	54.0	25.3	20.7
Gold vs Transformer+copy	61.2	23.6	15.2

Table 5.4: Human evaluation results of language fluency.

whether the generation correctly matches the meaning of the logical form, we invite human experts (two computer science graduate students) to perform the evaluation. We sample 200 examples from each method and ask them to verify whether the description correctly presents the meaning of the logic form. Each example is examined by both students, and the decision is made after discussion. The second row of Table 5.3 shows the evaluation results.

As we can observe from all evaluation results, the GPT-2 model gives big improvements on both fidelity preserving and language fluency, but there's still a gap, especially on semantic correctness. We believe our dataset can serve as a valuable resource posing such a challenge on high-fidelity generation with complex semantics.

5.5.3 Importance of the Logical Form

We conduct experiments without using the logical form, i.e., to generate arbitrary logically correct descriptions solely based on the table, which is the task setting of [127]. The generation is evaluated with all descriptions of the same table as multi-references, as in their setting. The best performing model of [127] obtains a BLEU-4 score of 20.17 and a factual correctness rate of 20.2% based on human evaluation of 500 samples. In contrast, the generations of our best -performing baseline can obtain a factual correctness rate of 82.4% shown in Table 5.3, which demonstrates the great importance of the logical form on high-fidelity generation. Note that the automatic scores are not directly comparable, since, in our task setting, each generation maps to a unique logical form and is evaluated with a single reference.

5.5.4 Component-Wise Ablation

Models	B-4	R-1	R-2	R-4	R-L
GPT-2	31.44	64.16	39.48	17.46	53.99
-w/o caption	21.67	54.26	29.16	9.99	45.70
-w/o header	29.86	62.98	38.46	16.64	52.57
-w/o content	30.42	64.17	38.89	16.79	53.63

Table 5.5: Ablation study on other input components.

We perform ablation studies on other input components: the table caption, header, and content, using the best-performing GPT-2 model. As shown in Table 5.5, both the table caption and header provide strong context information for generation, and the table content also brings a slight improvement.

5.5.5 Few-Shot Setting

Considering that acquiring a large amount of (logical form, description) pairs in real-world cases is expensive, we also include a few-shot learning task for our dataset, where the model is only provided with hundreds of paired examples. Previous works have shown that the pre-trained language models obtain strong NLG performance even with a handful of fine-tuning instances [116]. Therefore we still use the best-performing GPT-2 model for this study. In our dataset, the amount of unseen logical form structures increases with the reduction of training instances. As shown in Table 5.6, while there's still a gap with the fully-supervised result, the result with 1,000 training instances using GPT-2 is comparable to some other baselines with the full training data. This demonstrates the potential of incorporating generative pre-training for the few-shot learning task.

# of examples	B-4	R-1	R-2	R-4	R-L
Full	31.44	64.16	39.48	17.46	53.99
100	17.09	48.26	23.52	7.47	38.74
200	19.98	51.99	27.02	9.42	41.86
500	23.04	56.64	30.99	11.35	46.86
1000	24.57	57.81	32.64	12.21	47.67

Table 5.6: Results for few-shot learning setting with 100, 200, 500, and 1000 training examples, using GPT-2.

Chapter 6

Knowledge-Enriched Task-Oriented Dialogue

6.1 Introduction

Dialogue systems have achieved substantial progress [144, 64, 145] due to recent success in language model pre-training [98, 146, 147]. One major type of dialogue being studied is task-oriented dialogue (TOD) [62, 63, 148, 64], where the system aims to collect user intents/goals to complete certain tasks (e.g. restaurant-booking). In most of TOD systems, the system responses are concise and templated, as we only focus on the success of task completion but not providing a natural and engaging conversational experience. The latter is the target of another kind of popularly studied dialogue - knowledge-grounded chit-chat [86, 149, 150, 151]. Knowledge-grounded chit-chat enables dialog systems to access external knowledge so that they can provide more engaging and knowledgeable conversations and in the same time reduce hallucinations [152].

Existing studies mostly focus on one specific type of dialogue, either task-oriented dialogue or knowledge-grounded chit-chat. However, the ultimate goal of Conversational

AI is a human-like, unified system capable of conversing with the users naturally and seamlessly among all kinds of dialogues. Current TOD systems can hardly make interesting and engaging conversations only with templated functional responses. Few previous works like ACCENTOR [153] have studied the combination of TOD and chit-chat, but their chit-chat augmentation is largely limited to simple general responses like 'you're welcome', 'sounds good to me'. In this work, we propose to enrich TOD with knowledge-grounded chit-chat, as one step further towards the ultimate goal of building a human-like, unified system (See Figure 6.1 for an example). We believe that the proposed knowledge-enriched TOD system can conduct more social, natural, and engaging conversations.

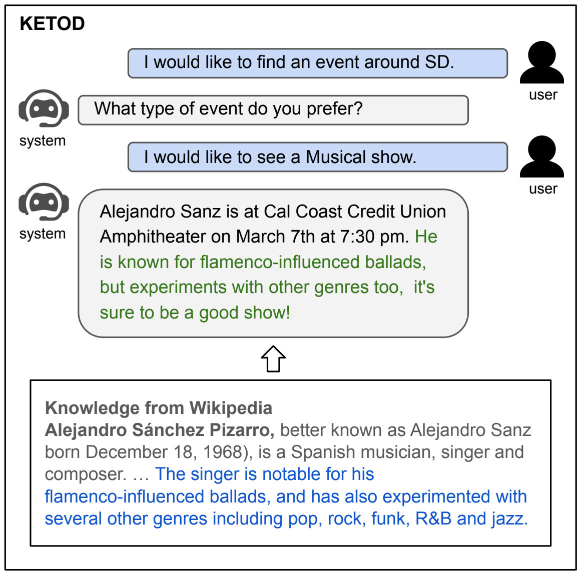

KETOD
I would like to find an event around SD.
user
What type of event do you prefer?
system
I would like to see a Musical show.
user
Alejandro Sanz is at Cal Coast Credit Union Amphitheater on March 7th at 7:30 pm. He is known for flamenco-influenced ballads, but experiments with other genres too, it's sure to be a good show!
system
Knowledge from Wikipedia
Alejandro Sánchez Pizarro, better known as Alejandro Sanz born December 18, 1968), is a Spanish musician, singer and composer. ... The singer is notable for his flamenco-influenced ballads, and has also experimented with several other genres including pop, rock, funk, R&B and jazz.

Figure 6.1: An example from the KETOD dataset: the green text is our enriched chit-chat based on the entity knowledge of *Alejandro Sanz* in the original TOD. Such knowledge-grounded chit-chat makes the dialogue more natural and engaging.

To this end, we propose a new dataset, KETOD (Knowledge-Enriched Task-Oriented Dialogue). In order to obtain natural and high-quality knowledge-grounded chit-chat, we design the dataset construction framework by augmenting existing TODs and using the relevant entity knowledge to make the chit-chat enrichment. Specifically, for a given TOD, 1) extracting the entities from the dialogue states and actions; 2) retrieving the knowledge associated with the entities from external knowledge sources; 3) asking the human annotators to enrich the system responses with chit-chat using the retrieved knowledge. We demonstrate that the knowledge-enriched dialogues constructed with the proposed framework are consistently preferred by human judges across all axes of engagingness, interestingness, knowledge, and humanness.

We propose two models, and study the challenges and insights of our new dataset. The first model is an end-to-end language model that jointly learns and generates both the TOD results (dialogue states and actions) and the knowledge-enriched responses. The second model is a pipeline that first generates the TOD results, then uses another response generation model to generate the knowledge-enriched responses. We run comprehensive experiments to demonstrate the improvement over the baselines, and show that our models can generate better knowledge-enriched responses while maintaining competitive performance on the TOD tasks. To summarize, we make the following major contributions:

- We propose the task of combining TOD and knowledge-grounded chit-chat.
- We construct a new large-scale dataset, KETOD, with high-quality, manually annotated dialogue responses enriched with knowledge-grounded chit-chat. The dataset is publically available.
- We propose two models for our dataset, and carry comprehensive experiments to study the challenges and insights. We believe our dataset should be a valuable resource for building a human-like conversational assistant.

6.2 Related Work

Task-oriented dialogue. Task-oriented dialogue (TOD) has been one of the most popular types of dialogue in the research community. There have been many works on building each component of the TOD system, such as dialogue state tracking, action prediction, and response generation [75, 154, 155, 20, 21, 156, 72, 157]. Later works begin to investigate building end-to-end systems [158, 156, 159, 160]. Most recent works on TOD also apply such language model pre-training style methods on building end-to-end systems [64, 161, 162], achieving top performances on various datasets. Popular datasets in TOD include the DSTC challenge series [163], MultiWOZ [63], SGD [148], etc. As the primary goal of TOD is the successful completion of the functional tasks, the system responses are mostly concise and templated.

Chit-chat dialogue. Another type of popular studied dialogue is chit-chat, with the goal of making a natural and engaging conversation. Apart from the ‘pure’ simple chit-chat that mostly covers plain and general responses, more works focus on knowledge groundings to achieve better specificity and engagingness, such as using user profiles [149], social media contexts [164], or knowledge graphs [150, 165], etc. In this work, our enriched chit-chat is grounded on open-domain knowledge, similar as the Topical-Chat [166] and the WOW dataset [151], where the system converses with the users about certain topics involving entity knowledge in an open-ended setting. In contrast, their datasets specifically focus on knowledge-grounded chit-chat, while our dataset combines TOD and such chit-chat.

Combination of task-oriented dialogue and chit-chat. ACCENTOR [153] proposes to combine TOD with chit-chat by prepending or appending chit-chat to the TOD system responses. But their chit-chat is mostly general responses like ’sounds good!’, ’you’re welcome’. FusedChat [167] proposes to insert chit-chat turns into TOD as well as re-writing TOD turns, but their chit-chat is still mostly general responses or based on commonsense knowledge. Kim et al. [168] propose to insert additional turns into TOD, where the system needs to respond based on the knowledge from domain FAQs. The DSTC10 task 2 [169] is based on the dataset from [168] with a similar focus. HyKnow [170] also proposes to insert turns into TOD grounded on knowledge from unstructured documents. These datasets focus on the challenge of detecting those turns requiring external knowledge and selecting the knowledge to generate the responses. In contrast, our dataset focuses on injecting knowledge-grounded chit-chats into the original TOD responses, to make the dialogue more natural and engaging. Our dataset poses more challenges in selecting knowledge based on the dialogue context and generating the responses with both the correct TOD information and the chit-chat seamlessly.

6.3 The KETOD Dataset

6.3.1 Dataset Construction

In this section, we describe our framework to construct the KETOD dataset. We start from existing TOD datasets and employ human annotators to augment the functional system responses with knowledge-grounded chit-chat. The proposed approach is demonstrated to give natural, contextual-relevant knowledge enrichment, and meanwhile easy to scale to different datasets. Figure 6.2 gives an overview of the dataset construction pipeline.

Data preparation. We build upon the SGD dataset [148], with TOD spanning 16 domains, such as `Restaurant, Wheather`, etc. Given each TOD, to obtain the knowledge relevant to the dialogue context, we first extract all the entities from the dialogue states and actions. We exclude the domains `Alarm`, `Banks`, and `Payment` as there are mostly

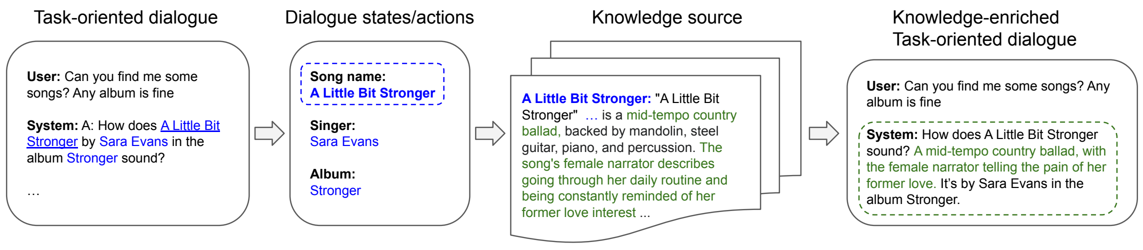

Task-oriented dialogue
User: Can you find me some songs? Any album is fine
System: A: How does A Little Bit Stronger by Sara Evans in the album Stronger sound?
...
Dialogue states/actions
Song name: A Little Bit Stronger
Singer: Sara Evans
Album: Stronger
Knowledge source
A Little Bit Stronger: "A Little Bit Stronger" ... is a mid-tempo country ballad, backed by mandolin, steel guitar, piano, and percussion. The song's female narrator describes going through her daily routine and being constantly reminded of her former love interest ...
Knowledge-enriched Task-oriented dialogue
User: Can you find me some songs? Any album is fine
System: How does A Little Bit Stronger sound? A mid-tempo country ballad, with the female narrator telling the pain of her former love. It's by Sara Evans in the album Stronger.

Figure 6.2: The pipeline of dataset construction: for each task-oriented dialogue, we first extract all the entities from the dialogue states and actions. Then we retrieve the knowledge associated with each entity from external knowledge sources (Wikipedia). At last, we ask human annotators to enrich the TOD system responses with chit-chat grounded on the retrieved knowledge.

no entities involved in these domains; Also, to simplify the human annotation process in the next step, we remove the dialogues with over 10 entities involved.

Knowledge retrieval. For each entity, we use the concatenation of the domain name and entity name as the query to retrieve Wikipedia articles. We use the DrQA retriever [171] to retrieve the top 2 Wikipedia articles and take the first 2 paragraphs of each article as the knowledge candidates associated with each entity. Then we break the retrieved articles into sentences, with each sentence as one knowledge snippet.

Response enrichment. In this step, we employ human annotators to enrich the system responses in the original TOD based on the dialogue context and the retrieved knowledge. For each TOD, we present to the annotators the full dialogue, as well as all the knowledge snippets associated with the entities in the dialogue. The annotators can click on each entity name to see the associated knowledge snippets in an expanded textbox. See E for our annotation interface.

The annotation process is as follows: 1) Read the full dialog first to have an overall story in mind, as well as the relevant knowledge snippets, then to decide how many turns to enrich with chit-chat and which turn(s) to enrich; If there is no way to make a natural chit-chat enrichment, skip the example. 2) After deciding the turn(s) to enrich with the chit-chat, select the knowledge snippets used to make the enrichment (at most 3 snippets for each turn); 3) Rewrite the system response to enrich with chit-chat grounded on the selected knowledge snippets; The functional information in the original response should be maintained, while may be rephrased to make the enriched response more natural.

To ensure the dataset quality, we first interview the annotators to select the appropriate hires through a few test examples. Then we launch a training session for all the annotators to learn the task and the annotation interface. We launch the official batches after the annotators can well-master the task. During annotation, we specifically emphasize the contextualization of the knowledge-grounded chit-chat - the enrichment should be contextualized closely on the dialogue context, but not a plain restatement of the knowledge snippets.

6.3.2 Dataset Statistics and Analysis

We end up with 5,324 dialogues with enriched system responses. We make the split of 4,247/545/532 as the train/dev/test set. Table 6.1 shows the statistics of the KETOD dataset. Around 12.1% of the turns (which indicates mostly 1 or 2 turns in one dialogue) are enriched with knowledge-grounded chit-chat. This intuitively complies with our goal of making the whole dialogue natural and engaging, since too frequent chit-chat may result in redundancy and unnaturalness.

Quality assessment of the annotation. During the annotation process, around 12% of the dialogues cannot be enriched with any turns and thus discarded. It takes around 100 seconds for the annotators to finish each dialogue. To assess the quality of the annotation, we sample 5% of the annotated dialogues and distribute them to linguistics to check: 1) If the chit-chat enrichment is relevant and natural; 2) If the knowledge snippets are accurately selected corresponding to the enrichment. We end up with a correct rate of 87.0%.

Dialogues	5,324
Vocabulary	27k
All turns	52,063
Turns enriched with chit-chat	6,302
All entities	4,639
All knowledge snippets	33,761
Avg. # turns per dialogue	9.78
Avg. # tokens in enriched responses	28.07
Avg. # entities per dialogue	4.98
Avg. # knowledge snippets per dialogue	70.50

Table 6.1: General statistics of KETOD.

Justification of the chit-chat enrichment. To demonstrate that our proposed knowledge-enriched TOD can be more natural and engaging, we conduct human evaluations to compare KETOD dialogues and their corresponding original TOD dialogues without chit-chat enrichment (SGD). We follow [172] to make pairwise comparisons of the full dialogues over the following four axes: engagingness, interestingness, knowledge, and humanness. The results in Figure 6.3 show the superiority of KETOD over all axes.

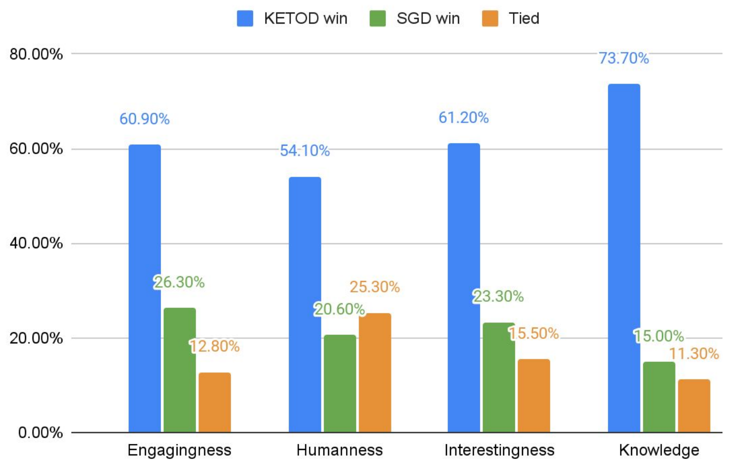


Figure 6.3: Results of pairwise comparison of KETOD vs SGD.

6.4 Approaches

In this section, we will describe the proposed two models for the KETOD dataset.

6.4.1 Overview and Formulations

For each dialogue turn, denote the dialogue context (history) as C, belief states as B, database search results as D, actions as A, the knowledge snippets used for chit-chat enrichment as K, the response as T. Then we formulate the problem as: given the dialogue context C and a knowledge source (Wikipedia in this dataset), the target is to generate the belief states B, actions A, and the response T, which may be enriched with chit-chat grounded on the knowledge based on the context. The goal of the optimization on KETOD is two-folded: 1) Optimizing the generation of knowledge-enriched responses; 2) Maintaining the task performances;

In this work, we propose the following modeling framework on KETOD: 1) given the dialogue context, generate the belief states and actions; 2) extract the entities in the belief states and actions, then use these entities to retrieve knowledge candidates (similar as in the dataset construction process); 3) conditioned on the dialogue context, use a knowledge selection model to select knowledge snippets from the knowledge candidates retrieved; 4) generate the knowledge-enriched response conditioned on both the dialogue context and the selected knowledge snippets.

Based on the above general framework, we propose two architectural approaches, **SimpleToDPlus** and **Combiner**, respectively in §6.4.3 and §6.4.4.

6.4.2 Knowledge Selection

After the generation of belief states and actions, we retrieve the knowledge snippet candidates from Wikipedia using the entities in the belief states and actions. The average

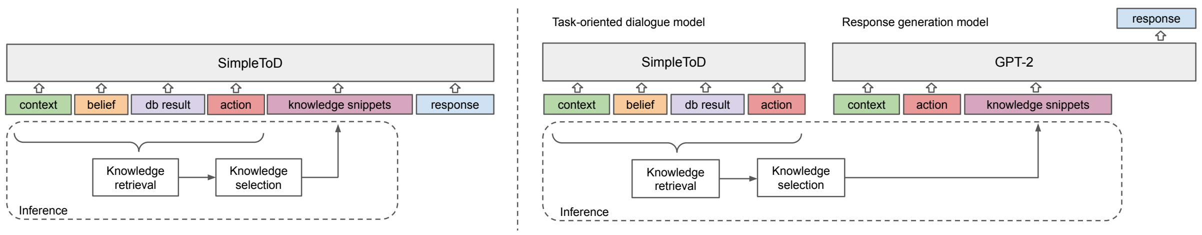


Figure 6.4: Illustration of the models. *Left*: the SimpleToDPlus model; *Right*: the Combiner model;

number of knowledge snippets candidates retrieved for each dialogue is around 70. It is impractical to input all of them into the models due to the large amount. As we have the annotation for the ground truth knowledge snippets used for each chit-chat enrichment, we train a knowledge selection model to select the top knowledge snippets most appropriate for chit-chat enrichment. Specifically, we concatenate the dialogue context with each knowledge snippet as the input. Then we use BERT [39] to train a simple classifier to rank all the knowledge snippets candidates. We take the top 3 ones as the knowledge selection results. We use the same knowledge selection model for both architectures.

6.4.3 SimpleToDPlus

SimpleToD [173] is a recent popular approach on TOD, which uses one single language model to sequentially generate the belief states, actions, and responses. It has achieved strong performances in all the above functional tasks. In this work, we propose its extension, **SimpleToDPlus**, to generate knowledge-enriched responses for TOD. The left part of Figure 6.4 shows the overview of SimpleToDPlus. We formulate the training sequence as:

$$[C, B, D, A, K, \text{¡chitchat¿}, T] \tag{6.1}$$

Where ¡chitchat¿ is a tag to indicate the decision of whether to enrich the response with knowledge grounded chit-chat or not. If the response is not enriched, we insert the tag ¡nochitchat¿. Since the number of the gold knowledge snippets varies from 1 to 3 (as in the dataset construction), to be compatible with inference time, here we first run the knowledge selection model on all training instances. Then we construct the knowledge snippets K as the merge of the gold knowledge snippets and the knowledge selection model results, truncated to 3 ones. If the response is not enriched with chit-chat, i.e., no gold knowledge snippets, we still put 3 snippets from the knowledge selection model ranking results here during training.

In the inference time, we first sequentially generate the belief states and actions. Then we extract the entities from the generated belief states and actions, and apply the same process of knowledge retrieval as in dataset construction. Next, we run the knowledge selection model on the retrieved knowledge candidates and take the top 3 knowledge snippets as the model input followed by the generated actions. At last, the model generates the decision to make chit-chat enrichment or not, followed by the final response.

Since the knowledge-enriched response is conditioned on the entity knowledge from the belief states and actions, we need to directly include the entities in the actions and responses during generation, instead of generating a delexicalized result first and then lexicalizing in the post-process as in the original SimpleToD. To simplify, we use the oracle database search results for all the experiments.

6.4.4 Combiner

SimpleToDPlus models all the generations in an end-to-end manner. In **Combiner**, we use a pipeline of a TOD model followed by a response generation model to separate the

Models	**Joint GA**	**Avg GA**	**Act-Slot F1**	**BLEU-4$_{\text{aug}}$**	**BLEU-4$_{\text{orig}}$**	**BLEU-4$_{\text{all}}$**
SimpleToD-ref	27.6	54.2	67.6	-	-	-
SimpleToD	23.7	50.1	62.7	4.8	10.7	10.0
SimpleToDPlus	**28.6**	**52.2**	**66.9**	6.3	**11.7**	**11.0**
Combiner	24.5	51.5	64.5	**6.5**	9.9	9.5

Table 6.2: Main experiment results: Both SimpleToDPlus and Combiner outperform the baseline. Overall SimpleToDPlus obtains better response generation performance while maintaining competitive TOD performance.

TOD part (belief states, actions) with the generation of knowledge-enriched responses. The goal is to study whether an independent model can better learn each task with less interference from the other. The overview of the architecture is shown on the right of Figure 6.4.

For the TOD model, we use SimpleToD to generate the belief states and actions, with the training sequence as:

$$[C, B, D, A] \tag{6.2}$$

We find that including the knowledge-enriched responses during training degrades the task performance, indicating the disturbance from the ungrounded knowledge in the responses.

For the response generation model, we use GPT-2 [98] with the concatenation of the dialogue context, actions, and the knowledge snippets as the prompt:

$$T = \text{GPT-2}(C, A, K) \tag{6.3}$$

We use the same way of constructing the merged knowledge snippets during training, and the same process of knowledge retrieval and selection during inference as in SimpleToDPlus.

6.5 Experiments

Baseline model. We use SimpleToD [173] as our baseline model, i.e., with the training sequence as [C, B, D, A, T], without the injection of knowledge snippets. Therefore the knowledge-grounded chit-chat in the responses T do not have any knowledge groundings - we aim to show the necessity of knowledge grounding for our task, as well as the effectiveness of our proposed models to incorporate knowledge.

Experimental setups and evaluations. Check E for details of model training and parameter settings. For the TOD performances, we evaluate the belief states with joint goal accuracy (Joint GA) and average goal accuracy (Avg GA), and the actions with act-slot F1, same as [153]. For the automatic evaluations of response generation, we use three BLEU-4 scores: BLEU-4_{aug} for evaluating the responses enriched with knowledge; BLEU-4_{orig} for evaluating the responses not enriched with knowledge; BLEU-4_{all} for evaluating all responses;

6.5.1 Main Results

Performance on response generation. Table 6.2 shows our main experiment results. For the performances on response generation, we can see that both of our proposed models, SimpleToDPlus and Combiner, improve on the knowledge-enriched response generation (BLEU-4_{aug}) over the SimpleToD baseline. Since in the baseline, we do not include the knowledge snippets in the input, the generated responses are mostly enriched with random knowledge or frequent knowledge in the training data. The improvements demonstrate the necessity of knowledge grounding and the effectiveness of the proposed knowledge enrichment methods. Combiner performs slightly better on knowledge-enriched responses than SimpleToDPlus but falls short on the responses without knowledge-enrichment (i.e., original TOD responses). This is partially because of its pipeline nature - a separated response generation module can better learn the knowledge enrichment without the disturbance of other tasks, but the error cascading from the generated actions degrades the performance of the TOD responses part.

Performances on belief states and actions. To better study how the knowledge enrichment affects the TOD performances, we first train SimpleToD on our dataset without the knowledge enrichment, i.e., replace all the knowledge-enriched responses with the original responses in SGD. We name it as SimpleToD-ref in Table 6.2, serving as a reference of the original TOD performances. The SimpleToD baseline gives largely degraded performances due to the disturbance from the ungrounded knowledge in the responses during training. Therefore in Combiner, we do not include the responses in the training sequences of the TOD model (specified in section 6.4.4), and obtain better scores. SimpleToDPlus achieves the best TOD performances, which are nearly competitive with SimpleToD-ref. This is partially due to the enhancement of language modeling ability brought by the training on the responses grounded on the input knowledge.

Human evaluations. In order to get the more comprehensive measure of the response generation performances, we conduct human evaluations for both dialogue-level pairwise comparison and turn-level factualness evaluation. For dialogue-level pairwise comparison, we randomly sample 200 dialogues from the test set and apply the same process as in dataset evaluation (6.3.2). For each model, we construct the full dialogue results by concatenating the generated response for each turn given the gold dialogue context. Table 6.3 shows the results of pairwise comparison between the SimpleToDPlus model and the Combiner model, demonstrating SimpleToDPlus is more performant. Table 6.4 shows the results of pairwise comparison between SimpleToDPlus and the gold reference, indicating there is still a large room for further improvements. See E for the human evaluation results of comparing both methods to the baseline. For turn-level factualness evaluation, we randomly sample one turn with chit-chat enrichment from each dialogue,

Metrics	**SimpleToDPlus win (%)**	**Combiner win (%)**	**Tied (%)**
Engagingness	47.8	24.5	27.8
Interestingness	34.5	19.0	46.5
Knowledge	29.5	26.3	44.3
Humanness	43.3	23.8	33.0

Table 6.3: Human evaluation of SimpleToDPlus vs. Combiner.

Metrics	**SimpleToDPlus win (%)**	**Gold win (%)**	**Tied (%)**
Engagingness	16.8	60.5	22.8
Interestingness	12.0	51.0	37.0
Knowledge	14.5	44.8	40.8
Humanness	17.3	58.0	24.8

Table 6.4: Human evaluation of SimpleToDPlus vs. Gold.

and present both the generated response and the selected knowledge snippets to the annotators. The annotators are asked to check whether the chit-chat in the responses are factually correct based on the knowledge snippets. SimpleToDPlus and Combiner obtain the factualness correct rate of 64.2% and 66.1%, respectively. In summary, Combiner achieves better factualness of knowledge enrichment since its independent response generation model can better focus on the learning of knowledge groundings. But its error cascading due to the pipeline nature may degrade the overall consistency and human-likeness of the generated dialogue.

As we have two optimization goals on KETOD 1) Optimizing the generation of knowledge-enriched responses; 2) Maintaining the task performances, we consider SimpleToDPlus as a better model regarding the overall performances. We will use the results of SimpleToDPlus for the ablations and other analyses in the rest of the experiments.

	BLEU-4$_{\text{aug}}$	**BLEU-4$_{\text{all}}$**
Given gold TOD results, decision, and knowledge		
SimpleToD	6.5	13.1
SimpleToDPlus	9.7	14.6
Combiner	14.6	15.1
Given gold TOD results		
SimpleToD	6.3	12.8
SimpleToDPlus	7.4	14.0
Combiner	9.6	13.9

Table 6.5: Analysis of different inference stages: we provide the models with gold results up to certain stages, and investigate the performances for the inferences on following stages.

	BLEU-4$_{\text{aug}}$	**BLEU-4$_{\text{all}}$**	**Knowledge selection recall (%)**
Gold	9.7	14.6	100.0
BERT selection	7.8	14.4	52.7
TF-IDF selection	6.6	13.7	14.1

Table 6.6: SimpleToDPlus response generation performance with varying knowledge selection strategies.

6.5.2 Ablations and Analysis

Analysis of different inference stages. There are several inference stages for this task - the TOD results (belief states and actions), the selection of knowledge snippets, and the final response generation, where each stage is conditioned on previous results. Therefore the errors accumulate through all the stages leading to the final performances. Here we run another two sets of experiments to study such error accumulations and compare the two models. Specifically, first, we feed the models with the gold TOD results, chit-chat decisions, and knowledge snippets, to solely test the abilities to generate the knowledge-enriched responses; Second, we feed the models with the gold TOD results to test the following stages of knowledge selection and the response generation. The results are shown in Table 6.5. Compared with the full inference results in Table 6.2, we can see

	BLEU-4$_{\text{aug}}$	BLEU-4$_{\text{all}}$	Enrichment decision F1 (%)
Gold decision	9.7	14.6	100.0
Predicted decision	8.0	14.1	58.7

Table 6.7: SimpleToDPlus response generation performance using (1) the gold set of turns to enrich with chit-chat, and (2) the predicted set of turns.

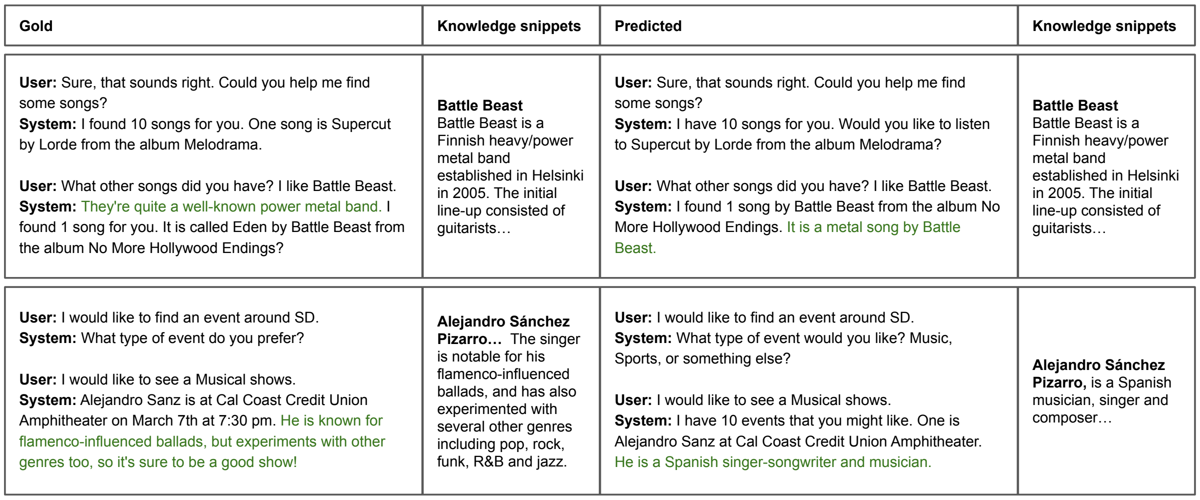
	All	Hotels	Movies	Restaurant	Music
BLEU-4$_{\text{aug}}$	6.3	7.1	5.2	5.1	7.7
BLEU-4$_{\text{all}}$	11.0	10.3	12.2	14.0	12.3

Table 6.8: Domain breakdown of SimpleToDPlus response generation performances.

that the Combiner model largely outperforms SimpleToDPlus if provided with more gold results for previous stages. However, it gradually falls behind SimpleToDPlus when moving to fully end-to-end inference due to the error cascading of its pipeline nature.

Importance of knowledge selection strategies. To demonstrate the importance of the knowledge selection strategies (and their subsequent recall performance), we run SimpleToDPlus with 1) gold knowledge snippets; 2) predicted knowledge snippets (with BERT); 3) knowledge snippets selected by heuristics (we use TF-IDF matching between the current dialogue turn and the knowledge snippets). To eliminate the influences brought by other inference stages, we feed the model with gold TOD results (dialogue states and actions). The results are shown in Table 6.6. There exists a certain level of variance for knowledge selection, e.g., when recommending a song for the user, you may talk about its genre, its singer, or the album.

Learning *when* to inject knowledge-enriched chit-chat. In all models, we use the special token '¡chitchat¿' and '¡nochitchat¿' to indicate the decision to inject knowledge enrichment for the responses. To study the effect of the chit-chat injection decision-making accuracy on the overall dialogue tasks, we run SimpleToDPlus (1) with the



Gold	Knowledge snippets	Predicted	Knowledge snippets
User: Sure, that sounds right. Could you help me find some songs? **System:** I found 10 songs for you. One song is Supercut by Lorde from the album Melodrama. **User:** What other songs did you have? I like Battle Beast. **System:** They're quite a well-known power metal band. I found 1 song for you. It is called Eden by Battle Beast from the album No More Hollywood Endings?	**Battle Beast** Battle Beast is a Finnish heavy/power metal band established in Helsinki in 2005. The initial line-up consisted of guitarists…	**User:** Sure, that sounds right. Could you help me find some songs? **System:** I have 10 songs for you. Would you like to listen to Supercut by Lorde from the album Melodrama? **User:** What other songs did you have? I like Battle Beast. **System:** I found 1 song by Battle Beast from the album No More Hollywood Endings. It is a metal song by Battle Beast.	**Battle Beast** Battle Beast is a Finnish heavy/power metal band established in Helsinki in 2005. The initial line-up consisted of guitarists…
User: I would like to find an event around SD. **System:** What type of event do you prefer? **User:** I would like to see a Musical shows. **System:** Alejandro Sanz is at Cal Coast Credit Union Amphitheater on March 7th at 7:30 pm. He is known for flamenco-influenced ballads, but experiments with other genres too, so it's sure to be a good show!	**Alejandro Sánchez Pizarro…** The singer is notable for his flamenco-influenced ballads, and has also experimented with several other genres including pop, rock, funk, R&B and jazz.	**User:** I would like to find an event around SD. **System:** What type of event would you like? Music, Sports, or something else? **User:** I would like to see a Musical shows. **System:** I have 10 events that you might like. One is Alejandro Sanz at Cal Coast Credit Union Amphitheater. He is a Spanish singer-songwriter and musician.	**Alejandro Sánchez Pizarro,** is a Spanish musician, singer and composer…

Figure 6.5: Case studies: two examples of comparing the generation from SimpleToDPlus (right) with the gold reference (left), together with the knowledge snippets selected. Overall our model can mostly generate reasonable knowledge enrichment, but still falls short on engagingness and consistency compared to the gold references.

ground-truth information of turns to enrich with chit-chat, and (2) with the predicted decisions, using the gold TOD results. Table 6.7 shows the performance gap, which highlights the importance of knowing *when* to inject knowledge-enriched chit-chat. While such decisions are conditioned on the dialogue history, e.g., we may tend to not enrich a turn if many of the previous turns are enriched to avoid redundancy, there also exists some variance. In a real system, we may consider specifying the turns to make the chit-chat enrichment instead of letting the model make the decision.

Domain analysis. We investigate the model performance for each domain in Table 6.8. We observe that the performance differences may depend on the variance of the enriched knowledge. Domains with larger variance on selected knowledge tend to have lower automatic scores. For example, in `Hotels` domain, mostly the chit-chat is about the locations since there are mostly location entities involved in this domain. But for the `restaurants` domain, the enriched knowledge can be about the food, the restaurant, as well as the location. The selected knowledge shows more diversity and variance.

We provide case studies in Figure 6.5 to compare the predicted results with the gold references.

Chapter 7

Conclusion and Future Work

In this thesis, I elaborated on my work throughout my Ph.D. study toward the goal of building an advanced, human-like AI assistant. Specifically, I contributed in both the directions of natural language understanding and natural language generation. In this chapter, I will make conclusions and summarize future directions for each division.

7.1 Natural Language Understanding

First, in [18], we introduce FinQA, a new expert-annotated QA dataset that aims to tackle numerical reasoning over real-world financial data. The questions in FinQA pose great challenge for existing models to resolve domain-specific knowledge, as well as to acquire complex numerical reasoning abilities. We propose baseline frameworks and conduct comprehensive experiments and analysis. The results show that current large pre-trained models still fall far behind the human expert performance. We believe FinQA should serve as a valuable resource for the research community. This encourages potential future work on for such realistic, complex application domains:

- How to develop pre-training tasks to improve numerical reasoning abilities of language models still remains a challenging problem.

- Recent prompting-based methods demonstrate the potential to improve numerical reasoning by proper prompt design of large language models. It is also an interesting direction to explore what is the principle way to model numerical reasoning problems, the symbolic-based approaches or prompting large language models.

Second, in [29], we investigate task-oriented dialogue and take the first step toward building the user-centric dialogue system. We propose a new representation to model fine-grained user preferences, and build a new dataset using our proposed representation. sStarting from our dataset, NUANCED, we believe the user-centric dialogue system is an open-ended problem and the following directions are worth pursuing:

- Preliminary experimental results indicate that to improve performance, it is promising to incorporate external domain texts into pre-trained models, for example, pre-training the model on domain corpora like restaurant descriptions and reviews.
- Although our dataset collects a large set of domain entity knowledge, we still cannot guarantee that it will cover the vast amount of unknown entities in the future. One idea is to incorporate a knowledge base (KB) in the form of data augmentation or modeling.
- Through our large-scale dataset, although one can learn a general agreement of estimated distributions from the crowds, a more user-specific distribution would be more desirable. We believe providing a personalized solution is another proper next step to consider.

7.2 Natural Language Generation

First, in [30], we propose the new research problem of few-shot natural language generation. Our approach is simple, easy to implement, while achieving strong performance on various domains. Our basic idea of acquiring language modeling prior can be potentially extended to a broader scope of generation tasks, based on various input structured data, such as knowledge graphs, SQL queries, etc. The deduction of manual data curation efforts for such tasks is of great potential and importance for many real-world applications.

Second, in [31], we study natural language generation with logical inferences. We formulate the problem of logical-level NLG as generation from logical forms in order to obtain controllable and high-fidelity generations. We propose a new dataset named Logic2Text. Potential future directions include the followings:

- Among the baselines, pre-trained language model obtains the best result but still brings factual and logical errors. It's a challenging task for the neural models to understand and generalize on such semantic forms.
- Human evaluations are precise but expensive. Our dataset can be used in the reverse direction to train a semantic parser, to assist parsing-based evaluations.
- In this work, we primarily focus on the step to generate descriptions based on the logical form. In a real-world NLG system, the logical forms should be produced based on the end applications and user interests. Another potential future direction could be the content selections, i.e., how to select and organize the logical forms to construct a discourse plan based on user interests.

At last, in [32], we dive into the real application of dialogue assistants. We propose to combine task-oriented dialogue with knowledge-grounded chit-chat, and construct a new dataset named KETOD, with manually composed knowledge-enriched system responses. We conduct comprehensive experiments on our new dataset to study the insights and challenges. We believe that our proposed task is an important step towards the ultimate goal to build a unified, human-like conversational AI. Our new dataset KETOD, annotated by experts, will greatly facilitate the research in this direction. In real-world conversations, there are many ways that different kinds of dialogues are fused together. In this work, we start from the more straightforward type of directly enriching the TOD responses with knowledge-grounded chit-chat. Potential future works could be:

- Investigating other fusion strategies to combine the two kinds of dialogues, for example, interleaving TOD dialogue turns with chit-chat turns, or integrating multiple fusion strategies.
- Investigating multiple kinds of dialogues, for example, chit-chat grounded on multiple types of knowledge sources, or multi-modality knowledge sources, interactions, etc.

7.3 Summary

My long-term research goal is to build an advanced AI assistant that can conduct conversations and complete tasks like human beings. There are still many open problems to be addressed by the research community, given the current research progress. Here I emphasize two of them to be pursued next:

The Complex Reasoning Tasks We anticipate that the tasks requiring complex reasoning beyond surface semantic understanding should be the next research focus. The recent rapid advancement in large pre-trained language models has achieved near-human performances in many traditional NLP tasks that mostly focus on language patterns or simple semantic understanding. It is becoming the major trend to train bigger language models to capture stronger language patterns from massive corpus to provide solutions to such traditional NLP tasks. However, when considering tasks requiring complex cognitive reasoning, large language models often show a flat learning curve with the model size. Our work also demonstrates such challenges. One direction for reasoning tasks is the neural symbolic reasoning approach, as commonly studied by the community. In the other direction, with the development of large language models these years, the prompting-based approach also shows great potential. In the future, I am interested in studying the principle way of imitation for human reasoning ability.

The Realization of Abstract Problems For many research problems, the research community mostly studies abstract problem settings that extract a representative challenge aspect from the real-world problems, and formulate it into an abstract problem. But when building real-world systems, we need to consider integration, realization, and all kinds of constraints to bring together such abstract problems. This is one of the most important challenges toward the ultimate goal of building a human-like assistant. In this thesis, we take some preliminary steps to study the realistic settings of the research problems, such as the new representations for fine-grained user preferences, the new model with fewer training data, and the combination of task-oriented dialogue and knowledge-grounded chitchat. In the future, I will keep working on bridging the gap between abstract problems and real-world applications.

Appendix A

FinQA: A Dataset of Numerical Reasoning over Financial Data

A.1 Operation Definitions

We describe all the operations in Table A.1.

Name	Arguments	Output	Description
add	number1, number2	number	add two numbers: $number1 + number2$
subtract	number1, number2	number	subtract two numbers: $number1 - number2$
multiply	number1, number2	number	multiply two numbers: $number1 \cdot number2$
divide	number1, number2	number	multiply two numbers: $number1/number2$
exp	number1, number2	number	exponential: $number1^{number2}$
greater	number1, number2	bool	comparison: $number1 > number2$
table-sum	table header	number	the summation of one table row
table-average	table header	number	the average of one table row
table-max	table header	number	the maximum number of one table row
table-min	table header	number	the minimum number of one table row

Table A.1: Definitions of all operations

A.2 Experiment Details

All the validation results of the baselines are shown in Table A.2. The trainings of all models are conducted on TITAN RTX GPUs. All the implementation and pre-trained models are based on the huggingface transformers library. We use the Adam optimizer [61]. The parameter settings are the following:

Retriever The learning rate is set as 3e-5, with batch size of 16.

TF-IDF + Single Op We use the TF-IDF from the Scikit-learn library.

FinQANet The learning rate is set as 1e-5. For Bert-base, Roberta-base, and finBert we use batch size of 32; For Bert-large and RoBerta-large we use batch size of 16 due to GPU memory constraints.

Retriever + Seq2seq A bidirectional LSTM is used for encoding the input, then an LSTM is used for decoding with attention. Learning rate is set as 1e-3, hidden size as 100.

Retriever + NeRd The parameter settings are the same as FinQANet.

Pre-Trained Longformer We truncate the maximum input length as 2,000. The learning rate is set as 2e-5, with batch size of 16 due to GPU memory constraints.

For more modeling details refer to our released code.

A.3 Case Studies

Here we provide more case studies with the full input reports. For all the examples the gold evidence is highlighted in blue.

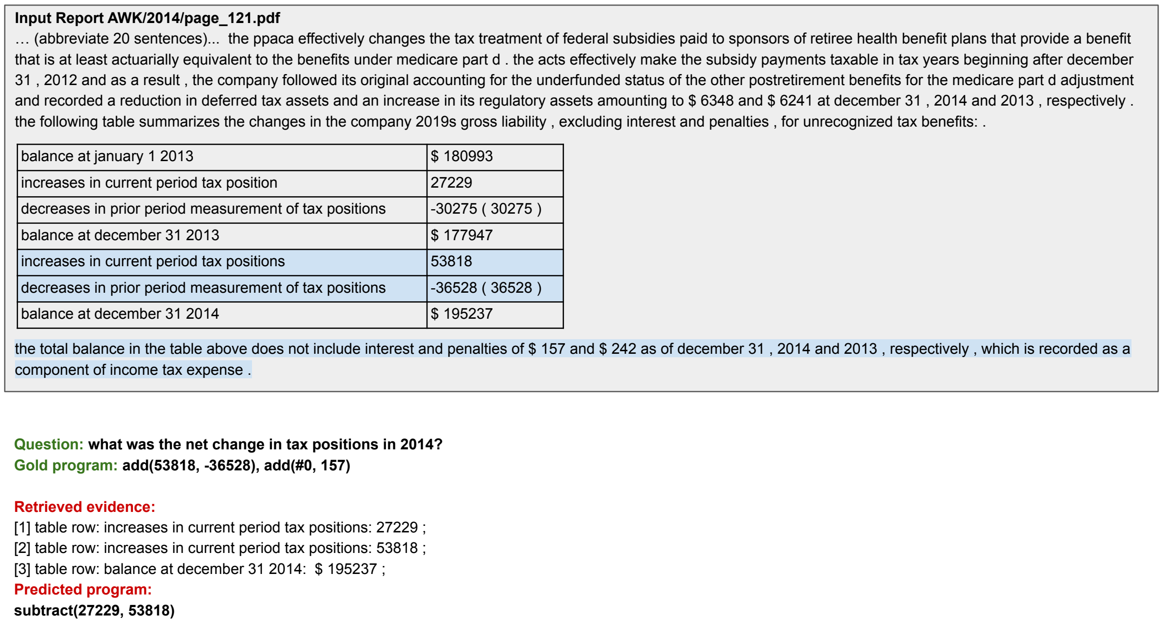
Baselines	**Execution Accuracy (%)**	**Program Accuracy (%)**
TF-IDF + Single Op	1.65	1.65
Retriever + Direct Generation	0.87	-
Pre-Trained Longformer (base)	23.83	22.56
Retriever + Seq2seq	18.76	17.52
Retriever + NeRd (BERT-base)	47.53	45.37
FinQANet (FinBert)	46.64	44.11
FinQANet (BERT-base)	49.91	47.15
FinQANet (BERT-large)	53.86	50.95
FinQANet (RoBerta-base)	56.27	53.49
FinQANet (RoBerta-large)	61.22	58.05

Table A.2: Results on validation set

A.4 Annotation Interface

We use Turkle[1] to build our annotation platform, which is a Django-based web application that can run in a local server. Figure A.3 and Figure A.4 show our annotation interface. After the annotators finish one example, they will use the validation check button to automatically check the validity of their inputs.

[1]https://github.com/hltcoe/turkle



Input Report AWK/2014/page_121.pdf
… (abbreviate 20 sentences)... the ppaca effectively changes the tax treatment of federal subsidies paid to sponsors of retiree health benefit plans that provide a benefit that is at least actuarially equivalent to the benefits under medicare part d . the acts effectively make the subsidy payments taxable in tax years beginning after december 31 , 2012 and as a result , the company followed its original accounting for the underfunded status of the other postretirement benefits for the medicare part d adjustment and recorded a reduction in deferred tax assets and an increase in its regulatory assets amounting to $ 6348 and $ 6241 at december 31 , 2014 and 2013 , respectively . the following table summarizes the changes in the company 2019s gross liability , excluding interest and penalties , for unrecognized tax benefits: .

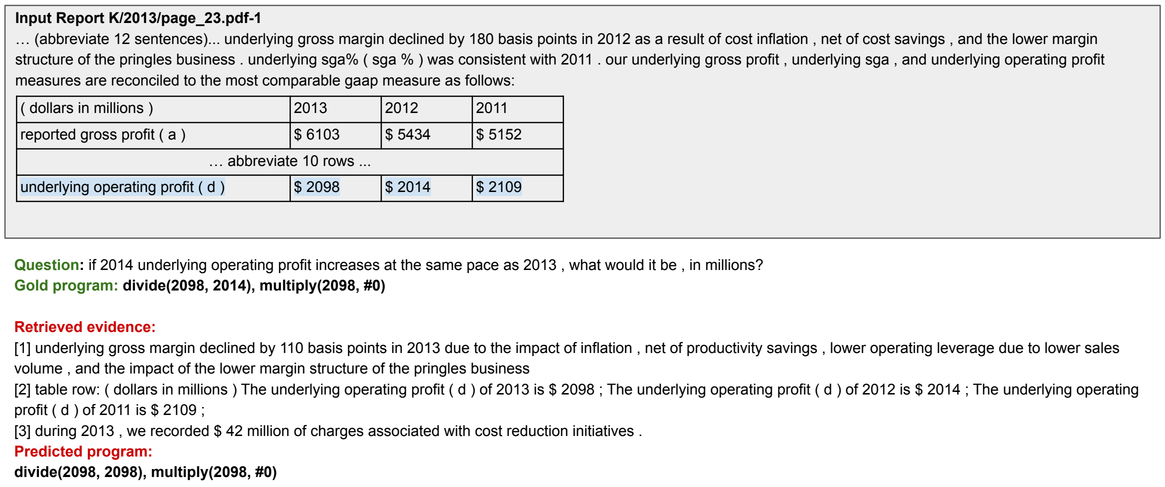
balance at january 1 2013	$ 180993
increases in current period tax position	27229
decreases in prior period measurement of tax positions	-30275 (30275)
balance at december 31 2013	$ 177947
increases in current period tax positions	53818
decreases in prior period measurement of tax positions	-36528 (36528)
balance at december 31 2014	$ 195237

the total balance in the table above does not include interest and penalties of $ 157 and $ 242 as of december 31 , 2014 and 2013 , respectively , which is recorded as a component of income tax expense .

Question: what was the net change in tax positions in 2014?
Gold program: add(53818, -36528), add(#0, 157)

Retrieved evidence:
[1] table row: increases in current period tax positions: 27229 ;
[2] table row: increases in current period tax positions: 53818 ;
[3] table row: balance at december 31 2014: $ 195237 ;
Predicted program:
subtract(27229, 53818)

Figure A.1: Error case study 1: The net change in the tax position is the sum of the increase and the decrease plus the penalties and interest. The model lacks this finance knowledge, thus the retriever fails to retrieve the correct table rows and sentences. Another challenging point is the table understanding, since in this case, it's hard to distinguish the retrieved two table rows for the year 2013 or 2014, using our method that regards each table row as basic unit. The model needs to look at the full table to get this global information.



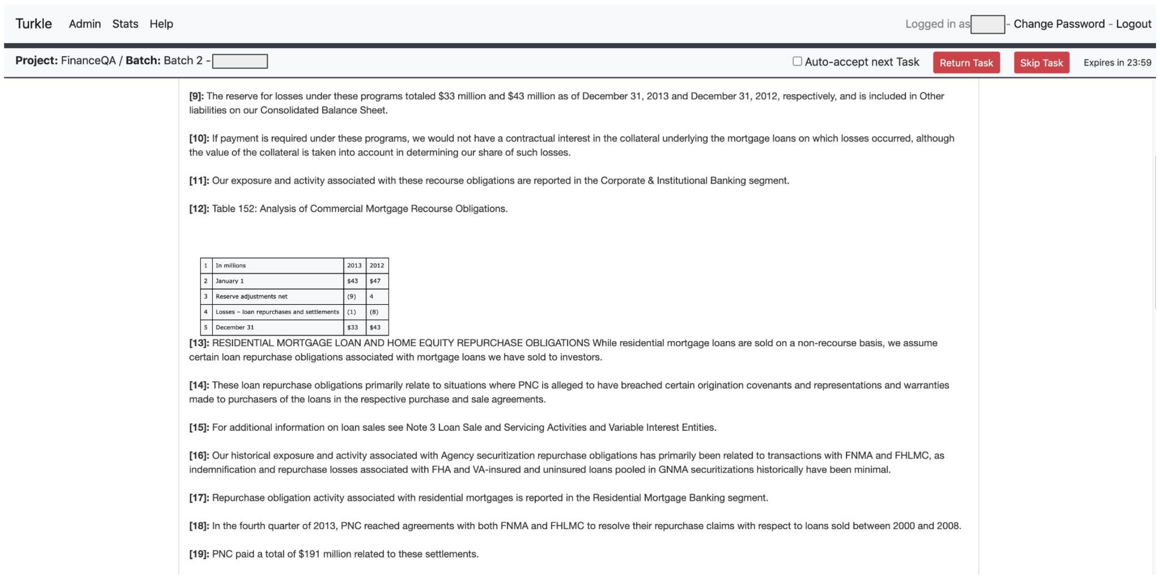
Input Report K/2013/page_23.pdf-1
… (abbreviate 12 sentences)... underlying gross margin declined by 180 basis points in 2012 as a result of cost inflation , net of cost savings , and the lower margin structure of the pringles business . underlying sga% (sga %) was consistent with 2011 . our underlying gross profit , underlying sga , and underlying operating profit measures are reconciled to the most comparable gaap measure as follows:

(dollars in millions)	2013	2012	2011
reported gross profit (a)	$ 6103	$ 5434	$ 5152
… abbreviate 10 rows ...			
underlying operating profit (d)	$ 2098	$ 2014	$ 2109

Question: if 2014 underlying operating profit increases at the same pace as 2013 , what would it be , in millions?
Gold program: divide(2098, 2014), multiply(2098, #0)

Retrieved evidence:
[1] underlying gross margin declined by 110 basis points in 2013 due to the impact of inflation , net of productivity savings , lower operating leverage due to lower sales volume , and the impact of the lower margin structure of the pringles business
[2] table row: (dollars in millions) The underlying operating profit (d) of 2013 is $ 2098 ; The underlying operating profit (d) of 2012 is $ 2014 ; The underlying operating profit (d) of 2011 is $ 2109 ;
[3] during 2013 , we recorded $ 42 million of charges associated with cost reduction initiatives .
Predicted program:
divide(2098, 2098), multiply(2098, #0)

Figure A.2: Error case study 2: Complex numerical reasoning.



Turkle Admin Stats Help Logged in as - Change Password - Logout

Project: FinanceQA / **Batch:** Batch 2 -

Auto-accept next Task Return Task Skip Task Expires in 23:59

[9]: The reserve for losses under these programs totaled $33 million and $43 million as of December 31, 2013 and December 31, 2012, respectively, and is included in Other liabilities on our Consolidated Balance Sheet.

[10]: If payment is required under these programs, we would not have a contractual interest in the collateral underlying the mortgage loans on which losses occurred, although the value of the collateral is taken into account in determining our share of such losses.

[11]: Our exposure and activity associated with these recourse obligations are reported in the Corporate & Institutional Banking segment.

[12]: Table 152: Analysis of Commercial Mortgage Recourse Obligations.

1	In millions	2013	2012
2	January 1	$43	$47
3	Reserve adjustments net	(9)	4
4	Losses – loan repurchases and settlements	(1)	(8)
5	December 31	$33	$43

[13]: RESIDENTIAL MORTGAGE LOAN AND HOME EQUITY REPURCHASE OBLIGATIONS While residential mortgage loans are sold on a non-recourse basis, we assume certain loan repurchase obligations associated with mortgage loans we have sold to investors.

[14]: These loan repurchase obligations primarily relate to situations where PNC is alleged to have breached certain origination covenants and representations and warranties made to purchasers of the loans in the respective purchase and sale agreements.

[15]: For additional information on loan sales see Note 3 Loan Sale and Servicing Activities and Variable Interest Entities.

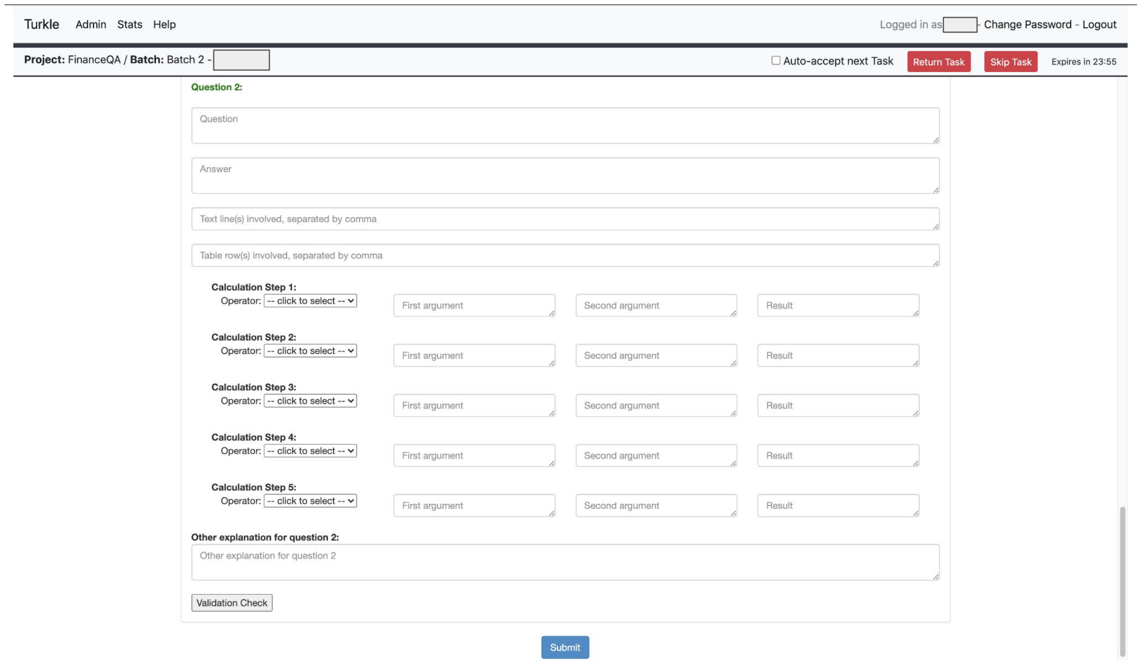
[16]: Our historical exposure and activity associated with Agency securitization repurchase obligations has primarily been related to transactions with FNMA and FHLMC, as indemnification and repurchase losses associated with FHA and VA-insured and uninsured loans pooled in GNMA securitizations historically have been minimal.

[17]: Repurchase obligation activity associated with residential mortgages is reported in the Residential Mortgage Banking segment.

[18]: In the fourth quarter of 2013, PNC reached agreements with both FNMA and FHLMC to resolve their repurchase claims with respect to loans sold between 2000 and 2008.

[19]: PNC paid a total of $191 million related to these settlements.

Figure A.3: Annotation interface: Display report.



Turkle Admin Stats Help Logged in as - Change Password - Logout

Project: FinanceQA / **Batch:** Batch 2 -

Auto-accept next Task Return Task Skip Task Expires in 23:55

Question 2:

Question

Answer

Text line(s) involved, separated by comma

Table row(s) involved, separated by comma

Calculation Step 1:
Operator: -- click to select -- First argument Second argument Result

Calculation Step 2:
Operator: -- click to select -- First argument Second argument Result

Calculation Step 3:
Operator: -- click to select -- First argument Second argument Result

Calculation Step 4:
Operator: -- click to select -- First argument Second argument Result

Calculation Step 5:
Operator: -- click to select -- First argument Second argument Result

Other explanation for question 2:

Other explanation for question 2

Validation Check

Submit

Figure A.4: Annotation interface: Annotator input fields.

Appendix B

NUANCED: Natural Utterance Annotation for Nuanced Conversation with Estimated Distributions

B.1 Model Implementation and Training Details

Figure B.1 presents the architecture of the BERT baseline. For each turn, we concatenate each slot with the current turn and the dialogue context as the input. On the `[CLS]` output, we add one head for slot prediction as binary classification, i.e., whether the input slot is updated in the current turn. For each slot, we add a specific head for value prediction. We use cross entropy loss for slot prediction, and mean squared loss for value distribution prediction. The overall loss is a weighted combination of the two losses. We set the weight for value prediction as 20.0. The threshold for value prediction in NUANCED-reduced is set as 0.5. We use BERT-base uncased model from the official

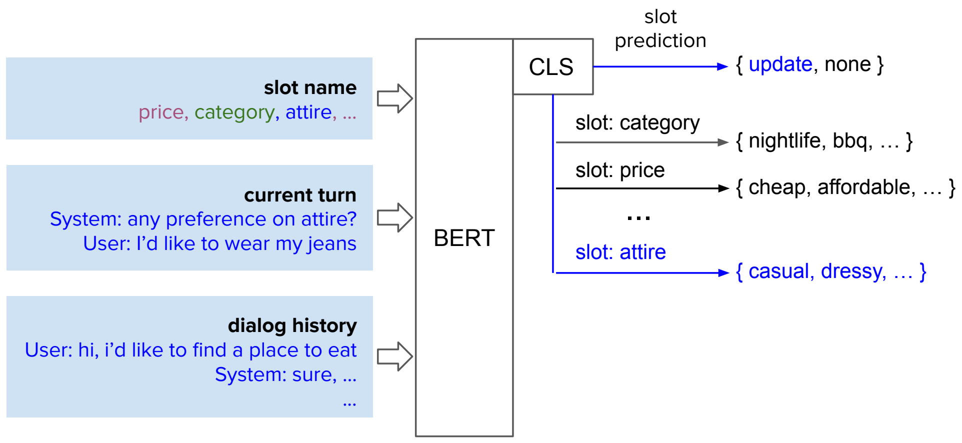


Figure B.1: Illustration of the BERT baseline

release[1] with 110M parameters; The learning rate is set as 3e-5, batch size as 32. We take the results based on the performance on validation set. For NUANCED-reduced, the training takes around 25,000 gradient steps; For NUANCED, the training takes around 40,000 steps. For the transformer model, to achieve best performance we use 6 layers and hidden size 300. All training is done on a single NVIDIA TESLA M40 card with 11G memory.

Note that for the slot "food category", some values are commonly observed in the dataset such as "American food", "nightlife", while some others are less frequently such as "Thai". During training we employ up-sampling for the less frequent ones.

In the construction of NUANCED, we sample a subset of the user history and aggregate to get the ground truth preference distributions. Because the number of viable values of each slot is different, for those slots with relatively more values the distribution generally presents 'long tail', we only take the top 3 value distributions for each slot. Correspondingly, during the model evaluation, we also take the top 3 predicted value distributions to calculate the MAE.

[1]https://github.com/google-research/bert

B.2 Analysis on Slots

We also study how the model performs on each slot in the domain, shown in Table B.1. Generally, slots that may involve more factoid knowledge or more choices of values are harder to learn, such as `food category`, `parking`. These may require learning long-tailed knowledge from external data.

Slot	**food category**	**price**	**parking**	**noise**
Mean MAE(1e-2)	15.48	15.29	16.94	13.34
Slot	**ambience**	**alcohol**	**wifi**	**attire**
Mean MAE(1e-2)	15.04	13.88	12.30	8.95

Table B.1: Performance for each slot of our dataset.

B.3 Case Studies

Table B.2 provides some case studies with ground truth and the BERT model predictions.

Dialogue Turns	**NUANCED-reduced**	**NUANCED**
Assistant: any preference on attire? **User**: I like shorts and a loose tee shirt in this heat.	**Gold labels**: Attire (casual= 1, dressy= 0, formal= 0)	**Gold Distributions**: Attire (casual= 1.00, dressy= 0.00, formal= 0.00)
	BERT predictions: Attire (casual= 1, dressy= 0, formal= 0)	**BERT predictions**: Attire (casual= 0.99, dressy= 0.01, formal= 0)
Assistant: what type of food would you like? **User**: Ribs would be perfect.	**Gold labels**: Category (traditional_american= 1.0, bbq= 1.0, nightlife= 0.0)	**Gold Distributions**: Category (traditional_american= 0.50, bbq= 0.50, nightlife= 0.00)
	BERT predictions: category (traditional_american= 1.0, nightlife= 1.0, new_american= 0.0)	**BERT predictions**: Category (traditional_american= 0.20, nightlife= 0.08, new_american= 0.09)
Assistant: any preference on alcohol? **User**: I really want a G&T or a Riesling, but I could also have a tonic water.	**Gold labels**: alcohol (full_bar= 1.0, beer_and_wine= 1.0, don't_serve= 1.0)	**Gold Distributions**: alcohol (full_bar= 0.78, beer_and_wine= 0.33, don't_serve= 0.11)
	BERT predictions: alcohol (full_bar= 1.0, beer_and_wine= 1.0, don't_serve= 1.0)	**BERT predictions**: alcohol (full_bar= 0.55, beer_and_wine= 0.47, don't_serve= 0.09)
Assistant: what parking option would you like? **User**: I need something fuss-free and out of the rain for my car, Also, I really want a gin and tonic, but it's not a complete deal-breaker if I can't have it.	**Gold labels**: parking (garage= 1.0, valet= 0.0, validated= 0.0) alcohol (full_bar= 1.0, beer_and_wine= 1.0, don't_serve= 1.0)	**Gold Distributions**: parking (garage= 0.86, valet= 0.00, validated= 0.00) alcohol (full_bar= 0.93, beer_and_wine= 0.21, don't_serve= 0.14)
	BERT predictions: parking (garage= 1.0, valet= 1.0, lot= 1.0) alcohol (full_bar= 1.0, beer_and_wine= 1.0, don't_serve= 1.0)	**BERT predictions**: parking (garage= 0.78, valet= 0.41, lot= 0.34) alcohol (full_bar= 0.79, beer_and_wine= 0.17, don't_serve= 0.12)
(after some turns)		
Assistant: here're the recommendations. **User**: You know what, if it's going to be a fancier place then I don't mind dealing with more complicated parking after all.	**Gold labels**: parking (garage= 1.0, valet= 1.0, validated= 1.0)	**Gold Distributions**: parking (garage= 0.86, valet= 0.64, validated= 0.21)
	BERT predictions: parking (garage= 1.0, lot= 1.0, validated= 1.0)	**BERT predictions**: parking (garage= 0.67, valet= 0.48, lot= 0.40)

Table B.2: Some case studies. the last example shows two turns in a dialogue and corresponding distributions for each turn. The user updates the preference in a later turn based on a previous turn.

Appendix C

Few-Shot NLG with Pre-Trained Language Model

C.1 Implementation Details

We use the Adam optimizer [61] with learning rate set to 0.0003. The mini-batch size is set to 40 and the weight λ of the copy loss term to 0.7. The dimension of the position embedding is set to 5. For attribute name with multiple words, we average their word embeddings as the attribute name embedding. Refer to our released code and data at `https://github.com/czyssrs/Few-Shot-NLG` for more details.

C.2 ROUGE-4 Results

Following previous work [26], we conduct automatic evaluations using BLEU-4 and ROUGE-4 (F-measure)[1]. Table C.1, C.2 and C.3 show the ROUGE-4 results for three domains *Humans*, *Books* and *Songs*, respectively.

[1]We use standard scripts NIST mteval-v13a.pl (for BLEU), and rouge-1.5.5 (for ROUGE)

Domain	Humans				
# of training instances	-	50	100	200	500
Template	5.1	-	-	-	-
Base-original	-	0.1	0.4	0.5	0.6
Base	-	0.1	0.4	0.8	1.5
Base+switch	-	4.9	6.3	9.8	12.5
Base+switch+LM-scratch	-	1.0	2.8	4.7	7.1
Base+switch+LM (Ours)	-	**14.1**	**16.2**	**22.1**	**28.3**

Table C.1: ROUGE-4 results on *Humans* domain

Domain	Books				
# of training instances	-	50	100	200	500
Template	15.0	-	-	-	-
Base-original	-	1.1	1.6	2.1	1.5
Base	-	1.7	1.5	2.1	2.4
Base+switch	-	12.8	15.0	18.1	20.7
Base+switch+LM-scratch	-	2.4	4.2	6.5	10.7
Base+switch+LM (Ours)	-	**22.5**	**23.1**	**25.0**	**27.6**

Table C.2: ROUGE-4 results on *Books* domain

Domain	Songs				
# of training instances	-	50	100	200	500
Template	24.5	-	-	-	-
Base-original	-	3.4	4.2	4.7	4.8
Base	-	4.1	5.1	4.7	5.8
Base+switch	-	20.2	21.7	23.2	24.8
Base+switch+LM-scratch	-	5.4	8.0	12.0	15.0
Base+switch+LM (Ours)	-	**26.2**	**28.6**	**30.1**	**32.6**

Table C.3: ROUGE-4 results on *Songs* domain

C.3 Human Evaluation Details

We conduct human evaluation studies using Amazon Mechanical Turk, based on two aspects: *Factual correctness* and *Language naturalness*. For both studies, we evaluate the results trained with 200 training instances of *Humans* domain. We randomly sample 500 instances from the test set, together with the texts generated with different methods. Each evaluation unit is assigned to 3 workers to eliminate human variance.

The first study attempts to evaluate how well a generated text can correctly convey information in the table. Each worker is present with *both the input table and a generated text*, and asked to count how many facts in the generated text are supported by the table, and how many are contradicting with or missing from the table, similar as in [95]. The we calculate the average number of supporting and contradicting facts for the texts generated by each method.

The second study aims to evaluate whether the generated text is grammatically correct and fluent in terms of language, regardless of factual correctness. Each worker is present with a pair of texts generated from the same input table, by two different methods, then asked to select the better one only according to language naturalness, or "Tied" if the two texts are of equal quality. *The input table is not shown to the workers.* Each time a generated text is chosen as the better one, we assign score of 1.0. If two texts are tied, we assign 0.5 for each. We then calculate the average score for the texts generated by each method, indicating its superiority in pairwise comparisons with all other methods.

The significance test is conducted respectively on all three measures: number of supporting facts and number of contradicting facts for the first study; the assigned score for the second study. We use the Tukey HSD post-hoc analysis of an ANOVA with the worker's response as the dependent variable, the method and worker id as independent variables.

Appendix D

Logic2Text: High-Fidelity Natural Language Generation from Logical Forms

D.1 Logic Type Definitions & Logical Form Annotation

D.1.1 Logic Type Definitions

We define all 7 logic types in our dataset and provide examples based on the following table in Figure D.1.

`Count`: counting some rows in the table based on the values in one column, with the scope of all table rows or a subset of rows.

Example descriptions: "in opec 2012, there were 4 countries from africa.", "in opec 2012, among the countries from africa, 2 of them joined after 1970.", etc.

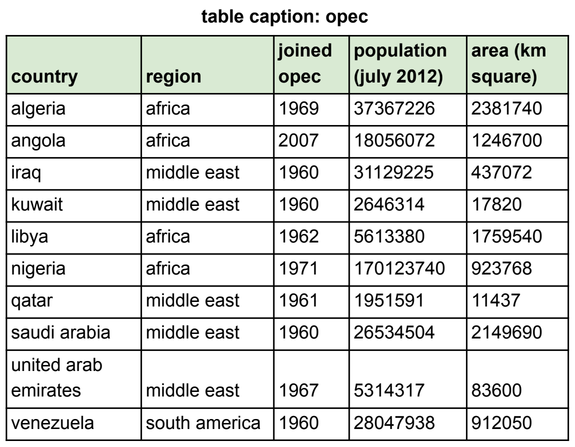


table caption: opec

country	region	joined opec	population (july 2012)	area (km square)
algeria	africa	1969	37367226	2381740
angola	africa	2007	18056072	1246700
iraq	middle east	1960	31129225	437072
kuwait	middle east	1960	2646314	17820
libya	africa	1962	5613380	1759540
nigeria	africa	1971	170123740	923768
qatar	middle east	1961	1951591	11437
saudi arabia	middle east	1960	26534504	2149690
united arab emirates	middle east	1967	5314317	83600
venezuela	south america	1960	28047938	912050

Figure D.1: Example table

`Superlative`: Describing the maximum or minimum value in a column, with the scope of all table rows or a subset of rows. You may also talk about other columns on this row with the superlative value.
Example descriptions: “in opec in 2012, angola, from africa, was the latest country to join.”, “among the member countries in opec in 2012 from the middle east, qatar was the smallest in area.”, etc.

`Ordinal`: Describing the n-th maximum or minimum value in a column, with the scope of all table rows or a subset of rows. You may also talk about other columns on this row with the n-th maximum or minimum value.
Example descriptions: “in opec in 2012, qatar was the 5th country to join.”, “Among the africa member countries, algeria was the 2nd earliest to join.”, etc.

`Comparative`: Comparing two rows in the table, regarding their values in one column. You may also talk about other columns on these two rows.
Example descriptions: “in opec in 2012, libiya joined 2 years later than kuwait.”, “in opec in 2012, algeria, from africa, had a larger population than iraq from the middle east.”

`Aggregation`: Describing the sum or average value over a column, with the scope of all table rows or a subset of rows.

Example descriptions: “in opec 2012, the countries from africa had an average population of around 57,800,000.”, etc.

`Unique`: Describing one unique row, regarding one column, with the scope of all table rows or a subset of rows. You may also talk about other columns on this unique row.

Example descriptions: “in opec 2012, angola was the only country to join after 2000.”, “in 2012, among the member countries from africa, the only one to join opec after 2000 is angola.”, etc.

`Majority`: Describing the majority values (most or all) over one column, with the scope of all table rows or a subset of rows.

Example descriptions: “in opec 2012, most countries joined before 2000.”, “in opec 2012, all of the africa member countries had an area larger than 900,000.”, etc.

Logical Form Annotation

Here we provide the question sets for annotating each logical type.

`Count`: (1). Choose whether the counting is performed on the scope of all table rows, or on a subset of all rows. (2). Select the table column that the counting is performed on. (3). Select the criterion, based on which we filter the table records to be counted. Here we consider the following criterion: ”equal”, ”not equal”, ”less than”, ”less than or equal to”, ”greater than”, ”greater than or equal to”, ”fuzzily match”, ”all” (or ”other” if none of the above is correct). (4). Based on the selected criterion, write the value to be filtered for counting. (5). Write down the result of the counting.

`Superlative`: (1). Is the superlative action performed on the scope of all table rows, or on a subset of all rows? (2). What is the table column that the superlative action is performed on? (3). Is the superlative action taking the numerical maximum, or minimum value among the records? (4). What is the table row containing this superlative value? (5). On this row with the superlative value, what are the other column(s) mentioned? If not any other column is mentioned, write 'n/a'. (6). Is this superlative value itself mentioned in the statement?

`Aggregation`: (1). Choose whether the aggregation is performed on the scope of all table rows, or on a subset of all rows. (2). Select the table column that the aggregation is performed on. (3). What is the type of this aggregation, sum or average? (4). What is the result of this aggregation?

`Comparative`: (1). Which column is the statement comparing? (2). What is the first row to be compared? (3). What is the second row to be compared? (4). What is the relationship comparing the records numerically in the first row with the second? (choose from "greater", "less", "equal", "not equal", "difference value", or "other" if not any of the above. Here we consider the relationship between actual numerical values between two records, NOT the relationship expressed in the statement) (5). Is the compared records itself mentioned in the statement? (6). What are the other column(s) of these two rows mentioned in the statement?

`Majority`: (1). What is the scope of this majority? (2). Which column the statement is describing? (3). Is the statement describing all the records or most frequent records within the scope? (4). Select the criterion, based on which we filter records to describe the majority. Here we consider the following criterion: "equal", "not equal", "less than", "less than or equal to", "greater than", "greater than or equal to", "fuzzily match" (or "other" if none of the above is correct). (5). Based on the selected criterion, write the value to be filtered for describing the majority.

`Ordinal`: (1). What is the scope that the ordinal description is performed on? (all rows or a subset of rows) (2). What is the table column that the ordinal description is based on? (3). Is the ordinal description based on a numerically max to min or min to max ranking of the column records? (4). What is the order described in the statement, based on this ranking? (5). What is the table row containing this n-th record ? (6). On this row, what are the other column(s) mentioned? If not any other column is mentioned, write 'n/a'. (7). Is this n-th record itself mentioned in the statement?

`Unique`: (1). What is the scope of this statement describing unique row? (2). What is this unique row? (3). Write the table column that shows the uniqueness of this row (4). Select the criterion, based on which we filter records in this column to find the unique row. Here we consider the following criterion: "equal", "not equal", "less than", "greater than", "fuzzily match" (or "other" if none of the above is correct). (5). Based on the selected criterion, write the value to be filtered for the unqiue row. (6). On this unique row, what are the other column(s) mentioned (except the column describing the scope)? If not any other column is mentioned, write 'n/a'.

D.2 Function Definitions

Here we list the function definitions and descriptions for our logical form in table D.1. Note that since the tables in WikiTables are not standard database table, but semi-structured tables, the cell values are often not well-formatted with a lot of mixed strings and numbers, dates in different formats, etc. Therefore for some functions involving arithmetic operations on table cell values, we only specify a coarse "object" type for the arguments, and then parse the numerical or date type values in the function implementations. Refer to our released code for detailed implementations.

Name	Arguments	Output	Description
count	view	number	returns the number of rows in the view
only	view	bool	returns whether there is exactly one row in the view
hop	row, header string	object	returns the value under the header column of the row
and	bool, bool	bool	returns the boolean operation result of two arguments
max/min/avg/sum	view, header string	number	returns the max/min/average/sum of the values under the header column
nth_max/nth_min	view, header string	number	returns the n-th max/n-th min of the values under the header column
argmax/argmin	view, header string	row	returns the row with the max/min value in header column
nth_argmax/nth_argmin	view, header string	row	returns the row with the n-th max/min value in header column
eq/not_eq	object, object	bool	returns if the two arguments are equal
round_eq	object, object	bool	returns if the two arguments are roughly equal under certain tolerance
greater/less	object, object	bool	returns if argument 1 is greater/less than argument 2
diff	object, object	object	returns the difference between two arguments
filter_eq/not_eq	view, header string, object	view	returns the subview whose values under the header column is equal/not equal to argument 3
filter_greater/less	view, header string, object	view	returns the subview whose values under the header column is greater/less than argument 3
filter_greater_eq /less_eq	view, header string, object	view	returns the subview whose values under the header column is greater/less or equal than argument 3
filter_all	view, header string	view	returns the view itself for the case of describing the whole table
all_eq/not_eq	view, header string, object	bool	returns whether all the values under the header column are equal/not equal to argument 3
all_greater/less	view, header string, object	bool	returns whether all the values under the header column are greater/less than argument 3
all_greater_eq/less_eq	view, header string, object	bool	returns whether all the values under the header column are greater/less or equal to argument 3
most_eq/not_eq	view, header string, object	bool	returns whether most of the values under the header column are equal/not equal to argument 3
most_greater/less	view, header string, object	bool	returns whether most of the values under the header column are greater/less than argument 3
most_greater_eq/less_eq	view, header string, object	bool	returns whether most of the values under the header column are greater/less or equal to argument 3

Table D.1: Function definitions

D.3 Model Implementation Details

Here we provide some implementation details of the baseline models.

Template Some example templates are listed below. Texts in braces is optional depending on the logical form.

`count`:

in [table_caption], (among the ones whose [scope_column] are [equal to/greater than/...] [scope_value]), there are [result] ones whose [column_name] are [equal to/greater than/...] [value] .

`superlative`:

in [table_caption], (among the ones whose [scope_column] are [equal to/greater than/...] [scope_value]), the [max/minimum] [column_name] is [value].

in [table_caption], (among the ones whose [scope_column] are [equal to/greater than/...] [scope_value]), [subject], with ([other_col1] [other_val];...), has the [max/minimum] [column_name], ([value]).

`ordinal`:

similar as `superlative`, replace max/minimum as n-th max/minimum.

`comparative`:

in [table_caption], [subject1] has [greater/less/...] [column_name] than [subject2].

in [table_caption], [subject1] has [diff_value] [column_name] [greater/less/...] than [subject2].

in [table_caption], [subject1], with ([other_col1] [other_val];...), has [greater/less/...] [column_name] than [subject2], with ([other_col1] [other_val];...).

`unique`:

in [table_caption], (among the ones whose [scope_column] are [equal to/greater than/...] [scope_value]), there is only one of them whose [column_name] is [greater/less /...] than [value].

in [table_caption], (among the ones whose [scope_column] are [equal to/greater than/...] [scope_value]), the only one whose [column_name] is [greater/less/...] than [value] is for [subject], with ([other_col1] [other_val];...).

`aggregation`:

in [table_caption], (among the ones whose [scope_column] are [equal to/greater than/...] [scope_value]), the [average/sum] of [column_name] is [result].

`majority`:

in [table_caption], (among the ones whose [scope_column] are [equal to/greater than/...] [scope_value]), [most/all] of them has [column_name] [equal to/greater than/ ...] [majority_value].

For all neural models we use Byte-Pair Encoding (BPE) [114] and the subword vocabulary used in [98]. We use the pre-trained word embeddings from [98] and project to certain smaller dimensions (300) as the word embeddings. The batch size of all models are set to 32. The beam size is set to 3. As the table content only serves as context information for generation, to save GPU memory we set the maximum length of the table content as 200. The hyperparameters are chosen based on manual tuning regarding the BLEU score on the validation set.

Seq2seq+att & pointer-generator The learning rate is set to 0.001. For seq2seq, the training takes around 16000 gradient steps. For pointer generator, training takes around 5000 steps.

Graph2seq+copy we reuse the code skeleton from the released code from [140]. The table caption and header are first fed into a seq2seq, then the final hidden state is used to initialize the nodes of the graph encoder. When applying attention and copy, for graph nodes, we concatenate the token embedding and the embedding of its node as the embedding for the token. The learning rate is set to 0.0005. Training takes around 11000 steps.

Transformer+copy we mostly follow the structure setting in the original Transformer model [83]. We use 4 attention heads and 6 layers. The final hidden layer is used for calculating the attention score and the copy switch. We also add the segment embeddings for different input components similar as [39]. The learning rate is set to 0.0005. training takes around 32000 steps.

GPT-2 We use the GPT-2 small 117M model from the released code and pre-trained model from [98]. Word embeddings are fixed during training. The learning rate is set to 0.0003. The training takes around 500 steps to converge.

All the experiments are run on GeForce GTX 1080Ti GPU. Table D.2 shows the validation performance of different baselines.

Models	B-4	R-1	R-2	R-4	R-L
Template	17.81	51.16	24.89	6.68	38.12
Seq2seq+att	12.26	35.44	15.68	4.81	30.36
Pointer generator	25.43	57.35	31.97	12.33	48.11
Graph2seq+copy	25.65	57.65	31.98	12.29	48.28
Transformer+copy	27.20	59.70	34.06	14.03	48.71
GPT-2	**32.98**	**64.86**	**40.02**	**18.38**	**54.59**

Table D.2: Automatic evaluation results for validation set.

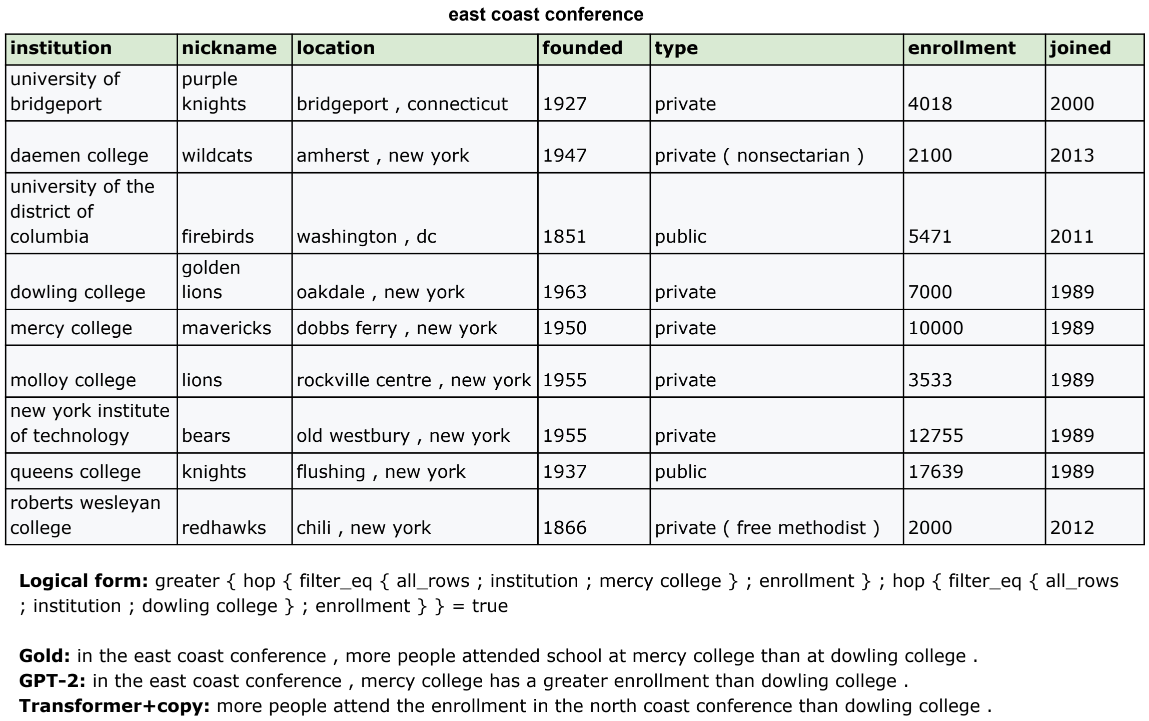
D.4 Human Evaluation Details

Human Evaluations on AMT We randomly sample 500 examples from the top two best performing methods (GPT-2 and Transformer+copy), and the gold references. The evaluations are conducted on two axes: *factual correctness* and *language fluency*. For factual correctness, we provide the workers with both the table and the description, and ask them to verify whether the description is factually correct based on the table. If the description contains too many grammar errors to be readable, the worker is instructed to select "incorrect". Minor grammar errors can be accepted, as long as the worker can understand the meanings. For language fluency, we conduct pairwise comparison between the three methods. For this evaluation we only present the pair of descriptions to the worker, and ask them to select a better one only based on language fluency (a better description should be fluent, coherent, and free of grammar errors), or select "Tied" if the two descriptions are of similar quality. For both evaluations we distribute each task to 3 workers to eliminate human variance.

Human Expert Evaluation To conduct precise evaluation of semantic correctness, i.e., whether the generation correctly matches the meaning of the logical form, we invite human experts (two computer science graduate students) to perform the evaluation. We sample 200 examples from each method and ask them to verify whether the description correctly presents the meaning of the logic form, with neither insufficient nor redundant information. The description should also be fluent and free of grammar errors. Therefore this evaluation can be seen as a comprehensive evaluation of the generation quality. Each example is examined by both students and the decision is made after discussion.

D.5 Generation Examples

We provide 2 examples of generations in Figure D.2 and Figure D.3.



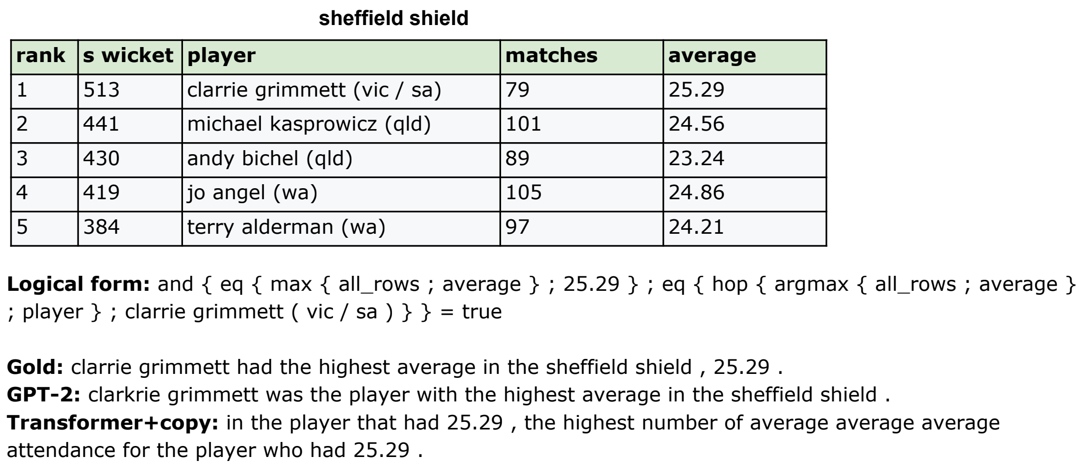
east coast conference

institution	nickname	location	founded	type	enrollment	joined
university of bridgeport	purple knights	bridgeport , connecticut	1927	private	4018	2000
daemen college	wildcats	amherst , new york	1947	private (nonsectarian)	2100	2013
university of the district of columbia	firebirds	washington , dc	1851	public	5471	2011
dowling college	golden lions	oakdale , new york	1963	private	7000	1989
mercy college	mavericks	dobbs ferry , new york	1950	private	10000	1989
molloy college	lions	rockville centre , new york	1955	private	3533	1989
new york institute of technology	bears	old westbury , new york	1955	private	12755	1989
queens college	knights	flushing , new york	1937	public	17639	1989
roberts wesleyan college	redhawks	chili , new york	1866	private (free methodist)	2000	2012

Logical form: greater { hop { filter_eq { all_rows ; institution ; mercy college } ; enrollment } ; hop { filter_eq { all_rows ; institution ; dowling college } ; enrollment } } = true

Gold: in the east coast conference , more people attended school at mercy college than at dowling college .
GPT-2: in the east coast conference , mercy college has a greater enrollment than dowling college .
Transformer+copy: more people attend the enrollment in the north coast conference than dowling college .

Figure D.2: Example generations.



sheffield shield

rank	s wicket	player	matches	average
1	513	clarrie grimmett (vic / sa)	79	25.29
2	441	michael kasprowicz (qld)	101	24.56
3	430	andy bichel (qld)	89	23.24
4	419	jo angel (wa)	105	24.86
5	384	terry alderman (wa)	97	24.21

Logical form: and { eq { max { all_rows ; average } ; 25.29 } ; eq { hop { argmax { all_rows ; average } ; player } ; clarrie grimmett (vic / sa) } } = true

Gold: clarrie grimmett had the highest average in the sheffield shield , 25.29 .
GPT-2: clarkrie grimmett was the player with the highest average in the sheffield shield .
Transformer+copy: in the player that had 25.29 , the highest number of average average average attendance for the player who had 25.29 .

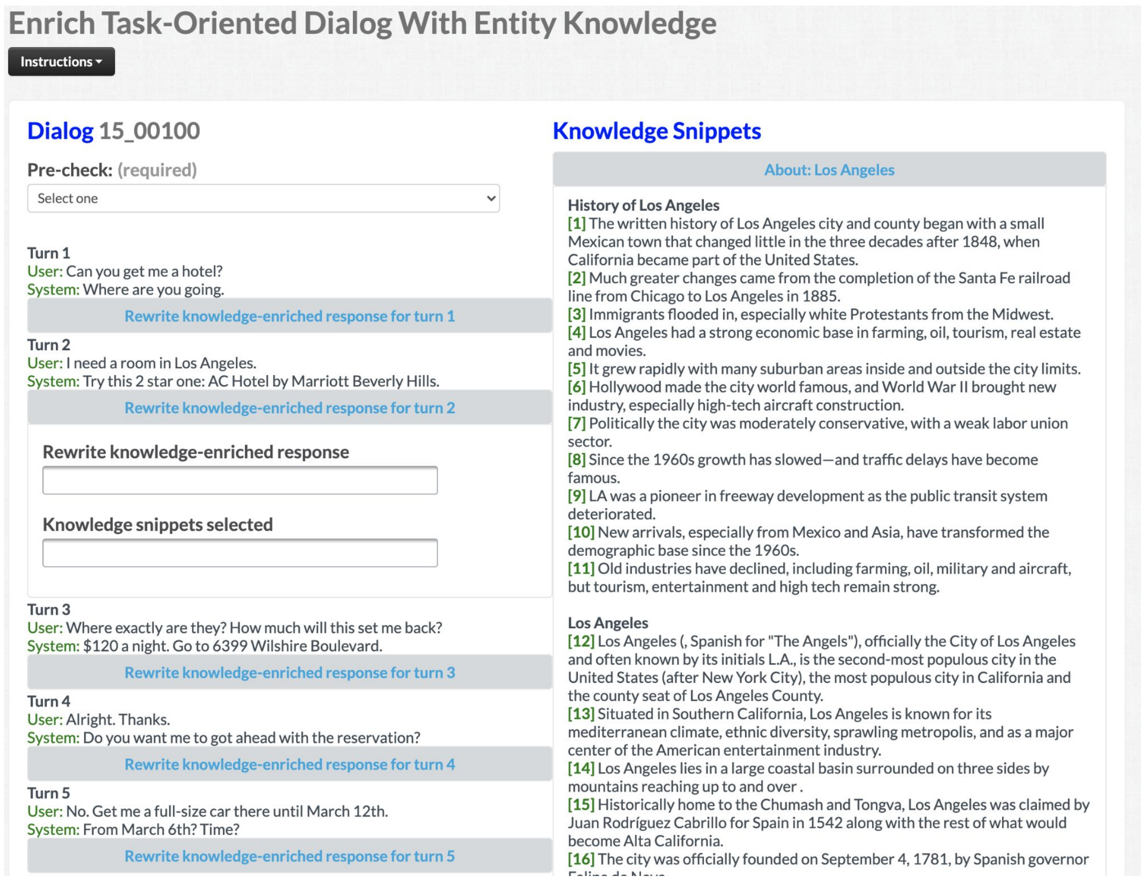
Figure D.3: Example generations.

Appendix E

KETOD: Knowledge-Enriched Task-Oriented Dialogue

E.1 Dataset Construction

Figure E.1 shows our annotation interface to add knowledge-grounded chit-chat to TOD. The left part shows the full dialogue, where the annotators can click and expand each turn to make the chit-chat enrichment. The right part shows all the entities with the associated knowledge snippets. The annotators can click on each entity name to expand the textbox to see the knowledge snippets. We add index number to each knowledge snippet (shown in green brackets), and the annotators are asked to write down the indexes of the knowledge snippets they used for writing the knowledge grounded chit-chat. Figure E.2 shows one example annotation turn using our interface.


Enrich Task-Oriented Dialog With Entity Knowledge

Dialog 15_00100

Pre-check: (required)

Turn 1
User: Can you get me a hotel?
System: Where are you going.

Rewrite knowledge-enriched response for turn 1

Turn 2
User: I need a room in Los Angeles.
System: Try this 2 star one: AC Hotel by Marriott Beverly Hills.

Rewrite knowledge-enriched response for turn 2

Rewrite knowledge-enriched response

Knowledge snippets selected

Turn 3
User: Where exactly are they? How much will this set me back?
System: $120 a night. Go to 6399 Wilshire Boulevard.

Turn 4
User: Alright. Thanks.
System: Do you want me to got ahead with the reservation?

Turn 5
User: No. Get me a full-size car there until March 12th.
System: From March 6th? Time?

Knowledge Snippets

About: Los Angeles

History of Los Angeles
[1] The written history of Los Angeles city and county began with a small Mexican town that changed little in the three decades after 1848, when California became part of the United States.
[2] Much greater changes came from the completion of the Santa Fe railroad line from Chicago to Los Angeles in 1885.
[3] Immigrants flooded in, especially white Protestants from the Midwest.
[4] Los Angeles had a strong economic base in farming, oil, tourism, real estate and movies.
[5] It grew rapidly with many suburban areas inside and outside the city limits.
[6] Hollywood made the city world famous, and World War II brought new industry, especially high-tech aircraft construction.
[7] Politically the city was moderately conservative, with a weak labor union sector.
[8] Since the 1960s growth has slowed—and traffic delays have become famous.
[9] LA was a pioneer in freeway development as the public transit system deteriorated.
[10] New arrivals, especially from Mexico and Asia, have transformed the demographic base since the 1960s.
[11] Old industries have declined, including farming, oil, military and aircraft, but tourism, entertainment and high tech remain strong.

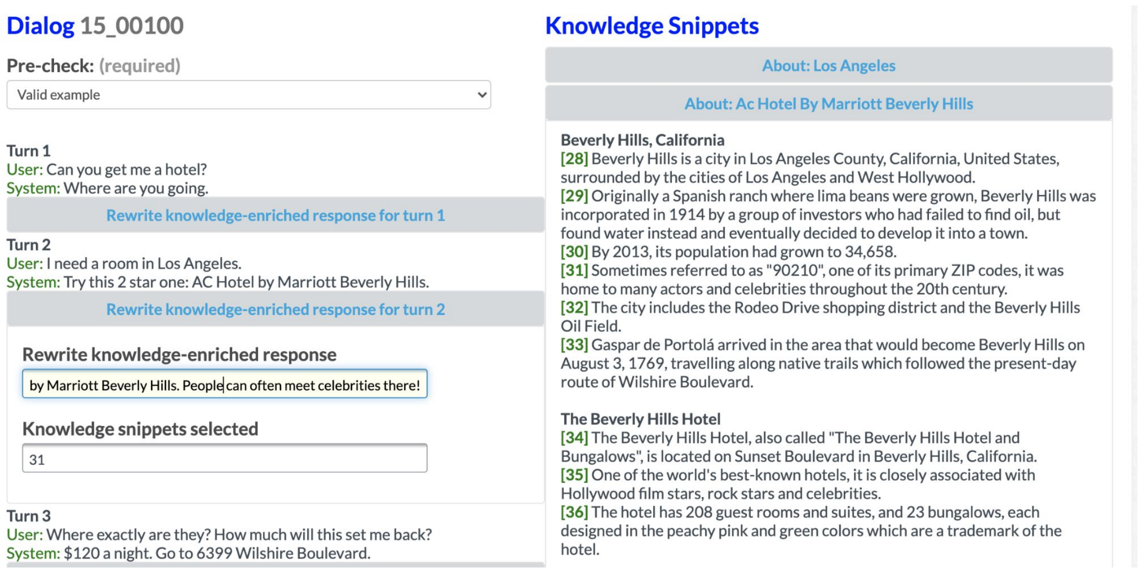
Los Angeles
[12] Los Angeles (, Spanish for "The Angels"), officially the City of Los Angeles and often known by its initials L.A., is the second-most populous city in the United States (after New York City), the most populous city in California and the county seat of Los Angeles County.
[13] Situated in Southern California, Los Angeles is known for its mediterranean climate, ethnic diversity, sprawling metropolis, and as a major center of the American entertainment industry.
[14] Los Angeles lies in a large coastal basin surrounded on three sides by mountains reaching up to and over .
[15] Historically home to the Chumash and Tongva, Los Angeles was claimed by Juan Rodríguez Cabrillo for Spain in 1542 along with the rest of what would become Alta California.
[16] The city was officially founded on September 4, 1781, by Spanish governor

Figure E.1: Our annotation interface example 1.

E.2 Model and Training Details

All the implementations are based on the Huggingface Transformers library[1]. For all models, we use the Adam optimizer [61]. For the knowledge selection model, we use BERT-base with learning rate of 3e-5 and batch size of 16. For the baseline SimpleToD model, SimpleToDPlus model, and Combiner model, we all use learning rate of 1e-4 and batch size of 16. All the experiments are done using TESLA M40 GPU cards.

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


Dialog 15_00100

Pre-check: (required)

Valid example

Turn 1
User: Can you get me a hotel?
System: Where are you going.

Rewrite knowledge-enriched response for turn 1

Turn 2
User: I need a room in Los Angeles.
System: Try this 2 star one: AC Hotel by Marriott Beverly Hills.

Rewrite knowledge-enriched response for turn 2

Rewrite knowledge-enriched response

by Marriott Beverly Hills. People can often meet celebrities there!

Knowledge snippets selected

31

Turn 3
User: Where exactly are they? How much will this set me back?
System: $120 a night. Go to 6399 Wilshire Boulevard.

Knowledge Snippets

About: Los Angeles

About: Ac Hotel By Marriott Beverly Hills

Beverly Hills, California
[28] Beverly Hills is a city in Los Angeles County, California, United States, surrounded by the cities of Los Angeles and West Hollywood.
[29] Originally a Spanish ranch where lima beans were grown, Beverly Hills was incorporated in 1914 by a group of investors who had failed to find oil, but found water instead and eventually decided to develop it into a town.
[30] By 2013, its population had grown to 34,658.
[31] Sometimes referred to as "90210", one of its primary ZIP codes, it was home to many actors and celebrities throughout the 20th century.
[32] The city includes the Rodeo Drive shopping district and the Beverly Hills Oil Field.
[33] Gaspar de Portolá arrived in the area that would become Beverly Hills on August 3, 1769, travelling along native trails which followed the present-day route of Wilshire Boulevard.

The Beverly Hills Hotel
[34] The Beverly Hills Hotel, also called "The Beverly Hills Hotel and Bungalows", is located on Sunset Boulevard in Beverly Hills, California.
[35] One of the world's best-known hotels, it is closely associated with Hollywood film stars, rock stars and celebrities.
[36] The hotel has 208 guest rooms and suites, and 23 bungalows, each designed in the peachy pink and green colors which are a trademark of the hotel.

Figure E.2: Our annotation interface example 2.

Metrics	SimpleToDPlus win (%)	SimpleToD win (%)	Tied (%)
Engagingness	40.0	30.3	29.8
Interestingness	31.8	19.5	48.8
Knowledge	38.0	18.3	43.8
Humanness	38.3	26.8	35.0

Table E.1: Human evaluation of SimpleToDPlus vs. SimpleToD.

E.3 Evaluation Details

Table E.1 and E.2 show the human evaluation results of SimpleToDPlus vs. SimpleToD, and Combiner vs. SimpleToD, respectively.

Metrics	**Combiner win (%)**	**SimpleToD win (%)**	**Tied (%)**
Engagingness	34.8	33.5	31.8
Interestingness	27.0	22.5	50.5
Knowledge	32.5	23.0	44.5
Humanness	27.8	32.5	39.8

Table E.2: Human evaluation of Combiner vs. SimpleToD.

Bibliography

[1] M. A. C. Soares and F. S. Parreiras, *A literature review on question answering techniques, paradigms and systems*, *J. King Saud Univ. Comput. Inf. Sci.* **32** (2020), no. 6 635–646.

[2] A. M. N. Allam and M. H. Haggag, *The question answering systems: A survey*, *International Journal of Research and Reviews in Information Sciences (IJRRIS)* **2** (2012), no. 3.

[3] K. Höffner, S. Walter, E. Marx, R. Usbeck, J. Lehmann, and A. N. Ngomo, *Survey on challenges of question answering in the semantic web*, *Semantic Web* **8** (2017), no. 6 895–920.

[4] D. Chen, *Neural reading comprehension and beyond.* Stanford University, 2018.

[5] S. Liu, X. Zhang, S. Zhang, H. Wang, and W. Zhang, *Neural machine reading comprehension: Methods and trends*, *CoRR* **abs/1907.01118** (2019) [arXiv:1907.0111].

[6] D. Diefenbach, V. López, K. D. Singh, and P. Maret, *Core techniques of question answering systems over knowledge bases: a survey*, *Knowl. Inf. Syst.* **55** (2018), no. 3 529–569.

[7] X. Yao and B. V. Durme, *Information extraction over structured data: Question answering with freebase*, in *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers*, pp. 956–966, The Association for Computer Linguistics, 2014.

[8] F. Zhu, W. Lei, C. Wang, J. Zheng, S. Poria, and T. Chua, *Retrieving and reading: A comprehensive survey on open-domain question answering*, *CoRR* **abs/2101.00774** (2021) [arXiv:2101.0077].

[9] Y. Yang, W. Yih, and C. Meek, *Wikiqa: A challenge dataset for open-domain question answering*, in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September*

17-21, 2015 (L. Màrquez, C. Callison-Burch, J. Su, D. Pighin, and Y. Marton, eds.), pp. 2013–2018, The Association for Computational Linguistics, 2015.

[10] D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner, *DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs*, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), pp. 2368–2378, Association for Computational Linguistics, 2019.

[11] A. Amini, S. Gabriel, S. Lin, R. Koncel-Kedziorski, Y. Choi, and H. Hajishirzi, *Mathqa: Towards interpretable math word problem solving with operation-based formalisms*, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), pp. 2357–2367, Association for Computational Linguistics, 2019.

[12] Q. Jin, Z. Yuan, G. Xiong, Q. Yu, H. Ying, C. Tan, M. Chen, S. Huang, X. Liu, and S. Yu, *Biomedical question answering: A survey of approaches and challenges*, *ACM Comput. Surv.* **55** (2023), no. 2 35:1–35:36.

[13] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. H. Chi, Q. Le, and D. Zhou, *Chain of thought prompting elicits reasoning in large language models*, *CoRR* **abs/2201.11903** (2022) [arXiv:2201.1190].

[14] X. Qiu, T. Sun, Y. Xu, Y. Shao, N. Dai, and X. Huang, *Pre-trained models for natural language processing: A survey*, *CoRR* **abs/2003.08271** (2020) [arXiv:2003.0827].

[15] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig, *Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing*, *CoRR* **abs/2107.13586** (2021) [arXiv:2107.1358].

[16] P. Rajpurkar, R. Jia, and P. Liang, *Know what you don't know: Unanswerable questions for squad*, in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 2: Short Papers* (I. Gurevych and Y. Miyao, eds.), pp. 784–789, Association for Computational Linguistics, 2018.

[17] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, H. F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young, E. Rutherford, T. Hennigan, J. Menick, A. Cassirer, R. Powell, G. van den Driessche, L. A. Hendricks, M. Rauh, P. Huang, A. Glaese, J. Welbl, S. Dathathri, S. Huang, J. Uesato,

J. Mellor, I. Higgins, A. Creswell, N. McAleese, A. Wu, E. Elsen, S. M. Jayakumar, E. Buchatskaya, D. Budden, E. Sutherland, K. Simonyan, M. Paganini, L. Sifre, L. Martens, X. L. Li, A. Kuncoro, A. Nematzadeh, E. Gribovskaya, D. Donato, A. Lazaridou, A. Mensch, J. Lespiau, M. Tsimpoukelli, N. Grigorev, D. Fritz, T. Sottiaux, M. Pajarskas, T. Pohlen, Z. Gong, D. Toyama, C. de Masson d'Autume, Y. Li, T. Terzi, V. Mikulik, I. Babuschkin, A. Clark, D. de Las Casas, A. Guy, C. Jones, J. Bradbury, M. Johnson, B. A. Hechtman, L. Weidinger, I. Gabriel, W. S. Isaac, E. Lockhart, S. Osindero, L. Rimell, C. Dyer, O. Vinyals, K. Ayoub, J. Stanway, L. Bennett, D. Hassabis, K. Kavukcuoglu, and G. Irving, *Scaling language models: Methods, analysis & insights from training gopher*, *CoRR* **abs/2112.11446** (2021) [arXiv:2112.1144].

[18] Z. Chen, W. Chen, C. Smiley, S. Shah, I. Borova, D. Langdon, R. Moussa, M. Beane, T. Huang, B. R. Routledge, and W. Y. Wang, *Finqa: A dataset of numerical reasoning over financial data*, in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021* (M. Moens, X. Huang, L. Specia, and S. W. Yih, eds.), pp. 3697–3711, Association for Computational Linguistics, 2021.

[19] A. Rastogi, D. Hakkani-Tür, and L. P. Heck, *Scalable multi-domain dialogue state tracking*, in *2017 IEEE Automatic Speech Recognition and Understanding Workshop, ASRU 2017, Okinawa, Japan, December 16-20, 2017*, pp. 561–568, IEEE, 2017.

[20] V. Zhong, C. Xiong, and R. Socher, *Global-locally self-attentive encoder for dialogue state tracking*, in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers* (I. Gurevych and Y. Miyao, eds.), pp. 1458–1467, Association for Computational Linguistics, 2018.

[21] M. Eric, R. Goel, S. Paul, A. Sethi, S. Agarwal, S. Gao, A. Kumar, A. K. Goyal, P. Ku, and D. Hakkani-Tür, *Multiwoz 2.1: A consolidated multi-domain dialogue dataset with state corrections and state tracking baselines*, in *Proceedings of The 12th Language Resources and Evaluation Conference, LREC 2020, Marseille, France, May 11-16, 2020* (N. Calzolari, F. Béchet, P. Blache, K. Choukri, C. Cieri, T. Declerck, S. Goggi, H. Isahara, B. Maegaard, J. Mariani, H. Mazo, A. Moreno, J. Odijk, and S. Piperidis, eds.), pp. 422–428, European Language Resources Association, 2020.

[22] L. Ren, K. Xie, L. Chen, and K. Yu, *Towards universal dialogue state tracking*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* (E. Riloff,

D. Chiang, J. Hockenmaier, and J. Tsujii, eds.), pp. 2780–2786, Association for Computational Linguistics, 2018.

[23] E. Nouri and E. Hosseini-Asl, *Toward scalable neural dialogue state tracking model*, *CoRR* **abs/1812.00899** (2018) [arXiv:1812.0089].

[24] P. Xu and Q. Hu, *An end-to-end approach for handling unknown slot values in dialogue state tracking*, in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers* (I. Gurevych and Y. Miyao, eds.), pp. 1448–1457, Association for Computational Linguistics, 2018.

[25] E. Reiter and R. Dale, *Building applied natural language generation systems*, *Natural Language Engineering* **3** (1997), no. 1 57–87.

[26] T. Liu, K. Wang, L. Sha, B. Chang, and Z. Sui, *Table-to-text generation by structure-aware seq2seq learning*, in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 4881–4888, 2018.

[27] S. Wiseman, S. M. Shieber, and A. M. Rush, *Learning neural templates for text generation*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018*, pp. 3174–3187, 2018.

[28] R. Puduppully, L. Dong, and M. Lapata, *Data-to-text generation with content selection and planning*, *CoRR* **abs/1809.00582** (2018) [arXiv:1809.0058].

[29] Z. Chen, H. Liu, H. Xu, S. Moon, H. Zhou, and B. Liu, *NUANCED: natural utterance annotation for nuanced conversation with estimated distributions*, in *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021* (M. Moens, X. Huang, L. Specia, and S. W. Yih, eds.), pp. 4016–4024, Association for Computational Linguistics, 2021.

[30] Z. Chen, H. Eavani, W. Chen, Y. Liu, and W. Y. Wang, *Few-shot NLG with pre-trained language model*, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020* (D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, eds.), pp. 183–190, Association for Computational Linguistics, 2020.

[31] Z. Chen, W. Chen, H. Zha, X. Zhou, Y. Zhang, S. Sundaresan, and W. Y. Wang, *Logic2text: High-fidelity natural language generation from logical forms*, in

Findings of the Association for Computational Linguistics: EMNLP 2020, Online Event, 16-20 November 2020 (T. Cohn, Y. He, and Y. Liu, eds.), vol. EMNLP 2020 of *Findings of ACL*, pp. 2096–2111, Association for Computational Linguistics, 2020.

[32] Z. Chen, B. Liu, S. Moon, C. Sankar, P. A. Crook, and W. Y. Wang, *KETOD: knowledge-enriched task-oriented dialogue*, *CoRR* **abs/2205.05589** (2022) [arXiv:2205.0558].

[33] M. Jerven, *Poor numbers: how we are misled by African development statistics and what to do about it.* Cornell University Press, 2013.

[34] D. MacKenzie, *An engine, not a camera: How financial models shape markets.* Mit Press, 2008.

[35] D. MacKenzie, D. Beunza, Y. Millo, and J. P. Pardo-Guerra, *Drilling through the allegheny mountains: Liquidity, materiality and high-frequency trading*, *Journal of cultural economy* **5** (2012), no. 3 279–296.

[36] A.-C. Lange, M. Lenglet, and R. Seyfert, *Cultures of high-frequency trading: Mapping the landscape of algorithmic developments in contemporary financial markets*, *Economy and Society* **45** (2016), no. 2 149–165.

[37] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. W. Cohen, R. Salakhutdinov, and C. D. Manning, *Hotpotqa: A dataset for diverse, explainable multi-hop question answering*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* (E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, eds.), pp. 2369–2380, Association for Computational Linguistics, 2018.

[38] X. Zheng, D. Burdick, L. Popa, P. Zhong, and N. X. R. Wang, *Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context*, *Winter Conference for Applications in Computer Vision (WACV)* (2021).

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

[40] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, *Roberta: A robustly optimized BERT pretraining approach*, *CoRR* **abs/1907.11692** (2019) [arXiv:1907.1169].

[41] A. Talmor and J. Berant, *The web as a knowledge-base for answering complex questions*, in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)* (M. A. Walker, H. Ji, and A. Stent, eds.), pp. 641–651, Association for Computational Linguistics, 2018.

[42] P. Pasupat and P. Liang, *Compositional semantic parsing on semi-structured tables*, 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 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*, pp. 1470–1480, The Association for Computer Linguistics, 2015.

[43] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, Z. Zhang, and D. R. Radev, *Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* (E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, eds.), pp. 3911–3921, Association for Computational Linguistics, 2018.

[44] W. Chen, H. Wang, J. Chen, Y. Zhang, H. Wang, S. Li, X. Zhou, and W. Y. Wang, *Tabfact: A large-scale dataset for table-based fact verification*, in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net, 2020.

[45] W. Chen, H. Zha, Z. Chen, W. Xiong, H. Wang, and W. Y. Wang, *Hybridqa: A dataset of multi-hop question answering over tabular and textual data*, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020* (T. Cohn, Y. He, and Y. Liu, eds.), pp. 1026–1036, Association for Computational Linguistics, 2020.

[46] R. Koncel-Kedziorski, S. Roy, A. Amini, N. Kushman, and H. Hajishirzi, *MAWPS: A math word problem repository*, in *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016* (K. Knight, A. Nenkova, and O. Rambow, eds.), pp. 1152–1157, The Association for Computational Linguistics, 2016.

[47] B. Kim, K. S. Ki, D. Lee, and G. Gweon, *Point to the expression: Solving algebraic word problems using the expression-pointer transformer model*, in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language*

Processing, EMNLP 2020, Online, November 16-20, 2020 (B. Webber, T. Cohn, Y. He, and Y. Liu, eds.), pp. 3768–3779, Association for Computational Linguistics, 2020.

[48] K. Chen, Q. Huang, H. Palangi, P. Smolensky, K. D. Forbus, and J. Gao, *Mapping natural-language problems to formal-language solutions using structured neural representations*, in *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, vol. 119 of *Proceedings of Machine Learning Research*, pp. 1566–1575, PMLR, 2020.

[49] X. Chen, C. Liang, A. W. Yu, D. Zhou, D. Song, and Q. V. Le, *Neural symbolic reader: Scalable integration of distributed and symbolic representations for reading comprehension*, in *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*, OpenReview.net, 2020.

[50] J. Han, U. Barman, J. Hayes, J. Du, E. Burgin, and D. Wan, *Nextgen AML: distributed deep learning based language technologies to augment anti money laundering investigation*, in *Proceedings of ACL 2018, Melbourne, Australia, July 15-20, 2018, System Demonstrations* (F. Liu and T. Solorio, eds.), pp. 37–42, Association for Computational Linguistics, 2018.

[51] W. Wang, J. Zhang, Q. Li, C. Zong, and Z. Li, *Are you for real? detecting identity fraud via dialogue interactions*, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019* (K. Inui, J. Jiang, V. Ng, and X. Wan, eds.), pp. 1762–1771, Association for Computational Linguistics, 2019.

[52] A. Nourbakhsh and G. Bang, *A framework for anomaly detection using language modeling, and its applications to finance*, *CoRR* **abs/1908.09156** (2019) [arXiv:1908.0915].

[53] M. Day and C. Lee, *Deep learning for financial sentiment analysis on finance news providers*, in *2016 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2016, San Francisco, CA, USA, August 18-21, 2016* (R. Kumar, J. Caverlee, and H. Tong, eds.), pp. 1127–1134, IEEE Computer Society, 2016.

[54] C. Wang, M. Tsai, T. Liu, and C. Chang, *Financial sentiment analysis for risk prediction*, in *Sixth International Joint Conference on Natural Language Processing, IJCNLP 2013, Nagoya, Japan, October 14-18, 2013*, pp. 802–808, Asian Federation of Natural Language Processing / ACL, 2013.

[55] M. S. Akhtar, A. Kumar, D. Ghosal, A. Ekbal, and P. Bhattacharyya, *A multilayer perceptron based ensemble technique for fine-grained financial*

sentiment analysis, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017* (M. Palmer, R. Hwa, and S. Riedel, eds.), pp. 540–546, Association for Computational Linguistics, 2017.

[56] Z. Liu, D. Huang, K. Huang, Z. Li, and J. Zhao, *Finbert: A pre-trained financial language representation model for financial text mining*, in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020* (C. Bessiere, ed.), pp. 4513–4519, ijcai.org, 2020.

[57] Y. Yang, M. C. S. Uy, and A. Huang, *Finbert: A pretrained language model for financial communications*, *CoRR* **abs/2006.08097** (2020) [arXiv:2006.0809].

[58] D. Araci, *Finbert: Financial sentiment analysis with pre-trained language models*, *CoRR* **abs/1908.10063** (2019) [arXiv:1908.1006].

[59] C. Alberti, K. Lee, and M. Collins, *A BERT baseline for the natural questions*, *CoRR* **abs/1901.08634** (2019) [arXiv:1901.0863].

[60] I. Beltagy, M. E. Peters, and A. Cohan, *Longformer: The long-document transformer*, *CoRR* **abs/2004.05150** (2020) [arXiv:2004.0515].

[61] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (Y. Bengio and Y. LeCun, eds.), 2015.

[62] T. Wen, Y. Miao, P. Blunsom, and S. J. Young, *Latent intention dialogue models*, in *Proceedings of the 34th International Conference on Machine Learning, ICML 2017, Sydney, NSW, Australia, 6-11 August 2017* (D. Precup and Y. W. Teh, eds.), vol. 70 of *Proceedings of Machine Learning Research*, pp. 3732–3741, PMLR, 2017.

[63] P. Budzianowski, T. Wen, B. Tseng, I. Casanueva, S. Ultes, O. Ramadan, and M. Gasic, *Multiwoz - A large-scale multi-domain wizard-of-oz dataset for task-oriented dialogue modelling*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* (E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, eds.), pp. 5016–5026, Association for Computational Linguistics, 2018.

[64] E. Hosseini-Asl, B. McCann, C. Wu, S. Yavuz, and R. Socher, *A simple language model for task-oriented dialogue*, *CoRR* **abs/2005.00796** (2020) [arXiv:2005.0079].

[65] Y. Sun and Y. Zhang, *Conversational recommender system*, in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018* (K. Collins-Thompson, Q. Mei, B. D. Davison, Y. Liu, and E. Yilmaz, eds.), pp. 235–244, ACM, 2018.

[66] Y. Zhang, X. Chen, Q. Ai, L. Yang, and W. B. Croft, *Towards conversational search and recommendation: System ask, user respond*, in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management, CIKM 2018, Torino, Italy, October 22-26, 2018* (A. Cuzzocrea, J. Allan, N. W. Paton, D. Srivastava, R. Agrawal, A. Z. Broder, M. J. Zaki, K. S. Candan, A. Labrinidis, A. Schuster, and H. Wang, eds.), pp. 177–186, ACM, 2018.

[67] D. Adiwardana, M.-T. Luong, D. R. So, J. Hall, N. Fiedel, R. Thoppilan, Z. Yang, A. Kulshreshtha, G. Nemade, Y. Lu, *et. al.*, *Towards a human-like open-domain chatbot*, *arXiv preprint arXiv:2001.09977* (2020).

[68] S. Roller, E. Dinan, N. Goyal, D. Ju, M. Williamson, Y. Liu, J. Xu, M. Ott, K. Shuster, E. M. Smith, *et. al.*, *Recipes for building an open-domain chatbot*, *arXiv preprint arXiv:2004.13637* (2020).

[69] B. Liu and I. Lane, *Attention-based recurrent neural network models for joint intent detection and slot filling*, in *Interspeech 2016, 17th Annual Conference of the International Speech Communication Association, San Francisco, CA, USA, September 8-12, 2016* (N. Morgan, ed.), pp. 685–689, ISCA, 2016.

[70] R. Gangadharaiah and B. Narayanaswamy, *Joint multiple intent detection and slot labeling for goal-oriented dialog*, in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)* (J. Burstein, C. Doran, and T. Solorio, eds.), pp. 564–569, Association for Computational Linguistics, 2019.

[71] M. Heck, C. van Niekerk, N. Lubis, C. Geishauser, H. Lin, M. Moresi, and M. Gasic, *Trippy: A triple copy strategy for value independent neural dialog state tracking*, in *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020* (O. Pietquin, S. Muresan, V. Chen, C. Kennington, D. Vandyke, N. Dethlefs, K. Inoue, E. Ekstedt, and S. Ultes, eds.), pp. 35–44, Association for Computational Linguistics, 2020.

[72] B. Peng, X. Li, L. Li, J. Gao, A. Çelikyilmaz, S. Lee, and K. Wong, *Composite task-completion dialogue policy learning via hierarchical deep reinforcement learning*, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11,*

2017 (M. Palmer, R. Hwa, and S. Riedel, eds.), pp. 2231–2240, Association for Computational Linguistics, 2017.

[73] P. Su, M. Gasic, N. Mrksic, L. M. Rojas-Barahona, S. Ultes, D. Vandyke, T. Wen, and S. J. Young, *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 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, The Association for Computer Linguistics, 2016.

[74] O. Dusek, J. Novikova, and V. Rieser, *Findings of the E2E NLG challenge*, in *Proceedings of the 11th International Conference on Natural Language Generation, Tilburg University, The Netherlands, November 5-8, 2018* (E. Krahmer, A. Gatt, and M. Goudbeek, eds.), pp. 322–328, Association for Computational Linguistics, 2018.

[75] T. Wen, M. Gasic, N. Mrksic, P. Su, D. Vandyke, and S. J. Young, *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 2015, Lisbon, Portugal, September 17-21, 2015* (L. Màrquez, C. Callison-Burch, J. Su, D. Pighin, and Y. Marton, eds.), pp. 1711–1721, The Association for Computational Linguistics, 2015.

[76] W. Chen, J. Chen, P. Qin, X. Yan, and W. Y. Wang, *Semantically conditioned dialog response generation via hierarchical disentangled self-attention*, in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers* (A. Korhonen, D. R. Traum, and L. Màrquez, eds.), pp. 3696–3709, Association for Computational Linguistics, 2019.

[77] Z. Fu, Y. Xian, Y. Zhang, and Y. Zhang, *Tutorial on conversational recommendation systems*, in *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020* (R. L. T. Santos, L. B. Marinho, E. M. Daly, L. Chen, K. Falk, N. Koenigstein, and E. S. de Moura, eds.), pp. 751–753, ACM, 2020.

[78] H. Xu, S. Moon, H. Liu, B. Liu, P. Shah, B. Liu, and P. S. Yu, *User memory reasoning for conversational recommendation*, *CoRR* **abs/2006.00184** (2020) [arXiv:2006.0018].

[79] W. Lei, G. Zhang, X. He, Y. Miao, X. Wang, L. Chen, and T. Chua, *Interactive path reasoning on graph for conversational recommendation*, in *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020* (R. Gupta, Y. Liu, J. Tang, and B. A. Prakash, eds.), pp. 2073–2083, ACM, 2020.

[80] S. Li, W. Lei, Q. Wu, X. He, P. Jiang, and T. Chua, *Seamlessly unifying attributes and items: Conversational recommendation for cold-start users*, *CoRR* **abs/2005.12979** (2020) [arXiv:2005.1297].

[81] G. Penha and C. Hauff, *What does BERT know about books, movies and music? probing BERT for conversational recommendation*, in *RecSys 2020: Fourteenth ACM Conference on Recommender Systems, Virtual Event, Brazil, September 22-26, 2020* (R. L. T. Santos, L. B. Marinho, E. M. Daly, L. Chen, K. Falk, N. Koenigstein, and E. S. de Moura, eds.), pp. 388–397, ACM, 2020.

[82] J. D. Williams, M. Henderson, A. Raux, B. Thomson, A. W. Black, and D. Ramachandran, *The dialog state tracking challenge series*, *AI Mag.* **35** (2014), no. 4 121–124.

[83] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, *Attention is all you need*, in *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, 4-9 December 2017, Long Beach, CA, USA* (I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, eds.), pp. 5998–6008, 2017.

[84] A. Gatt and E. Krahmer, *Survey of the state of the art in natural language generation: Core tasks, applications and evaluation*, *Journal of Artificial Intelligence Research* **61** (2018) 65–170.

[85] H. He, A. Balakrishnan, M. Eric, and P. Liang, *Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings*, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pp. 1766–1776, 2017.

[86] M. Ghazvininejad, C. Brockett, M. Chang, B. Dolan, J. Gao, W. Yih, and M. Galley, *A knowledge-grounded neural conversation model*, in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 5110–5117, 2018.

[87] Y. Su, H. Sun, B. M. Sadler, M. Srivatsa, I. Gur, Z. Yan, and X. Yan, *On generating characteristic-rich question sets for QA evaluation*, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pp. 562–572, 2016.

[88] A. Saha, V. Pahuja, M. M. Khapra, K. Sankaranarayanan, and S. Chandar, *Complex sequential question answering: Towards learning to converse over linked*

question answer pairs with a knowledge graph, in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 705–713, 2018.

[89] J. Yin, X. Jiang, Z. Lu, L. Shang, H. Li, and X. Li, *Neural generative question answering*, in *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*, pp. 2972–2978, 2016.

[90] S. A. Hasan and O. Farri, *Clinical natural language processing with deep learning*, in *Data Science for Healthcare*, pp. 147–171. Springer, 2019.

[91] A. J. Cawsey, B. L. Webber, and R. B. Jones, *Natural language generation in health care*, 1997.

[92] C. DiMarco, H. Covvey, D. Cowan, V. DiCiccio, E. Hovy, J. Lipa, D. Mulholland, *et. al.*, *The development of a natural language generation system for personalized e-health information*, in *Medinfo 2007: Proceedings of the 12th World Congress on Health (Medical) Informatics; Building Sustainable Health Systems*, p. 2339, IOS Press, 2007.

[93] J. Novikova, O. Dusek, and V. Rieser, *The E2E dataset: New challenges for end-to-end generation*, in *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue, Saarbrücken, Germany, August 15-17, 2017*, pp. 201–206, 2017.

[94] P. Liang, M. I. Jordan, and D. Klein, *Learning semantic correspondences with less supervision*, in *ACL 2009, Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP, 2-7 August 2009, Singapore*, pp. 91–99, 2009.

[95] S. Wiseman, S. M. Shieber, and A. M. Rush, *Challenges in data-to-document generation*, in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*, pp. 2253–2263, 2017.

[96] R. Lebret, D. Grangier, and M. Auli, *Neural text generation from structured data with application to the biography domain*, in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pp. 1203–1213, 2016.

[97] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson, *One billion word benchmark for measuring progress in statistical language modeling*, *arXiv preprint arXiv:1312.3005* (2013).

[98] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, *Language models are unsupervised multitask learners*, .

[99] M. A. Walker, O. Rambow, and M. Rogati, *Spot: A trainable sentence planner*, in *Language Technologies 2001: The Second Meeting of the North American Chapter of the Association for Computational Linguistics, NAACL 2001, Pittsburgh, PA, USA, June 2-7, 2001*, 2001.

[100] W. Lu, H. T. Ng, and W. S. Lee, *Natural language generation with tree conditional random fields*, in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing, EMNLP 2009, 6-7 August 2009, Singapore, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 400–409, 2009.

[101] I. Konstas and M. Lapata, *Unsupervised concept-to-text generation with hypergraphs*, in *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 3-8, 2012, Montréal, Canada*, pp. 752–761, 2012.

[102] I. Konstas and M. Lapata, *A global model for concept-to-text generation*, *J. Artif. Intell. Res.* **48** (2013) 305–346.

[103] C. Gardent, A. Shimorina, S. Narayan, and L. Perez-Beltrachini, *The webnlg challenge: Generating text from RDF data*, in *Proceedings of the 10th International Conference on Natural Language Generation, INLG 2017, Santiago de Compostela, Spain, September 4-7, 2017*, pp. 124–133, 2017.

[104] H. ElSahar, C. Gravier, and F. Laforest, *Zero-shot question generation from knowledge graphs for unseen predicates and entity types*, in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)* (M. A. Walker, H. Ji, and A. Stent, eds.), pp. 218–228, Association for Computational Linguistics, 2018.

[105] S. Ma, P. Yang, T. Liu, P. Li, J. Zhou, and X. Sun, *Key fact as pivot: A two-stage model for low resource table-to-text generation*, in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers* (A. Korhonen, D. R. Traum, and L. Màrquez, eds.), pp. 2047–2057, Association for Computational Linguistics, 2019.

[106] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, *Distributed representations of words and phrases and their compositionality*, in *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings of a meeting held December 5-8, 2013, Lake Tahoe, Nevada, United States.*, pp. 3111–3119, 2013.

[107] J. Pennington, R. Socher, and C. D. Manning, *Glove: Global vectors for word representation*, in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pp. 1532–1543, 2014.

[108] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, *Deep contextualized word representations*, in *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)* (M. A. Walker, H. Ji, and A. Stent, eds.), pp. 2227–2237, Association for Computational Linguistics, 2018.

[109] Q. V. Le and T. Mikolov, *Distributed representations of sentences and documents*, in *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, pp. 1188–1196, 2014.

[110] R. Kiros, Y. Zhu, R. Salakhutdinov, R. S. Zemel, R. Urtasun, A. Torralba, and S. Fidler, *Skip-thought vectors*, in *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 3294–3302, 2015.

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

[112] A. See, P. J. Liu, and C. D. Manning, *Get to the point: Summarization with pointer-generator networks*, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pp. 1073–1083, 2017.

[113] B. Dhingra, M. Faruqui, A. P. Parikh, M. Chang, D. Das, and W. W. Cohen, *Handling divergent reference texts when evaluating table-to-text generation*, in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers* (A. Korhonen, D. R. Traum, and L. Màrquez, eds.), pp. 4884–4895, Association for Computational Linguistics, 2019.

[114] R. Sennrich, B. Haddow, and A. Birch, *Neural machine translation of rare words with subword units*, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

[115] M. Freitag and S. Roy, *Unsupervised natural language generation with denoising autoencoders*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* (E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, eds.), pp. 3922–3929, Association for Computational Linguistics, 2018.

[116] Z. Chen, H. Eavani, Y. Liu, and W. Y. Wang, *Few-shot NLG with pre-trained language model*, *CoRR* **abs/1904.09521** (2019) [arXiv:1904.0952].

[117] W. Chen, H. Wang, J. Chen, Y. Zhang, H. Wang, S. Li, X. Zhou, and W. Y. Wang, *Tabfact: A large-scale dataset for table-based fact verification*, *CoRR* **abs/1909.02164** (2019) [arXiv:1909.0216].

[118] S. H. Lee, *Natural language generation for electronic health records*, *NPJ digital medicine* **1** (2018), no. 1 63.

[119] O. Dušek, J. Novikova, and V. Rieser, *Evaluating the state-of-the-art of end-to-end natural language generation: The E2E NLG Challenge*, *arXiv preprint arXiv:1901.11528* (Jan., 2019).

[120] A. Gatt and E. Krahmer, *Survey of the state of the art in natural language generation: Core tasks, applications and evaluation*, *J. Artif. Intell. Res.* **61** (2018) 65–170.

[121] H. Gong, X. Feng, B. Qin, and T. Liu, *Table-to-text generation with effective hierarchical encoder on three dimensions (row, column and time)*, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019* (K. Inui, J. Jiang, V. Ng, and X. Wan, eds.), pp. 3141–3150, Association for Computational Linguistics, 2019.

[122] K. Song, X. Tan, T. Qin, J. Lu, and T. Liu, *MASS: masked sequence to sequence pre-training for language generation*, in *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA* (K. Chaudhuri and R. Salakhutdinov, eds.), vol. 97 of *Proceedings of Machine Learning Research*, pp. 5926–5936, PMLR, 2019.

[123] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, *Exploring the limits of transfer learning with a unified text-to-text transformer*, *CoRR* **abs/1910.10683** (2019) [arXiv:1910.1068].

[124] R. Tian, S. Narayan, T. Sellam, and A. P. Parikh, *Sticking to the facts: Confident decoding for faithful data-to-text generation*, *CoRR* **abs/1910.08684** (2019) [arXiv:1910.0868].

[125] A. P. Parikh, X. Wang, S. Gehrmann, M. Faruqui, B. Dhingra, D. Yang, and D. Das, *Totto: A controlled table-to-text generation dataset*, *CoRR* **abs/2004.14373** (2020) [arXiv:2004.1437].

[126] F. Korn, X. Wang, Y. Wu, and C. Yu, *Automatically generating interesting facts from wikipedia tables*, in *Proceedings of the 2019 International Conference on Management of Data, SIGMOD Conference 2019, Amsterdam, The Netherlands, June 30 - July 5, 2019* (P. A. Boncz, S. Manegold, A. Ailamaki, A. Deshpande, and T. Kraska, eds.), pp. 349–361, ACM, 2019.

[127] W. Chen, J. Chen, Y. Su, Z. Chen, and W. Y. Wang, *Logical natural language generation from open-domain tables*, *arXiv preprint arXiv:2004.10404* (Apr., 2020).

[128] M. White, *Efficient realization of coordinate structures in combinatory categorial grammar*, *Research on Language and Computation* **4** (2006), no. 1 39–75.

[129] C. Gardent and A. Plainfossé, *Generating from a deep structure*, in *COLNG 1990 Volume 2: Papers presented to the 13th International Conference on Computational Linguistics*, 1990.

[130] L. Song, Y. Zhang, Z. Wang, and D. Gildea, *A graph-to-sequence model for amr-to-text generation*, in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers* (I. Gurevych and Y. Miyao, eds.), pp. 1616–1626, Association for Computational Linguistics, 2018.

[131] J. D. Phillips, *Generation of text from logical formulae*, *Mach. Transl.* **8** (1993), no. 4 209–235.

[132] J. Calder, M. Reape, and H. Zeevat, *An algorithm for generation in unification categorial grammar*, in *EACL 1989, 4th Conference of the European Chapter of the Association for Computational Linguistics, April 10-12, 1989, University of Manchester, Institute of Science and Technology, Manchester, England* (H. L. Somers and M. M. Wood, eds.), pp. 233–240, The Association for Computer Linguistics, 1989.

[133] S. M. Shieber, G. van Noord, F. C. N. Pereira, and R. C. Moore, *Semantic-head-driven generation*, *Comput. Linguistics* **16** (1990), no. 1 30–42.

[134] J. Bos, *The groningen meaning bank*, in *Proceedings of the Joint Symposium on Semantic Processing. Textual Inference and Structures in Corpora, JSSP 2013, Trento, Italy, November 20-22, 2013*, p. 2, Association for Computational Linguistics, 2013.

[135] J. May, *Semeval-2016 task 8: Meaning representation parsing*, in *Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016* (S. Bethard, D. M. Cer, M. Carpuat, D. Jurgens, P. Nakov, and T. Zesch, eds.), pp. 1063–1073, The Association for Computer Linguistics, 2016.

[136] D. Flickinger, Y. Zhang, and V. Kordoni, *Deepbank. a dynamically annotated treebank of the wall street journal*, in *Proceedings of the 11th International Workshop on Treebanks and Linguistic Theories*, pp. 85–96, 2012.

[137] C. S. Bhagavatula, T. Noraset, and D. Downey, *Methods for exploring and mining tables on wikipedia*, in *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics, IDEA@KDD 2013, Chicago, Illinois, USA, August 11, 2013* (D. H. Chau, J. Vreeken, M. van Leeuwen, and C. Faloutsos, eds.), pp. 18–26, ACM, 2013.

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

[139] D. Marcheggiani and L. Perez-Beltrachini, *Deep graph convolutional encoders for structured data to text generation*, in *Proceedings of the 11th International Conference on Natural Language Generation, Tilburg University, The Netherlands, November 5-8, 2018* (E. Krahmer, A. Gatt, and M. Goudbeek, eds.), pp. 1–9, Association for Computational Linguistics, 2018.

[140] K. Xu, L. Wu, Z. Wang, Y. Feng, and V. Sheinin, *Sql-to-text generation with graph-to-sequence model*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018* (E. Riloff, D. Chiang, J. Hockenmaier, and J. Tsujii, eds.), pp. 931–936, Association for Computational Linguistics, 2018.

[141] Y. Lin, Y. C. Tan, and R. Frank, *Open sesame: Getting inside bert's linguistic knowledge*, *CoRR* **abs/1906.01698** (2019) [arXiv:1906.0169].

[142] A. Rogers, O. Kovaleva, and A. Rumshisky, *A primer in bertology: What we know about how BERT works*, *CoRR* **abs/2002.12327** (2020) [arXiv:2002.1232].

[143] M. Mager, R. F. Astudillo, T. Naseem, M. A. Sultan, Y. Lee, R. Florian, and S. Roukos, *Gpt-too: A language-model-first approach for amr-to-text generation*, *CoRR* **abs/2005.09123** (2020) [arXiv:2005.0912].

[144] Y. Zhang, S. Sun, M. Galley, Y. Chen, C. Brockett, X. Gao, J. Gao, J. Liu, and B. Dolan, *DIALOGPT : Large-scale generative pre-training for conversational response generation*, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations, ACL 2020, Online, July 5-10, 2020* (A. Celikyilmaz and T. Wen, eds.), pp. 270–278, Association for Computational Linguistics, 2020.

[145] C. Tao, C. Chen, J. Feng, J. Wen, and R. Yan, *A pre-training strategy for zero-resource response selection in knowledge-grounded conversations*, in *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021* (C. Zong, F. Xia, W. Li, and R. Navigli, eds.), pp. 4446–4457, Association for Computational Linguistics, 2021.

[146] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, *Exploring the limits of transfer learning with a unified text-to-text transformer*, *J. Mach. Learn. Res.* **21** (2020) 140:1–140:67.

[147] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, *BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension*, in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020* (D. Jurafsky, J. Chai, N. Schluter, and J. R. Tetreault, eds.), pp. 7871–7880, Association for Computational Linguistics, 2020.

[148] A. Rastogi, X. Zang, S. Sunkara, R. Gupta, and P. Khaitan, *Towards scalable multi-domain conversational agents: The schema-guided dialogue dataset*, in *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp. 8689–8696, AAAI Press, 2020.

[149] S. Zhang, E. Dinan, J. Urbanek, A. Szlam, D. Kiela, and J. Weston, *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, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers* (I. Gurevych and Y. Miyao, eds.), pp. 2204–2213, Association for Computational Linguistics, 2018.

[150] Y. Tuan, Y. Chen, and H. Lee, *Dykgchat: Benchmarking dialogue generation grounding on dynamic knowledge graphs*, in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019* (K. Inui, J. Jiang, V. Ng, and X. Wan, eds.), pp. 1855–1865, Association for Computational Linguistics, 2019.

[151] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, and J. Weston, *Wizard of wikipedia: Knowledge-powered conversational agents*, in *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*, OpenReview.net, 2019.

[152] K. Shuster, S. Poff, M. Chen, D. Kiela, and J. Weston, *Retrieval augmentation reduces hallucination in conversation*, in *Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 16-20 November, 2021* (M. Moens, X. Huang, L. Specia, and S. W. Yih, eds.), pp. 3784–3803, Association for Computational Linguistics, 2021.

[153] K. Sun, S. Moon, P. A. Crook, S. Roller, B. Silvert, B. Liu, Z. Wang, H. Liu, E. Cho, and C. Cardie, *Adding chit-chat to enhance task-oriented dialogues*, in *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2021, Online, June 6-11, 2021* (K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, eds.), pp. 1570–1583, Association for Computational Linguistics, 2021.

[154] T. Wen, D. Vandyke, N. Mrksic, M. Gasic, L. M. Rojas-Barahona, P. Su, S. Ultes, and S. J. Young, *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 2017, Valencia, Spain, April 3-7, 2017, Volume 1: Long Papers* (M. Lapata, P. Blunsom, and A. Koller, eds.), pp. 438–449, Association for Computational Linguistics, 2017.

[155] N. Mrksic, D. Ó. Séaghdha, T. Wen, B. Thomson, and S. J. Young, *Neural belief tracker: Data-driven dialogue state tracking*, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers* (R. Barzilay and M. Kan, eds.), pp. 1777–1788, Association for Computational Linguistics, 2017.

[156] B. Liu, G. Tür, D. Hakkani-Tür, P. Shah, and L. P. Heck, *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 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers) (M. A. Walker, H. Ji, and A. Stent, eds.), pp. 2060–2069, Association for Computational Linguistics, 2018.

[157] G. Zhou, P. Luo, R. Cao, F. Lin, B. Chen, and Q. He, *Mechanism-aware neural machine for dialogue response generation*, in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA* (S. P. Singh and S. Markovitch, eds.), pp. 3400–3407, AAAI Press, 2017.

[158] A. Bordes, Y. Boureau, and J. Weston, *Learning end-to-end goal-oriented dialog*, in *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, OpenReview.net, 2017.

[159] B. Liu, G. Tür, D. Hakkani-Tür, P. Shah, and L. P. Heck, *End-to-end optimization of task-oriented dialogue model with deep reinforcement learning*, *CoRR* **abs/1711.10712** (2017) [arXiv:1711.1071].

[160] H. Xu, H. Peng, H. Xie, E. Cambria, L. Zhou, and W. Zheng, *End-to-end latent-variable task-oriented dialogue system with exact log-likelihood optimization*, *World Wide Web* **23** (2020), no. 3 1989–2002.

[161] B. Peng, C. Li, J. Li, S. Shayandeh, L. Liden, and J. Gao, *SOLOIST: few-shot task-oriented dialog with A single pre-trained auto-regressive model*, *CoRR* **abs/2005.05298** (2020) [arXiv:2005.0529].

[162] Y. Su, L. Shu, E. Mansimov, A. Gupta, D. Cai, Y. Lai, and Y. Zhang, *Multi-task pre-training for plug-and-play task-oriented dialogue system*, *CoRR* **abs/2109.14739** (2021) [arXiv:2109.1473].

[163] J. D. Williams, A. Raux, and M. Henderson, *The dialog state tracking challenge series: A review*, *Dialogue Discourse* **7** (2016), no. 3 4–33.

[164] A. Sordoni, M. Galley, M. Auli, C. Brockett, Y. Ji, M. Mitchell, J. Nie, J. Gao, and B. Dolan, *A neural network approach to context-sensitive generation of conversational responses*, in *NAACL HLT 2015, The 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, USA, May 31 - June 5, 2015* (R. Mihalcea, J. Y. Chai, and A. Sarkar, eds.), pp. 196–205, The Association for Computational Linguistics, 2015.

[165] S. Moon, P. Shah, A. Kumar, and R. Subba, *Opendialkg: Explainable conversational reasoning with attention-based walks over knowledge graphs*, in *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long*

Papers (A. Korhonen, D. R. Traum, and L. Màrquez, eds.), pp. 845–854, Association for Computational Linguistics, 2019.

[166] K. Gopalakrishnan, B. Hedayatnia, Q. Chen, A. Gottardi, S. Kwatra, A. Venkatesh, R. Gabriel, and D. Hakkani-Tür, *Topical-chat: Towards knowledge-grounded open-domain conversations*, in *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association, Graz, Austria, 15-19 September 2019* (G. Kubin and Z. Kacic, eds.), pp. 1891–1895, ISCA, 2019.

[167] T. Young, F. Z. Xing, V. Pandelea, J. Ni, and E. Cambria, *Fusing task-oriented and open-domain dialogues in conversational agents*, *CoRR* **abs/2109.04137** (2021) [arXiv:2109.0413].

[168] S. Kim, M. Eric, K. Gopalakrishnan, B. Hedayatnia, Y. Liu, and D. Hakkani-Tür, *Beyond domain apis: Task-oriented conversational modeling with unstructured knowledge access*, in *Proceedings of the 21th Annual Meeting of the Special Interest Group on Discourse and Dialogue, SIGdial 2020, 1st virtual meeting, July 1-3, 2020* (O. Pietquin, S. Muresan, V. Chen, C. Kennington, D. Vandyke, N. Dethlefs, K. Inoue, E. Ekstedt, and S. Ultes, eds.), pp. 278–289, Association for Computational Linguistics, 2020.

[169] S. Kim, Y. Liu, D. Jin, A. Papangelis, K. Gopalakrishnan, B. Hedayatnia, and D. Hakkani-Tur, *"how robust r u?": Evaluating task-oriented dialogue systems on spoken conversations*, *CoRR* **abs/2109.13489** (2021) [arXiv:2109.1348].

[170] S. Gao, R. Takanobu, W. Peng, Q. Liu, and M. Huang, *Hyknow: End-to-end task-oriented dialog modeling with hybrid knowledge management*, in *Findings of the Association for Computational Linguistics: ACL/IJCNLP 2021, Online Event, August 1-6, 2021* (C. Zong, F. Xia, W. Li, and R. Navigli, eds.), vol. ACL/IJCNLP 2021 of *Findings of ACL*, pp. 1591–1602, Association for Computational Linguistics, 2021.

[171] D. Chen, A. Fisch, J. Weston, and A. Bordes, *Reading wikipedia to answer open-domain questions*, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers* (R. Barzilay and M. Kan, eds.), pp. 1870–1879, Association for Computational Linguistics, 2017.

[172] M. Li, J. Weston, and S. Roller, *ACUTE-EVAL: improved dialogue evaluation with optimized questions and multi-turn comparisons*, *CoRR* **abs/1909.03087** (2019) [arXiv:1909.0308].

[173] E. Hosseini-Asl, B. McCann, C. Wu, S. Yavuz, and R. Socher, *A simple language model for task-oriented dialogue*, in *Advances in Neural Information Processing*

Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), 2020.