UnifiedSKG: Unifying and Multi-Tasking Structured Knowledge Grounding with Text-to-Text Language Models

Tianbao Xie[*1,17] **Chen Henry Wu**[*2] **Peng Shi**[3] **Ruiqi Zhong**[4] **Torsten Scholak**[5]
Michihiro Yasunaga[6] **Chien-Sheng Wu**[7] **Ming Zhong**[8] **Pengcheng Yin**[9] **Sida Wang**[10] **Victor Zhong**[16]
Bailin Wang[11] **Chengzu Li**[12] **Connor Boyle**[16] **Ansong Ni**[13] **Ziyu Yao**[14] **Dragomir Radev**[13]
Caiming Xiong[7] **Lingpeng Kong**[12,17] **Rui Zhang**[15] **Noah A. Smith**[16] **Luke Zettlemoyer**[10,16] **Tao Yu**[16,17]

[1]Harbin Institute of Technology [2]CMU [3]University of Waterloo [4]UC Berkeley [5]ElementAI
[6]Stanford University [7]Salesforce Research [8]UIUC [9]Google Research [10]Meta AI
[11]University of Edinburgh [12]Shanghai AI Lab [13]Yale University [14]George Mason University
[15]Penn State University [16]University of Washington [17]The University of Hong Kong

Abstract

Structured knowledge grounding (SKG) leverages structured knowledge to complete user requests, such as semantic parsing over databases and question answering over knowledge bases. Since the inputs and outputs of SKG tasks are heterogeneous, they were historically studied in separate by different communities, which limits systematic and compatible research on SKG. In this paper, we overcome this limitation by proposing the UnifiedSKG framework, which unifies 21 SKG tasks into the text-to-text format, aiming to promote systematic SKG research, instead of being exclusive to a single task, domain, or dataset. We use UnifiedSKG to benchmark T5 with different sizes and show that T5, with simple modifications when necessary, achieves state-of-the-art performance on almost all 21 tasks. We demonstrate that multi-task prefix-tuning improves the performance on most tasks, largely improving the overall performance. UnifiedSKG facilitates the investigation of zero-shot and few-shot learning, and we show that T0, GPT-3, and Codex struggle in zero-shot and few-shot learning for SKG. We also use UnifiedSKG to conduct a series of controlled experiments on structured knowledge encoding variants across SKG tasks. UnifiedSKG is easily extensible to more tasks, and is open-sourced at https://github.com/hkunlp/unifiedskg.[1]

1 Introduction

Structured knowledge (e.g., web tables, knowledge graphs, and databases) stores large amounts of data in an organized structure and forms a basis for a wide range of applications, e.g., medical records, personal assistants, and customer relations management. Accessing and searching data in structured knowledge typically requires query languages or professional training. To promote the efficiency of data access, structured knowledge grounding (SKG) grounds user requests in structured knowledge and produces various outputs, including computer programs (e.g., SQL and SPARQL), cell values, and natural language responses (Figure 1). For example, semantic parsing (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005) converts natural language questions into formal programs; question answering (Berant et al., 2013) derives answers from tables or knowledge graphs.

SKG has attracted significant interest and has been studied through different tasks defined by different NLP communities. Many recent developments in tasks, models, and datasets for SKG have led to task-specialized modeling advances, making each task's progress seemingly unique and incompatible. A main reason is that SKG tasks are *heterogeneous*. Different types of structured knowledge, such as databases or knowledge graphs, require highly specialized encoders (Lin et al., 2019; Herzig et al., 2020; Wang et al., 2020; Yasunaga et al., 2021). Some SKG tasks, like semantic parsing, also use customized decoders to generate programs (Yin and Neubig, 2018; Ren et al., 2021). Therefore, instead of solving common challenges in SKG research, improvements in SKG are prone to be exclusive to a single task, domain, or dataset.

In this paper, we propose the UnifiedSKG framework to advocate for a unifying view of 21 SKG tasks across six task families and multiple domains (Table 1). UnifiedSKG standardizes datasets, models, code, experiments, and evaluation metrics into a single framework. By cast-

*Equal contribution. Work done while interning with the University of Hong Kong. Author contributions listed at end of paper.

[1]More up-to-date collections are available at https://unifiedskg.com

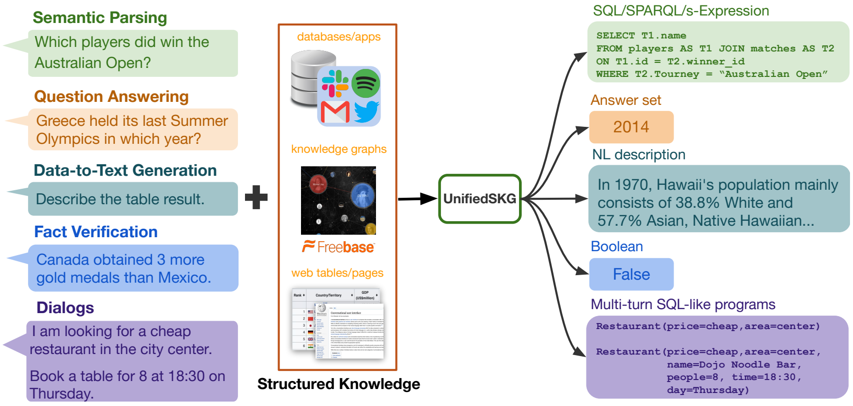


Figure 1: Structured knowledge grounding (SKG) leverages structured knowledge to complete user requests. By casting inputs and outputs into the text-to-text format (e.g., T5), UNIFIEDSKG standardizes datasets, code, and evaluation metrics in a single framework for 21 SKG tasks. We benchmark UNIFIEDSKG's performance at scale, finding that T5, with simple modification when necessary, is able to achieve SOTA on all tasks.

ing user requests, structured knowledge, and outputs into the text-to-text formats (Raffel et al., 2020), UNIFIEDSKG promotes model advances because new models can be easily applied to diverse SKG tasks, and new tasks can be easily framed using the standardized abstraction. While previous works also cast SKG tasks into the text-to-text format (Hosseini-Asl et al., 2020; Shaw et al., 2021; Liu et al., 2021), their independent choices of pretrained language models (PLMs), input-output formats, and frameworks make our unification non-trivial. UNIFIEDSKG is easily extensible to more SKG tasks, and is open-sourced to promote community-wide progress.

Using UNIFIEDSKG as a benchmark, we demonstrate that finetuning T5 (with simple modification when necessary) on individual tasks achieves state-of-the-art (SOTA) results on almost all 21 tasks, establishing a powerful, unified, and reproducible starting point for SKG research. T5 performance also increases with model size on most SKG tasks.

UNIFIEDSKG also facilitates multi-task learning on SKG, enabling knowledge sharing and cross-task generalization. We first show that naive multi-task learning does not improve the overall performance. We then demonstrate that multi-task learning with prefix-tuning (Li and Liang, 2021) outperforms single-task learning on most tasks, on both T5-base and T5-large, largely improving the overall performance.

UNIFIEDSKG is a challenging testbed for few-shot (Brown et al., 2020; Ye et al., 2021a) and zero-shot learning (Zhong et al., 2021; Wei et al., 2021; Sanh et al., 2021) with PLMs. Models like T0 (Sanh et al., 2021) struggle in zero-shot learning on SKG tasks, and GPT-3 (Brown et al., 2020; Shin et al., 2021a) and Codex (Chen et al., 2021a) struggle in few-shot learning on SKG tasks.

UNIFIEDSKG further enables a series of controlled experiments on the importance of structured knowledge encoding choices across SKG tasks. We find that T5 (Raffel et al., 2020) is sensitive to structured knowledge encoding variations, and the sensitivity varies across tasks. We hope UNIFIEDSKG promote the progress of more general, robust structured knowledge encoding methods. Finally, we conduct a comprehensive error analysis to shed light on where current PLMs fall short on different SKG tasks. Although the number of errors made by PLMs decreases with model size, T5-3B may still generate invalid or nonsense outputs.

In summary, we 1) unify and benchmark 21 SKG tasks under the UNIFIEDSKG framework to evaluate SKG across diverse grounding goals and structured knowledge sources, 2) demonstrate (near) SOTA performance of T5 on all SKG tasks with maximum simplicity and generality, 3) show knowledge sharing across SKG tasks via multi-task prefix-tuning, and 4) analyze recent modeling contributions (zero-shot, few-shot, and structured knowledge encoding) across these tasks. We hope UNIFIEDSKG facilitates the community to design new models and learning algorithms that generalize to a diverse set of SKG tasks and to identify their associated challenges.

2 Related Work

SKG with PLMs PLMs have been applied to several SKG tasks. To encode structured knowl-

Task Family	Task	Knowledge Input	User Input	Output
Semantic Parsing	Spider (Yu et al., 2018)	Database	Question	SQL
	GrailQA (Gu et al., 2021)	Knowledge Graph	Question	s-Expression
	WebQSP (Yih et al., 2016)	Knowledge Graph	Question	s-Expression
	MTOP (Li et al., 2021)	API Calls	Question	TOP representation
Question Answering	WikiSQL (Zhong et al., 2017)	Table	Question	Answer
	WikiTQ (Pasupat and Liang, 2015)	Table	Question	Answer
	CompWebQ (Talmor and Berant, 2018)	Knowledge Graph	Question	Answer
	HybridQA (Chen et al., 2020d)	Table + Text Passage	Question	Answer
	MultiModalQA (Talmor et al., 2021)	Table + Text + Image	Question	Answer
	FeTaQA (Nan et al., 2021a)	Table	Question	Free-Form Answer
Data-to-Text	DART (Nan et al., 2021b)	Triple	None	Text
	ToTTo (Parikh et al., 2020)	Highlighted Table	None	Text
Conversational	MultiWoZ2.1 (Eric et al., 2019)	Ontology	Dialog	Dialog State
	KVRET (Eric et al., 2017)	Table	Dialog	Response
	SParC (Yu et al., 2019b)	Database	Multi turn	SQL
	CoSQL (Yu et al., 2019a)	Database	Dialog	SQL
	SQA (Iyyer et al., 2017)	Table	Multi turn	Answer
Fact Verification	TabFact (Chen et al., 2020c)	Table	Statement	Boolean
	FEVEROUS (Aly et al., 2021)	Table + Text	Statement	Boolean
Formal-Language-to-Text	SQL2Text (Shu et al., 2021)	Optional Database	SQL	Text
	Logic2Text (Chen et al., 2020e)	Table Schema	Python-like program	Text

Table 1: We unify 21 SKG tasks with different knowledge input, user input, and output, covering six task families.

edge, prior work linearized the structured knowledge and concatenated it with the text (Hwang et al., 2019; Liu et al., 2020; Hosseini-Asl et al., 2020; Liu et al., 2021), which has been augmented by positional encoding (e.g., row/column embedding) for tables (Herzig et al., 2020; Yin et al., 2020a) and template-based linearization for tables and knowledge graphs (Chen et al., 2020b,c; Oguz et al., 2021). More recently, cell-column alignment is modeled by manipulating the attention matrix of transformers (Eisenschlos et al., 2021; Zhang et al., 2020). Hierarchical encoding is another way to represent the structure, e.g., Wang et al. (2021b) leveraged tree-based transformers to represent the structure of the tables; Iida et al. (2021) used transformers to encode row and column representations; Chen et al. (2020a) used hierarchical transformers to encode KG triples. SKG's outputs include, but are not limited to, formal language, dialogue state, natural language, set of answers, and boolean values. Among them, the formal language is challenging for PLMs trained with natural language. To bridge the gap, Shin et al. (2021b) adopted the insights from Berant and Liang (2014) and Marzoev et al. (2020) and proposed to convert the formal language into English-like representations, which is mapped back to formal language automatically. We refrain from exploring encoding architectures and decoding algorithms. Instead, we systematically study how well vanilla PLMs perform and how sensitive they are to encoding variations.

Task format unification Recent years witnessed the trend of unifying related but different tasks into a shared format. McCann et al. (2018) unified various tasks as question answering, Yin et al. (2020b); Wang et al. (2021a) unified few-shot learning as textual entailment, and PLUR (Chen et al., 2021b) unified program learning, understanding, and repair tasks into a graph-to-sequence format. In this paper, we focus on the text-to-text format (Raffel et al., 2020), given its flexibility. Different from unifying tasks that only take text as input, a core challenge in unifying SKG tasks into the text-to-text format is to linearize structured knowledge into sequences. Related to our work, UnifiedQA (Khashabi et al., 2020) examined the feasibility of unifying QA tasks, an important task family that belongs to SKG, while UNIFIEDSKG consists of six task families for systematic exploration.

Cross-task generalization with PLMs Going beyond task boundaries, multi-task learning and transfer learning view different tasks as related, which have been shown to outperform single-task learning (Aghajanyan et al., 2021a; Vu et al., 2021). Recently, large PLMs show potential for zero-shot and few-shot learning, e.g., GPT-2 (Radford et al., 2019) and GPT-3 (Brown et al., 2020), which can be further improved by multi-task and transfer learning (Zhong et al., 2021), e.g., FLAN (Wei et al., 2021), T0 (Sanh et al., 2021), and CrossFit (Ye et al., 2021a). ExT5 (Aribandi et al., 2021) shows that scaling up the number of tasks in multi-task learning helps improve pretraining sample efficiency and downstream performances. UNIFIEDSKG facilitates the investigation of multi-task, zero-shot, and few-shot learning in SKG.

3 The UNIFIEDSKG Framework

The goal of this paper is not to innovate on model architectures from the modeling perspective. Our contribution is to demonstrate that a single UNIFIEDSKG framework is able to achieve (near) state-of-the-art performance on all 21 SKG tasks but with maximum simplicity and generality.

3.1 Task Unification

The guiding principle of UNIFIEDSKG's task selection is diversity. To this end, we unify 21 SKG tasks across six task families and multiple domains (Table 1). Our task families include:

- **Semantic parsing** converts questions to a logical form (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005).
- **Question answering** derives answers to natural language questions based on structured data (Berant et al., 2013).
- **Data-to-text generation** summarizes structured data in natural language (Novikova et al., 2017).
- **Fact verification** checks if a statement is true based on structured data (Chen et al., 2020c)
- **Conversational tasks** requires understanding of not only the user's last request but also the full interaction history between user and machine (Eric et al., 2019; Yu et al., 2019a).
- **Formal language to text translation** describes formal language in natural language (Chen et al., 2020e).

Tasks in all these families generally take as input x a user request, a structured knowledge input, and an optional (dialogue) context to predict an output y. Figure 2 illustrates and exemplifies how we convert the input x to an input sequence $\tilde{x}$ and the output y to an output sequence $\tilde{y}$ by means of "linearization" (Liu et al., 2021). Linearization enables the unification of diverse forms of structured knowledge, which is key to UNIFIEDSKG. We provide more details, examples, and input length analysis in the Appendices E and F. Code implementation for UNIFIEDSKG uses Huggingface's Transformers (Wolf et al., 2020) and Datasets (Lhoest et al., 2021) toolkits.

3.2 Modeling

The simplest usage of UNIFIEDSKG is to train text-to-text PLMs on individual tasks. In this case, training minimizes the negative log-likelihood loss that we average over tokens contained in each batch, as done by the Transformers (Wolf et al., 2020) toolkit. For decoding, we use beam search by default. UNIFIEDSKG also facilitates exploration of multi-task learning, few-shot, and zero-shot learning with PLMs, and details are introduced in the corresponding parts in Section 4.

4 Experiments and Analysis

4.1 Experiments and Results on Individual Tasks

As a first step, we apply T5-family models (Raffel et al., 2020) on each individual task in UNIFIEDSKG. For model training, we set the maximum number of epochs as 50–200, depending on dataset size for each task. We use early stopping and model selection on the development set. More details are provided in Appendix C.1. For each task, we select one metric commonly used by previous work, and results of full metrics are presented in Appendix A.

Comparison with previous SOTA Table 2 shows that, except for some semantic parsing tasks, vanilla T5-3B outperforms almost all previous SOTA not trained on extra unsupervised in-domain data. Some semantic parsing SOTA models, denoted as $^+$ in Table 2, are also T5 with *post hoc* modification, e.g., constrained decoding (Scholak et al., 2021) or reranking (Ye et al., 2021b). We conclude that T5, with simple modification when necessary, achieves SOTA on almost all tasks. This shows that a generalist architecture like T5, when scaling up to a certain size, could be as good as task-specific architecture on SKG tasks, bring more confidence for researchers to continue exploring larger PLMs.

Model scalability In general, T5 performance increases with model size, but this trend varies for different task families. Semantic parsing, QA, and fact verification tasks have large benefits from increasing model size, while text generation has marginal benefits from increasing model size. See Section 4.5 for a human evaluation for generation tasks. A general observation is that the gap between T5-base and T5-large is larger than that between T5-large and T5-3B.

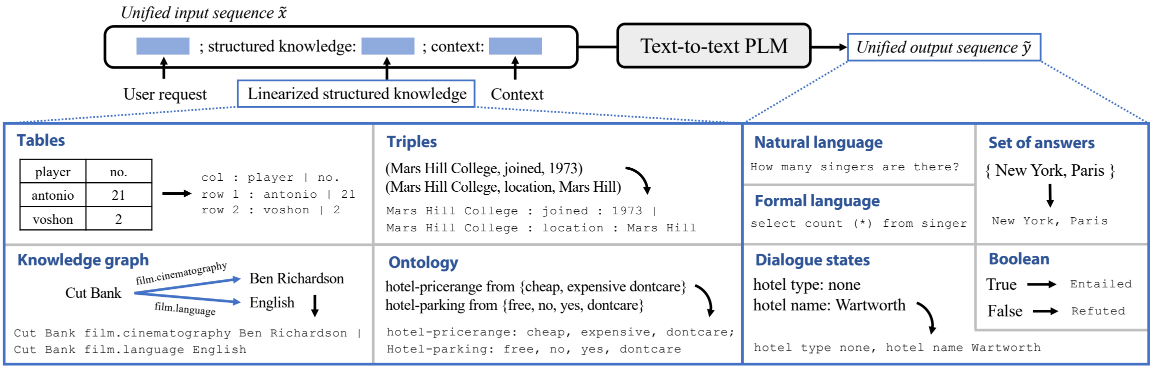


Figure 2: We unify SKG tasks with heterogeneous inputs and outputs into the text-to-text format.

	Metric	T5-base	T5-large	T5-3B	Previous SOTA (w/o extra)	Previous SOTA (w/ extra)
Spider (dev)	Match	58.12	66.63	71.76	**75.5**$^{+}$ (Scholak et al., 2021)	74.7 (Rubin and Berant, 2021)
GrailQA	Match	62.39	67.30	70.11	**83.8**$^{+}$ (Ye et al., 2021b)	—
WebQSP	F1	78.83	79.45	80.70	**83.6**$^{+}$ (Ye et al., 2021b)	—
MTOP	Match	85.49	86.17	**86.78**	86.36 (Pasupat et al., 2021)	—
WikiTQ	Acc	35.76	43.22	**49.29**	44.5 (Wang et al., 2019)	57.5 (Liu et al., 2021)
WikiSQL	Acc	82.63	84.80	**85.96**	85.8 (Liu et al., 2021)	89.5 (Liu et al., 2021)
CompWebQ	Acc	68.43	71.38	**73.26**	70.4‡ (Das et al., 2021)	—
HybridQA (dev)	Acc	54.07	56.95	59.41	**60.8**‡ (Eisenschlos et al., 2021)	63.4‡ (Eisenschlos et al., 2021)
MultiModalQA$^{\top}$(dev)	F1	75.51	81.84	**85.28**	82.7 (Yoran et al., 2021)	83.8 (Yoran et al., 2021)
FeTaQA	BLEU	29.91	32.45	**33.44**	30.54 (Nan et al., 2021a)	—
DART	BLEU	46.22	**46.89**	46.66	46.89† (Nan et al., 2021b)	47.2 (Aghajanyan et al., 2021b)
ToTTo (dev)	BLEU	48.29	**48.95**	**48.95**	48.95† (Kale and Rastogi, 2020)	—
MultiWoZ2.1	Joint Acc	54.64	54.45	55.42	60.61* (Dai et al., 2021)	60.48 (Yu et al., 2021)
KVRET	Micro F1	66.45	65.85	**67.88**	63.6 (Gou et al., 2021)	—
SParC (dev)	Match	50.54	56.69	**61.51**	54.1 (Hui et al., 2021)	62.2 (Yu et al., 2021)
CoSQL (dev)	Match	42.30	48.26	54.08	**56.9**$^{+}$ (Scholak et al., 2021)	52.1 (Yu et al., 2021)
SQA	Overall Acc	52.91	61.28	**62.37**	58.6 (Liu et al., 2021)	74.5 (Liu et al., 2021)
TabFact	Acc	76.13	80.85	**83.68**	74.4 (Yang et al., 2020)	84.2 (Liu et al., 2021)
FEVEROUS$^{\top}$(dev)	Acc	75.05	79.81	**82.40**	82.38$^{\diamond}$ (Aly et al., 2021)	—
SQL2Text	BLEC	93.52	93.68	**94.78**	93.7 (Shu et al., 2021)	—
Logic2Text	BLEC	90.66	90.57	**91.39**	88.6 (Shu et al., 2021)	—

Table 2: Test or dev set performance of models trained on individual tasks. Vanilla T5 or T5 with simple modification (constrained decoding or reranking, used by models marked by $^{+}$) achieve SOTA on nearly all tasks. The best result without extra pretraining is shown in **bold**. More detailed results in full metrics and with standard variances can be found in Table 11 and 12 in Appendix. Human evaluation for generation tasks is in Section 4.5. *w/ (w/o) extra* means with (without) extra pretraining on unsupervised structured data (e.g., web tables).[2]

	Spider	WikiTQ	DART	MWoZ	TabFact	SQL2Text
T5-3B	**71.76**	**50.65**	**50.38**	58.46	83.97	92.71
T0-3B	68.09	50.62	50.16	**60.20**	**85.51**	**92.93**

Table 3: Comparison between T5-3B and T0-3B. T0-3B is initialized from language model-adapted T5 and further pretrained on a large number of non-SKG tasks. We finetune both models on individual tasks. T0-3B under-performs T5-3B on semantic parsing (Spider) and outperforms T5-3B on dialogue state tracking (MWoZ) and fact verification (TabFact).

Effect of structured knowledge pretraining Some smaller models pretrained on structured knowledge (Liu et al., 2021) show competitive performance as T5-3B, suggesting that pretraining with structured data is beneficial for SKG tasks. This observation calls for structured knowledge pretraining that generalizes to different SKG tasks across domains, which can be systematically explored using UNIFIEDSKG.

Effect of pretraining on non-SKG tasks T0-3B (Sanh et al., 2021) is initialized from T5-3B and pretrained on multiple tasks, most of which do not use structured knowledge as input (we refer to them as non-SKG tasks). Exploring the performance of T0-3B on SKG tasks helps us understand the relationship between SKG tasks and non-SKG tasks. Specifically, we finetune T5-3B and T0-3B on individual tasks and report their performance in Table 3. We find that T0-3B under-performs T5-

	T5-base				T5-large	
	ST-F	ST-P	MT-F	MT-P	ST-F	MT-P
Spider	58.12	58.61	58.90	**59.86**	66.63	**67.60**
GrailQA	60.00	61.33	56.00	**62.67**	**67.00**	65.33
WebQSP	72.50	73.81	67.25	**74.77**	73.96	**74.92**
MTOP	**83.89**	82.93	78.79	82.77	**84.70**	84.34
WikiTQ	36.94	36.42	**41.15**	39.74	43.30	**50.90**
WikiSQL	**84.50**	83.09	81.85	84.44	86.27	**87.45**
CompWQ	66.71	67.85	68.28	**69.70**	68.85	**71.27**
HybridQA	54.07	**54.93**	53.52	54.88	56.95	**57.33**
MMQA	75.51	75.50	**76.63**	76.40	81.84	**84.59**
FeTaQA	29.00	28.03	**31.85**	29.33	30.94	**32.48**
DART	50.62	50.33	49.74	**50.68**	**51.72**	50.82
ToTTo	**48.29**	45.70	45.29	45.21	**48.95**	47.90
MWoZ	**57.52**	56.67	53.19	57.06	58.23	**59.24**
KVRET	20.04	19.68	18.53	**21.32**	18.84	**20.76**
SParC	50.54	51.04	**51.70**	51.29	56.69	**59.02**
CoSQL	42.30	44.39	43.59	**45.68**	48.26	**51.64**
SQA	49.49	44.81	**51.48**	48.43	**59.12**	58.15
TabFact	76.34	75.74	71.19	**77.86**	81.4	**83.62**
FEVER.	75.05	75.33	76.85	**78.02**	79.81	**82.05**
SQL2Text	93.69	**94.50**	93.57	93.79	93.35	**93.93**
Logic2Text	92.15	**95.25**	92.24	94.70	92.88	**93.61**
Total para.	$21T$	$T+21P$	T	$T+21P$	$21T$	$T+21P$
Avg. score	60.82	60.76	60.08	**61.84**	64.27	**65.57**

Table 4: Multi-task learning results. ST and MT stand for single-task and multi-task. F and P stand for finetuning and prefix-tuning. For total parameters, T denotes the number of T5 parameters, and P denotes the number of prefix parameters. Notably, $P \ll T$. Multi-task learning with prefix improves the performance on most tasks, largely improving the overall performance. We report results on the development set.

3B on semantic parsing (Spider) and outperforms T5-3B on dialogue state tracking (MultiWoZ) and fact verification (TabFact). We notice that T0-3B is pretrained on dialogue-related tasks (e.g., dialogue QA and summarization) and NLI tasks; therefore, pretraining on non-SKG tasks might not be useful for SKG unless we add similar SKG tasks to the pretraining set.

4.2 Multi-Task Learning

UNIFIEDSKG facilitates the exploration of multi-task learning. In this part, we systematically study multi-task learning on all 21 unified tasks. We conduct full analysis using T5-base and then report our multi-task prefix-tuning results on T5-large. Our observation is that SKG tasks benefit from multi-task prefix-tuning on both T5-base and T5-large, showing the benefits brought by multi-task learning is scalable in terms of the model size.

Baselines We setup the following baselines for multi-task leaning:

- Single-task finetuning (ST-F), which is the same as Section 4.1.
- Single-task prefix-tuning (ST-P) (Li and Liang, 2021), which learns lightweight task-specific parameters while keeping the PLM fixed. We set the prefix length as 10.
- Multi-task finetuning (MT-F), which combines the training data of all datasets with the temperature mixing (Raffel et al., 2020; after hyperparameter tuning with a few training steps, we set the temperature as 2). We choose the checkpoint with the highest average metric on all tasks.

Table 4 shows that prefix-tuning is comparable to finetuning on nearly all tasks. However, we find that it takes about 5–10 times as many training steps to reach comparable performance, which is similarly observed for prompt-tuning (Lester et al., 2021) (See Appendix D). We also observe that multi-task finetuning leads to mixed results. For many tasks, naive multi-task learning is even worse than single-task learning.

Multi-task prefix-tuning (MT-P) Our explanation for the mixed results of naive multi-task finetuning is that the inputs of SKG tasks contain different structured knowledge from diverse domains, making it difficult to learn shared parameters effectively. To address this challenge, we train a prefix on all tasks, freezing T5 and using the same temperature sampling as multi-task finetuning. We then initialize each task's prefix with this pretrained prefix and optimize the prefix while freezing T5. This initialization step is similar to the prompt transfer explored in Vu et al. (2021). Following single-task prefix-tuning, we set the prefix length as 10. Results in Table 4 show that multi-task prefix-tuning outperforms single-task finetuning and single-task prefix-tuning on most tasks, and it largely outperforms the naive multi-task learning baseline. It demonstrates that SKG tasks can be studied together to share data and knowledge.

Exploring task knowledge transfer Text-to-text unification facilitates studying knowledge transfer between tasks. Given two tasks, *task A* and *task B*, we first train the model on task A and then continue training on task B. From Table 5, we observe

[2]‡We use gold linking, but the SOTA does not. †We report our result if the SOTA is also vanilla T5. *It also uses vanilla T5, but its text-to-text format's cost scales linearly with the number of dialogue state slots. ◇We apply the model in Aly et al. (2021) on the validation subset where at least one table is used as evidence. ⊤We only keep the examples that include at least one table, and no image involves.

Task A	Task B	Type	B only	A to B
WikiSQL	TabFact	same source	81.43	82.76
TabFact	WikiTQ	same source	43.30	45.88
WikiSQL	FeTaQA	same source	30.94	31.19
Spider	GrailQA	parallel tasks	67.00	67.00
Spider	WikiTQ	subtask	43.30	41.68
Spider	TabFact	weakly related	81.43	80.39

Table 5: Task knowledge transfer. We use T5-large in this table. *B only* means training the model on task B; *A to B* means to train the model on task A and then to finetune the model on task B. In both settings, we report task B's development set performance. We find that tasks benefit from other tasks with the same data source.

	T5-3B	T0 3B	GPT-3 175B		Codex 175B	
	finetune	*zero-shot*	*select*	*random*	*select*	*random*
Spider	71.76	0.00	20.00	$18.33_{3.78}$	40.72	$43.23_{4.16}$
WikiTQ	50.65	12.68	32.00	$29.33_{9.04}$	26.21	$20.46_{4.21}$
DART	50.38	23.42	40.23	$34.21_{4.50}$	42.13	$36.54_{1.67}$
MWoZ	58.46	0.00	18.00	$0.02_{0.02}$	23.47	$0.06_{0.03}$
TabFact	83.97	52.45	51.00	$49.67_{3.79}$	50.97	$51.58_{1.59}$
SQL2Text	92.71	39.64	94.00	$85.00_{2.65}$	90.64	$88.31_{1.61}$

Table 6: Zero-shot and few-shot learning for SKG. Subscripts show the standard deviation with three runs. *select* means to select the most similar training samples as few-shot examples, while *random* means to randomly select training samples as few-shot examples. T0 performs poorly on all the tasks in the zero-shot setting. Codex outperforms GPT3 on tasks that generate structured programs (Spider and MultiWoZ).

that tasks benefit from other tasks with the same data source (e.g., WikiSQL, TabFact, WikiTQ, and FeTaQA all use Wikipedia tables as the structured knowledge input). On the other hand, we do not observe positive transfer between *parallel tasks* (e.g., Spider and GrailQA are both semantic parsing, while they have different structured knowledge and different output) and *subtask* (e.g., question answering can be viewed as the execution semantic parses) when data sources are different. Compared to the positive results in Table 4, results in this part indicate that manually selecting source and target tasks may not be efficient for multi-task learning.

4.3 Zero-Shot and Few-Shot Learning

The text-to-text unification of UNIFIEDSKG enables us to investigate zero/few-shot learning on SKG with large, probably black-box, PLMs.

Zero-shot learning setting Zero-shot learning enables models to solve tasks with descriptions without training samples. We adopt T0 (Sanh et al., 2021) and follow Sanh et al. (2021) to create similar instructions for the unseen tasks. Our instructions are provided in Appendix C.3.

Few-shot learning settings Brown et al. (2020) showed that large PLMs could be few-shot learners by encoding a few training samples and the test input as "context" and directly generating the answer without gradient updates. We use GPT-3 (Brown et al., 2020) and Codex (Chen et al., 2021a) to explore such few-shot learning for SKG. For GPT3, we randomly sampled 100 data points from the development set to stay within our budget. Specifically, we explore two settings.

In the first setting, we randomly sample few-shot examples, which are shared for all development set samples, denoted as *random* in Table 6. For linearized sequences that are too long for Codex (4096) and GPT3 (2048), we use as many samples as possible and make sure that there is at least one few-shot example using truncation.

In the second setting, we follow Gao et al. (2021) to select few-shot examples from the training set. We call this setting *few-shot with example selection*, denoted as *select* in Table 6. Specifically, we use the pretrained SBERT (Reimers and Gurevych, 2020) to obtain the sentence embeddings for user request input (for tasks that only have structured input, we embed the linearized structured input), and sample 5 most similar examples by cosine similarity. Further details (prompts and task instructions) on GPT3 and Codex are provided in Appendix C.4.

SKG is challenging for zero/few-shot learning. Table 6 shows that zero-shot performance is very poor for most tasks, with Spider and MultiWoZ even being 0. Table 6 shows a large gap between few-shot learning (both the true few-shot and few-shot with example selection) and finetuning for Spider, WikiTQ, MWoZ, and TabFact, while the gap is smaller for generation tasks. Also, few-shot with example selection outperforms few-shot learning without example selection, but the gap is usually smaller than 10 points (out of 100). It is also interesting to compare the results between synthesis tasks like Spider that require predicting programs, and *induction*-style tasks like WikiTQ and TabFact, where a model directly outputs answers using latent representations of programs (Devlin et al., 2017). We find language models generally struggle more when adapting to induction tasks (e.g., close to random-guess on the binary classification task TabFact), reminiscent of recent attempts in program synthesis and induction using PLMs (Austin et al., 2021). To prompt GPT3 and Codex, we employed prompts that are similar to our UNIFIEDSKG lin-

	Spider	WikiTQ	MultiWoZ2.1	TabFact
rs(c)	$66.63_{2.31}$	$43.30_{0.25}$	$58.23_{0.39}$	$81.43_{0.16}$
sr	64.12	38.78	—	80.98
rcs	—	—	58.89	—

Table 7: Ordering of inputs. Subscripts show the standard deviation with three runs. s, r, and c stand for the structured knowledge, request input, and conversational context. Placing r before s is always better, and placing c between r and s is better for dialogue state tracking (MultiWoZ2.1).

	Spider	WikiTQ	DART	MultiWoZ2.1
Same Order	$66.63_{2.31}$	$43.30_{0.25}$	$51.72_{0.15}$	$58.23_{0.39}$
Reversed Order	64.80	37.80	48.47	13.59

Table 8: Order-sensitivity of structured knowledge. Subscripts show the standard deviation with three runs. *Same Order* is the default benchmark setting. *Reversed Order* means to reverse the structured knowledge ordering on the development set. Tasks with cross-domain tables (in WikiTQ), databases (in Spider), and triples (in DART) are less order-sensitive, while pre-defined ontology (in MultiWoZ2.1) is highly order-sensitive.

earized inputs. We expect better task prompting design might result in better performance with GPT3 and Codex.

4.4 Structured Knowledge Encoding

Structured knowledge encoding methods have been explored in many prior studies (Bogin et al., 2019; Lin et al., 2019; Yin et al., 2020a; Herzig et al., 2020; Agarwal et al., 2020; Saxena et al., 2020; Yasunaga et al., 2021; Oguz et al., 2021). UNIFIEDSKG facilitates the systematic study of structured knowledge encoding. We hope UNIFIEDSKG promote the progress of general structured knowledge encoding methods. Focused on the conversion of structured knowledge into sequences, we frame this part into the following questions.

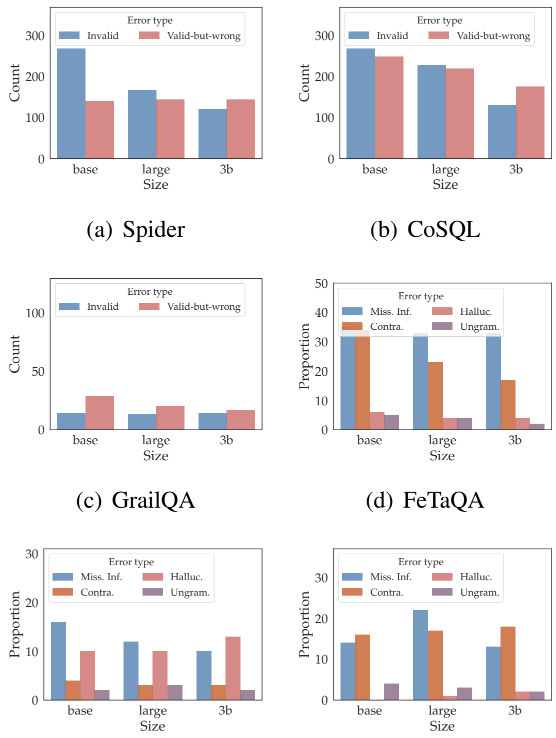
Does the ordering of user input, structured knowledge, and context matter? Our sequence input in Section 3.1 requires a specification of the order of user input, structured knowledge, and context. To explore the effect of such ordering, we switch the ordering of user input, structured knowledge, and the context in both the training and development set and re-train and re-evaluate T5-large. Table 7 show that placing the text before structured knowledge (*rs*) is better than the opposite (*sr*) on Spider and WikiTQ. Interestingly, this finding is consistently held across different SKG tasks. A possible reason is that the position of the text is relatively fixed in *rs*, helping the decoder to learn stable attention over the text. Also, placing the context in between the text and structured knowledge has better results.

	Spider	WikiSQL	TabFact
Linearization	40.23	59.21	58.77
Natural Language	38.59	63.16	58.56

Table 9: Converting structured knowledge into natural language for low-resource learning. A large improvement is observed on question answering (WikiSQL), but not on text2SQL semantic parsing (Spider) and fact verification (TabFact).

Is T5 sensitive to structured knowledge ordering? Order-insensitivity is common for most structured knowledge, e.g., permutation of columns in a table preserves the meaning. However, T5 with structured knowledge linearization is perhaps sensitive to ordering. To study this insensitivity, we evaluate T5-large on a manipulated development set where items in the structured knowledge inputs are reversed. We reverse the order of each schema for the database, columns for the table, and slots and values for the ontology, respectively. Results in Table 8 show that tasks with cross-domain tables and databases as input are order-sensitive, while models are very sensitive to the ordering of ontology. More explorations on robustness to other manipulations will be an open question in UNIFIEDSKG.

Is it beneficial to represent structured knowledge as natural language? SKG data are not typically included in pretraining data for PLMs. Given ample training data, PLMs adapt well to SKG tasks, as shown in Table 2. However, under the low-resource setting, converting the structured data to natural language might be helpful. For Spider, we use a shared template to convert structured knowledge to natural language descriptions. For TabFact and WikiSQL, we randomly selected 236 tables shared by both datasets and manually created templates to convert each row as a sentence. Examples of the templates are shown in Appendix H. These templates produce about 1000 examples for each dataset, and we divide them for training and test. We observe that in WikiSQL, the conversion to natural language stabilizes and accelerates the training process, while in TabFact and Spider, it doesn't help much by a large margin. Table 9 shows that the effect of conversion to natural lan-



(a) Spider (b) CoSQL

(c) GrailQA (d) FeTaQA

(e) ToTTo (f) KVRET

Figure 3: Error analysis. For semantic parsing, we plot the number of invalid/valid-but-wrong predictions. For generation, we plot the proportion of missing-information/contradiction/hallucination/ungrammatical errors among all predictions (one prediction may have multiple errors). Full visualization is in Appendix A.

guage is different across tasks. For table input, conversion to natural language improves the performance on WikiSQL but not on TabFact. For the database schema input, conversion to natural language slightly degrades the performance.

4.5 Human Evaluation for Generation Tasks

For each generation task, we randomly sample 100 development set samples and ask human annotators to judge the correctness of each output, using a 0-1 score. Eight of this paper's authors participated, and two of them are native speakers of English. They are familiar with the tasks being evaluated, and the human evaluation guideline is shown in Appendix C.5. Each sample is annotated by three annotators, and we average their scores.[3] Table 10 shows that automatic metrics do not always reflect human evaluation. BLEC (Shu et al., 2021), a 0-1 score based on keyword matching, is over-optimistic about the correctness. These observations call for better automatic metrics to truly reflect the model's ability on generation tasks. Also, larger models are not always better at generation. Detailed error analysis is provided below.

[3]For FeTaQA, DART, ToTTo, KVRET, SQL2Text, and Logic2Text, the overall agreement / free-marginal kappa is 82.89%/0.66, 89.78%/0.80, 81.56%/0.63, 81.56%/0.63, 86.67%/0.73, and 79.33%/0.59.

	Metric	T5-base	T5-large	T5-3B
FeTaQA	BLEU	29.00	30.94	31.73
	Human*†	36.0%	51.3%	57.3%
DART	BLEU	50.62	51.72	50.38
	Human	90.7%	91.7%	87.7%
ToTTo	BLEU	48.29	48.95	48.95
	Human	78.7%	80.0%	81.3%
KVRET	BLEU	20.04	18.84	17.75
	Human†	72.3%	66.3%	75.0%
SQL2Text	BLEC	93.69	93.35	92.71
	Human*	83.7%	90.3%	84.7%
Logic2Text	BLEC	92.15	92.88	91.69
	Human†	77.2%	81.5%	84.2%

Table 10: Automatic metrics and human evaluation on the development set of generation tasks. $^{*}p < 0.05$ for "the rank-1 model is better than the rank-2 model". $^{\dagger}p < 0.05$ for "the rank-2 model is better than the rank-3 model". Generation metrics do not always reflect human evaluation. Larger models are not always better.

4.6 Error Analysis

Error analysis based on output validity Unconstrained decoding from PLMs may generate *invalid outputs*. For semantic parsing, we divide wrong outputs into two groups 1) invalid outputs (i.e., not executable when the output is SQL, and not parse-able when the output is s-expression or TOP-representation), 2) valid but wrong answers. These two groups are denoted as invalid and valid-but-wrong in Figure 3. For SQL semantic parsing (Spider, SParC, and CoSQL), a large number of errors is caused by invalid predictions, and invalid predictions gradually decrease with the increase of model size. This phenomenon is also observed by Scholak et al. (2021), who further used the PICARD method to improve output validity, largely improving the parsing performance. For s-expression semantic parsing (GrailQA and WebQSP), invalid predictions take up 30%-50% of all incorrect predictions, and increasing the model size does not significantly reduce invalidity. Moreover, T5 has few invalid predictions on MTOP, for which constrained decoding may not be very beneficial. For fact verification tasks, invalid outputs are those other than "entailed" and "refuted". We observe that all sizes of T5 trained on individual datasets (TabFact, FEVEROUS) generate valid outputs. For Question Answering tasks, we do not have error analysis in terms of validity since the validity check for an answer is non-trivial and could be imprecise.

Error analysis for text generation tasks For generation tasks, we consider four types of errors: 1) missing information where required information is not shown in the prediction, 2) contradiction where the prediction is contradictory to the structured knowledge or user request 3) hallucination where the prediction contains information that cannot be verified by the structured knowledge or user request, and 4) ungrammatical where the prediction is not grammatical. Figure 3 shows that the proportion of ungrammatical predictions is generally less than 5%, suggesting that T5 is good at generating grammatical sentences. Missing information and contradiction are common errors made by T5, and performance gains generally come from reducing contradiction. Hallucination is not a common error made by T5 except for the highlighted-table-to-text task (ToTTo), where T5 tends to output information of non-highlighted cell values.

Case study We discuss some representative examples of model prediction and summarize interesting observations (More in Appendix G). Compared with T5-base and T5-large, T5-3B's predictions tend to be more diverse and creative as shown in Appendix G.2 and G.7. Also, T5-3B sometimes leverages domain knowledge to summarize facts (e.g., describing *rating 5 out of 5* as *low*) in some tasks such as DART, while the other two models would just copy the original expressions in the input, as shown in Appendix G.5 and G.6. However, this ability puts T5-3B in the risk of manipulating information and twisting the meaning of user request (Appendix G.3.2 and G.4). Moreover, we observe that T5-base and T5-large sometimes omit the punctuation marks in the outputs.

5 Conclusions and Future Directions

In this paper, we propose the UnifiedSKG framework to promote systematic research on structured knowledge grounding by unifying 21 SKG tasks from six task families into the text-to-text format. Using UnifiedSKG as a benchmark, we demonstrate that finetuning T5 on individual tasks achieves near or above SOTA results on almost all 21 tasks, establishing it as a powerful and reproducible starting point for SKG research. We demonstrate that multi-task prefix-tuning improves the performance on most SKG tasks, largely improving the overall performance. UnifiedSKG enabled controlled experiments on the structured knowledge encoding variants, and we find that the effects of these variations depend on the task being studied. UnifiedSKG also facilitates the investigation of multi-task learning. Moreover, UnifiedSKG is a challenging testbed for zero-shot and few-shot learning, shown by the poor results of powerful, large PLMs.

Future directions With UnifiedSKG, new models can be easily applied to diverse SKG tasks, and new tasks can be easily framed in terms of the standardized abstraction. It promotes a systematic study on more general and robust advances in structured knowledge encoding, multi-task learning, zero-shot learning, and few-shot learning for SKG tasks. It also would be interesting to explore general pretraining methods within UnifiedSKG, which potentially benefit all the unified tasks.

When the structured knowledge inputs are too long for current GPUs' capacity, we truncate them based on heuristic rules. Although we expect future increases in GPU capacity may decrease the importance of structured knowledge truncation, it is still beneficial to jointly study SKG and retrieval since structured knowledge can be arbitrarily long.

Since we select popular tasks from each of the six task families, we put ourselves at the risk of being disproportionate in terms of the data domain and population, and we actively welcome new and diverse tasks to be added into UnifiedSKG. Additionally, our error analysis can be further improved to be more fine-grained, and we hope our findings can promote future work on systematically studying and decomposing the behavior of PLMs on SKG tasks. As mentioned by de Vries et al. (2020), training and evaluation data should reflect the intents and linguistic phenomena in real-world applications, suggesting more realistic tasks to be added into UnifiedSKG.

Furthermore, UnifiedSKG provides the correct type of structured knowledge for each task. However, how a system searches for correct structured knowledge resources, takes appropriate action, and integrates information and results from multiple structured sources given a user request is still under-explored, which is necessary to build a unified multi-purpose SKG system.

6 Contributions

Code implementation Tianbao Xie and Chen Henry Wu implemented the code base of the UNIFIEDSKG framework and experiment pipeline. The code of PICARD and advice from Torsten Scholak sped up the implementation.

Task unification Tianbao Xie, Peng Shi, Michihiro Yasunaga, Chen Henry Wu, and Ming Zhong implemented the 21 tasks into the text-to-text format, adapted the metrics, and verified the performances.

Paper writing Chen Henry Wu and Tianbao Xie finished most part of the paper. Michihiro Yasunaga, Peng Shi, and Chengzu Li added results and analysis for their corresponding parts. Peng Shi drafted related work on SKG with PLMs. Torsten Scholak, Pengcheng Yin, Rui Zhang, Ruiqi Zhong, Victor Zhong, Michihiro Yasunaga, Connor Boyle, Chien-Sheng Wu, Sida Wang, Bailin Wang, Ansong Ni, Ziyu Yao, Lingpeng Kong, Caiming Xiong, Dragomir Radev, Noah A. Smith, and Luke Zettlemoyer carefully reviewed the paper and gave feedback for multiple rounds.

Experiments Chen Henry Wu, Tianbao Xie, and Chien-Sheng Wu conducted experiments on individual tasks and multi-task learning. Tianbao conducted the zero-shot learning experiments. Chengzu Li and Tianbao Xie conducted the few-shot learning experiments. Tianbao Xie conducted experiments on the ordering of sequence inputs and order-sensitivity. Chengzu Li, Connor Boyle, and Peng Shi conducted the experiments on converting structured knowledge into natural language.

Human evaluation Chen Henry Wu organized the human evaluation. Torsten Scholak, Rui Zhang, Chengzu Li, Connor Boyle, Tianbao Xie, Peng Shi, Tao Yu, and Chen Henry Wu were the human participants.

Error analysis and case study Tianbao Xie, Chen Henry Wu, and Michihiro Yasunaga designed and conducted the error analysis for semantic parsing and generation tasks. Authors who participated in the human annotation selected the cases for case study.

Discussion We had three separate weekly meetings, and everyone in the project attended one of them. Torsten Scholak, Ruiqi Zhong, Pengcheng Yin, Victor Zhong, Peng Shi, Rui Zhang, Sida Wang, and Lingpeng Kong actively provided advice. Torsten Scholak provided signals that prefix-tuning would be comparable to fine-tuning. Ruiqi Zhong gave advice on analyzing the effect of model size, Pengcheng Yin and Peng Shi gave advice on analysis on converting structured knowledge into natural language. Pengcheng Yin helped interpret experimental results. Ziyu Yao suggested that we report both SOTA (w/ extra) and SOTA (w/o extra) for a fair comparison. Victor Zhong and Bailin Wang gave valuable suggestions on multi-task learning and task transfer analysis. Luke Zettlemoyer, Noah A. Smith, Caiming Xiong, and Dragomir Radev gave valuable comments on research questions and experimental design.

Computing resources We thank Salesforce Research, an Amazon Research Reward, ElementAI, and Yale NLP for providing computing resources generously.

Tao Yu designed and led the research.

Acknowledgement

We thank Yifei Min and Libo Qin for early stage discussion. We thank Qian Liu for TAPEX code and advice on question answering tasks. We thank wandb for free logging and OpenAI for free Codex usage.

References

Oshin Agarwal, Heming Ge, Siamak Shakeri, and Rami Al-Rfou. 2020. Knowledge graph based synthetic corpus generation for knowledge-enhanced language model pre-training. *arXiv preprint arXiv:2010.12688*.

Armen Aghajanyan, Anchit Gupta, Akshat Shrivastava, Xilun Chen, Luke Zettlemoyer, and Sonal Gupta. 2021a. Muppet: Massive multi-task representations with pre-finetuning. In *Proceedings of EMNLP 2021*, pages 5799–5811, Online and Punta Cana, Dominican Republic.

Armen Aghajanyan, Dmytro Okhonko, Mike Lewis, Mandar Joshi, Hu Xu, Gargi Ghosh, and Luke Zettlemoyer. 2021b. Htlm: Hyper-text pre-training and prompting of language models.

Rami Aly, Zhijiang Guo, Michael Sejr Schlichtkrull, James Thorne, Andreas Vlachos, Christos Christodoulopoulos, Oana Cocarascu, and Arpit Mittal. 2021. The fact extraction and VERification over unstructured and structured information (FEVEROUS) shared task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 1–13.

Vamsi Aribandi, Yi Tay, Tal Schuster, Jinfeng Rao, Huaixiu Steven Zheng, Sanket Vaibhav Mehta, Honglei Zhuang, Vinh Q. Tran, Dara Bahri, Jianmo Ni, Jai Gupta, Kai Hui, Sebastian Ruder, and Donald Metzler. 2021. Ext5: Towards extreme multi-task scaling for transfer learning.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc V. Le, and Charles Sutton. 2021. Program synthesis with large language models. *ArXiv*, abs/2108.07732.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *EMNLP 2013*, pages 1533–1544.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of ACL 2014*, pages 1415–1425.

Ben Bogin, Matt Gardner, and Jonathan Berant. 2019. Global reasoning over database structures for text-to-sql parsing. In *Proceedings of EMNLP 2019*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021a. Evaluating large language models trained on code.

Sanxing Chen, Xiaodong Liu, Jianfeng Gao, Jian Jiao, Ruofei Zhang, and Yangfeng Ji. 2020a. Hitter: Hierarchical transformers for knowledge graph embeddings. *arXiv preprint arXiv:2008.12813*.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. 2020b. Logical natural language generation from open-domain tables. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7929–7942, Online. Association for Computational Linguistics.

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020c. Tabfact : A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. 2020d. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *Findings of EMNLP 2020*.

Zhiyu Chen, Wenhu Chen, Hanwen Zha, Xiyou Zhou, Yunkai Zhang, Sairam Sundaresan, and William Yang Wang. 2020e. Logic2Text: High-fidelity natural language generation from logical forms. In *Findings of the Association for Computational Linguistics: EMNLP 2020*.

Zimin Chen, Vincent Josua Hellendoorn, Pascal Lamblin, Petros Maniatis, Pierre-Antoine Manzagol, Daniel Tarlow, and Subhodeep Moitra. 2021b. PLUR: A unifying, graph-based view of program learning, understanding, and repair. In *Thirty-Fifth Conference on Neural Information Processing Systems*.

Yinpei Dai, Hangyu Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, and Xiaodan Zhu. 2021. Preview, attend and review: Schema-aware curriculum learning for multi-domain dialogue state tracking. In *Proceedings ACL-IJCNLP 2021 (Volume 2: Short Papers)*, pages 879–885, Online.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of EMNLP 2021*, pages 9594–9611, Online and Punta Cana, Dominican Republic.

Harm de Vries, Dzmitry Bahdanau, and Christopher D. Manning. 2020. Towards ecologically valid research on language user interfaces. *ArXiv*.

Jacob Devlin, Jonathan Uesato, Surya Bhupatiraju, Rishabh Singh, Abdel rahman Mohamed, and Pushmeet Kohli. 2017. Robustfill: Neural program learning under noisy i/o. In *ICML*.

Julian Martin Eisenschlos, Maharshi Gor, Thomas Müller, and William W Cohen. 2021. Mate: Multi-view attention for table transformer efficiency. *arXiv preprint arXiv:2109.04312*.

Mihail Eric, Rahul Goel, Shachi Paul, Abhishek Sethi, Sanchit Agarwal, Shuyag Gao, and Dilek Hakkani-Tur. 2019. Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines. *arXiv preprint arXiv:1907.01669*.

Mihail Eric, Lakshmi. Krishnan, François Charette, and Christopher D. Manning. 2017. Key-value retrieval networks for task-oriented dialogue. In *SIGDIAL Conference*.

Tianyu Gao, Adam Fisch, and Danqi Chen. 2021. Making pre-trained language models better few-shot learners. In *Association for Computational Linguistics (ACL)*.

Yanjie Gou, Yinjie Lei, Lingqiao Liu, Yong Dai, and Chunxu Shen. 2021. Contextualize knowledge bases with transformer for end-to-end task-oriented dialogue systems. In *Proceedings of the EMNLP 2021*, pages 4300–4310, Online and Punta Cana, Dominican Republic.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*.

Jonathan Herzig, P. Nowak, Thomas Müller, Francesco Piccinno, and Julian Martin Eisenschlos. 2020. Tapas: Weakly supervised table parsing via pre-training. In *Proceedings of ACL*.

Ehsan Hosseini-Asl, Bryan McCann, Chien-Sheng Wu, Semih Yavuz, and Richard Socher. 2020. A simple language model for task-oriented dialogue. In *Proceedings of Conference on Neural Information Processing Systems (NeurIPS)*.

Binyuan Hui, Ruiying Geng, Qiyu Ren, Binhua Li, Yongbin Li, Jian Sun, Fei Huang, Luo Si, Pengfei Zhu, and Xiaodan Zhu. 2021. Dynamic hybrid relation network for cross-domain context-dependent semantic parsing.

Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. 2019. A comprehensive exploration on wikisql with table-aware word contextualization. *ArXiv*, abs/1902.01069.

Hiroshi Iida, Dung Thai, Varun Manjunatha, and Mohit Iyyer. 2021. Tabbie: Pretrained representations of tabular data. *arXiv preprint arXiv:2105.02584*.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. 2017. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1821–1831, Vancouver, Canada.

Mihir Kale and Abhinav Rastogi. 2020. Text-to-text pre-training for data-to-text tasks. In *Proceedings of INLG 2020, Dublin, Ireland, December 15-18, 2020*, pages 97–102.

D. Khashabi, S. Min, T. Khot, A. Sabhwaral, O. Tafjord, P. Clark, and H. Hajishirzi. 2020. Unifiedqa: Crossing format boundaries with a single qa system. *EMNLP - findings*.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021*, pages 3045–3059.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander M. Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing.

Haoran Li, Abhinav Arora, Shuohui Chen, Anchit Gupta, Sonal Gupta, and Yashar Mehdad. 2021. MTOP: A comprehensive multilingual task-oriented semantic parsing benchmark. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2950–2962, Online.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 4582–4597, Online.

Bill Yuchen Lin, Xinyue Chen, Jamin Chen, and Xiang Ren. 2019. Kagnet: Knowledge-aware graph networks for commonsense reasoning. In *Proceedings of EMNLP-IJCNLP*.

Zhaojiang Lin, Bing Liu, Seungwhan Moon, Paul A Crook, Zhenpeng Zhou, Zhiguang Wang, Zhou Yu, Andrea Madotto, Eunjoon Cho, and Rajen Subba. 2021. Leveraging slot descriptions for zero-shot cross-domain dialogue statetracking. In *Proceedings of NAACL 2021*, pages 5640–5648.

Qian Liu, Bei Chen, Jiaqi Guo, Zeqi Lin, and Jian guang Lou. 2021. Tapex: Table pre-training via learning a neural sql executor.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. In *AAAI*.

Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2018. Mem2seq: Effectively incorporating knowledge bases into end-to-end task-oriented dialog systems. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1468–1478.

Alana Marzoev, Samuel Madden, M Frans Kaashoek, Michael Cafarella, and Jacob Andreas. 2020. Unnatural language processing: Bridging the gap between synthetic and natural language data. *arXiv preprint arXiv:2004.13645*.

Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. The natural language decathlon: Multitask learning as question answering. *CoRR*, abs/1806.08730.

Linyong Nan, Chiachun Hsieh, Ziming Mao, Xi Victoria Lin, Neha Verma, Rui Zhang, Wojciech Kryściński, Nick Schoelkopf, Riley Kong, Xiangru Tang, Murori Mutuma, Ben Rosand, Isabel Trindade, Renusree Bandaru, Jacob Cunningham, Caiming Xiong, and Dragomir Radev. 2021a. Fetaqa: Free-form table question answering. *TACL*.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, Yangxiaokang Liu, Nadia Irwanto, Jessica Pan, Faiaz Rahman, Ahmad Zaidi, Murori Mutuma, Yasin Tarabar, Ankit Gupta, Tao Yu, Yi Chern Tan, Xi Victoria Lin, Caiming Xiong, Richard Socher, and Nazneen Fatema Rajani. 2021b. Dart: Open-domain structured data record to text generation. In *NAACL*.

Jekaterina Novikova, Ondrej Dusek, and Verena Rieser. 2017. The E2E dataset: New challenges for end-to-end generation. In *SIGDial 2017*, pages 201–206.

Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Scott Yih. 2021. Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering. *arXiv preprint arXiv:2012.14610*.

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of EMNLP*.

Panupong Pasupat and Percy Liang. 2015. 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 (Volume 1: Long Papers)*, pages 1470–1480, Beijing, China.

Panupong Pasupat, Yuan Zhang, and Kelvin Guu. 2021. Controllable semantic parsing via retrieval augmentation. In *Proceedings of EMNLP 2021*, pages 7683–7698, Online and Punta Cana, Dominican Republic.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Belgium, Brussels.

Libo Qin, Xiao Xu, Wanxiang Che, Yue Zhang, and Ting Liu. 2020. Dynamic fusion network for multi-domain end-to-end task-oriented dialog. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6344–6354, Online.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Nils Reimers and Iryna Gurevych. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of EMNLP 2020*.

Hongyu Ren, Hanjun Dai, Bo Dai, Xinyun Chen, Michihiro Yasunaga, Haitian Sun, Dale Schuurmans, Jure Leskovec, and Denny Zhou. 2021. Lego: Latent execution-guided reasoning for multi-hop question answering on knowledge graphs. In *International Conference on Machine Learning (ICML)*.

Ohad Rubin and Jonathan Berant. 2021. SmBoP: Semi-autoregressive bottom-up semantic parsing. In *Proceedings of NAACL 2021*, pages 311–324, Online.

Victor Sanh, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Fevry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush. 2021. Multitask prompted training enables zero-shot task generalization.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Association for Computational Linguistics (ACL)*.

Torsten Scholak, Nathan Schucher, and Dzmitry Bahdanau. 2021. PICARD: Parsing incrementally for constrained auto-regressive decoding from language models. In *Proceedings of EMNLP 2021*, pages 9895–9901.

Peter Shaw, Ming-Wei Chang, Panupong Pasupat, and Kristina Toutanova. 2021. Compositional generalization and natural language variation: Can a semantic parsing approach handle both? In *ACL/IJCNLP*.

Richard Shin, Christopher H. Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021a. Constrained language models yield few-shot semantic parsers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*.

Richard Shin, Christopher H Lin, Sam Thomson, Charles Chen, Subhro Roy, Emmanouil Antonios Platanios, Adam Pauls, Dan Klein, Jason Eisner, and Benjamin Van Durme. 2021b. Constrained language models yield few-shot semantic parsers. *arXiv preprint arXiv:2104.08768*.

Chang Shu, Yusen Zhang, Xiangyu Dong, Peng Shi, Tao Yu, and Rui Zhang. 2021. Logic-consistency text generation from semantic parses. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*.

A. Talmor and J. Berant. 2018. The web as a knowledge-base for answering complex questions. In *North American Association for Computational Linguistics (NAACL)*.

Alon Talmor, Ori Yoran, Amnon Catav, Dan Lahav, Yizhong Wang, Akari Asai, Gabriel Ilharco, Hannaneh Hajishirzi, and Jonathan Berant. 2021. Multimodal{qa}: complex question answering over text, tables and images. In *International Conference on Learning Representations*.

Tu Vu, Brian Lester, Noah Constant, Rami Al-Rfou, and Daniel Cer. 2021. Spot: Better frozen model adaptation through soft prompt transfer.

Bailin Wang, Richard Shin, Xiaodong Liu, Oleksandr Polozov, and Matthew Richardson. 2020. Rat-sql: Relation-aware schema encoding and linking for text-to-sql parsers. In *ACL*.

Bailin Wang, Ivan Titov, and Mirella Lapata. 2019. Learning semantic parsers from denotations with latent structured alignments and abstract programs. In *Proceedings of EMNLP-IJCNLP 2019*, pages 3774–3785, Hong Kong, China.

Quan Wang, Zhendong Mao, Bin Wang, and Li Guo. 2017. Knowledge graph embedding: A survey of approaches and applications. *IEEE Transactions on Knowledge and Data Engineering*, 29(12):2724–2743.

Sinong Wang, Han Fang, Madian Khabsa, Hanzi Mao, and Hao Ma. 2021a. Entailment as few-shot learner. *CoRR*, abs/2104.14690.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. 2021b. Tuta: Tree-based transformers for generally structured table pre-training. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 1780–1790.

Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. 2021. Finetuned language models are zero-shot learners.

Jason D Williams, Antoine Raux, and Matthew Henderson. 2016. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP 2020: System Demonstrations*, pages 38–45, Online.

Chien-Sheng Wu, Richard Socher, and Caiming Xiong. 2019. Global-to-local memory pointer networks for task-oriented dialogue. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Xiaoyu Yang, Feng Nie, Yufei Feng, Quan Liu, Zhigang Chen, and Xiaodan Zhu. 2020. Program enhanced fact verification with verbalization and graph attention network. In *Proceedings of EMNLP 2020*, pages 7810–7825, Online.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics.

Qinyuan Ye, Bill Yuchen Lin, and Xiang Ren. 2021a. Crossfit: A few-shot learning challenge for cross-task generalization in nlp. In *Proceedings of EMNLP*.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2021b. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. *arXiv preprint arXiv:2109.08678*.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Pengcheng Yin and Graham Neubig. 2018. TRANX: A transition-based neural abstract syntax parser for semantic parsing and code generation. In *EMNLP 2018: System Demonstrations*, pages 7–12.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020a. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Wenpeng Yin, Nazneen Fatema Rajani, Dragomir R. Radev, Richard Socher, and Caiming Xiong. 2020b. Universal natural language processing with limited annotations: Try few-shot textual entailment as a start. In *Proceedings of EMNLP 2020, Online, November 16-20, 2020*, pages 8229–8239.

Ori Yoran, Alon Talmor, and Jonathan Berant. 2021. Turning tables: Generating examples from semi-structured tables for endowing language models with reasoning skills. *arXiv preprint arXiv:2107.07261*.

Tao Yu, Rui Zhang, Heyang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. 2019a. CoSQL: A conversational text-to-SQL challenge towards cross-domain natural language interfaces to databases. In *Proceedings of EMNLP 2019*, pages 1962–1979, Hong Kong, China.

Tao Yu, Rui Zhang, Oleksandr Polozov, Christopher Meek, and Ahmed Hassan Awadallah. 2021. SCoRE: Pre-training for context representation in conversational semantic parsing. In *International Conference on Learning Representations*.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. 2018. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of EMNLP 2018*, Brussels, Belgium.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Irene Li Heyang Er, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Vincent Zhang Jonathan Kraft, Caiming Xiong, Richard Socher, and Dragomir Radev. 2019b. Sparc: Cross-domain semantic parsing in context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big Bird: Transformers for Longer Sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297.

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In *AAAI 1996*, pages 1050–1055.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *UAI*.

Hongzhi Zhang, Yingyao Wang, Sirui Wang, Xuezhi Cao, Fuzheng Zhang, and Zhongyuan Wang. 2020. Table fact verification with structure-aware transformer. In *Proceedings of EMNLP 2020*, pages 1624–1629.

Ruiqi Zhong, Kristy Lee, Zheng Zhang, and Dan Klein. 2021. Adapting language models for zero-shot learning by meta-tuning on dataset and prompt collections. In *Findings of EMNLP*.

Ruiqi Zhong, Tao Yu, and Dan Klein. 2020. Semantic evaluation for text-to-sql with distilled test suite. In *EMNLP 2020*.

Victor Zhong, Caiming Xiong, and Richard Socher. 2017. Seq2sql: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103.

A Results with Full Metrics

	Metric	T5-base	T5-large	T5-3B
Spider	Match	$58.12_{1.46}$	$66.63_{2.31}$	71.76
	Exec	$60.06_{0.54}$	$68.28_{1.61}$	74.37
	Test suite	$56.22_{0.73}$	$64.12_{1.28}$	68.38
GrailQA	Match	60.00	67.00	69.00
WebQSP	F1	72.50	73.96	75.97
MTOP	Match	83.89	84.70	84.88
	Template	88.85	88.32	88.86
WikiTQ	Acc	$36.94_{0.19}$	$43.30_{0.25}$	50.65
WikiSQL	Acc	84.50	86.27	87.34
CompWebQ	Acc	66.71	68.85	70.27
	F1	80.02	81.05	81.43
	Hits@1	83.64	85.49	86.20
HybridQA	Acc	54.07	56.95	59.41
	F1	61.85	64.62	66.76
MMQA	Acc	67.29	74.08	78.48
	F1	75.51	81.84	82.28
FeTaQA	BLEU	29.00	30.94	31.73
DART	BLEU	$50.62_{0.72}$	$51.72_{0.15}$	50.38
ToTTo	BLEU	48.29	48.95	48.95
MultiWoZ2.1	Joint Acc	$57.52_{0.96}$	$58.23_{0.39}$	58.46
KVRET	BLEU	20.04	18.84	17.75
SParC	Match	50.54	56.69	61.51
	Exec	53.95	60.60	67.33
	Match (interact)	31.28	37.44	41.94
	Exec (interact)	34.36	41.23	46.45
CoSQL	Match	42.30	48.26	54.08
	Exec	49.26	56.01	62.23
	Match (interact)	12.63	16.72	22.78
	Exec (interact)	16.04	20.14	26.16
SQA	Overall Acc	49.49	59.12	60.93
	Pos 0 Acc	57.65	63.65	59.18
	Pos 1 Acc	47.96	57.02	61.73
	Pos 2 Acc	43.27	56.73	62.57
	Pos 3 Acc	40.52	55.56	61.44
TabFact	Acc	$76.34_{0.36}$	$81.40_{0.16}$	83.97
FEVEROUS	Acc	75.05	79.81	82.40
SQL2Text	BLEC	$93.69_{0.29}$	$93.35_{0.29}$	92.71
Logic2Text	BLEC	92.15	92.88	91.69

Table 11: Development set performance with full metrics. We do three experiments with different random seeds on representative task of each family and report their averages and standard variances format as avr_{var}.

For the KVRET dataset, instead of the version used in our main tables, we re-run another more widely used pre-processed version (Madotto et al., 2018; Wu et al., 2019; Qin et al., 2020) on T5-base, T5-large and T5-3b. Results are shown in Table 13.

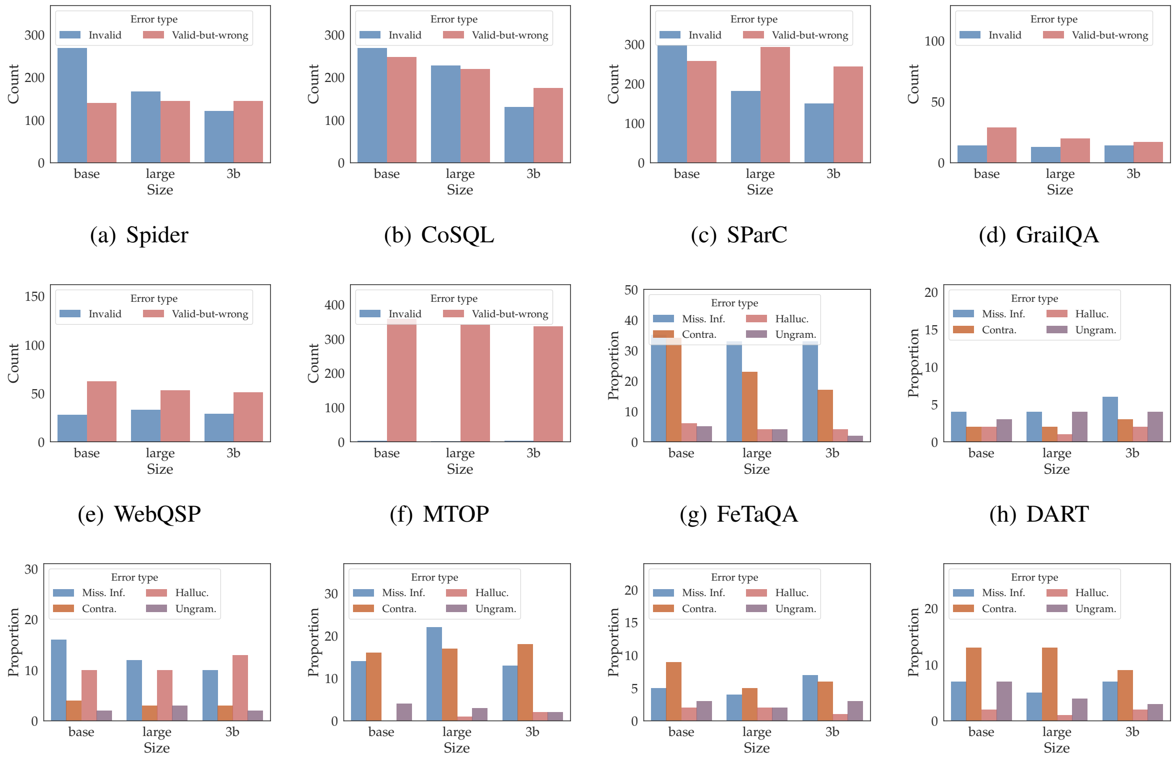
	Metric	T5-base	T5-large	T5-3B
GrailQA	Match	62.39	67.30	70.11
WebQSP	F1	78.83	79.45	80.70
MTOP	Match	85.49	86.17	86.78
	Template	87.52	89.53	90.20
WikiTQ	Acc	$35.76_{0.66}$	$43.22_{0.65}$	49.29
WikiSQL	Acc	82.63	84.80	85.96
CompWebQ	Acc	68.43	71.38	73.26
	F1	80.20	81.76	82.58
	Hits@1	83.70	85.40	86.08
FeTaQA	BLEU	29.91	32.45	33.44
	ROUGE-1-Fmeasure	61.77	64.01	65.21
	ROUGE-2-Fmeasure	39.44	42.26	43.09
	ROUGE-L-Fmeasure	51.93	54.29	55.31
	METEOR	48.53	50.80	51.23
	BertScore-F1	0.92	0.93	0.93
	BLEURT	-0.01	0.06	0.09
DART	BLEU	$46.22_{0.66}$	$46.89_{0.53}$	46.66
	TER	$61.80_{0.20}$	$60.97_{0.31}$	60.70
	METEOR	$55.09_{0.35}$	$55.76_{0.25}$	55.67
	BertScore-F1	$0.95_{0.00}$	$0.95_{0.00}$	0.95
	BLEURT	$0.2833_{0.0057}$	$0.30_{0.00}$	0.30
MultiWoZ2.1	Joint Acc	$54.64_{0.22}$	$54.45_{0.20}$	55.42
KVRET	BLEU	17.41	17.27	15.45
	F1 micro all	66.45	65.85	67.88
	F1 micro schedule	73.48	75.90	77.99
	F1 micro navigate	64.89	62.72	65.47
	F1 micro weather	63.78	62.80	64.01
	F1 macro all	65.90	64.19	67.13
	F1 macro schedule	71.27	75.95	79.14
	F1 macro navigate	64.42	59.92	62.66
	F1 macro weather	64.60	63.07	66.19
SQA	Overall Acc	52.91	61.28	62.37
	Pos 0 Acc	62.93	67.80	59.51
	Pos 1 Acc	44.43	55.08	60.25
	Pos 2 Acc	50.44	61.88	68.77
	Pos 3 Acc	53.71	58.08	65.07
TabFact	Acc	$76.13_{0.39}$	$80.85_{0.24}$	83.68
SQL2Text	BLEC	$93.52_{1.00}$	$93.68_{1.12}$	94.78
Logic2Text	BLEC	90.66	90.57	91.39

Table 12: Test set performance with full metrics (for tasks with a publicly available test set). We do three experiments with different random seeds on representative task of each family and report their averages and standard variances format as avr_{var}.

B Input and Output Length Analysis

Linearization of large structured knowledge input (e.g., large tables and KGs) can be arbitrarily long, which needs to be truncated to fit in GPUs with a limited size. The input and output are tokenized by T5Tokenizer in Huggingface's Transformers[4]. We visualize the length distribution in Figure 5, and details are presented in Table 14. Among the datasets with very long inputs, we choose WikiTableQuestion to study the impact of input length. We visualize the table length distribution and performances with different input truncation lengths in

[4]https://huggingface.co/t5-base/tree/main



(a) Spider (b) CoSQL (c) SParC (d) GrailQA

(e) WebQSP (f) MTOP (g) FeTaQA (h) DART

(i) ToTTo (j) KVRET (k) SQL2Text (l) Logic2Text

Figure 4: Error analysis. For semantic parsing, we show the number of invalid/valid-but-wrong predictions. For generation tasks, we show the proportion of missing-information/contradiction/hallucination/ungrammatical predictions among all predictions (one prediction may have multiple errors).

Metric	T5-base	T5-large	T5-3B
BLEU(dev)	22.80	23.07	22.71
BLEU(test)	21.21	22.36	20.40
F1 micro all(test)	67.49	68.03	70.07
F1 micro schedule(test)	79.39	79.47	78.54
F1 micro navigate(test)	62.87	63.59	65.34
F1 micro weather(test)	61.43	62.61	66.74
F1 macro all(test)	65.91	64.87	66.07
F1 macro schedule(test)	78.73	77.23	76.02
F1 macro navigate(test)	59.53	58.99	60.47
F1 macro weather(test)	64.05	62.58	65.78

Table 13: Baselines results are higher in pre-processed KVRET dataset. It doesn't change our conclusion on T5 with simple modification when necessary achieves SOTA on almost all tasks.

Figure 6. We observe that the accuracy increases as the input becomes longer, motivating future work to study how to effectively encode large structured input, e.g., leveraging sparse attention (Zaheer et al., 2020).

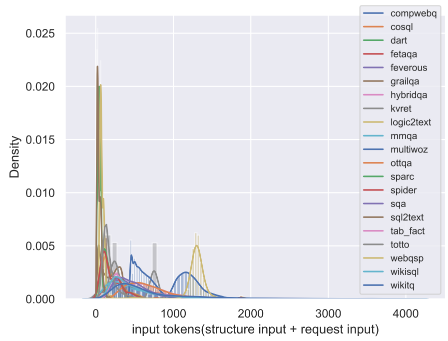


Figure 5: Input token distribution(<4096) in train set from different tasks. We exclude MTOP since it concentrates on a relatively small field which would make this figure unreadable. In general, 1024 is a good length for practice, and for most tasks, 2048 can hold all its inputs.

C Experimental Setup

C.1 Implementation Details

We use T5 (Raffel et al., 2020) as our backbone language model. Each experiment For T5-3B ex-

Distribution(%)	Structure Input Tokens			Text Input Tokens			Structure Input + Text Input Tokens			Sequence Output Tokens		
	[0, 512)	[512, 1024)	[1024, ∞)	[0, 512)	[512, 1024)	[1024, ∞)	[0, 512)	[512, 1024)	[1024, ∞)	[0, 128)	[128, 256)	[256, ∞)
Spider	97.01	1.81	1.17	100.00	0.00	0.00	95.47	3.35	1.17	98.81	1.18	0.0
GRAILQA	100.00	0.00	0.00	100.00	0.00	0.00	99.96	0.04	0.00	99.97	0.03	0.00
WebQsp	3.40	2.32	94.28	100.00	0.00	0.00	3.18	2.47	94.35	99.81	0.19	0.00
MTOP	0.00	100.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	99.97	0.03	0.00
WikiTableQuestions	48.32	27.48	24.18	100.00	0.00	0.00	46.03	29.43	24.52	99.98	0.01	0.01
WikiSQL	63.38	25.33	11.29	100.00	0.00	0.00	61.50	26.79	11.70	99.97	0.02	0.01
ComWebQ	1.18	14.52	84.30	100.00	0.00	0.00	1.09	11.28	87.63	99.59	0.39	0.01
HybridQA	35.53	50.63	13.8	100.00	0.00	0.00	31.77	53.35	14.86	100.00	0.00	0.0
MultiModalQA	63.02	25.67	11.30	100.00	0.00	0.00	60.54	27.26	12.18	99.99	0.01	0.00
FeTaQA	60.36	28.62	11.01	100.00	0.00	0.00	58.46	29.85	11.68	100.00	0.00	0.0
DART	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	99.99	0.01	0.0
ToTTo	95.80	2.87	1.31	100.00	0.00	0.00	95.80	2.87	1.31	99.99	0.01	0.0
MultiWoZ	100.00	0.00	0.00	98.77	1.21	0.01	54.76	45.09	0.13	0.00	100.00	0.0
KVRET	65.08	34.91	0.00	100.00	0.00	0.00	65.08	34.91	0.00	99.97	0.03	0.0
SParC	96.70	2.02	1.28	100.00	0.00	0.00	95.10	3.62	1.28	99.34	0.66	0.00
CoSQL	96.03	2.23	1.73	100.00	0.00	0.00	93.98	4.28	1.73	99.06	0.93	0.0
SQA	64.54	29.74	5.71	100.00	0.00	0.00	60.96	33.11	5.92	95.12	4.19	0.67
TabFact	63.22	28.19	8.58	100.00	0.00	0.00	60.68	30.20	9.10	100.00	0.00	0.0
FEVEROUS	61.37	22.24	16.39	100.00	0.00	0.00	57.53	25.07	17.40	100.00	0.00	0.00
SQL2Text	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.0	100.00	0.00	0.0
Logic2Text	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.0	100.00	0.00	0.0

Table 14: Input and output length for each task's train set.

Distribution(%)	Structure Input Tokens			Text Input Tokens			Structure Input + Text Input Tokens			Sequence Output Tokens		
	[0, 512)	[512, 1024)	[1024, ∞)	[0, 512)	[512, 1024)	[1024, ∞)	[0, 512)	[512, 1024)	[1024, ∞)	[0, 128)	[128, 256)	[256, ∞)
Spider	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	99.23	0.77	0.00
GRAILQA	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00
WebQsp	3.56	1.29	95.15	100.00	0.00	0.00	3.56	1.29	95.15	99.68	0.32	0.00
Russ	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00
MTOP	0.00	100.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	100.00	0.00	0.00
WikiTableQuestions	49.56	28.65	21.79	100.00	0.00	0.00	48.60	29.11	22.29	99.93	0.07	0.00
WikiSQL	63.90	25.88	10.22	100.00	0.00	0.00	62.06	26.99	10.95	100.00	0.00	0.00
ComWebQ	0.28	15.79	83.93	100.00	0.00	0.00	0.28	12.66	87.06	99.00	1.00	0.00
HybridQA	38.37	52.63	9.00	100.00	0.00	0.00	34.16	56.00	9.84	100.00	0.00	0.00
MultiModalQA	66.22	25.72	8.06	100.00	0.00	0.00	64.02	27.38	8.59	100.00	0.00	0.00
FeTaQA	67.03	27.47	5.49	100.00	0.00	0.00	64.84	29.57	5.59	100.00	0.00	0.00
DART	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00
ToTTo	95.82	2.92	1.26	100.00	0.00	0.00	95.82	2.92	1.26	100.00	0.00	0.00
MultiWoZ	100.00	0.00	0.00	99.16	0.84	0.00	25.07	74.68	0.24	0.00	100.00	0.00
KVRET	65.76	34.24	0.00	100.00	0.00	0.00	65.76	34.24	0.00	99.79	0.21	0.00
SParC	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	99.26	0.74	0.00
CoSQL	100.00	0.00	0.00	100.00	0.00	0.00	99.62	0.38	0.00	99.23	0.77	0.00
SQA	60.09	33.38	6.53	100.00	0.00	0.00	56.91	36.42	6.67	94.17	5.39	0.44
TabFact	62.17	29.31	8.52	100.00	0.00	0.00	59.95	30.91	9.14	100.00	0.00	0.00
FEVEROUS	61.56	23.71	14.73	100.00	0.00	0.00	57.57	26.58	15.85	100.00	0.00	0.00
SQL2Text	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00
Logic2Text	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00

Table 15: Input and output length for each task's development set.

periments, we use Deepspeed [5] to save memory. We use batch size 32 as default, except WikiTQ, WikiSQL, and TabFact, for which for use batch size 128 because we found it to work significantly better. We use the Adafactor optimizer for T5-base and T5-large, and AdamW for T5-3b. We evaluate on the development set for each 500 steps and use the average development set metric for best checkpoint selection. For all tasks, we set learning rate to 5e-5 and used linear learning rate decay. All experiments are done on NVIDIA Tesla V100 and NVIDIA Tesla A100.

C.2 Metric Details

For most semantic parsing tasks, we report the exact match accuracy of logical forms, and for task has test suite (Zhong et al., 2020), we add test suite metric to represent model's performance; an exception is WebQSP, for which we follow previous work to execute the parses and report the F1 score. For QA, we report the exact match accuracy of answer sets. For data-to-text generation, we report sacre-BLEU (Post, 2018).[6] We use each task's representative metric used by previous works. For fact verification, we report the accuracy. For high-fidelity NLG, we report BLEC (Shu et al., 2021),

[5] https://github.com/microsoft/DeepSpeed

[6] Signature: BLEU + case.lc + numrefs.1 + smooth.exp + tok.13a + version.1.4.0

	Structure Input Tokens			Text Input Tokens			Structure Input + Text Input Tokens			Sequence Output Tokens		
Distribution(%)	[0, 512)	[512, 1024)	[1024, ∞)	[0, 512)	[512, 1024)	[1024, ∞)	[0, 512)	[512, 1024)	[1024, ∞)	[0, 128)	[128, 256)	[256, ∞)
Spider	-	-	-	-	-	-	-	-	-	-	-	-
GRAILQA	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	99.98	0.02	0.00
WebQsp	3.48	1.95	94.57	100.00	0.00	0.00	3.36	2.07	94.57	100.00	0.00	0.00
Russ	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00
MTOP	0.00	100.00	0.00	100.00	0.00	0.00	0.00	100.00	0.00	100.00	0.00	0.00
WikiTableQuestions	48.00	31.15	20.86	100.00	0.00	0.00	47.08	31.70	21.22	99.98	0.02	0.00
WikiSQL	61.49	26.00	12.51	100.00	0.00	0.00	59.57	27.43	13.00	99.96	0.03	0.01
ComWebQ	0.85	16.02	83.13	100.00	0.00	0.00	0.85	13.07	86.08	99.43	0.57	0.00
HybridQA	-	-	-	-	-	-	-	-	-	-	-	-
FeTaQA	65.40	28.01	6.59	100.00	0.00	0.00	63.26	29.51	7.24	100.00	0.00	0.00
DART	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00
ToTTo	-	-	-	-	-	-	-	-	-	-	-	-
MultiWoZ	100.00	0.00	0.00	98.71	1.29	0.00	24.82	74.93	0.24	0.00	100.00	0.00
KVRET	66.14	33.86	0.00	100.00	0.00	0.00	66.14	33.86	0.00	100.00	0.00	0.00
SParC	-	-	-	-	-	-	-	-	-	-	-	-
CoSQL	-	-	-	-	-	-	-	-	-	-	-	-
SQA	62.54	30.92	6.54	100.00	0.00	0.00	61.37	32.05	6.58	93.69	5.68	0.63
TabFact	64.59	28.01	7.40	100.00	0.00	0.00	62.55	29.35	8.10	100.00	0.00	0.00
FEVEROUS	-	-	-	-	-	-	-	-	-	-	-	-
SQL2Text	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00
Logic2Text	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00	100.00	0.00	0.00

Table 16: Input and output length for each task's test set.

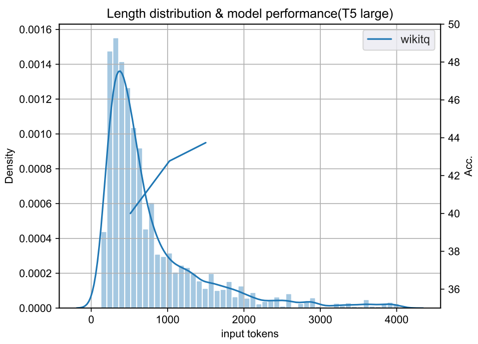


Figure 6: Length effect on WikiTableQuestion.

which is the exact match between keywords in the formal language and the natural language. Unless specified, we use T5-large and report the development set performance.

C.3 T0 Zero-shot Experiments Details

For each task in UNIFIEDSKG we search Sanh et al. (2021) for the most similar instructions(if there is no one for use, we create one follow their writing style), make our input in that format and directly test on T0 3B. The specific instructions are shown below.

```
Spider
Given database schema "[linearized
database schema]". Can you tell me the
SQL for "[request]"?

WikiTQ
I know that the answer to "[request]" is
 in "[linearized table]". Can you tell
me what it is?

DART
Put the triples together to form a
sentence: [relation triples]

MultiWoZ
Known ontology "[ontology]", the
dialogue state when "[dialogue history
and current request]" is given

TabFact
Suppose "[linearized table]" Can we
infer that "[statement]"?

SQL2Text
Paraphrase "[SQL]" to natural language:
```

C.4 GPT3 and Codex Details

C.4.1 Hyperparameter Setting

Temperature For GPT3 and Codex, we set the decoding temperature to 0 (i.e., greedy decoding without sampling) for Spider, WikiTQ, MultiWoZ and TabFact. We observe a drop of 10% in the exact match metric when set the temperature to 1 by default in OpenAI. For Codex, we tune the temperature from 0 to 1 in a step of 0.1 for DART, SQL2Text, and no significant difference is observed. For GPT3, we do not tune on that to stay within our budget.

Max output length We set max output length to 256 for Spider, WikiTQ, MultiWoZ and SQL2Text, while 4 for TabFact to contain more length in the input side(the concept of max length in GPT3 and Codex is the sum of input tokens length and output tokens length). We set "\n" as the stop token.

C.4.2 Prompts

We use simple prompt words for each task to concat the request, linearized structured knowledge, and context together. For example, for each example in WikiTQ, we format it as "*examples*\n\n[linearized table] || Write a answer for [request] \nThe answer is:", and make GPT3 and Codex make the completion as prediction. We do experiments on Spider with different format of forming structured knowledge(e.g., linearization, description), but get the similar result. Better usage of GPT3 and Codex under UNIFIEDSKG Framework is an interesting direction.

C.5 Human Evaluation

Participants of our human evaluation are eight of the authors of this paper. They are familiar with the tasks being evaluated. The human evaluation guideline is shown below.

```
## General Guideline
1. Each line is a dev set sample, with
some inputs (detailed below), a human
reference (seq_out) shown in blue, and
three model outputs named model1, model2,
 and model3.
2. Each model output receives a 0-1
score (0 stands for incorrect, and 1
stands for correct). By "correct" we
mean "responding to the user request
properly and correctly, without grammar
or wording mistakes".
3. When an output is incorrect, you
specify the type(s) of error, e.g., 1)
missing information, 2) contradiction,
3) hallucination, and 4) ungrammatical.

## Task-Specific Details
### DART
1. Task: triples-to-text generation.
2. struct_in: a set of relation-triples
joined by ``|``. Each relation-triple is
 of form ``entityA : relation : entityB
``.

### FeTaQA
1. Task: free-form QA
2. question: a question about the table.
3. table: a table represented as a
dictionary: {"header": [header item,
...], "rows": [[cell value, ...], ...]}.
4. meta: table_page_title |
table_section_title

### KVRET
1. Task: dialogue system
2. dialogue: a dialogue represented as a
 dictionary: {"driver": [request1, ...],
 "assistant": [response1, ...]}, the
last response of the assistant is the
human reference.
3. kb: a knowledge base represented as a
 dictionary: {"header": [header item,
...], "rows": [[cell value, ...], ...]}.

### Logic2Text
1. Task: logic expression to text
translation
2. table: a table represented as a
dictionary: {"caption": table caption, "
header": [header item, ...], "rows": [[
cell value, ...], ...]}.
3. logic_str: logic expression of a
statement.

### SQL2Text
1. Task: SQL to text translation
2. query: SQL.

### ToTTo
1. Task: highlighted-table-to-text
generation.
2. table_page_title and section: table
meta information.
3. Visualization of highlighted tables
is provided in ``totto_vis/``.
```

C.6 Hyper-parameters

Shown in Table 17. For semantic parsing tasks, the decoding was done under the greedy search, where we set the beam size to 1 specially. For tasks with long linearized sequence, we used 1024 as input length to hold the maximum of input, reasons are explained in B.

D Training Details

Here we show comparisons of finetuning and prefix-tuning on aspect of training. For prefix-tuning, we use the random initialization as did in paper (Li and Liang, 2021). In general, Prefix-tuning needs more steps than finetuning but has the ability to reach comparable results as long as continue training it.

E Task Unification

E.1 Term Definition

Highlighted tables Highlighted table contain table, table metadata (such as the title), and set of highlighted cells which supports the sentence. A phrase is supported by the table if it is either directly stated in the cell contents or meta-data, or can be logically inferred by them (Parikh et al., 2020).

Relation-triples Relation triples are a set of subject-predicate-object triples to capture rich relationships in the data. Many data-to-text tasks such as DART(Nan et al., 2021b) take these relation triples as inputs and generate natural language from them.

Task type	Task	Input length	Batch size	Beam size
Semantic Parsing	Spider (Yu et al., 2018)	512	32	1
	GrailQA (Gu et al., 2021)	512	32	4
	WebQSP (Yih et al., 2016)	1024	32	4
	MTOP (Li et al., 2021)	1024	32	4
Question Answering	WikiSQL (Zhong et al., 2017)	1024	128	4
	WikiTQ (Pasupat and Liang, 2015)	1024	128	4
	CompWebQ (Talmor and Berant, 2018)	1024	32	4
	HybridQA (Chen et al., 2020d)	1024	32	4
	MultiModalQA (Talmor et al., 2021)	1024	32	4
	FeTaQA (Nan et al., 2021a)	512	32	4
Data-to-Text	DART (Nan et al., 2021b)	512	32	4
	ToTTo (Parikh et al., 2020)	512	32	4
Conversational	MultiWoZ2.1 (Eric et al., 2019)	1024	32	4
	KVRET (Eric et al., 2017)	1024	32	4
	SParC (Yu et al., 2019b)	512	32	1
	CoSQL (Yu et al., 2019a)	512	32	1
	SQA (Iyyer et al., 2017)	1024	128	4
Fact Verification	TabFact (Chen et al., 2020c)	1024	128	4
	FEVEROUS (Aly et al., 2021)	1024	32	4
High-fidelity NLG	SQL2Text (Shu et al., 2021)	512	32	4
	Logic2Text (Chen et al., 2020e)	512	32	4

Table 17: Hyperparameters for each SKG task.

Knowledge Graph A Knowledge Graph is a multi-relational graph composed of entities (nodes) and relations (different types of edges). Each edge is represented as a triple of the form (head entity, relation, tail entity), also called a fact, indicating that two entities are connected by a specific relation.(Wang et al., 2017)

Dialogue State and Ontology A dialogue state s_t at any turn t in a dialogue comprises the summary of the dialogue history until turn t, such that s_t contains all sufficient information for the system to choose the next action. (Williams et al., 2016) Specifically, it captures the user goals in the conversation in the form of (slot, value) pairs. The set of possible slots is predefined in the Ontology O, typically domain-dependent, while the values assumed by each slots are provided by the user as a dialogue goal.

E.2 Linearization

- **Tables.** Following Liu et al. (2021), we linearize the table into a sequence. By inserting several special tokens to indicate the table boundaries, a linearized table can be represented as "col: c_1, ..., c_N row 1 : r_1 row 2 : r_2... r_M ", N and M are the number of columns and rows.

- **Highlighted tables.** Following Parikh et al. (2020), we represent each highlighted cell by concatenating its value, column headers, and row headers. The table is represented as the concatenation of the page title, section title, and representations of all highlighted cells.

- **Relation-triples and knowledge graphs.** Following Nan et al. (2021b), each relation-triple is linearized as "*sub* : *rela* : *obj*", and different triples are joined by " | ". The subgraph retrieved from the knowledge graph is treated as a list of relation-triples and we use the same formulation.

- **Ontology.** Following Hosseini-Asl et al. (2020) and Lin et al. (2021), for each slot in ontology, each slot along with its all possible values is formatted as "*slot* : *value*$_1$, ... *value*$_{slot_n}$ ", different slot-values are joined by " | "

E.3 Output Format

When the output is *natural language* or *formal language* we do not modify it because it is already in sequence format; a *set of answers*, we use a comma followed by a space to join the answers; a *Boolean value*, we map True to "entailed" and False to "refuted"; a *dialogue state*, we follow Hosseini-Asl et al. (2020) to place its slot-value pairs sequentially.

Task	Finetune	Prefix-tuning
Spider	16500	100000
GrailQA	17000	78000
WebQSP	1500	8000
MTOP	30000	60000
WikiSQL	8500	80000
WikiTQ	1500	16000
CompWebQ	3500	27000
HybridQA	7000	30000
MultiModalQA	6000	40000
FeTaQA	11000	20000
DART	7000	250000
ToTTo	12000	>250000
MultiWoZ2.1	6000	40000
KVRET	4000	40000
SParC	2000	6400
CoSQL	38000	100000
SQA	27000	>250000
TabFact	8000	210000
FEVEROUS	1200	40000
SQL2Text	3000	10000
Logic2Text	3500	10000

Table 18: The comparison of approximate training steps finetuning and prefix-tuning used to reach the decent performance on T5 base. >250000 means we stop the training due to time limitation. Prefix-tuning needs more steps to converge and converges to comparable performances.

F Input and Output Examples for Each Task

F.1 Spider

Structured Input:

```
| concert_singer | stadium : stadium_id ,
 location , name , capacity , highest ,
lowest , average | singer : singer_id ,
name , country , song_name ,
song_release_year , age , is_male |
concert : concert_id , concert_name ,
theme , stadium_id , year |
singer_in_concert : concert_id ,
singer_id
```

Request Input:

```
How many singers do we have?
```

Sequence Output:

```
select count(*) from singer
```

F.2 GRAILQA

Structured Input:

```
soviet red army: m.06dr9 | organization.
organization.founders government.
governmental_body.jurisdiction
organization.organization_founder.
organizations_founded military.
military_service.military_person
government.political_party_tenure
government.national_anthem_of_a_country
visual_art.art_subject.
artwork_on_the_subject government.
government_agency government.
governmental_jurisdiction.government
people.deceased_person.place_of_burial
people.deceased_person.date_of_death
people.person.children people.person.
parents people.person.height_meters
government.government_position_held.
office_holder government.government
people.person people.person.sibling_s
people.person.quotations people.person.
gender
```

Request Input:

```
the person who founded the soviet red
army also founded what government agency
?
```

Sequence Output:

```
(AND government.government_agency (JOIN
organization.organization.founders (JOIN
 (R organization.organization.founders)
m.06dr9)))
```

F.3 CompWebQ

Structured Input:

```
Liam Hemsworth celebrities.celebrity.
sexual_relationships..celebrities.
romantic_relationship.celebrity Liam
Hemsworth | Liam Hemsworth type.object.
type tv.tv_actor | Liam Hemsworth film.
actor.film..film.performance.character
Billy The Kid | Liam Hemsworth film.
actor.film..film.performance.film The
Hunger Games |
(omitted to save space)
| Liam Hemsworth type.object.type
celebrities.celebrity | Cut Bank film.
film.cinematography Ben Richardson | Cut
 Bank film.film.language English
Language | Cut Bank film.film.edited_by
Carol Littleton | Liam Hemsworth film.
actor.film..film.performance.film The
Hunger Games: Mockingjay, Part 1 | Liam
Hemsworth film.actor.film..film.
performance.film Timeless
```

Request Input:

```
What movie was produced by Dan Cohen and
 features Liam Hemsworth as an actor?
```

Sequence Output:

```
Cut Bank
```

F.4 WebQsp

Structured Input:

```
Spain: m.06mkj | m.06mkj location.
location.contains m.0g3qgy | m.06mkj
location.location.contains m.02qf5mh | m
.0j5_3sv government.
```

government_position_held.
office_position_or_title m.0j5_3sz | m
.06mkj location.location.contains m.02
zb43k |
(omitted to save space)
| m.06mkj government.
governmental_jurisdiction.
governing_officials m.010wswjc | m.06mkj
location.location.contains m.09k5hy | m
.010wswjc government.
government_position_held.
office_position_or_title m.0j5_3sz | m
.06mkj location.location.contains m.02
z98t5 | m.06mkj location.location.
contains m.03qcr60

Request Input:

what is the king of spain's name?

Sequence Output:

(JOIN (R government.
government_position_held.office_holder)
(AND (JOIN government.
government_position_held.time_macro
2015^^http://www.w3.org/2001/XMLSchema#
date) (AND (JOIN government.
government_position_held.
office_position_or_title m.0j5_3sz) (
JOIN (R government.
governmental_jurisdiction.
governing_officials) m.06mkj))))

F.5 MTOP

Structured Input:

IN:GET: MESSAGE, WEATHER, ALARM,
INFO_RECIPES, STORIES_NEWS, REMINDER,
RECIPES, EVENT, CALL_TIME, LIFE_EVENT,
INFO_CONTACT, CONTACT, TIMER,
REMINDER_DATE_TIME, AGE, SUNRISE,
EMPLOYER, EDUCATION_TIME, JOB,
AVAILABILITY,
(omitted to save space)
IN:PREVIOUS: TRACK_MUSIC | IN:HOLD:
CALL | IN:SKIP: TRACK_MUSIC | IN:LIKE:
MUSIC | IN:RESTART: TIMER | IN:RESUME:
TIMER, CALL, MUSIC | IN:MERGE: CALL |
IN:REPLAY: MUSIC | IN:LOOP: MUSIC | IN:
STOP: MUSIC, SHUFFLE_MUSIC | IN:UNLOOP:
MUSIC | IN:CANCEL: MESSAGE, CALL | IN:
REWIND: MUSIC | IN:REPEAT: ALL_MUSIC,
ALL_OFF_MUSIC | IN:FAST: FORWARD_MUSIC
| IN:DISLIKE: MUSIC | IN:DISPREFER: |
IN:HELP: REMINDER | IN:FOLLOW: MUSIC

Request Input:

call Nicholas and Natasha

Sequence Output:

[IN:CREATE_CALL [SL:CONTACT Nicholas] [
SL:CONTACT Natasha]]

F.6 WikiSQL

Structured Input:

col : player | no. | nationality |
position | years in toronto | school/
club team row 1 : antonio lang | 21 |
united states | guard-forward |
1999-2000 | duke row 2 : voshon lenard |
2 | united states | guard | 2002-03 |
minnesota row 3 : martin lewis | 32, 44
| united states | guard-forward |
1996-97 | butler cc (ks) row 4 : brad
lohaus | 33 | united states | forward-
center | 1996 | iowa row 5 : art long |
42 | united states | forward-center |
2002-03 | cincinnati row 6 : john long |
25 | united states | guard | 1996-97 |
detroit row 7 : kyle lowry | 3 | united
states | guard | 2012-present |
villanova

Request Input:

what position does the player who played
for butler cc (ks) play?

Sequence Output:

guard-forward

F.7 WikiTableQuestions

Structured Input:

col : team | county | wins | years won
row 1 : greystones | wicklow | 1 | 2011
row 2 : ballymore eustace | kildare | 1
| 2010 row 3 : maynooth | kildare | 1 |
2009 row 4 : ballyroan abbey | laois | 1
| 2008 row 5 : fingal ravens | dublin |
1 | 2007 row 6 : confey | kildare | 1 |
2006 row 7 : crettyard | laois | 1 |
2005 row 8 : wolfe tones | meath | 1 |
2004 row 9 : dundalk gaels | louth | 1 |
2003

Request Input:

which team won previous to crettyard?

Sequence Output:

wolfe tones

F.8 HybridQA

Structured Input:

col : position | athlete | nationality |
time row 1 : 1 | patrick makau musyoki
| kenya | 2:03.38 row 2 : 2 | stephen
kwelio chemlany | kenya | 2:07.55 row 3 :
3 | edwin kimaiyo | kenya | 2:09.50 row
4 : 4 | felix limo | kenya | 2:10.38
row 5 : 5 | scott overall | united
kingdom | 2:10.55 row 6 : 6 | ricardo
serrano | spain | 2:13.32 row 7 : 7 |
pedro nimo | spain | 2:13.34 row 8 : 8 |
simon munyutu | france | 2:14.20 row 9 :
9 | driss el himer | france | 2:14.46
row 10 : 10 | hendrick ramaala | south
africa | 2:16.00passages: ricardo
serrano (athlete): at the 2011 iaaf
world cross country championships he was

89th overall . his marathon debut followed later that year and he was sixth at the 2011 berlin marathon with a time of 2:13.32 hours . | spain: with an area of 505,990 km2 (195,360 sq mi) , spain is the largest country in southern europe , the second largest country in western europe and the european union , and the fourth largest country in the european continent . by population (about 47 million) , spain is the sixth largest in europe and the fifth in the european union . |

Request Input:

what place was achieved by the person who finished the berlin marathon in 2:13.32 in 2011 the first time he competed in a marathon ?

Sequence Output:

sixth

F.9 MultiModalQA

Structured Input:

ben piazza | filmography col : year | title | role | notes row 1 : 1957 | a dangerous age | david | row 2 : 1959 | the hanging tree | rune | row 3 : 1962 | no exit | camarero | row 4 : 1970 | tell me that you love me, junie moon | jesse | row 5 : 1972 | the outside man | desk clerk | row 6 : 1973 | the candy snatchers | avery | row 7 : 1976 | the bad news bears | bob whitewood | row 8 : 1977 | i never promised you a rose garden | jay blake | row 9 : 1979 | nightwing | roger piggott | row 10 : 1979 | the concorde ... airport ’79 | associate | tv version, uncredited row 11 : 1980 | the blues brothers | father | row 12 : 1982 | waltz across texas | bill wrather | row 13 : 1985 | mask | mr. simms | row 14 : 1988 | clean and sober | kramer | row 15 : 1990 | rocky v | doctor | uncredited row 16 : 1991 | guilty by suspicion | darryl zanuck |

Request Input:

for which film did ben piazza play the role of mr. simms?

Sequence Output:

mask

F.10 FeTaQA

Structured Input:

andy karl | awards and nominations col : year | award | category | work | result row 1 : 2013 | drama desk award | outstanding featured actor in a musical | the mystery of edwin drood | nominated row 2 : 2013 | broadway.com audience awards | favorite onstage pair (with jessie mueller) | the mystery of edwin drood | nominated row 3 : 2014 | (omitted to save space) | groundhog day | nominated row 15 : 2017 | tony award | best actor in a musical | groundhog day | nominated row 16 : 2017 | drama desk award | outstanding actor in a musical | groundhog day | won row 17 : 2017 | drama league award | distinguished performance | groundhog day | nominated row 18 : 2017 | outer critics circle award | outstanding actor in a musical | groundhog day | won

Request Input:

when did andy karl win the olivier award and for which of his work?

Sequence Output:

for his performance in groundhog day, andy karl received the 2017 olivier award for best actor in a musical.

F.11 DART

Structured Input:

Mars Hill College : joined : 1973 | Mars Hill College : location : Mars Hill, North Carolina

Request Input:

None

Sequence Output:

A school from Mars Hill, North Carolina, joined in 1973.

F.12 ToTTo

Structured Input:

<page_title> List of Governors of South Carolina </page_title> <section_title> Governors under the Constitution of 1868 </section_title> <table> <cell> 76 <col_header> # </col_header> <col_header> 74 </col_header> <col_header> 75 </col_header> </cell> <cell> Daniel Henry Chamberlain <col_header> Governor </col_header> <row_header> 76 </row_header> </cell> <cell> December 1, 1874 <col_header> Took Office </col_header> <row_header> 76 </row_header> </cell> </table>

Request Input:

None

Sequence Output:

Daniel Henry Chamberlain was the 76th Governor of South Carolina from 1874.

F.13 MultiWoZ2.1

Structured Input:

hotel-pricerange: cheap, dontcare, expensive, moderate; hotel-type: guesthouse, hotel; hotel-parking: dontcare, free, no, yes; hotel-book day: friday, monday, saturday, sunday, thursday, tuesday, wednesday; hotel-book people: 1, 2, 3, 4, 5, 6, 7, 8; hotel-book stay: 1, 2, 3, 4, 5, 6, 7, 8; hotel-area: centre, dontcare, east, north, south, west; hotel-stars: 0, 1, 2, 3, 4, 5, dontcare; hotel-internet: dontcare, no, yes; hotel-name: none; train-destination: none; train-day: dontcare, friday, monday, saturday, sunday, thursday, tuesday, wednesday; train-departure: none; train-arriveby: none; train-book people: 0, 1, 10, 15, 2, 3, 4, 5, 6, 7, 8, 9; taxi-destination: none; taxi-departure: none; taxi-leaveat: none ; train-leaveat: none; attraction-area: cambridge, centre, dontcare, east, north, south, west; restaurant-pricerange: cheap, dontcare, expensive, moderate; restaurant-area: centre, east, north, south, west; restaurant-food: none; attraction-name: none; restaurant-name: none; attraction-type: architecture, boat, church, cinema, college, concerthall, entertainment, hotspot, multiple sports, museum, nightclub, park, special, swimmingpool, theatre; restaurant-book day: friday, monday, saturday, sunday, thursday, tuesday, wednesday; restaurant-book people: 1, 2, 3, 4, 5, 6, 7, 8; restaurant-book time: none; taxi-arriveby: none;

Request Input:

i am looking to book a train that is leaving from cambridge to bishops stortford on friday .

Context:

booking was successful . reference number is : bmukptg6 . can i help you with anything else today ? | friday and can you book it for me and get a reference number ? | what day will you be staying ? | could you book the wartworth for 1 night , 1 person ? | if you would like something cheap , i recommend the allenbell . for something moderate -ly priced , i would recommend the warkworth house . | that does not matter as long as it has free wifi and parking . | i can help you with that . what is your price range ? | i need to book a hotel in the east that has 4 stars . | none

Sequence Output:

hotel pricerange none, hotel type none, hotel parking yes, hotel book day friday, hotel book people 1, hotel book stay 1, hotel area east, hotel stars 4, hotel internet yes, hotel name wartworth, train destination bishops stortford, train day friday, train departure cambridge, train arriveby none, train book people none, taxi destination none, taxi departure none, taxi leaveat none, train leaveat none, attraction area none, restaurant pricerange none, restaurant area none, restaurant food none, attraction name none, restaurant name none, attraction type none, restaurant book day none, restaurant book people none, restaurant book time none, taxi arriveby none

F.14 KVRET

Structured Input:

col : poi | poi_type | address | distance | traffic_info row 1 : sigona farmers market | grocery store | 638 amherst st | 3 miles | car collision nearby row 2 : cafe venetia | coffee or tea place | 269 alger dr | 1 miles | car collision nearby row 3 : 5672 barringer street | certain address | 5672 barringer street | 5 miles | no traffic row 4 : valero | gas station | 200 alester ave | 2 miles | road block nearby row 5 : stanford childrens health | hospital | 899 ames ct | 5 miles | moderate traffic row 6 : palo alto garage r | parking garage | 481 amaranta ave | 1 miles | moderate traffic row 7 : teavana | coffee or tea place | 145 amherst st | 1 miles | road block nearby row 8 : willows market | grocery store | 409 bollard st | 5 miles | no traffic

Request Input:

ok, please give me directions via a route that avoids all heavy_traffic.

Context:

valero is located at 200_alester_ave. | address to the gas_station

Sequence Output:

since there is a road_block_nearby, i found another route for you and i sent it on your screen

F.15 SParC

Structured Input:

| flight_2 | airlines : uid , airline , abbreviation , country | airports : city , airportcode , airportname , country , countryabbrev | flights : airline , flightno , sourceairport , destairport

Request Input:

What is the country corresponding it?

Context:

```
Of these, which is Jetblue Airways? |
What are all the airlines?
```

Sequence Output:

```
select country from airlines where
airline = "JetBlue Airways"
```

F.16 CoSQL

Structured Input:

```
| car_1 | continents : contid ,
continent | countries : countryid ,
countryname , continent | car_makers :
id , maker , fullname , country |
model_list : modelid , maker , model |
car_names : makeid , model , make |
cars_data : id , mpg , cylinders ,
edispl , horsepower , weight ,
accelerate , year
```

Request Input:

```
Which models are the most popular?
```

Context:

```
How many car models are produced by each
 maker? List the count and the maker
full name.
```

Sequence Output:

```
select t1.model from model_list as t1
join car_makers as t2 on t1.maker = t2.
id group by t2.id order by count ( * )
desc limit 1
```

F.17 SQA

Structured Input:

```
col : date | number | type | captain |
casualties | sunk by row 1 : 4 may 1943
| u-209 | viic | heinrich brodda | 46 |
catalina flying-boat of no. 5 squadron
row 2 : 5 may 1943 | u-638 | viic |
oskar staudinger | 44 | hms sunflower
row 3 : 5 may 1943 | u-531 | ixc/40 |
herbert neckel | 54 | hms vidette row 4 :
 6 may 1943 | u-192 | ixc/40 | werner
happe | 55 | hms loosestrife row 5 : 6
may 1943 | u-125 | ixc | ulrich folkers
| 54 | "hms oribi | hms snowflake" row 6
 : 6 may 1943 | u-630 | viic | werner
winkler | 47 | hms vidette row 7 : 6 may
 1943 | u-438 | viic | heinrich hensohn
| 48 | hms pelican
```

Request Input:

```
which captain was not oskar staudinger?
```

Context:

```
who were the captains of those boats? |
what boats were lost on may 5?
```

Sequence Output:

```
herbert neckel
```

F.18 TabFact

Structured Input:

```
col : round | clubs remaining | clubs
involved | winners from previous round |
 new entries this round | leagues
entering at this round row 1 : first
round | 156 | 86 | none | 86 | tff third
 league & turkish regional amateur
league row 2 : second round | 113 | 108
| 43 | 65 | s\u00fcper lig & tff first
league & tff second league row 3 : third
 round | 59 | 54 | 54 | none | none row
4 : fourth round | 32 | 32 | 27 | 5 | s\
u00fcper lig row 5 : fifth round | 16 |
16 | 16 | none | none row 6 : group
stage | 8 | 8 | 8 | none | none row 7 :
semi - finals | 4 | 4 | 4 | none | none
row 8 : final | 2 | 2 | 2 | none | none
```

Request Input:

```
during the third round of the turkish
cup , there be no new entry during that
stage
```

Sequence Output:

```
entailed
```

F.19 FEVEROUS

Structured Input:

```
col : no. | title | narrator | aired
between | original air date | us viewers
 row 1 : 1 | "magic is coming" |
giancarlo esposito | "a land without
magic" "broken" | september 30, 2012
(2012-09-30) | 6.04 row 2 : 2 | "the
price of magic" | alan dale | "selfless,
 brave and true" "lacey" | april 14,
2013 (2013-04-14) | 5.17 row 3 : 3 | "
journey to neverland" | alfred molina |
"and straight on 'til morning" "the
heart of the |
(omitted to save space)
| "dark swan rises: a once upon a time
fan | howard parker | "operation
mongoose, part 2" "the | september 27,
2015 (2015-09-27) | 3.20 row 8 : 8 | "
evil reigns once more" | howard parker |
 "an untold story" "the savior" |
september 25, 2016 (2016-09-25) | 2.86
row 9 : 9 | "the final battle begins" |
howard parker | "the song in your heart"
 "the final battle" | may 14, 2017
(2017-05-14) | 2.82
```

Request Input:

```
list of once upon a time (debuted
october 23, 2011) episodes consists of
10 specials, one of the narrators alan
dale.
```

Sequence Output:

```
refuted
```

F.20 SQL2Text

Structured Input:

```
None
```

Request Input:

```
SELECT Fname FROM FACULTY WHERE Rank = "
Professor" ORDER BY Fname
```

Sequence Output:

```
What are the first names for all faculty
 professors, ordered by first name?
```

F.21 Logic2Text

Structured Input:

```
Caption: list of tallest structures in
the world - 300 to 400 metres | planned
pinnacle height, expected year of
completion, structural type, country,
town
```

Request Input:

```
eq { count { filter_eq { all_rows ;
expected year of completion ; 2006 } } ;
 3 } = true
```

Sequence Output:

```
three of the tallest structures will be
completed in 2006 .
```

G Case study

G.1 Spider case

G.1.1 Case 1:

Structured Input:

```
| concert_singer | stadium : stadium_id ,
 location , name , capacity , highest ,
lowest , average | singer : singer_id ,
name , country , song_name ,
song_release_year , age , is_male |
concert : concert_id , concert_name ,
theme , stadium_id , year |
singer_in_concert : concert_id ,
singer_id
```

Request Input:

```
What are the names of all stadiums that
did not have a concert in 2014?
```

Sequence Output:

```
select name from stadium except select
t2.name from concert as t1 join stadium
as t2 on t1.stadium_id = t2.stadium_id
where t1.year = 2014
```

T5-base prediction (incorrect):

```
select name from stadium except select
stadium_name from concert where year =
2014
```

T5-large prediction (correct):

```
select name from stadium except select
t2.name from concert as t1 join stadium
as t2 on t1.stadium_id = t2.stadium_id
where t1.year = 2014
```

T5-3B prediction (correct):

```
select name from stadium except select
t2.name from concert as t1 join stadium
as t2 on t1.stadium_id = t2.stadium_id
where t1.year = 2014
```

G.1.2 Case 2:

Structured Input:

```
| concert_singer | stadium : stadium_id ,
 location , name , capacity , highest ,
lowest , average | singer : singer_id ,
name , country , song_name ,
song_release_year , age , is_male |
concert : concert_id , concert_name ,
theme , stadium_id , year |
singer_in_concert : concert_id ,
singer_id
```

Request Input:

```
What is the name and capacity for the
stadium with highest average attendance?
```

Sequence Output:

```
select name, capacity from stadium order
 by average desc limit 1
```

T5-base prediction (incorrect):

```
select name, capacity from stadium order
 by avg(amount) desc limit 1
```

T5-large prediction (correct):

```
select name, capacity from stadium order
 by average desc limit 1
```

T5-3B prediction (correct):

```
select name, capacity from stadium order
 by average desc limit 1
```

G.1.3 Case 3:

Structured Input:

```
| pets_1 | student : stuid , lname ,
fname , age , sex , major , advisor ,
city_code | has_pet : stuid , petid |
pets : petid , pettype ( cat , dog ) ,
pet_age , weight
```

Request Input:

```
Find the first name of students who have
 cat or dog pet.
```

Sequence Output:

```
select distinct t1.fname from student as
 t1 join has_pet as t2 on t1.stuid = t2.
stuid join pets as t3 on t3.petid = t2.
petid where t3.pettype = 'cat' or t3.
pettype = 'dog'
```

T5-base prediction (incorrect):

```
select t1.fname from student as t1 join
has_pet as t2 on t1.stuid = t2.stuid
where t2.pettype = "cat" or t2.pettype =
 "dog"
```

T5-large prediction (incorrect):

```
select t1.fname from student as t1 join
has_pet as t2 on t1.stuid = t2.stuid
where t2.pettype = "cat" or t2.pettype =
 "dog"
```

T5-3B prediction (correct):

```
select t1.fname from student as t1 join
has_pet as t2 on t1.stuid = t2.stuid
join pets as t3 on t2.petid = t3.petid
where t3.pettype = "cat" or t3.pettype =
 "dog"
```

G.2 FeTaQA case

Structured Input:

```
te aroha (new zealand electorate) | 1890
 election col : party | party |
candidate | votes | % | [non utf-8 token
] row 1 : - | independent | william
shepherd allen | 786 | 56.34 | - row 2 :
 - | liberal | william fraser | 609 |
43.65 | - row 3 : majority | majority |
majority | 175 | 12.54 | - row 4 :
turnout | turnout | turnout | 1,395 |
48.60 | - row 5 : registered electors |
registered electors | registered
electors | 2,870 | - | -
```

Request Input:

```
what were the voting results in the te
aroha election by the candidate?
```

Sequence Output:

```
votes for allen and fraser were 786 and
609 respectively.
```

T5-base prediction (incorrect):

```
the te aroha election was won by william
 shepherd allen with a majority of 175.
```

T5-large prediction (incorrect):

```
in the te aroha election, allen won with
 a majority of 175 votes.
```

T5-3B prediction (correct):

```
in the 1890 te aroha election, the
result was a victory for the independent
 candidate, william shepherd allen, who
received 786 votes against 609 for the
liberal candidate, william fraser.
```

G.3 KVRET case

G.3.1 Case 1:

Structured Input:

```
col : location | monday | tuesday |
wednesday | thursday | friday | saturday
 | sunday | today row 1 : danville |
clear skies, low of 30f, high of 50f |
cloudy, low of 20f, high of 40f |
raining, low of 80f, high of 100f | dew,
 low of 80f, high of 90f | clear skies,
low of 70f, high of 80f | frost, low of
40f, high of 60f | overcast, low of 20f,
 high of 30f | monday row 2 :... row 6 :
 alhambra | windy, low of 90f, high of
100f | hail, low of 90f, high of 100f |
clear skies, low of 80f, high of 90f |
rain, low of 70f, high of 80f | stormy,
low of 70f, high of 80f | rain, low of
60f, high of 70f | hail, low of 40f,
high of 60f | monday ...
```

Request Input:

```
alhambra please.
```

Context Input:

```
what city are you inquiring about? | is
it going to be raining this weekend
```

Sequence Output:

```
it will be raining on saturday and
hailing on sunday in alhambra.
```

T5-base prediction (correct):

```
it will be raining on saturday and
hailing on sunday in alhambra.
```

T5-large prediction (incorrect):

```
it will not rain in alhambra this
weekend.
```

T5-3B prediction (incorrect):

```
there will be no rain this weekend in
alhambra.
```

G.3.2 Case 2:

Structured Input:

```
col : event | time | date | room |
agenda | party row 1 : tennis activity |
 5pm | wednesday | - | - | mother row 2 :
 taking medicine | 9am | the 3rd | - | -
 | sister row 3 : swimming activity | 11
am | friday | - | - | father row 4 :
conference | 6pm | the 8th | 100
conference room | go over budget |
infrastructure team row 5 : meeting | 4
pm | the 10th | conference room 102 | go
 over quarterly report | infrastructure
team row 6 : dentist appointment | 10am
| the 13th | - | - | jeff row 7 :
dentist appointment | 11am | the 5th | -
 | - | jeff row 8 : lab appointment | 2
pm | the 4th | - | - | sister
```

Request Input:

```
when is the date and time of my
dentist_appointment?
```

Context Input:

```
None
```

Sequence Output:

```
your dentist_appointment is at 10am on
the_13th.
```

T5-base prediction (correct):

```
you have two dentist_appointments
scheduled. one on the_5th at 11am and
another on the_13th at 10am.
```

T5-large prediction (correct):

```
your dentist_appointment is on the_13th
at 10 am.
```

T5-3B prediction (incorrect):

```
you have three dentist_appointments
scheduled. one on the_5th at 11am,
another on the_13th at 10am, and the
last is on the_5th at 11am.
```

G.4 SQL2Text case

Query:

```
SELECT company , main_industry FROM
company WHERE headquarters != 'USA'
```

Sequence Out:

```
Show the company name and the main
industry for all companies whose
headquarters are not from USA.
```

T5-base prediction (incorrect):

```
What are the companies and main
industries of companies whose
headquarters are not "USA"?
```

T5-large prediction (correct):

```
What are the companies and main
industries of the companies whose
headquarters are not located in the
United States?
```

T5-3B prediction (incorrect):

```
What are the main industries of
companies whose headquarters are not in
the USA?
```

G.5 DART case

Structured Input:

```
Cotto : eattype : coffee shop | Cotto :
food : English | Cotto : pricerange :
more than [non utf-8 token]30 | Cotto :
customer rating : high | Cotto : area :
city centre | Cotto : near : The
Portland Arms
```

Sequence Output:

```
The Cotto is a highly rated English
coffee shop in the city centre near The
Portland Arms, and prices range more
than [non utf-8 token]30.
```

T5-base prediction (correct):

```
Cotto is a coffee shop providing English
 food in the more than [non utf-8 token
]30 price range. It is located in the
city centre. It is near The Portland
Arms. Its customer rating is high.
```

T5-large prediction (correct):

```
Cotto is a coffee shop providing English
 food in the more than [non utf-8 token
]30 price range. It is located in the
city centre. It is near The Portland
Arms. Its customer rating is high.
```

T5-3B prediction (correct):

```
Cotto is a coffee shop located in the
city centre near The Portland Arms. It
serves English food with a price range
of more than [non utf-8 token]30 and has
 a high customer rating.
```

G.6 Logic2Text case

Structured Input:

```
Caption: 2008 san diego chargers season
| week, date, time, opponent, result,
game site, nfl recap, record
```

Request Input:

```
eq { count { filter_eq { filter_eq {
all_rows ; game site ; qualcomm stadium
} ; time ; 5:15 pm } } ; 3 } = true
```

Sequence Output:

```
in the 2008 san diego chargers season ,
among the games that were played in
qualcomm stadium , 3 of them started at
5:15 pm .
```

T5-base prediction (incorrect):

```
in the 2008 san diego chargers season,
when the game was at qualcomm stadium,
there were three times the time was 5:15
 pm.
```

T5-large prediction (incorrect):

```
in the 2008 san diego chargers season,
when the game was at qualcomm stadium,
there were 3 times the time was 5:15 pm.
```

T5-3B prediction (correct):

```
in the 2008 san diego chargers season,
among the games played at qualcomm
stadium, 3 of them started at 5:15 pm.
```

Alisson Perticheto

Section Title: For the Philippines
Table Section Text: *None*

International							
Event	**11-12**	**12-13**	**13-14**	**14-15**	**16-17**	**17-18**	**18-19**
Worlds			WD	34th		WD	
Four Continents			17th	16th			18th
CS Golden Spin							23rd
CS Lombardia						23rd	WD
CS Finlandia						17th	WD
CS Nebelhorn						11th	WD
Bavarian Open					13th		
Coupe Printemps					4th		
Cup of Nice			13th		20th		
Cup of Tyrol					9th		
EduSport Trophy							2nd
Egna/Gardena		5th			1st		
Bosphorus Cup							1st
Nebelhorn Trophy			18th				
SEA Games						3rd	
Skate Helena				2nd			
International: Junior							
Junior Worlds		18th	WD				
JGP Slovenia		13th					
Bavarian Open		2nd					
GP Bratislava					2nd		
National							
Philippine Champ.	1st J	1st J	1st			1st	1st
J = Junior level; WD = Withdrew							

Figure 7: Visualized highted table for ToTTo case 1.

G.7 ToTTo case

Structured Input: See Figure 7.

Sequence Output:

```
Alisson Perticheto placed 18th at the
2013 Junior Worlds, 17th at the 2014
Four Continents and 16th at the 2015
Four Continents.
```

T5-base prediction (incorrect):

```
Alisson Perticheto finished 18th at the
Junior Worlds and 17th at the Four
Continents.
```

T5-large prediction (incorrect):

```
Alisson Perticheto placed 17th at the
2014 Four Continents and 16th at the
2015 Junior Worlds.
```

T5-3B prediction (correct):

```
Alisson Perticheto finished 17th at the
2014 Four Continents, 16th at the 2015
Four Continents, and 18th at the 2013
Junior Worlds.
```

H Natural Language Template Examples

H.1 Spider Template

Overall Description Template:

```
{db id} contains tables such as {table1
name}, {table2 name}
```

Primary Key Template:

```
{primary key} is the primary key.
```

Table Description Template:

```
Table {table name} has column such as {
column 1 name}, {column 2 name}, ...
```

Foreign Keys Description Template:

```
The {column1 name} of {table 1} is the
foreign key of {column2 name} of {table
2}
```

H.2 TabFact Template

Template Examples:

```
Table 1-24143253-5:
{name} lost his spouse {deceased spouse}
 to {cause of death} on {date of spouses
 death} after {length of marriage} of
marriage; they had {children together}
together; he is currently {current
marital status}

Table 2-14978398-2:
The {version} of song Comme j'ai mal has
 a length of {length} in album {album}
remixed by {remixed by} in year {year}

Table 1-15187735-12:
On {date} in 1936 VFL Season, the home
team {home team} and away team {away
team} had a game at venue {venue} with a
 crowd of {crowd}; the home team score
is {home team score} and the away team
score is {away team score}
```

H.3 WikiSQL Template

Template Example:

```
Table 1-14240688-1:
in {year} were in division {division}, {
league} ranked {regular season}, made it
 to {playoffs} of the playoffs, made it
to <{open cup}> in the open cup, and
kept an average attendance of {avg
attendance}

Table 2-12997882-1:
On {date} in 2008 European Figure
Skating, the home team {home team} and
away team {away team} had a game at
venue {venue} with a crowd of {crowd};
the home team score is {home team score}
 and the away team score is {away team
score}

Table 1-13740746-1:
Episode number {ep no} of gerry anderson
 's new captain scarlet with a title of
{title} is directed by {director} and
written by {written by}; its original
air date is {original air date}; the
production number is {production no}
```