HiTab: A Hierarchical Table Dataset for Question Answering and Natural Language Generation

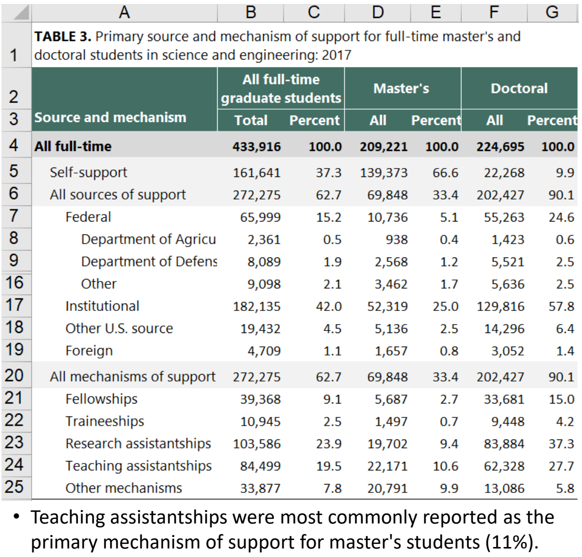
Zhoujun Cheng[1]*, **Haoyu Dong**[2]*†, **Zhiruo Wang**[3]*, **Ran Jia**[2], **Jiaqi Guo**[4] ,
Yan Gao[2], **Shi Han**[2], **Jian-Guang Lou**[2], **Dongmei Zhang**[2]
[1]Shanghai Jiao Tong University, [2]Microsoft Research Asia
[3]Carnegie Mellon University, [4]Xi'an Jiaotong University
blankcheng@sjtu.edu.cn, zhiruow@cs.cmu.edu
jasperguo2013@stu.xjtu.edu.cn
{hadong,jia.ran,yan.gao,shihan,jlou,dongmeiz}@microsoft.com

Abstract

Tables are often created with hierarchies, but existing works on table reasoning mainly focus on flat tables and neglect hierarchical tables. Hierarchical tables challenge table reasoning by complex hierarchical indexing, as well as implicit relationships of calculation and semantics. We present a new dataset, HiTab, to study question answering (QA) and natural language generation (NLG) over hierarchical tables. HiTab is a cross-domain dataset constructed from a wealth of statistical reports and Wikipedia pages, and has unique characteristics: (1) nearly all tables are hierarchical, and (2) questions are not proposed by annotators from scratch, but are revised from real and meaningful sentences authored by analysts. (3) To reveal complex numerical reasoning in analysis, we provide fine-grained annotations of quantity and entity alignment. Experimental results show that HiTab presents a strong challenge for existing baselines and a valuable benchmark for future research. Targeting hierarchical structure, we devise an effective hierarchy-aware logical form for symbolic reasoning over tables. Furthermore, we leverage entity and quantity alignment to explore partially supervised training in QA and conditional generation in NLG, which largely reduces spurious predictions in QA and meaningless descriptions in NLG. The dataset and code are available at https://github.com/microsoft/HiTab.

1 Introduction

In recent years, there are a flurry of works on reasoning over semi-structured tables, e.g., answering questions over tables (Yu *et al.*, 2018; Pasupat and Liang, 2015) and generating fluent and faithful text from tables (Lebret *et al.*, 2016; Parikh *et al.*, 2020).



	A	B	C	D	E	F	G
1	**TABLE 3.** Primary source and mechanism of support for full-time master's and doctoral students in science and engineering: 2017						
2		All full-time graduate students		Master's		Doctoral	
3	Source and mechanism	Total	Percent	All	Percent	All	Percent
4	**All full-time**	**433,916**	**100.0**	**209,221**	**100.0**	**224,695**	**100.0**
5	Self-support	161,641	37.3	139,373	66.6	22,268	9.9
6	All sources of support	272,275	62.7	69,848	33.4	202,427	90.1
7	Federal	65,999	15.2	10,736	5.1	55,263	24.6
8	Department of Agricu	2,361	0.5	938	0.4	1,423	0.6
9	Department of Defens	8,089	1.9	2,568	1.2	5,521	2.5
16	Other	9,098	2.1	3,462	1.7	5,636	2.5
17	Institutional	182,135	42.0	52,319	25.0	129,816	57.8
18	Other U.S. source	19,432	4.5	5,136	2.5	14,296	6.4
19	Foreign	4,709	1.1	1,657	0.8	3,052	1.4
20	All mechanisms of support	272,275	62.7	69,848	33.4	202,427	90.1
21	Fellowships	39,368	9.1	5,687	2.7	33,681	15.0
22	Traineeships	10,945	2.5	1,497	0.7	9,448	4.2
23	Research assistantships	103,586	23.9	19,702	9.4	83,884	37.3
24	Teaching assistantships	84,499	19.5	22,171	10.6	62,328	27.7
25	Other mechanisms	33,877	7.8	20,791	9.9	13,086	5.8

- Teaching assistantships were most commonly reported as the primary mechanism of support for master's students (11%).

Figure 1: A hierarchical table and accompanied descriptions in a National Science Foundation report.[1]

But they mainly focus on simple flat tables and neglect complex tables, e.g., hierarchical tables. A table is regarded as hierarchical if its header exhibits a multi-level structure (Lim and Ng, 1999; Chen and Cafarella, 2014; Wang *et al.*, 2020). Hierarchical tables are widely used, especially in data products, statistical reports, and research papers in government, finance, and science-related domains.

Hierarchical tables challenge QA and NLG due to: **(1) Hierarchical indexing.** Hierarchical headers, such as D2:G3 and A4:A25 in Figure 1, are informative and intuitive for readers, but make cell selection much more compositional than flat tables, requiring multi-level and bi-dimensional indexing. For example, to select the cell E5 ("66.6"), one needs to specify two top header cells, "Master's" and "Percent", and two left header cells, "All full-time" and "Self-support". **(2) Implicit calculation relationships among quantities.** In hierarchical tables, it is common to insert aggregated rows and columns without explicit indications, e.g., total (columns B,D,F and rows 4,6,7,20) and proportion (columns C,E,G), which challenge precise numeri-

[*] Equal contributions. Work done during Zhoujun and Zhiruo's internship at Microsoft Research Asia.

[†] Corresponding author.

[1] https://www.nsf.gov/statistics/2019/nsf19319/

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 1094 - 1110
May 22-27, 2022 ©2022 Association for Computational Linguistics

cal inference. **(3) Implicit semantic relationships among entities.** There are various cross-row, cross-column, and cross-level entity relationships, but lack explicit indications, e.g., "source" and "mechanism" in A2 describe A6:A19 and A20:A25 respectively, and D2 ("Master's") and F2 ("Doctoral") can be jointly described by a virtual entity, "Degree". How to identify semantic relationships and link entities correctly is also a challenge.

In this paper, we aim to build a dataset for hierarchical table QA and NLG. But without sufficient data analysts, it's hard to ensure questions and descriptions are meaningful and diverse (Gururangan *et al.*, 2018; Poliak *et al.*, 2018). Fortunately, large amounts of statistical reports are public from a variety of organizations (StatCan; NSF; Census; CDC; BLS; IMF), containing rich hierarchical tables and textual descriptions. Take Statistics Canada (StatCan) for example, it consists of $6,039$ reports in 27 domains authored by over 1,000 professionals. Importantly, since both tables and sentences are authored by domain experts, sentences are natural and reflective of real understandings of tables.

To this end, we propose a new dataset, HiTab, for QA and NLG on hierarchical tables. **(1)** All sentence descriptions of hierarchical tables are carefully extracted and revised by human annotators. **(2)** It shows that annotations of fine-grained and lexical-level entity linking significantly help table QA (Lei *et al.*, 2020; Shi *et al.*, 2020), motivating us to align entities in text with table cells. In addition to entity, we believe aligning quantities (Ibrahim *et al.*, 2019), especially composite quantities (computed by multiple cells), is also important for table reasoning, so we annotate underlying numerical relationships between quantities in text and table cells, as Table 1 shows. **(3)** Since real sentences in statistical reports are natural, diverse, and reflective of real understandings of tables, we devise a process to construct QA pairs based on existing sentence descriptions instead of asking annotators to propose questions from scratch.

HiTab presents a strong challenge to state-of-the-art baselines. For the QA task, MAPO (Liang *et al.*, 2018) only achieves 29.2% accuracy due to the ineffectiveness of the logical form customized for flat tables. To leverage the hierarchy for table reasoning, we devise a hierarchy-aware logical form for table QA, which shows high effectiveness. We propose partially supervised training given annotations of linked mentions and formulas, which helps models to largely reduce spurious predictions and achieve 45.1% accuracy. For the NLG task, models also have difficulties in understanding deep hierarchies and generate complex analytical texts. We explore controlled generation (Parikh *et al.*, 2020), showing that conditioning on both aligned cells and calculation types helps models to generate meaningful texts.

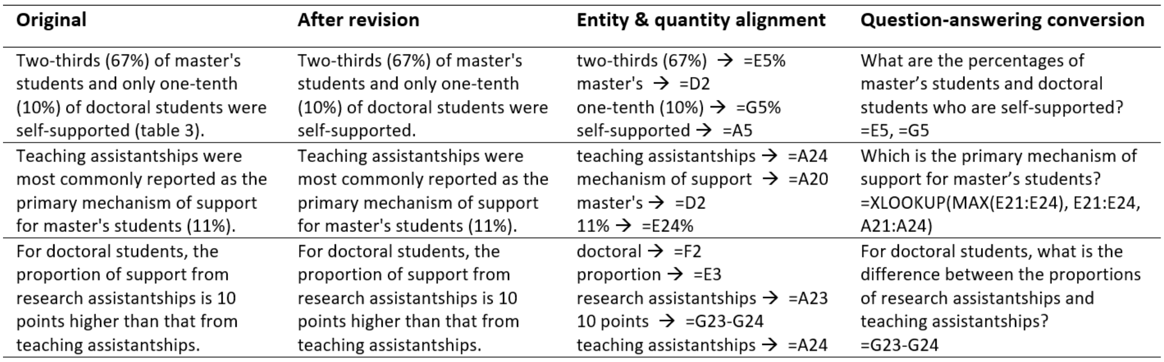
2 Dataset Construction and Analysis

We design an annotation process with six steps. To well-handle the annotation complexity, we recruit 18 students or graduates (13 females and 5 males) in computer science, finance, and English majors from top universities, and provide them with comprehensive online training, documents, and QAs. The annotation totally costs 2,400 working hours. We will discuss the ethical considerations in Section 8.

2.1 Hierarchical Table Collection

We select two representative organizations, Statistics Canada (StatCan) and National Science Foundation (NSF), that are rich of statistical reports. Different from Census; CDC; BLS; IMF that only provide PDF reports where table hierarchies are hard to extract precisely (Schreiber *et al.*, 2017), StaCan and NSF also provide reports in HTML, from which cell information such as text and formats can be extracted precisely using HTML tags.

First, we crawl English HTML statistical reports published in recent five years from StatCan ($1,083$ reports in 27 well-categorized domains) and NSF (208 reports from 11 organizations in science foundation domain). We merge StatCan and NSF and get the combination of various domains. In addition, ToTTo contains a small proportion (5.03%) of hierarchical tables, so we include them to cover more domains from Wikipedia. To keep the balance between statistical reports and Wikipedia pages, we include random $1,851$ tables (50% of our dataset) from ToTTo. Next, we transform HTML tables to spreadsheet tables using a preprocessing script. Since spreadsheet formula is easy to write, execute, and check, the spreadsheet is naturally a great annotation tool to align quantities and answer questions. To enable correct formula execution, we normalize quantities in data cells by excluding surrounding superscripts, internal commas, etc. Extremely small or large tables are filtered out (Appendix A.1 gives more details).



Original	After revision	Entity & quantity alignment	Question-answering conversion
Two-thirds (67%) of master's students and only one-tenth (10%) of doctoral students were self-supported (table 3).	Two-thirds (67%) of master's students and only one-tenth (10%) of doctoral students were self-supported.	two-thirds (67%) → =E5% master's → =D2 one-tenth (10%) → =G5% self-supported → =A5	What are the percentages of master's students and doctoral students who are self-supported? =E5, =G5
Teaching assistantships were most commonly reported as the primary mechanism of support for master's students (11%).	Teaching assistantships were most commonly reported as the primary mechanism of support for master's students (11%).	teaching assistantships → =A24 mechanism of support → =A20 master's → =D2 11% → =E24%	Which is the primary mechanism of support for master's students? =XLOOKUP(MAX(E21:E24), E21:E24, A21:A24)
For doctoral students, the proportion of support from research assistantships is 10 points higher than that from teaching assistantships.	For doctoral students, the proportion of support from research assistantships is 10 points higher than that from teaching assistantships.	doctoral → =F2 proportion → =E3 research assistantships → =A23 10 points → =G23-G24 teaching assistantships → =A24	For doctoral students, what is the difference between the proportions of research assistantships and teaching assistantships? =G23-G24

Table 1: Examples of the annotation process. All sentences describe the table in Figure 1.

2.2 Sentence Extraction and Revision

In this step, annotators manually go through statistical reports and extract sentence descriptions for each table. Sentences consisting of multiple semantic-independent sub-sentences will be carefully split into multiple ones. Annotators are instructed to eliminate redundancy and ambiguity in sentences through revisions including decontextualization and phrase deletion (Parikh *et al.*, 2020). Fortunately, most sentences in statistical reports are clean and fully supported by table data, so few revisions are needed to get high-quality text.

Operators	Formula template (ranges are placeholders)
opposite, percent	=-A5, =B2%
kth-argmax/argmin	=XLOOKUP(SMALL(D1:D3, k), D1:D3, A1:A3).
pair-argmax/argmin	=IF(B1>B2, A1, A2)[2]
sum, average	=SUM(D2:D4), =AVERAGE(D2:D4)
max, count	=MAX(D2:D4), =COUNT(D2:D4)
diff, div	=D3-D4, =D3/D4

Table 2: Example operators and formula templates.

2.3 Entity and Quantity Alignment

In this phase, annotators are instructed to align mentions in text with corresponding cells in tables. It has two parts, entity alignment and quantity alignment, as shown in Table 1. For entity alignment, we record the mappings from entity mentions in text to corresponding cells. Single-cell quantity mentions can be linked similar with entity mentions, but composite quantity mentions are calculated from two or more cells through operators like *max/sum/div/diff* (Table 2). The spreadsheet formula is powerful and easy-to-use for tabular data calculation, so we use the formula to record the calculations process of composite quantities in text, e.g., '10 points higher' (*=G23-G24*). Although quantities are often rounded in descriptions, we neglect rounding and refer to precise quantities in table cells.

2.4 Converting Sentences to QA Pairs

Existing QA datasets instruct annotators to propose questions from scratch, but it's hard to guarantee the meaningfulness and diversity of proposed questions. In HiTab, we simply revise declarative sentences into QA pairs. For each sentence, annotators need to identify a target key part to question about (according to the underlying logic), then convert it to the QA form. All questions are answered by formulas that reflect the numerical inference process. For example, the 'XLOOKUP' operator is frequently used to retrieve the header cells of superlatives, as shown in Table 1. To keep sentences as natural as they are, we do not encourage unnecessary sentence modification during the conversion. If an annotator finds multiple ways to question regarding a sentence, he/she only needs to choose one way that best reflects the overall meaning.

2.5 Regular Inspections and the Final Review

We ask the two most experienced annotators to perform regular inspections and the final review. (1) In the labeling process, they regularly sample annotations (about 10%) from all annotators to give timely feedback on labeling issues. (2) Finally, they review all annotations and fix labeling errors. Also, to assist the final review, we write a script to automatically identify spelling issues and formula issues. To double-check the labeling quality before the final review, we study the agreement of annotators by collecting and comparing annotations on randomly sampled 50 tables from two annotators. It shows 0.89 and 0.82 for quantity and entity alignment in Fleiss Kappa respectively, which are regarded as "almost perfect agreement" (Landis and Koch, 1977), and 64.5 in BLEU-4 after sentence revision, which also indicates high agreement. We further show annotation artifacts are substantially avoided

[2] For samples with XLOOKUP or IF formulas, we didn't explicitly provide the formulas in dataset because some reasoning logics are still too complex to be covered by them, e.g., the candidate cells are not on a continuous row/column. Instead, we manually check the answer cell(s) and provide the answer cell reference(s) for these samples.

Dataset	Tables	Data source			Fine-grained alignment		QA and NLG tasks				
		Table	Question or sentence	Real sentences revised per table	Entity	Quantity	QA	NLG	Questions	Words per question	Sentences
WTQ (Pasupat and Liang, 2015)	2,108	Wikipedia	Post-created	-	-	-	Yes	-	22,033	10.0	-
WikiSQL (Zhong *et al.*, 2017)	26,521	Wikipedia	Post-created	-	-	-	Yes	-	80,654	11.7	-
Spider (Yu *et al.*, 2018)	1,020	College data,WikiSQL	Post-created	-	-	-	Yes	-	10,181	13.2	-
HybridQA (Chen *et al.*, 2020b)	13,000	Wikipedia	Post-created	-	-	-	Yes	-	69,611	18.9	-
TAT-QA (Zhu *et al.*, 2021)	2,757	Financial reports (PDF)	Post-created	-	-	-	Yes	-	16,552	12.5	-
FinQA (Chen *et al.*, 2021)	2,776	Financial reports (PDF)	Post-created	-	-	-	Yes	-	8,281	16.6	-
DART (Nan *et al.*, 2020)	5,623	WTQ,WikiSQL,...	Post-created	-	-	-	-	Yes	-	-	82,191
LogicNLG (Chen *et al.*, 2020a)	7,392	Wikipedia	Post-created	-	-	-	-	Yes	-	-	37,015
ToTTo (Parikh *et al.*, 2020)	83,141	Wikipedia	Pre-existing	1.4	-	-	-	Yes	-	-	120,000
NumericNLG (Suadaa *et al.*, 2021)	1,300	Scientific papers (ACL)	Pre-existing	3.8	-	-	-	Yes	-	-	4,756
HiTab	**3,597**	**Stat. reports, Wiki.**	**Pre-existing**	**5.0 (reports)**	**Yes**	**Yes**	**Yes**	**Yes**	**10,672**	**16.5**	**10,672**

Table 3: Dataset statistics and comparison.

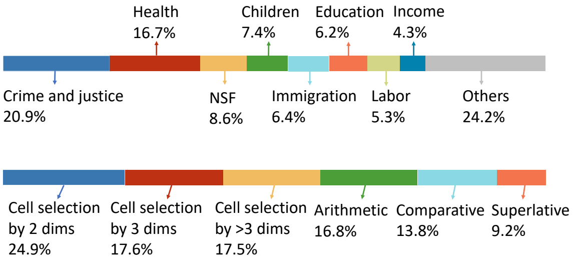


Figure 2: Distribution of domains and operations in StatCan and NSF. *Cell selection by k dims* means that header cells in *k* levels are used in cell selection.

in our dataset in Appendix A.2.

2.6 Hierarchy Extraction

We follow existing work (Lim and Ng, 1999; Chen and Cafarella, 2014; Wang *et al.*, 2020) and use the tree structure to model hierarchical headers. Since cell formats such as merging, indentation, and font bold, are commonly used to present hierarchies, we adapt heuristics in (Wang *et al.*, 2020) to extract top and left hierarchical trees, which has high accuracy. We go through 100 randomly sampled tables in HiTab, 94% of them are precisely extracted. Figure 8 in Appendix shows an illustration.

2.7 Dataset Statistics and Comparison

Table 3 shows a comprehensive comparison of related datasets. HiTab is not among the largest ones, but **(1)** it is the first dataset to study QA and NLG over hierarchical tables (accounting for 98.1% tables in HiTab) in-depth; **(2)** it is annotated with fine-grained entity and quantity alignment; **(3)** compared with TAT-QA, FinQA, and NumericNLG that are single-domain, HiTab has a wide coverage of different domains from statistical reports and Wikipedia, even wider than ToTTo or WTQ that only involves Wikipedia tables; **(4)** the number of real descriptions per table (5.0) in statistical reports (HiTab) is much richer than 1.4 in Wikipedia (ToTTo) and 3.8 in scientific papers, contributing more analytical aspects per table.

Figure 2 analyzes this dataset by domains and operations: domains are diverse, covering 28 domains from statistical reports (fully listed in Appendix A.3) and other open domains from Wikipedia; a large proportion of questions involves complex cell selection and numerical operations.

3 Hierarchical Table QA

Table QA is essential for table understanding, document retrieval, ad-hoc search, *etc*. Hierarchical tables are quite common in these scenarios like in webpages and reports, while current Table QA tasks and methods focus on simple flat tables.

Problem Statement Hierarchical Table QA is defined as follows: given a hierarchical table t and a question x in natural language, output answer y. The question-answer pair should be fully supported by the table. Our dataset $D = \{(x_i, t_i, y_i)\}, i \in [1, N]$ is a set of N question-table-answer triples.

Table QA is usually formulated as a semantic parsing problem (Pasupat and Liang, 2015; Liang *et al.*, 2017), where a parser converts the question into logical form, and an executor executes it to produce the answer. However, existing logical forms for Table QA (Pasupat and Liang, 2015; Liang *et al.*, 2017; Yin *et al.*, 2020) are customized for flat or database tables. The three challenges mentioned in Section 1 (hierarchical indexing, implicit indexing relationships, and implicit semantic relationships) make QA more difficult on hierarchical tables.

3.1 Hierarchy-aware Logical Forms

To this end, we propose a hierarchy-aware logical form that exploits table hierarchies to mitigate these challenges. Specifically, we define *region* as the operating object, and propose two functions for hierarchical region selection.

Definitions Given tree hierarchies of tables extracted in Section 2.6, we define *header* as a header cell (e.g., A7("Federal") in Figure 1), and *level* as a level in the left/top tree (e.g., A5,A6,A20 are on the same level). Existing logical forms on tables treat rows as operating objects and columns as attributes, and thus can not perform arithmetic operations on cells in the same row. However, a row in hierarchical tables is not necessarily a subject or record, thus operations can be applied on cells in the same row. Motivated by this, we define *region* as our operating object, which is a data region in table indexed by both left and top headers (e.g., B6:C19 is a rectangular region indexed by A6,B2). The logical form execution process is divided into two phases: region selection and region operation.

Region Selection We design two functions $(filter\_tree\ h)$ and $(filter\_level\ l)$ to do region selection, where h is a header, l is a level. Functions can be applied sequentially: the subsequent function applies on the return region of the previous function. $(filter\_tree\ h)$ selects a sub-tree region according to a header cell h: if h is a leaf header (e.g., A8), the selected region should be the row/column indexed by h (row 8); if h is a non-leaf header (e.g., A7), the selected region should be the rows/columns indexed by both h and its children headers (row 7-16). $(filter\_level\ l)$ selects a sub-tree from the input tree according to a level l and return the sub-region indexed by headers on level l. These two functions mitigate aforementioned three challenges: (1) hierarchical indexing is achieved by applying these two functions sequentially; (2) with $filter\_level$, data with different calculation types (e.g., rows 4-5) will not be co-selected, thus not incorrectly operated together; (3) level-wise semantics can be captured by aggregating header cell semantics (e.g., embeddings) on this level. Some logical form execution examples are shown in Appendix C.2.

Region Operation Operators are applied on the selected region to produce the answer. We define 19 operators, mostly following MAPO (Liang *et al.*, 2018), and further include some operators (e.g., *difference rate*) for hierarchical tables. Complete logical form functions are shown in Appendix C.1.

3.2 Experimental Setup

3.2.1 Baselines

We present baselines in two branches. One is logical form-based semantic parsing, and the other is end-to-end table parsing without logical forms.

Neural Symbolic Machine (Liang *et al.*, 2017) is a powerful semantic parsing framework consisting of a programmer to generate programs from NL and save intermediate results, and a computer to execute programs. We replace the LSTM encoder with BERT (Devlin *et al.*, 2018), and implement a lisp interpreter for our logical forms as executor. Table is linearized by placing headers in level order, which is shown in detail in Appendix C.4.

TaPas (Herzig *et al.*, 2020) is a state-of-the-art end-to-end table parsing model without generating logical forms. Its power to select cells and reason over tables is gained from its pretraining on millions of tables. To fit TaPas input, we convert hierarchical tables into flat ones following WTQ (Pasupat and Liang, 2015). Specifically, we unmerge the cells spanning many rows/columns on left/top headers and duplicate the contents into unmerged cells. The first top header row is specified as column names.

3.2.2 Weak Supervision

In weak supervision, the model is trained with QA pairs, without golden logical forms. For NSM, we compare three widely-studied learning paradigms:

MML (Dempster *et al.*, 1977) maximizes the marginal likelihood of observed programs. **REINFORCE** (Williams, 1992) maximizes the reward of on-policy samples. **MAPO** (Liang *et al.*, 2018) learns from programs both inside and outside buffer, and samples efficiently by systematic exploration.

Since these methods require consistent programs for learning or warm start, we randomly search $15,000$ programs per sample before training. The pruning rules are shown in Appendix C.3. Finally, 6.12 consistent programs are found per sample.

For TaPas, we use the pre-trained version and follow its weak supervised training process on WTQ.

3.2.3 Partial Supervision

Given labeled entity links, quantity links, and calculations (from the formula), we further explore to guide training in a *partially supervised* way. These three annotations indicate selected headers, region, and operators in QA[3]. For NSM, we exploit them to prune spurious programs, *i.e.*, incorrect programs that accidentally produce correct answers, in two ways. (1) When searching consistent programs, besides producing correct answers, programs are required to satisfy at least two constraints. In this way, the average consistent programs reduces from 6.12 to 2.13 per sample. (2) When training, satisfying each condition will add 0.2 to the original

[3] Entity and quantity alignments in text also occur in the question in most cases. In QA, we apply a simple n-gram matching algorithm to filter out the alignments not in questions.

Method	Dev	Test	%Spurious
Weak Supervision			
MAPO $w.$ original logical form	31.9	29.2	-
TaPas $w/o.$ logical form	39.7	38.9	-
MML $w.$ h.a. logical form	38.9	36.7	22.7
REINFORCE $w.$ h.a. logical form	42.7	38.4	39.3
MAPO $w.$ h.a. logical form	**43.5**	**40.7**	**19.0**
Partial Supervision			
TaPas $w/o.$ logical form	41.2	40.1	-
MML $w.$ h.a. logical form	**45.4**	**45.1**	**10.3**
REINFORCE $w.$ h.a. logical form	44.0	39.7	23.9
MAPO $w.$ h.a. logical form	44.8	44.3	10.7

Table 4: QA execution accuracy (*EA*) on dev/test and spurious program rate of 150 samples on dev. *h.a.* stands for *hierarchy-aware*.

binary 0/1 reward. Sampled programs with reward $r \geq 1.4$ are added to the program buffer.

For TaPas, we additionally provide answer coordinates and calculation types in training following its WikiSQL setting.

3.2.4 Evaluation Metrics

We use *Execution Accuracy* (*EA*) as our metric following (Pasupat and Liang, 2015), measuring the percentage of samples with correct answers. We also report *Spurious Program Rate* to study the percentage that incorrect logical forms produce correct answer. Since we do not have golden logical forms, we manually annotate logical forms for 150 random samples in dev set for evaluation.

3.2.5 Implementations

We split $3,597$ tables into train (70%), dev (15%) and test (15%) with no overlap. We download pre-trained models from huggingface [4]. For NSM, we utilize 'bert-base-uncased', and fine-tune 20K steps on HiTab. Beam size is 5 for both training and inference. To test MAPO original logical form, we convert flatten tables as we do for TaPas. For TaPas, we adopt the PyTorch (Paszke *et al.*, 2019) version in huggingface. We utilize 'tapas-base', and fine-tune 40 epochs on HiTab. All experiments are conducted on a server with four V100 GPUs.

3.3 Results

Table 4 summarizes our evaluation results.

Weak Supervision First, MAPO with our hierarchy-aware logical form outperforms that using its original logical form by a large margin 11.5%, indicating the necessity of designing a logical form leveraging hierarchies. Second, MAPO achieves the best *EA* (40.7%) with the lowest spurious rate (19%). But >50% questions are answered incorrectly, proving QA on HiTab is challenging. Third, though TaPas benefits from pretraining on tables, it performs worse than the best logical form-based method without table pretraining.

Partial Supervision From Table 4, we can conclude the effectiveness of partial supervision in two aspects. First, it improves *EA*. The model learns how to deal with more cases given high-quality programs. Second, it largely lowers *%Spurious*. The model learns to generate correct programs instead of some tricks. MML, whose performance highly depends on the quality of searched programs, benefits the most (36.7% to 45.1%), indicating partial supervision improves the quality of consistent programs by pruning spurious ones. However, TaPas does not gain much improvements from partial supervision, which we will discuss in the next paragraph.

Error Analysis For TaPas, 98.7% of success cases are cell selections, which means TaPas benefits little from partial supervision. This may be caused by: (1) TaPas does not support some common operators on hierarchical table like *difference*; (2) the coarse-to-fine cell selection strategy first selects columns then cells, but cells in different columns may also aggregate in hierarchical tables.

For MAPO under partial supervision, we analyze 100 error cases. Error cases fall into four categories: (1) entity missing (23%): the header to *filter* is not mentioned in question, where a common case is omitted *Total*; model failure, including (2) failing to select correct regions (38%) and (3) failing to generate correct operations (20%); (4) out of coverage (19%): question types unsolvable with the logical form, which is explained in Appendix C.1.

Spurious programs occur mostly in two patterns. In cell selection, there may exist multiple data cells with correct answers (e.g., G9,G16 in Figure 1), while only one is golden. In superlatives, the model can produce the target answer by operating on different regions (e.g., in both region B21:B25 and B23:B25, B23 is the largest).

Level-wise Analysis In Figure 3, we present level-wise accuracy of HiTab QA with MAPO and our hierarchy-aware logical form. *Level* here stands for sum of left and top header levels. As shown, the QA accuracy degrades when table level increases as table structure becomes more complex, except for level $= 2$, *i.e.,* tables with no hierarchies. The reason level $= 2$ performs relatively worse might be that only 1.9% tables without hierarchies are seen in HiTab. We also present an annotated table

[4] https://huggingface.co/

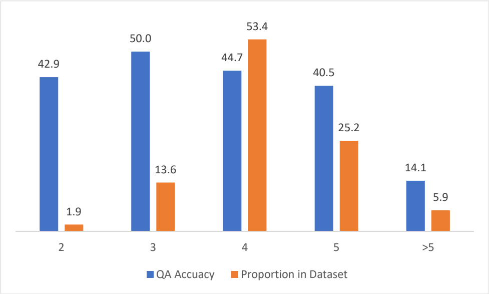


Figure 3: Level-wise QA accuracy and proportion of samples with MAPO and hierarchy-aware logical form.

example from our dataset to illustrate in detail the challenges mentioned in Section 1 that hierarchical tables bring in Appendix C.5.

4 Hierarchical Table-to-Text

4.1 Problem Statement

Some works formulate table-to-text as a summarization problem (Lebret *et al.*, 2016; Wiseman *et al.*, 2017). However, since a full table often contains quite rich information, there lack explicit signals on what to generate, which renders the task unconstrained and the evaluation difficult. On the other hand, some recent works propose *controlled* generation to enable more specific and logical generation: (1) LogicNLG generates a sentence conditioned on a logical form guiding symbolic operations over given cells, but writing correct logical forms as conditions is challenging for common users who are more experienced to write natural language directly, thus restricting the application to real scenario; (2) ToTTo generates a sentence given a table with a set of highlighted cells. In ToTTo's formulation, the condition of cell selection is much easier to specify than the logical form, but it neglects symbolic operations which are critical for generating some analytical sentences involving numerical reasoning in HiTab.

We place HiTab as a middle-ground of ToTTo and LogicNLG to make the task more controllable than ToTTo and closer to real application than LogicNLG. In our setting, given a table, the model generates a sentence conditioned on a group of selected cells (similar to ToTTo) and operators (much easier to be specified than logical forms). Although we use two strong conditions to guide symbolic operations over cells, there still leaves a considerable amount of content planning to be done by the model, such as retrieving contextual cells in a hierarchical table given selected cells, identifying how operators are applied on given cells, and composing sentences in a faithful and logical manner.

We now define our task as: given a hierarchical table T, highlighted cells C, and specified operators O, the goal is to generate a faithful description S. The dataset $H = (T_i, S_i), i \in [1, N]$ is a set of N table-description instances. Description S_i is a sentence about a table T_i and involves a series of operations $O_i = [O_{i1}, O_{i2}, \dots, O_{in}]$ on certain table cells $C_i = [c_{i1}, c_{i2}, \dots, c_{im}]$.

4.2 Controlled Generation

4.2.1 With Highlighted Cells

An entity or quantity in text can be supported by table cells if it is directly stated in cell contents, or can be logically inferred by them. Different from only taking data cells as highlighted cells (Parikh *et al.*, 2020), we also take header cells as highlighted cells, and it is usually the case for superlative ARG-type operations on a specific header level in hierarchical tables, e.g., "Teaching assistantships" is retrieved by ARGMAX in Figure 1. In our dataset, highlighted cells are extracted from annotations of the entity and quantity alignment.

4.2.2 With Operators

Highlighted cells can tell the target for text generation, but is not sufficient, especially for analytical descriptions involving cell operations in HiTab. So we propose to use operators as extra control. It contributes to text clarity and meaningfulness in two ways. (1) It clarifies the numerical reasoning intent on cells. For example, given the same set of data cells, applying SUM, AVERAGE, or COUNT conveys different meanings thus should yield different texts. (2) Operation results on highlighted cells can be used as additional input sources. Existing seq2seq models are not powerful enough to do arithmetic operations (Thawani *et al.*, 2021), e.g., adding up a group of numbers, and it greatly limits their ability to generate correct numbers in sentences. Explicitly pre-computing the calculation results is a promising alternative way to mitigate this gap in seq2seq models. Operators are extracted from annotations of formulas shown in Table 2.

4.2.3 Sub Table Selection and Serialization

Sub Table Selection Under controls of selected cells and operators, we devise a heuristic to retrieve all contextual cells as a sub table. (1) We start with highlighted cells extracted from our entity and quantity alignment, then use the extracted table hierarchy to group the selected cells into the top header, the left header, and the data region. (2) Based on the extracted table hierarchy, we use the source set of top and left header cells to include their indexed data cells, and we also use the source set of data cells to include corresponding header cells. (3) We also include their parent header cells in table hierarchy to construct a full set of headers. In the end, we take the union of of them as the result of sub table selection.

Serialization On each sub table, we do a row-turn traversal on linked cells and concatenate their cell strings using [SEP] tokens. Operator tokens and calculation results are also concatenated with the input sequence. We also experimented with other serialization methods, such as header-data pairing or template-based method, yet none reported superiority over the simple concatenation. Appendix B.1 gives an illustration.

4.3 Experiments

We conduct experiments by fine-tuning four state-of-the-art text generation methods on HiTab.

Pointer Generator (See *et al.*, 2017) A LSTM-based seq2seq model with copy mechanism. While originally designed for text summarization, it is also used in data-to-text (Gehrmann *et al.*, 2018).

BERT-to-BERT (Rothe *et al.*, 2020) A transformer encoder-decoder model (Vaswani *et al.*, 2017) initialized with BERT (Devlin *et al.*, 2018).

BART (Lewis *et al.*, 2019) A pre-trained denoising autoencoder with standard Transformer-based architecture and shows effectiveness in NLG.

T5 (Raffel *et al.*, 2019) A transformer-based pre-trained model. It converts all textual language problems into text-to-text and proves to be effective.

4.3.1 Evaluation Metrics

We use two automatic metrics, BLEU and PARENT. BLEU (Papineni *et al.*, 2002) is broadly used to evaluate text generation. PARENT (Dhingra *et al.*, 2019) is proposed specifically for data-to-text evaluation that additionally aligns n-grams from the reference and generated texts to the source table.

4.3.2 Experiment Setup

Samples are split into train (70%), dev (15%), and test (15%) sets just the same as the QA task. The maximum length of input/output sequence is set to 512/64. Implementation details of all baselines are given in Appendix B.2.

Model	Cell Highlight		Cell & Calculation	
	BLEU-4	PARENT	BLEU-4	PARENT
Pointer-Generator	5.8	8.8	9.0	10.8
BERT-to-BERT	11.4	16.7	11.7	15.4
BART	17.9	28.0	23.8	31.4
T5	**19.5**	**35.7**	**26.6**	**36.9**

Table 5: Results of hierarchical table-to-text.

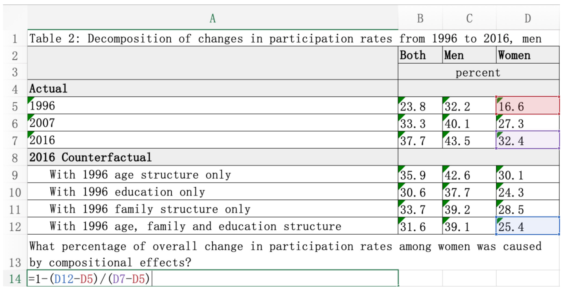
4.3.3 Experiment Result and Analysis

As shown in Table 5, **first**, from an overall point of view, both metrics are not scored high. This well proves the difficulty of HiTab. It could be caused by the hierarchical structure, as well as statements with logical and numerical complexity. **Second**, by comparing two controlled scenarios (cell highlights & both cell highlights and operators), we see that adding operators to conditions greatly help models to generate descriptions with higher scores, showing the effectiveness of our augmented conditional generation setting. **Third**, results on two controlled scenarios across baselines are quite consistent. Replacing the traditional LSTM with transformers shows large increasing. Leveraging seq2seq-like pretraining yields a rise of +6.5 BLEU and +11.3 PARENT. Lastly, between pretrained transformers, T5 reports higher scores over BART, probably for T5 is more extensively tuned during pre-training.

Further, to study the generation difficulty concerning **table hierarchy**, we respectively evaluate samples at different hierarchical depths, *i.e.*, table's maximum depths in top and left header trees. In groups of 2, 3, 4+ depth, BLEU scores 31.7, 26.5, and 21.3; PARENT scores 40.9, 36.5, and 31.6. The reason could be that, as the table header hierarchy grows deeper, the data indexing becomes increasingly compositional, rendering it harder to baseline models to configure entity relationships and compose logical sentences.

5 Related Work

Table-to-Text Existing datasets are restricted in flat tables or specific subjects (Liang *et al.*, 2009; Chen and Mooney, 2008; Wiseman *et al.*, 2017; Novikova *et al.*, 2016; Banik *et al.*, 2013; Lebret *et al.*, 2016; Moosavi *et al.*, 2021). The most related table-to-text dataset to HiTab is ToTTo (Parikh *et al.*, 2020), in which complex tables are also included. There are two main differences between HiTab and ToTTo: (1) in ToTTo, hierarchical tables only account for a small proportion (5%), and there are no indication and usage of table hierarchies. (2) in addition to cell highlights, Hitab conditions on



	A	B	C	D
1	Table 2: Decomposition of changes in participation rates from 1996 to 2016, men			
2		Both	Men	Women
3		percent		
4	Actual			
5	1996	23.8	32.2	16.6
6	2007	33.3	40.1	27.3
7	2016	37.7	43.5	32.4
8	2016 Counterfactual			
9	With 1996 age structure only	35.9	42.6	30.1
10	With 1996 education only	30.6	37.7	24.3
11	With 1996 family structure only	33.7	39.2	28.5
12	With 1996 age, family and education structure	31.6	39.1	25.4
13	What percentage of overall change in participation rates among women was caused by compositional effects?			
14	=1-(D12-D5)/(D7-D5)			

Figure 4: A meaningful but challenging case in HiTab.

Method	**Test Accuracy**	
MAPO w. partial supervision	32.6	
	BLEU	**PARENT**
T5 w. cell & calculation	16.9	28.8

Table 6: Results of cross-domain evaluation.

operators that reflect symbolic operations on cells.

Table QA mainly focuses on DB tables (Wang *et al.*, 2015; Yu *et al.*, 2018; Zhong *et al.*, 2017) and flat web tables (Pasupat and Liang, 2015; Sun *et al.*, 2016). Recently, there are some datasets on domain-specific table QA (Chen *et al.*, 2021; Zhu *et al.*, 2021) and jointly QA over tables and texts (Chen *et al.*, 2020b; Zhu *et al.*, 2021), but hierarchical tables still have not been studied in depth. CFGNN (Zhang, 2020) and GraSSLM (Zhang *et al.*, 2020) uses gragh neural networks to encode tables for QA, but all tables are database tables and relational web tables without hierarchies, respectively. Wang *et al.* (2021) include some hierarchical tables but only focuses on table search.

6 Discussion

HiTab also presents cross-domain and complicated-calculation challenges. (1) To explore cross-domain generalizability, we randomly split train/dev/test by domains for three times and present the average results of our best methods in Table 6. We found decreases in all metrics in QA and NLG. (2) Figure 4 shows a case that challenges existing methods: performing complicated calculations requires to jointly consider quantity relationships, header semantics, and hierarchies.

7 Conclusion

We present a new dataset, HiTab, that simultaneously supports QA and NLG on hierarchical tables, where tables are collected from statistical reports and Wikipedia in various domains. Importantly, we provide fine-grained annotations on entity and quantity alignment. In experiments, we introduce strong baselines and conduct detailed analysis on QA and NLG tasks on HiTab. Results suggest that HiTab can serve as a challenging and valuable benchmark for future research on complex tables.

8 Ethical Considerations

This work presents HiTab, a free and open English dataset for the research community to study table question-answering and table-to-text over hierarchical tables. Our dataset contains well-processed tables, annotations (QA pairs, target text, and bidirectionally mappings between entities and quantities in text and the corresponding cells in table), recognized table hierarchies, and source code. Data in HiTab are collected from two public organizations, StatCan and NSF. Both of them allow sharing and redistribution of their public reports, so there is no privacy issue. We collect tables and accompanied descriptive sentences from StatCan and NSF. We also include hierarchical tables in Wikipedia from ToTTo, which is a public dataset under MIT license, so there is no risk to use it. And in the labeling process, annotators need to check if there exist any names or uniquely identifies individual people or offensive content. They did not find any such sensitive information in our dataset. We recruit 18 students or graduates in computer science, finance, and English majors from top universities(13 females and 5 males). Each student is paid \$7.8 per hour (above the average local payment of similar jobs), totally spending 2,400 hours. We finally get 3,597 tables and 10,672 well-annotated sentences. And the data got approval from an ethics review board by an anonymous IT company. The details for our data collection and characteristics are introduced in Section 2.

References

Eva Banik, Claire Gardent, and Eric Kow. The kbgen challenge. In *the 14th European Workshop on Natural Language Generation (ENLG)*, pages 94–97, 2013.

BLS. U.s. bureau of labor statistics. `https://www.bls.gov` Accessed July 4, 2021.

CDC. Centers for disease control and prevention. `https://www.cdc.gov` Accessed July 4, 2021.

Census. Census bureau. `https://www.census.gov`. Accessed July 4, 2021.

Zhe Chen and Michael Cafarella. Integrating spreadsheet data via accurate and low-effort extraction. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1126–1135, 2014.

David L Chen and Raymond J Mooney. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, pages 128–135, 2008.

Wenhu Chen, Jianshu Chen, Yu Su, Zhiyu Chen, and William Yang Wang. Logical natural language generation from open-domain tables. *arXiv preprint arXiv:2004.10404*, 2020.

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Wang. Hybridqa: A dataset of multi-hop question answering over tabular and textual data. *arXiv preprint arXiv:2004.07347*, 2020.

Zhiyu Chen, Wenhu Chen, Charese Smiley, Sameena Shah, Iana Borova, Dylan Langdon, Reema Moussa, Matt Beane, Ting-Hao Huang, Bryan Routledge, et al. Finqa: A dataset of numerical reasoning over financial data. *arXiv preprint arXiv:2109.00122*, 2021.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

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

Bhuwan Dhingra, Manaal Faruqui, Ankur Parikh, Ming-Wei Chang, Dipanjan Das, and William W Cohen. Handling divergent reference texts when evaluating table-to-text generation. *arXiv preprint arXiv:1906.01081*, 2019.

Sebastian Gehrmann, Falcon Z Dai, Henry Elder, and Alexander M Rush. End-to-end content and plan selection for data-to-text generation. *arXiv preprint arXiv:1810.04700*, 2018.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324*, 2018.

Jonathan Herzig, Pawel Krzysztof Nowak, Thomas Mueller, Francesco Piccinno, and Julian Eisenschlos. Tapas: Weakly supervised table parsing via pretraining. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, 2020.

Yusra Ibrahim, Mirek Riedewald, Gerhard Weikum, and Demetrios Zeinalipour-Yazti. Bridging quantities in tables and text. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 1010–1021. IEEE, 2019.

IMF. International monetary fund. https://www.imf.org. Accessed July 4, 2021.

J Richard Landis and Gary G Koch. The measurement of observer agreement for categorical data. *biometrics*, pages 159–174, 1977.

Rémi Lebret, David Grangier, and Michael Auli. Neural text generation from structured data with application to the biography domain. *arXiv:1603.07771*, 2016.

Wenqiang Lei, Weixin Wang, Zhixin Ma, Tian Gan, Wei Lu, Min-Yen Kan, and Tat-Seng Chua. Re-examining the role of schema linking in text-to-sql. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6943–6954, 2020.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*, 2019.

Percy Liang, Michael I Jordan, and Dan Klein. Learning semantic correspondences with less supervision. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 91–99, 2009.

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D Forbus, and Ni Lao. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 23–33, 2017.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc V Le, and Ni Lao. Memory augmented policy optimization for program synthesis and semantic parsing. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018.

Seung-Jin Lim and Yiu-Kai Ng. An automated approach for retrieving hierarchical data from html tables. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 466–474, 1999.

Nafise Sadat Moosavi, Andreas Rücklé, Dan Roth, and Iryna Gurevych. Learning to reason for text generation from scientific tables. *arXiv:2104.08296*, 2021.

Linyong Nan, Dragomir Radev, Rui Zhang, Amrit Rau, Abhinand Sivaprasad, Chiachun Hsieh, Xiangru Tang, Aadit Vyas, Neha Verma, Pranav Krishna, et al. Dart: Open-domain structured data record to text generation. *arXiv preprint arXiv:2007.02871*, 2020.

Jekaterina Novikova, Oliver Lemon, and Verena Rieser. Crowd-sourcing nlg data: Pictures elicit better data. *arXiv preprint arXiv:1608.00339*, 2016.

NSF. National science foundation. https://www.nsf.gov. Accessed July 4, 2021.

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

Ankur P Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. Totto: A controlled table-to-text generation dataset. *arXiv preprint arXiv:2004.14373*, 2020.

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. *arXiv preprint arXiv:1508.00305*, 2015.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

Adam Poliak, Jason Naradowsky, Aparajita Haldar, Rachel Rudinger, and Benjamin Van Durme. Hypothesis only baselines in natural language inference. *arXiv preprint arXiv:1805.01042*, 2018.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv:1910.10683*, 2019.

Sascha Rothe, Shashi Narayan, and Aliaksei Severyn. Leveraging pre-trained checkpoints for sequence generation tasks. *Transactions of the Association for Computational Linguistics*, 8:264–280, 2020.

Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. Deepdesrt: Deep learning for detection and structure recognition of tables in document images. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 1162–1167. IEEE, 2017.

Abigail See, Peter J Liu, and Christopher D Manning. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*, 2017.

Tianze Shi, Chen Zhao, Jordan Boyd-Graber, Hal Daumé III, and Lillian Lee. On the potential of lexico-logical alignments for semantic parsing to sql queries. *arXiv:2010.11246*, 2020.

StatCan. Statistics canada. https://www150.statcan.gc.ca. Accessed July 4, 2021.

Lya Hulliyyatus Suadaa, Hidetaka Kamigaito, Kotaro Funakoshi, Manabu Okumura, and Hiroya Takamura. Towards table-to-text generation with numerical reasoning. 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 1451–1465, 2021.

Huan Sun, Hao Ma, Xiaodong He, Wen-tau Yih, Yu Su, and Xifeng Yan. Table cell search for question answering. In *Proceedings of the 25th International Conference on World Wide Web*, pages 771–782, 2016.

Avijit Thawani, Jay Pujara, Pedro A Szekely, and Filip Ilievski. Representing numbers in nlp: a survey and a vision. *arXiv preprint arXiv:2103.13136*, 2021.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017.

Yushi Wang, Jonathan Berant, and Percy Liang. Building a semantic parser overnight. 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 1332–1342, 2015.

Zhiruo Wang, Haoyu Dong, Ran Jia, Jia Li, Zhiyi Fu, Shi Han, and Dongmei Zhang. Structure-aware pre-training for table understanding with tree-based transformers. *arXiv:2010.12537*, 2020.

Fei Wang, Kexuan Sun, Muhao Chen, Jay Pujara, and Pedro Szekely. Retrieving complex tables with multi-granular graph representation learning. *arXiv preprint arXiv:2105.01736*, 2021.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052*, 2017.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. Tabert: Pretraining for joint understanding of textual and tabular data. *arXiv preprint arXiv:2005.08314*, 2020.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. *arXiv preprint arXiv:1809.08887*, 2018.

Yuchen Zhang, Panupong Pasupat, and Percy Liang. Macro grammars and holistic triggering for efficient semantic parsing. *arXiv preprint arXiv:1707.07806*, 2017.

Xingyao Zhang, Linjun Shou, Jian Pei, Ming Gong, Lijie Wen, and Daxin Jiang. A graph representation of semi-structured data for web question answering. *arXiv preprint arXiv:2010.06801*, 2020.

Xuanyu Zhang. Cfgnn: Cross flow graph neural networks for question answering on complex tables. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 9596–9603, 2020.

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

Fengbin Zhu, Wenqiang Lei, Youcheng Huang, Chao Wang, Shuo Zhang, Jiancheng Lv, Fuli Feng, and Tat-Seng Chua. Tat-qa: A question answering benchmark on a hybrid of tabular and textual content in finance. *arXiv preprint arXiv:2105.07624*, 2021.

A More Details on Dataset

A.1 Dataset Preprocessing

We filter tables using these constraints: (1) number of rows and columns are more than 2 and less than 64; (2) cell strings have no more than one non-ASCII character and 20 tokens; (3) hierarchies are successfully parsed via the method in 2.6. (4) hierarchies have no more than four levels on one side. Finally, 85% tables meet all constraints.

A.2 Annotation Artifacts

Annotation artifacts are common in large scale NLP datasets, which may raise unwanted statistical correlations making the task easier (Gururangan *et al.*, 2018). In HiTab, the annotation artifacts may come from homogeneous patterns of questions. To address this issue, we ask annotators to revise questions from the high-quality descriptions from statistical reports from 28 domains to guarantee the diversity and naturalness, and encourage them to choose the best way to raise question reflecting the overall meaning of the description. To further check whether and where artifacts may exist in our dataset, we conduct two experiments on QA and count the ratio of answer occurring in the question:

- Use table as only input without question, to see if there is a potential pattern between table and answer. We train BERT+MAPO for $10,000$ steps and TaPas for 10 epochs. Both methods can't converge under this setting, with 4.0% and 2.6% accuracy on the test set. The poor performance indicates model can't learn the answers by exploring and leveraging artifacts between the table and answer, and thus should learn to jointly inference the question and table.

- Shuffle the rows and columns of table randomly. Experiments show similar performance ($\pm 1\%$) between our original tables and shuffled tables. The result shows that the correlation between answer and table cell position is very little, thus model can't choose some specific positions, e.g., cell at the first row and first column, as a shortcut prediction.

- The ratio that answer occurs in the question is only 5.3%. Model that only learns to retrieve the question can't achieve high performance.

A.3 Domain Distribution

The full 29 domains of sample distribution in HiTab are shown in Figure 5.

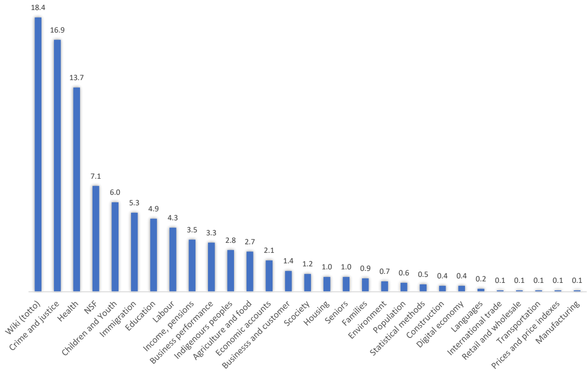


Figure 5: Proportion of samples in different 29 domains.

A.4 Annotation Interface

The annotation interface looks like Figure 6. Since spreadsheet formula is easy to write, execute, and check, the spreadsheet is naturally a great annotation tool. Annotators can use the Excel formula conveniently for cell linking and calculation in entity alignment and answering questions.

B Hierarchical Table-to-Text

B.1 Illustration on Controlled Generation in Hierarchical Table-to-Text.

Please find the illustration shown in Figure 7.

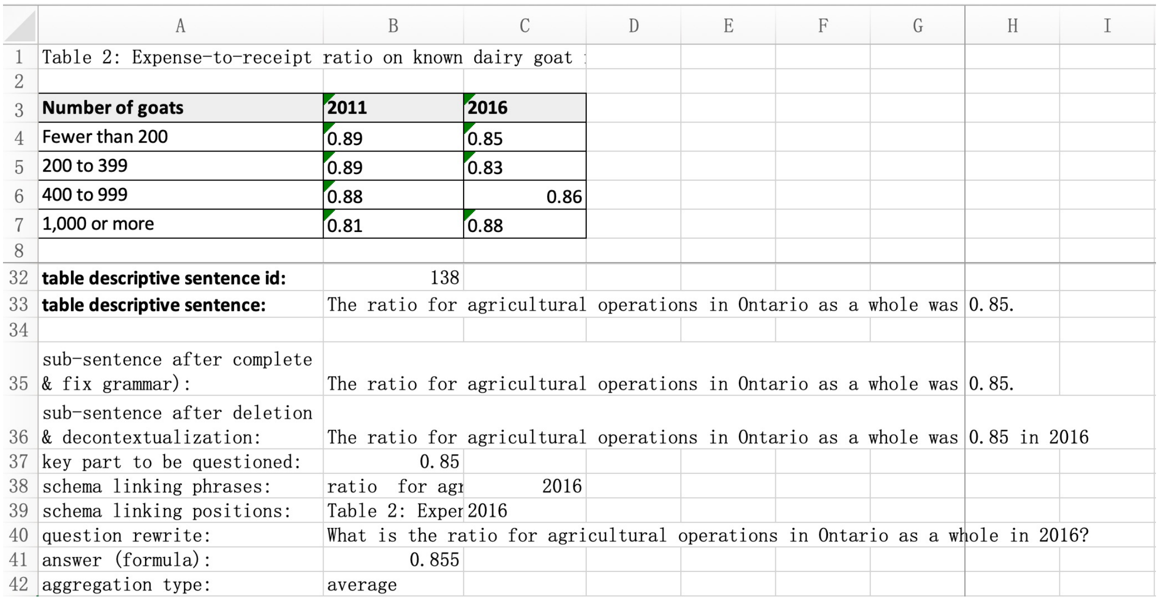
B.2 Baseline Implementation Details

We perform optimized tuning for baselines using the following settings.

Pointer Generator (See *et al.*, 2017) A LSTM-based seq2seq model with copy mechanism. The model uses two-layer bi-directional LSTMs for the encoder with 300-dim word embeddings and 300 hidden units. We perform fine-tuning using batch size 2, learning rate 0.05, and beam size 5.

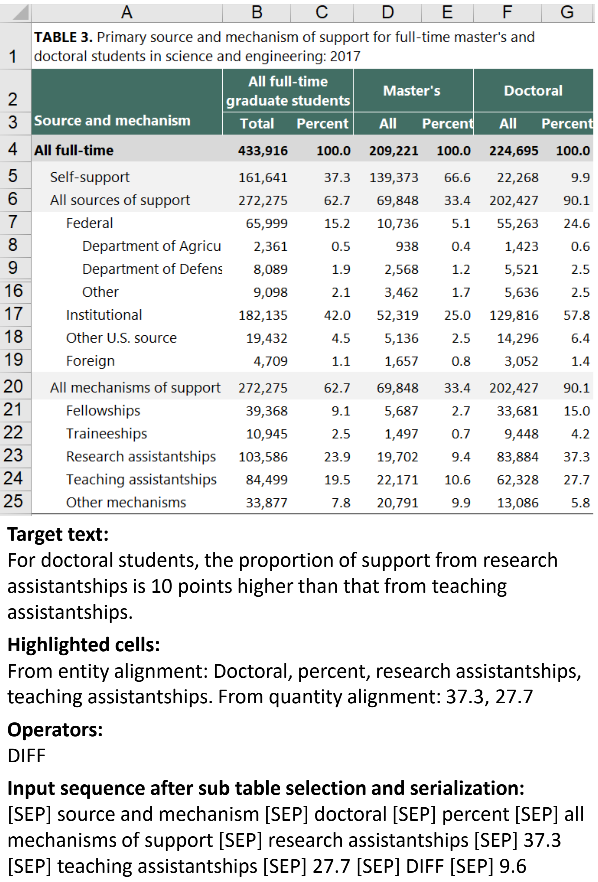
BERT-to-BERT (Rothe *et al.*, 2020) A transformer encoder-decoder model (Vaswani *et al.*, 2017) where the encoder and decoder are both initialized with BERT (Devlin *et al.*, 2018) by loading the checkpoint named 'bert-base-uncased' provided by the huggingface/transformers repository. We perform fine-tuning using batch-size 2 and learning rate $3e^{-5}$.

BART (Lewis *et al.*, 2019) BART is a pre-trained denoising autoencoder for seq2seq lan-



	A	B	C	D	E	F	G	H	I
1	Table 2: Expense-to-receipt ratio on known dairy goat								
2									
3	**Number of goats**	**2011**	**2016**						
4	Fewer than 200	0.89	0.85						
5	200 to 399	0.89	0.83						
6	400 to 999	0.88	0.86						
7	1,000 or more	0.81	0.88						
8									
32	**table descriptive sentence id:**	138							
33	**table descriptive sentence:**	The ratio for agricultural operations in Ontario as a whole was						0. 85.	
34									
35	sub-sentence after complete & fix grammar):	The ratio for agricultural operations in Ontario as a whole was						0. 85.	
36	sub-sentence after deletion & decontextualization:	The ratio for agricultural operations in Ontario as a whole was						0. 85 in 2016	
37	key part to be questioned:	0. 85							
38	schema linking phrases:	ratio for agr	2016						
39	schema linking positions:	Table 2: Exper	2016						
40	question rewrite:	What is the ratio for agricultural operations in Ontario as a whole in 2016?							
41	answer (formula):	0. 855							
42	aggregation type:	average							

Figure 6: Annotation interface in Excel.



	A	B	C	D	E	F	G
1	**TABLE 3.** Primary source and mechanism of support for full-time master's and doctoral students in science and engineering: 2017						
2		All full-time graduate students		Master's		Doctoral	
3	Source and mechanism	Total	Percent	All	Percent	All	Percent
4	**All full-time**	**433,916**	**100.0**	**209,221**	**100.0**	**224,695**	**100.0**
5	Self-support	161,641	37.3	139,373	66.6	22,268	9.9
6	All sources of support	272,275	62.7	69,848	33.4	202,427	90.1
7	Federal	65,999	15.2	10,736	5.1	55,263	24.6
8	Department of Agricu	2,361	0.5	938	0.4	1,423	0.6
9	Department of Defens	8,089	1.9	2,568	1.2	5,521	2.5
16	Other	9,098	2.1	3,462	1.7	5,636	2.5
17	Institutional	182,135	42.0	52,319	25.0	129,816	57.8
18	Other U.S. source	19,432	4.5	5,136	2.5	14,296	6.4
19	Foreign	4,709	1.1	1,657	0.8	3,052	1.4
20	All mechanisms of support	272,275	62.7	69,848	33.4	202,427	90.1
21	Fellowships	39,368	9.1	5,687	2.7	33,681	15.0
22	Traineeships	10,945	2.5	1,497	0.7	9,448	4.2
23	Research assistantships	103,586	23.9	19,702	9.4	83,884	37.3
24	Teaching assistantships	84,499	19.5	22,171	10.6	62,328	27.7
25	Other mechanisms	33,877	7.8	20,791	9.9	13,086	5.8

Target text:
For doctoral students, the proportion of support from research assistantships is 10 points higher than that from teaching assistantships.

Highlighted cells:
From entity alignment: Doctoral, percent, research assistantships, teaching assistantships. From quantity alignment: 37.3, 27.7

Operators:
DIFF

Input sequence after sub table selection and serialization:
[SEP] source and mechanism [SEP] doctoral [SEP] percent [SEP] all mechanisms of support [SEP] research assistantships [SEP] 37.3 [SEP] teaching assistantships [SEP] 27.7 [SEP] DIFF [SEP] 9.6

Figure 7: An illustration on controlled generation.

guage modeling. It uses standard Transformer-based architecture and shows effectiveness in NLG. We align model configuration with the BASE version of BART, and use the model 'facebook/bart-base' in huggingface/transformers. During fine-tuning, we use a batch size of 8 and a learning rate of $2e^{-4}$.

T5 (Raffel *et al.*, 2019) T5 is also a transformer-based pre-training LM. It trains extensively on text-to-text tasks and scores high on generation tasks. We use the pre-trained model 't5-base' in hugging-face/transformers. For fine-tuning, we set batch size to 8 and learning rate to $2e^{-4}$.

We use a beam size of 5 to search decoded outputs (sequence lengths range from 8 to 60 tokens)

C Hierarchical Table QA

C.1 Logical Form Function List

We list our logical form functions in Table 7.

Union selection is required for comparative and arithmetic operations. It is achieved by allowing variable number of headers in $filter\_tree$, where "variable" is one or two in practice.

In our implementation, a function by default takes the selected region of last function as input region to prune search space. We use grammars to filter left headers before top headers, and a $(filter\_level)$ is necessary after filtering one direction of tree even when only the leaf level is available. And we deactivate order relation functions (e.g., *eq* function) and the order argument *k*

Function	**Arguments**	**Returns**	**Description**
(**filter_tree** h)	**h**: a header	a region	Select a region indexed by sub-tree of the given header in the given region.
(**filter_level** l)	**l**: a level	a region	Select a region indexed by headers on the given level in the given region.
(**argmax** k) (**argmin** k)	**k**: a number	a list of headers	Find the header(s) with k-th largest/ smallest value in the region. [Input region should have one row or one column of data]
(**max** l) (**min** l) (**sum** l) (**average** l)	**l**: a level	a region	Maximum/minimum/sum/average of the given region, grouping by headers of the given level, *i.e.*, data values aggregate according to their header strings on the given level.
(**count** l)	**l**: a level	a number	Count the number of headers on the given level of given region.
(**difference**) (**proportion**) (**proportion_rev**) (**difference_rate**) (**difference_rate_rev**)		a number	Absolute difference, proportion and difference rate of given two elements a and b in region. rev means changing order of operands. e.g., $proportion$ applies b/a and $proportion\_rev$ applies a/b. [Input region should have two data elements]
(**greater_than** n) (**greater_eq_than** n) (**less_than** n) (**less_eq_than** n) (**eq** n) (**not_eq** n)	**n**: a number	a list of headers	Find the header(s) with data value(s) that have certain order relation with the given number. [Input region should have one row or one column of data]
(**opposite**)		a number	Take opposite value of data in a given region. [Input region should have one data element]

Table 7: Function list of hierarchy-aware logical form

Question	Logical Forms
Cell Selection Q: What is the GDP of China in 2012?	(filter_tree 2012) (filter_tree china) (filter_level LEFT_2) (filter_tree gdp) (filter_level TOP_1)
Superlative Q: Which country has the highest GDP in 2012?	(filter_tree 2012) (filter_level LEFT_2) (filter_tree gdp) (filter_level TOP_1) (argmax 1)
Q: How much more is U.S. GDP higher than China in 2013?	(filter_tree u.s. china) (filter_level LEFT_2) (filter_tree gdp) (filter_level TOP_1) (difference)

Table 8: Examples of our logical form. The table to be questioned is in Fig. 8. *LEFT_1* is a symbol for the first level on the left.

in *argmax/argmin* because there are few questions in these types and activating them will largely increase number of spurious programs when searching.

The logical form coverage after deactivation is 78.3% in 300 iterations of random exploration. Some typical question types that can not be covered are: (1) scale conversion, e.g., 0.984 to 98.4%, (2) operating data indexed by different levels of headers, e.g., proportion of total, (3) complex composite operations, e.g., Figure 4.

C.2 Examples of Logical Form Execution

Take the table in Figure 8 as input table, we demonstrate three types of questions with complete logical forms in Table 8.

C.3 Pruning Rules in Searching

We use trigger words and POS tags for some functions in random exploration, which is inspired by (Zhang *et al.*, 2017; Liang *et al.*, 2018). Functions are allowed to be selected only when triggers appear in the question. Triggers are listed in Table 9.

Function	Trigger Words
argmax argmin	JJR, JJS, RBR, RBS, top, first, bottom, and last.
max min	JJS, RBS
average	average, mean
sum	all, combine, total, sum
count	how, many, total, number
difference difference_rate difference_rate_rev	difference, more, than, change,compare, JJR RBR.
proportion proportion_rev	times, percent, percentage, fraction

Table 9: Trigger Words for Functions

C.4 Table Linearization

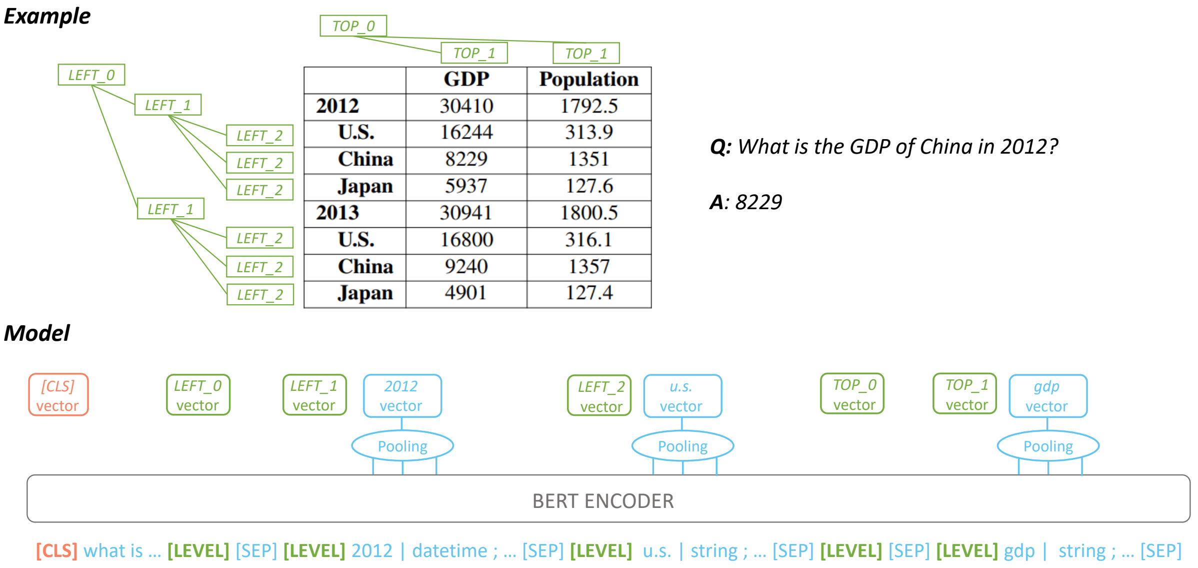
We linearize the question and table according to Figure 8.

The input is concatenation of question and table. Table is linearized by putting headers in level order. Each level is led by a *[LEVEL]* token to gather current level embedding. The first *[LEVEL]* token stands for level zero of left. Each header is linearized as *name* | *type*. *name* is the tokenized header string. *type* is the entity type parsed by Stanford CoreNLP, which includes “string”, “number”, “datetime” in our case. Headers with the same *name* will gather token embeddings by mean pooling.

C.5 Illustration on Challenges in Hierarchical Table

We present an annotated example in Figure 9 to show the challenges of hierarchical table introduced in Section 1.

To precisely answer the question in the figure, the model/method first needs to hierarchically index the grey region with “field in science” and “doctoral”, which requires understanding of textual and spatial semantics of the hierarchical table since the textual headers are spatially (seen as a tree) related with the region. Second, from the phrase “most enrolled”, it should further indexes “All” (column G) rather than “Percent” (column H) and infers *argmax* operation, , which calls for the ability to distinguish between different calculation relationships.

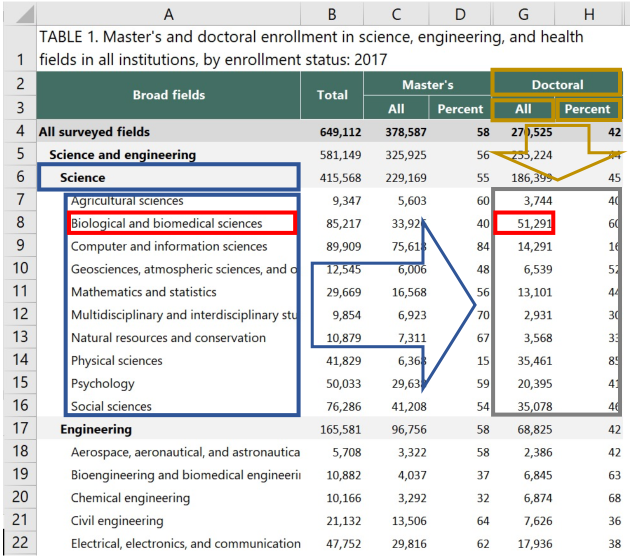


	GDP	Population
2012	30410	1792.5
U.S.	16244	313.9
China	8229	1351
Japan	5937	127.6
2013	30941	1800.5
U.S.	16800	316.1
China	9240	1357
Japan	4901	127.4

Q: *What is the GDP of China in 2012?*

A: *8229*

Figure 8: An QA example table with hierarchy and its linearized input to the encoder. Each level in the hierarchical header starts with a *LEVEL* token to learn a level representation. *LEFT_k* means the *k*th level in the left tree. Each header cell has a unique header cell representation.



	A	B	C	D	G	H
1	TABLE 1. Master's and doctoral enrollment in science, engineering, and health fields in all institutions, by enrollment status: 2017					
2	Broad fields	Total	Master's		Doctoral	
3			All	Percent	All	Percent
4	**All surveyed fields**	**649,112**	**378,587**	**58**	**270,525**	**42**
5	**Science and engineering**	581,149	325,925	56	255,224	44
6	**Science**	415,568	229,169	55	186,399	45
7	Agricultural sciences	9,347	5,603	60	3,744	40
8	Biological and biomedical sciences	85,217	33,92	40	51,291	60
9	Computer and information sciences	89,909	75,61	84	14,291	16
10	Geosciences, atmospheric sciences, and o	12,545	6,006	48	6,539	52
11	Mathematics and statistics	29,669	16,568	56	13,101	44
12	Multidisciplinary and interdisciplinary stu	9,854	6,923	70	2,931	30
13	Natural resources and conservation	10,879	7,311	67	3,568	33
14	Physical sciences	41,829	6,36	15	35,461	85
15	Psychology	50,033	29,63	59	20,395	41
16	Social sciences	76,286	41,208	54	35,078	46
17	**Engineering**	165,581	96,756	58	68,825	42
18	Aerospace, aeronautical, and astronautica	5,708	3,322	58	2,386	42
19	Bioengineering and biomedical engineerii	10,882	4,037	37	6,845	63
20	Chemical engineering	10,166	3,292	32	6,874	68
21	Civil engineering	21,132	13,506	64	7,626	36
22	Electrical, electronics, and communication	47,752	29,816	62	17,936	38

Source: National Science Foundation of U.S.

Question: what is the **most enrolled field in science** for **doctoral** students?

Textual semantic

- **science -> A6**
- **doctoral -> G2**

Hierarchical semantic

- **field in science -> A7:A16**
- **doctoral -> G3, H3**

Spatial semantic

- **Data -> G7:H16**

Numerical/calculational semantic

- **most enrolled -> argmax(G7:G16)**

Answer: Biological and biomedical sciences

Figure 9: A detailed annotated example to illustrate challenges in hierarchical table.