Parameter-Efficient Abstractive Question Answering over Tables and over Text

Anonymous ACL submission

Abstract

A long-term ambition of information seeking question answering (QA) systems is to reason over multi-modal contexts and generate natural answers to user queries. Today, memory intensive pre-trained language models are adapted to downstream tasks such as QA by fine-tuning the model on QA data in a specific modality like unstructured text or structured tables. To avoid training such memory-hungry models and utilizing a uniform architecture for each modality, parameter-efficient transfer learning techniques such as adapters add and train small task-specific bottle-neck layers between transformer layers. However, modality-specific adapter layers infused in a pre-trained transformer also require uniformity in the input sequence, which contradicts with existing work that trains structure-specific layers on multi-modal data. In this work, we study parameter-efficient abstractive QA in encoder-decoder models over structured tabular data and unstructured textual data using only 1.5% additional parameters for each modality. We retain table structure information by a hierarchy preserving transformation of complex hierarchical tables to 1-dimensional sequences, thus maintaining uniformity in the model input. We also ablate over adapter layers in both encoder and decoder modules and study the efficiency-performance trade-off and demonstrate that reducing additional trainable parameters down to 0.7%–1.0% leads to comparable results. Our models outperform current state-of-the-art models on tabular QA datasets such as Tablesum and FeTaQA and achieve comparable performance on a text QA dataset such as NarrativeQA using significantly less trainable parameters.

1 Introduction

Information seeking systems over diverse contexts necessitates model capabilities to reason over unstructured and structured data such as free-form text, tables and images (Agrawal et al., 2016; Vakulenko et al., 2019; Hudson and Manning, 2019; Zhang et al., 2020; Zhu et al., 2021; Deldjoo et al., 2021). Such systems might have the additional requirement of generating natural language responses if deployed as task-oriented conversational agents (Wen et al., 2015; Carnegie and Oh, 2000; Rambow et al., 2001; Ratnaparkhi, 2002). Recent work on open-domain question answering (QA) predominately addresses these challenges with fine-tuning massive pre-trained language models on the different modalities such as tables and text (Yin et al., 2020; Herzig et al., 2020, 2021; Katsis et al., 2021; Nan et al., 2021). However, each model trained on a specific input type is incompatible with other modalities and imposes huge constraints on storage efficient systems. For example, in tabular QA (Herzig et al., 2020), the structure of the table is learnt by training additional position (row and column identifiers) embeddings to identify which row and column a table cell belongs to. Multi-modal models (Zhu et al., 2021) can reason over both tables and text by concatenating the textual context and the flattened table, leading to longer input sequences and limiting the length of the context paragraph and the size of the table that can be en-

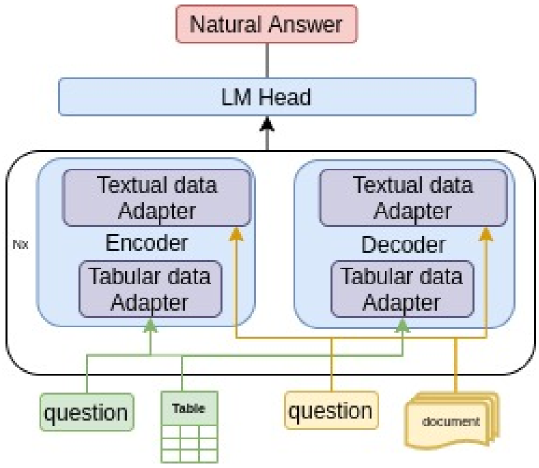


Figure 1: Parameter-efficient transfer learning using modality-specific (table/text) adapters for Abstractive Question Answering

coded. Moreover, they do not explicitly handle encoding complex hierarchical tabular structure in either the model (row and column embeddings) or in the input sequence (ambiguous association of table cells with table headers in complex tables).

To address these challenges, we study parameter-efficient transfer learning for abstractive QA over tables and text. We are motivated to use adapter-layers that inject small bottle-neck layers between frozen pre-trained transformer layers as adapters achieve comparable performance to fine-tuning on a variety of tasks such as multi-lingual translation (Pfeiffer et al., 2020b; Philip et al., 2020; Guo et al., 2020), classification (Houlsby et al., 2019a), language generation (Lin et al., 2020), domain-adaptation in dialogue state tracking and response generation (Hung et al., 2021).

Ablation studies on adapter-layers (Rucklé et al., 2020) on masked language models such as RoBERTa over the GLUE benchmark demonstrate that removing beginning adapter layers leads to a minimal drop in performance. Our task of abstractive QA is more challenging as it involves language generation in addition to natural language understanding (NLU) capability of the model. Further, extending adapter-layer ablation over separate encoders and decoders is non-trivial. Lin et al. (2020) explore the impact of the bottle-neck embedding size for various language generation tasks over n auto-regressive model such as GPT-2 (Radford et al., 2019). Our work deviates from theirs as we focus on language generation from multi-modal (structured or unstructured) input.

We propose a system, named *parameter-efficient abstractive question answering* (PeaQA), which learns to reason over unstructured and structured input using a *single*, *shared* pre-trained language model and modality-specific adapter layers. We also automatically transform hierarchical tables to regular tables to have a uniform representation without breaking association between table cells. In addition, we extend the study of ablating adapter-layers in a multi-modal setting over both encoder and decoder modules.

Our main contributions can be summarized as follows:

(1) We perform parameter-efficient transfer-learning for abstractive question answering multi-modal context consisting of semi-structured tables and unstructured text using only additional 1.5% of trainable parameters for each modality. Our model outperforms existing work by a large margin on tabular QA datasets (FeTaQA and Tablesum) and achieves comparable performance on text QA dataset (NarrativeQA) with significantly less parameters.

(2) We propose using a single, shared pre-trained language model and modality-specific adapter layers for different types of data. To do so, we introduce a 2-step transformation of hierarchical tables to 1-dimensional sequences which not only preserves table-cell association but also produces a uniform representation for our model.

(3) We study the impact of different adapter layers on performance in both encoder and decoder modules and show that beginning adapter layers can be eliminated without significant drop in performance. We also demonstrate that last encoder adapter layers are indispensable and have greater contribution than decoder layers at the same level.

2 Background and Related Work

Tabular question answering. Tabular QA systems aim to answer questions from tabular data. Such systems are required to reason over the structure of the table to perform numeric computations or extract cellular information. The structure of the table is usually encoded in the embedding layer of large language models by introducing table specific position information such as row id and column id. The table can then be flattened into a sequence without losing information about the table structure. This method of representing tables is utilized in (Herzig et al., 2020; Zhu et al., 2021; Katsis et al., 2021). Abstractive QA over tables poses additional challenges of generating natural answers by reasoning and aggregating various discontinuous facts from the table. Abstractive QA over tables has been explored in (Nan et al., 2021; Cheng et al., 2021), where the answer is generated with seq2seq models from the structured context. Nan et al. (2021) train a T5 model (Raffel et al., 2020) over a linearized table where each row is separated by a [SEP] token, whereas Cheng et al. (2021) explore generating answers from complex hierarchical tables using hierarchy-aware symbolic logic over a tree-based representation of the table.

Our work handles hierarchical tables by linearizing them, after which they can be treated as 1-

dimensional sequences with prompt <*context*> concatenated as a prefix, imposing uniformity in text and tabular QA encoding.

Textual question answering. Question Answering over text measures a system's ability to comprehend free-form text in the user question and context passage(s) and predict an answer. The answer predicted can be extractive (Lee et al., 2016; Seo et al., 2016; Rajpurkar et al., 2016; Pearce et al., 2021) in nature where the system identifies short text spans in the context passage to answer the user query or it can be abstractive (Yin et al., 2016; Mitra, 2017; Bauer et al., 2018; Reddy et al., 2019) in nature where the system is required to generate the answer in natural free-form text.

Our work focuses on generative/abstractive Question Answering using large pre-trained seq2seq models. More specifically, we focus on machine reading comprehension aspect of QA where the model is provided with the gold context passages from where the answer is generated.

Transfer learning. Transfer learning techniques such as fine-tuning large pre-trained models for downstream tasks, require a new set of model parameters to be learnt for each new task and domain. To avoid such memory intensive transfer learning methods, adapters have been proposed as a parameter-efficient method of adapting to new domains (Houlsby et al., 2019b; Pfeiffer et al., 2020b). A bottleneck adapter layer is injected after each sub-layer of the transformer. The total number of parameters added at each layer is limited by the size of the bottleneck embedding and reduces the total number of trainable parameters in the transformer. Adapters have been extended to language generation in a variety of generative tasks such as translation, summarization, multi-turn dialogue, and task-oriented natural language generation (Lin et al., 2020).

Our work attempts to reduce the additional parameters further to 0.7% by removing the beginning adapter layers from both encoder and decoder but still achieving comparable results.

3 Approach

Our approach utilizes a shared pre-trained language model across modalities and only learns modality-specific information in the adapter layers. To encode tables using a language model trained only on text imposes a transformation of the 2-dimensional tables to a linearized sequence which implicitly retains the structure information of the original data. The next sections describe the process of transforming hierarchical tables[1] into a linear representation, abstractive question answering using BART and details of the ablation study on adapter layers.

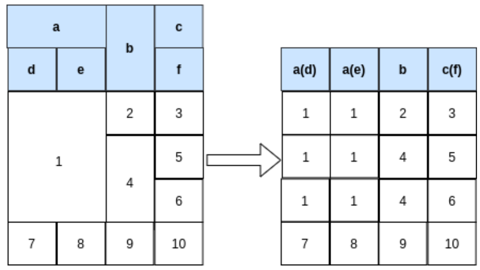


Figure 2: A multi-span table represented as a regular table.

3.1 Representation of tables

QA systems over structured data must parse tables that can be regular or hierarchical. Hierarchical tables can have header cells and body cells spanning across multiple rows and columns. Most existing work on tabular data (Nan et al., 2021; Zhang et al., 2020) is over regular tables. Concurrent to our work, recent work on hierarchical tables (Cheng et al., 2021) uses heuristics to extract hierarchies in tables and uses logical forms to perform operations of selected regions in the table. For our work, we choose to represent all tables uniformly in a 2 step process:

(1) Transformation of a hierarchical table into a regular table.

(2) Linearization of a regular table into a flattened sequence which can be encoded with a language model.

Linearize hierarchical table headers. Hierarchical table headers are linearized into a single row of headers by the following process. A header cell spanning multiple columns is duplicated and split into multiple number of cells. Next, the values of cells in all rows over which this header cell spans are concatenated with the entire split. For example, as shown in Figure 2, table header a spans across 2 other header columns d and e of the next header row. We first split header cell a into 2 columns a, a. Next, we concatenate the values of the next row over which the span is present, i.e., values d and e, to a linearized header $a(d)$, $b(e)$. Repeating this process over all the table header rows flattens the hierarchical header into a sequential one. In our running example, this process will yield the linear

[1]The tables in the datasets we studied contains only hierarchical column headers.

header $a(d)$, $a(d)$, b, $e(f)$.

Linearizing table body. Multi-span cells in the table body are parsed differently than the table header. All table body cells are associated with one or multiple header cells depending on whether the cell spans across one or multiple columns. For example, in Figure 2, cell at position $(1, 1)$ with value 1 is associated with 2 header columns $a(d)$ and $a(e)$, whereas the cell at position $(1, 2)$ is associated with 1 header column b. Cells which spans across multiple rows are associated with all the spanned rows in that column. For example, the cell with value 4 spans across 2 rows and can be treated as present in both rows in separate cells. This process leads to a regular table (Nan et al., 2021) which can then be interpreted as a sequence of keys (table headers) with associated values (table body). We flatten the regular table in row-major form, concatenating rows sequentially. Each row is a sequence of *(key, value)* pairs where a key is a column header and the value is the cell value of that column. The table in our running example is flattened to $a(d)$: 1 $b(e)$: 1 b : 2 $c(f)$: 3 $a(d)$: 1 $b(e)$: 1 b: 4 $c(f)$: 5 $a(d)$: 1 $b(e)$: 1 b: 4 $c(f)$: 6 $a(d)$: 7 $b(e)$: 8 b: 9 $c(f)$: 10.

3.2 Uniform representation of text and tables for abstractive question answering

We focus on encoder-decoder models for the task of abstractive question answering. We use BART (Lewis et al., 2019) encoder-decoder architecture which comprises of a bidirectional encoder and an auto-regressive decoder. The input sequence consists of the question, the context title and context sequence preceded with prompts indicating the beginning of the each sub-sequence. Formally, the input sequence is represented as *<question>* q_0 q_1 $\ldots$ q_m *<title>* t_1 t_2 $\ldots t_p$ *<context>* c_0 c_1 $\ldots$ c_n, where q_i is the i-th question token, t_j is the j-th title token, and c_k is the k-th context token. The context can either be a text passage or a flattened table as explained in the previous section. The parameters of the pre-trained BART model are frozen during training. Modality specific adapter layers added to the model are trained on either tabular context or textual context to generate natural answers.

3.3 Ablation study: Adapter pruning

Adapter-layer pruning has been explored on the GLUE benchmark in (Rücklé et al., 2020) which demonstrates that removing adapter layers from the beginning transformer layers leads to minimal performance drop. For encoder-decoder architectures, we hypothesize that this phenomenon should be observed on both the encoder and decoder modules. However, it is non-trivial how the adapter-layers in the encoder and decoder modules contribute to performance and whether the adapter layers of the encoder and decoder module have equal impact on the performance. To measure the impact of the adapter layers in different modules, we perform adapter ablation in both the encoder and decoder. We progressively remove adapter layers from both the encoder and decoder modules starting from the beginning layers and analyze performance drop caused by each successive elimination. We report our findings in Section 5. We observe minimal performance drop until the last few adapter-layers indicating that these layers contribute the most to task-specific representations.

Adapter-tune		**#Trainable parameters**
Encoder adapters removed	**Decoder adapters removed**	
–	–	$6,343,680$ (1.56%)
0	12	$5,815,040$ (1.43%)
$0-1$	$12-13$	$5,286,400$ (1.30%)
$0-2$	$12-14$	$4,757,760$ (1.17%)
$0-3$	$12-15$	$4,229,120$ (1.04%)
$0-4$	$12-16$	$3,700,480$ (0.91%)
$0-5$	$12-17$	$3,171,840$ (0.78%)
$0-6$	$12-18$	$2,643,200$ (0.65%)
$0-7$	$12-19$	$2,114,560$ (0.52%)
$0-8$	$12-20$	$1,585,920$ (0.39%)
$0-9$	$12-21$	$1,057,280$ (0.26%)
$0-10$	$12-22$	$528,640$ (0.13%)
$0-11$	$12-22$	$264,320$ (0.07%)
Fine-tune		$406,291,456$ (100%)

Table 1: Trainable parameters in the encoder and decoder. Encoder adapter layers are numbered from $0-11$ and decoder adapter layers are numbered from $12-23$. $x-y$ implies all adapter layers from x to y inclusive.

4 Experimental Setup

We seek to answer the following research questions with our experiments: (RQ1) How does adapter–tuning affect performance and can we achieve comparable results to fine-tuning in the context of multi-modal input? (RQ2) Do all adapter layers across the encoder and decoder contribute equally to performance across tasks/modalities?

Method	Model	Training	Rouge-1	Rouge-2	Rouge-L	BLEU
Tablesum (Lin et al., 2020)	CopyNet	Fine-tune	0.041	0.012	0.030	0.80
	GPT2		0.272	0.073	0.200	5.35
	T5		0.362	0.143	0.276	**10.43**
Ours (Pea-QA)	bart-large	Fine-tune	**0.410**	**0.188**	**0.32**	6.46
		Adapter-tune	0.391	0.183	0.309	6.64

Table 2: Results: Scores calculated on the test split of 20% random split of Tablesum.

Method	Model	Training	Rouge-1	Rouge-2	Rouge-L	BLEU
FeTaQA (Nan et al., 2021)	T5-small	Fine-tune	0.55	0.33	0.47	21.60
	T5-base		0.61	0.39	0.51	28.14
	T5-large		0.63	0.414	0.53	30.54
Ours (Pea-QA)	bart-large	Fine-tune	0.632	0.415	0.534	30.81
		Adapter-tune	**0.651**	**0.436**	**0.553**	**33.45**

Table 3: Results: Scores calculated on the test split of FeTaQA.

Method	Model	Training	Rouge-1	Rouge-2	Rouge-L	BLEU
Masque (Nishida et al., 2019)	–	Fine-tune	–	–	**0.547**	–
Ours (Pea-QA)	bart-large	Fine-tune	0.518	0.268	0.515	21.07
		Adapter-tune	0.51	0.27	0.50	20.08

Table 4: Results: Scores calculated on the test split of NarrativeQA. We use sacreBLEU [2] to measure BLEU score

4.1 Datasets

Tabular datasets. For abstractive QA over tables we use the Tablesum (Zhang et al., 2020) and FeTaQA (Nan et al., 2021) datasets. Tablesum consists of 200 unique Wikipedia tables over which questions and abstractive answers are manually annotated. Tablesum contains natural answers as well as summaries to tables as abstractive answers. 35% of samples are questions over complex hierarchical tables. For our experiments, we randomly split the samples into a 80%-20% split as the training and validation set.[3] FeTaQA is a larger abstractive tabular QA dataset consisting of question and free-form answers over $10,330$ regular tables. The dataset consists of $7,326$ samples in the training set, $1,001$ in the validation set, and $2,003$ in the test set. FeTaQA consists of human-annotated answers containing explanations involving entities and relations.

Text dataset. We train adapter layers for textual context on the NarrativeQA dataset (Kočiský et al., 2018). NarrativeQA is a complex abstractive question answering dataset over stories. The dataset contains $32,747$ samples in the training set, $3,461$ samples in the validation set and $10,557$ samples in the test set. For our task, we have selected the input context passage to be the human annotated *summary* of each sample which is the Wikipedia page summary of the story and represented as a paragraph. The input to the seq2seq model is the *question*, *title* and *summary* of each passage and the target is the abstractive answer associated with each sample. For samples with multiple answers, we treat each answer and the associated question and context as an independent sample.

Tablesum dataset's target answers are of summary-type where many questions require summarizaing the sections of the table. This leads to longer target sequences as observed in the maximum target length of 1579 compared to 338 for FeTaQA and 224 for NarrativeQA. Tablesum also contains larger tables of maximum 155 rows compared to 34 rows of FeTaQA. A summary of all datasets is presented in Appendix C.

4.2 Experiments

We perform all our experiments on a BART-Large variant of the BART model.[4] We add bottle-neck

[3]The original Tablesum dataset do not have Dev and Test splits. We choose to use the random Dev split for evaluaring our models due to limited number of samples in the dataset

[4]The results can be reproduced using code at `https://anonymous.4open.science/r/`

adapter layers from the Houlsby adapter configuration (Houlsby et al., 2019a). Each adapter layer has a bottle-neck embedding size of 64. We also fine-tune the model on each dataset for comparison. We sweep learning rates from $\{8e^{-4}, 6e^{-4}, 3e^{-4}, 1e^{-4}, 5e^{-5}, 4^{-5}, 3e^{-5}, 2e^{-5}, 1e^{-5}\}$ and select a learning rate of $6e^{-4}$ to train the adapter layers for the tabular QA datasets Tablesum and FeTaQA and use $1e^{-1}$ to train text QA dataset NarrativeQA. We select $4e^{-5}$ for fine-tuning on Tablesum, $8e^{-4}$ on FeTaQA datasets and $2e^{-5}$ to fine-tune NarrativeQA. We use a batch size of 4 and gradient accumulation of 8 to emulate an effective batch size of 32. The maximum prediction sequence length is set to 200 for the tabular QA datasets and to 100 for the text QA dataset. We train the model on each dataset for 15 epochs and evaluate on Rouge-2, Rouge-L and sacreBLEU metrics.[5] A summary of hyper-parameters are mentioned in Appendix B.

5 Results

We conduct experiments to answer the research questions as described in Section 4. The experimental results to answer our question are explained in the following sections.

5.1 Parameter efficient adapter-tuning

We address (RQ1) by comparing the performance of adapter-tuning to fine-tuning across the three datasets described in Section 4.1. The results of the experiments are shown in Table 2 for the Tablesum dataset, Table 3 for the FeTaQA dataset, and Table 4 for the NarrativeQA dataset.

We observe that for tabular QA, we outperform the state-of-art models. On the Tablesum dataset, we evaluate the model performance on 20% random split which we keep aside during training. We observe that the model outperforms the existing work by a large margin on both adapter-tuning and fine-tuning. Fine-tuning on Tablesum achieves 2% higher performance compared to adapter-tuning on Rouge-L,[6] 1% on Rouge-1 and is marginally higher on Rouge-2 scores. For the FeTaQA dataset, our model achieves higher gains in performance on adapter-tuning compared to fine-tuning. This can be attributed to catastrophic forgetting (French, 1999; Kirkpatrick et al., 2017; Chen et al., 2020) induced by differences in the distribution of downstream task data. Fine-tuning on FeTaQA achieves comparable performance with slight performance gain compared to the state-of-the-art model. For text QA, on the NarrativeQA dataset, adapter-tune performs comparable to fine-tune as shown in the F-scores of Table 4. Our fine-tuning results on NarrativeQA are lower than state-of-the-art models trained with sophisticated reasoning architectures, as our focus was primarily on comparing fine-tuning and adapter-tuning. We conclude that adapter-tuning achieves comparable performance to fine-tuning when using a standard pre-trained language model.

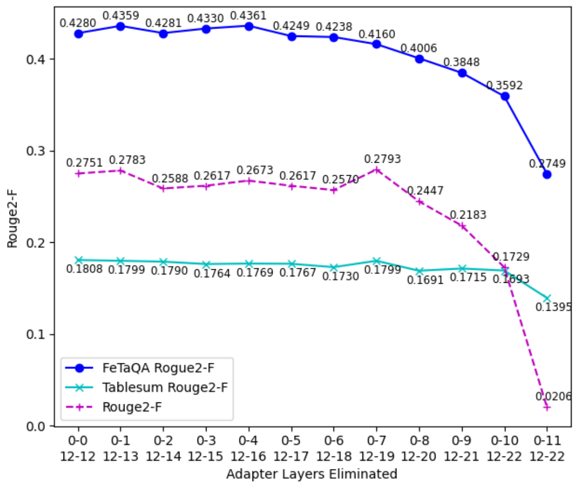


Figure 3: Adapter layer ablation Rouge-2 scores. X-axis represents encoder-adapter layers $(0-11)$ and decoder adapter layers $(12-23)$ deleted progressively. $x-y$ implies all adapter layers from x to y inclusive. Each point represents the Rouge-2 F-score for the model configuration that has encoder layers p to q deleted and decoder layers r to s deleted for x-tick $\frac{(x-y)}{(r-s)}$.

5.2 Ablation of adapter layers

We study (RQ2) by ablating adapter layers in both the encoder and decoder modules. We do so to study the impact of each adapter layer in the performance across different modality inputs. We progressively eliminate adapter layers from both the encoder and decoder starting from the first adapter layer from both the modules and finally deleting all layers from both the encoder and decoder. This leads to 12 experiments corresponding to 12 encoder adapter layer and 12 decoder adapter layer. We number the encoder adapter layers from $0-11$ and the decoder adapter layers from $12-23$. We measure the performance of the models using Rouge-2, Rouge-L and sacreBLEU scores. The

Pea-QA-0717/README.md

[5] We use Adapter-hub library to conduct all our experiments (Pfeiffer et al., 2020a)

[6] Rouge scores are calculated using https://pypi.org/project/rouge-score/

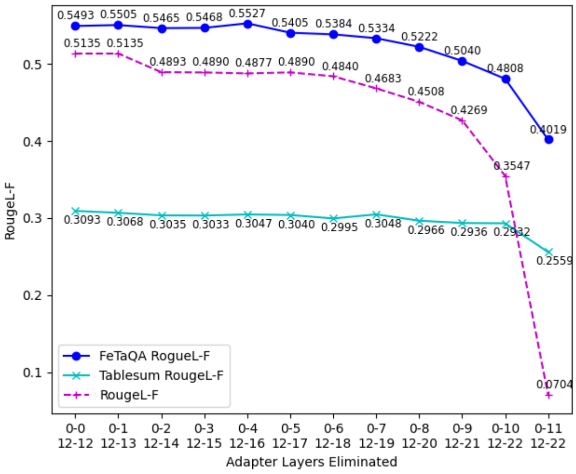


Figure 4: Adapter layer ablation Rouge-L scores. X-axis represents encoder-adapter layers $(0-11)$ and decoder adapter layers (12-23) deleted progressively. $x-y$ implies all adapter layers from x to y inclusive. Each point represents the Rouge-L F-score for the model configuration that has encoder layers p to q deleted and decoder layers r to s deleted for x-tick $\binom{(x-y)}{(r-s)}$.

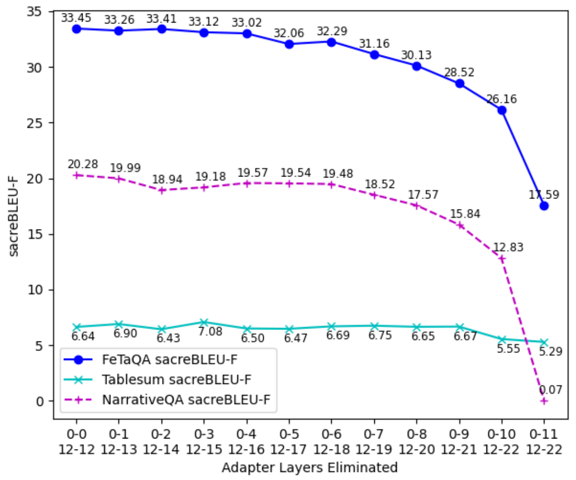
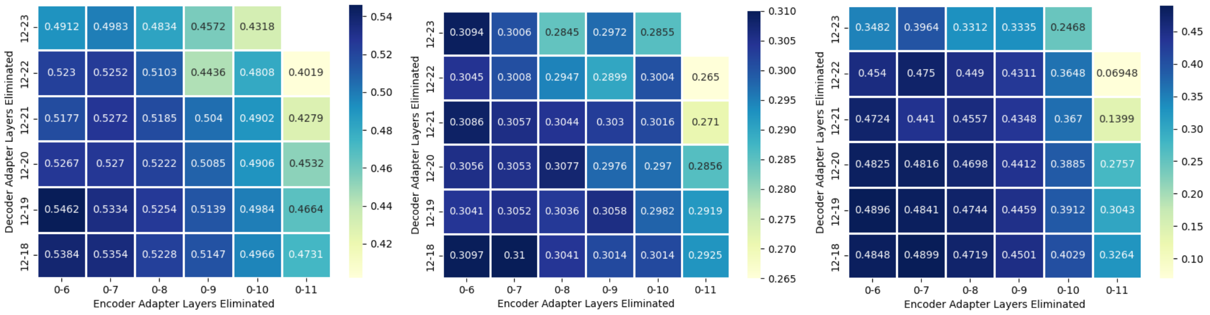


Figure 5: Adapter layer ablation sacreBLEU F-scores. X-axis represents encoder-adapter layers $(0-11)$ and decoder adapter layers $(12-23)$ deleted progressively. $x-y$ implies all adapter layers from $xtoy$ inclusive. Each point in the plot represents the F-score for a model configuration that has encoder layers p to q deleted and decoder layers r to s deleted for x-tick $\binom{(x-y)}{(r-s)}$.

F-scores for each dataset (NarrativeQA, Tablesum, FeTaQA) are shown in Figure 3, 4, and 5, respectively. We observe that as more adapter layers are eliminated, the performance drops across all datasets. However, the performance drop is minimal until the last adapter layers are also deleted. A steep drop is observed when the last few adapter layers from the encoder and decoder are deleted. This inflection point varies across dataset but is limited to the last 2 layers of the encoder and decoder. For the NarrativeQA dataset, this point is when all layers till the second last adapter layer from both the encoder and decoder are deleted. For the FeTaQA and Tablesum dataset, the performance drops sharply only when the last encoder and decoder layers are removed.

To analyze contribution of the i-th adapter layer of the encoder and the i-th layer of the decoder to the performance, we ablate over different combinations of eliminating adapter layers from the later half of both encoder and decoder, i.e., successively removing adapter layers $0-6$, $0-7$, $0-8$, $0-9$, $0-10$, $0-11$ from the encoder and adapter layers $12-18$, $12-19$, $12-20$, $12-21$, $12-22$, $12-23$ from the decoder (decoder layers are numbered from $12-23$). This leads to 36 different configurations of ablation study where a configuration $(p-q, r-s)$ represents all adapters from the p-th to the q-th layer in the encoder and from the r-th to the s-th in the decoder has been removed. The results are shown in Figure 6.

We observe that performance remains comparable as we progressively eliminate adapter layers from the encoder and decoder until layers at the end are removed. The drop in performance is steeper when we remove the last encoder and decoder adapter layers depicted towards the top-right corner of Figures 6a, 6b, and 6c. This reinforces the assumption that the last adapter layers learns most of the domain/task specific information. We also observe that the last adapter layers of the encoder and decoder contribute differently to performance. Removing the last encoder adapter layer (depicted in the last column $0-11$ across all rows) leads to a massive drop in all scores in-spite of decoder adapter layers. This pattern is observed across all datasets. This indicates that the last encoder adapter layer is indispensable for learning domain-specific knowledge. Dropping all decoder adapter layers till the last (represented by top row $12-23$) is comparable to dropping all adapter-layers till the 2-nd last encoder adapter layer (represented by column $0-10$). This pattern is observed for all ablation configurations till last 4 layers in the decoder (top 4 rows $12-20$, $12-21$, $12-22$, $12-23$ in Figure 6). The drop in performance stabilizes as we retain more end adapter layers and remove only beginning adapter layers. We observe that retaining just 50% of adapter layers by removing the beginning half of the adapters from both the encoder and decoder further increases parameter efficiency by introducing only 0.7% parameters as summarized in Table 1



(a) FeTaQA Rouge-L scores (b) Tablesum Rouge-L scores (c) NarrativeQA Rouge-L scores

Figure 6: Adapter layer ablation Rouge scores. X-axis represents range of encoder adapter layers deleted, Y-Axis represents range of decoder adapter layers deleted. $x - y$ implies all adapter layers from x to y inclusive. There are 36 model ablation configurations displayed. The ablation starts from 0 to 6 encoder adapter layers removal and 12 to 18 decoder adapter layer removal represented by the bottom left cell $((0-6),(12-18))$ and progressively increases deletion of encoder adapter layers along the X-axis and decoder adapter layers along the Y-axis.

but without significant compromise to performance. Further parameters can be removed to only 0.52% (removing $(0-7, 12-19)$) trainable parameters with only minor drop in performance. As an extreme case of performance-parameter trade-off, all but the last adapter layer of the decoder (2-nd last row in Figure 6) can be removed with any of the encoder configurations of $0-7$, $0-8$ and $0-9$.

5.3 Case studies

In Appendix F we detail a number of case studies. The main insights gained from these case studies are: (1) The mean length of predicted sequences is 193.27 for Tablesum dataset while that of target sequences is 259.76. The adapter-tuned model predicts shorter sequences than the target as observed in the case-study and in lower BLEU scores (precision) compared to the Rouge-L (recall) scores listed in Table 2 and ablation results in Figure 3, 4, and 5. This may be due to brevity penalty of BLEU. (2) The adapter-tuned model performs best on FeTaQA with gradual decrease in Rouge and BLEU across more ablations; the dataset has similar length targets in both training and test sets and scores are more stable across different configurations. (3) NarrativeQA samples contains diverse targets from very short phrases to longer sequences with a maximum target length of 224 characters. Many fact-based correct predictions have words reordered but reproduced correctly.

6 Conclusion

We study parameter-efficient transfer learning over tables and text in the context of abstractive question answering using small bottle-neck adapter layers. We achieve comparable performance to fine-tuning with only 1.5% training parameters across each modality. We propose a transformation from hierarchical tables to regular ones which can then be flattened to a 1-dimensional sequence for use in the shared pre-trained language model. This leads to a uniform pre-trained model with frozen weights across all modalities. We outperform results of state-of-the-art models on tabular QA datasets such as Tablesum and FeTaQA, and achieve comparable performance on the text QA dataset NarrativeQA. We extend an ablation study on adapter layers to encoder-decoder setting to study the impact of adapter layers from the respective modules. Encoder and decoder adapter layers at the same level contribute deferentially to performance. More specifically, we demonstrate that the adapter layers from the end of the encoder is indispensable to performance and contributes significantly more to encoding modality specific information than decoder adapter layers of the same level. Our results are useful for exploring scalability of QA models in memory constrained situations where comparable performance can be achieved with storing only one copy of a uniform language model while scaling across modalities using light-weight adapters.

One of the limitations of our work is that we only consider one way of representing the structure of the table by a hierarchical transformation process that linearizes the 2-dimensional table to a 1-dimensional sequence. Alternatively, one can encode the table structure in model embeddings using structure specific row and column ids. This is left as future work. We also do not perform explicit logical reasoning and cell aggregation over the tables. This might lead to better explainability of the model and is to be explored as future work.

References

Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C. Lawrence Zitnick, Dhruv Batra, and Devi Parikh. 2016. VQA: Visual question answering. *arXiv preprint arXiv:1505.00468*.

Lisa Bauer, Yicheng Wang, and Mohit Bansal. 2018. Commonsense for generative multi-hop question answering tasks. In *EMNLP*.

Alice Oh Carnegie and Alice H. Oh. 2000. Stochastic language generation for spoken dialogue systems. In *In Proc. of the ANLP/NAACL 2000 Wrkshp. on Conversational Systems*, pages 27–32.

Sanyuan Chen, Yutai Hou, Yiming Cui, Wanxiang Che, Ting Liu, and Xiangzhan Yu. 2020. Recall and learn: Fine-tuning deep pretrained language models with less forgetting. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7870–7881, Online. Association for Computational Linguistics.

Zhoujun Cheng, Haoyu Dong, Zhiruo Wang, Ran Jia, Jiaqi Guo, Yan Gao, Shi Han, Jian-Guang Lou, and Dongmei Zhang. 2021. Hitab: A hierarchical table dataset for question answering and natural language generation. *arXiv preprint arXiv:2108.06712*.

Yashar Deldjoo, Johanne R. Trippas, and Hamed Zamani. 2021. Towards multi-modal conversational information seeking. In *SIGIR '21: The 44th International ACM SIGIR Conference on Research and Development in Information Retrieval, Virtual Event, Canada, July 11-15, 2021*, pages 1577–1587. ACM.

Robert M. French. 1999. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135.

Junliang Guo, Zhirui Zhang, Linli Xu, Hao-Ran Wei, Boxing Chen, and Enhong Chen. 2020. Incorporating bert into parallel sequence decoding with adapters. In *Advances in Neural Information Processing Systems*, volume 33, pages 10843–10854. Curran Associates, Inc.

Jonathan Herzig, Thomas Müller, Syrine Krichene, and Julian Eisenschlos. 2021. Open domain question answering over tables via dense retrieval. *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

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

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019a. Parameter-efficient transfer learning for NLP. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 2790–2799. PMLR.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019b. Parameter-efficient transfer learning for NLP. *arXiv preprint arXiv:1902.00751*.

Drew A. Hudson and Christopher D. Manning. 2019. GQA: A new dataset for real-world visual reasoning and compositional question answering. *arXiv preprint arXiv:1902.09506*.

Chia-Chien Hung, Anne Lauscher, Simone Paolo Ponzetto, and Goran Glavaš. 2021. DS-TOD: Efficient domain specialization for task oriented dialog. *arXiv preprint arXiv:2110.08395*.

Yannis Katsis, Saneem Chemmengath, Vishwajeet Kumar, Samarth Bharadwaj, Mustafa Canim, Michael Glass, Alfio Gliozzo, Feifei Pan, Jaydeep Sen, Karthik Sankaranarayanan, and Soumen Chakrabarti. 2021. AIT-QA: Question answering dataset over complex tables in the airline industry. *arXiv preprint arXiv:2106.12944*.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. 2017. Overcoming catastrophic forgetting in neural networks. *arXiv preprint arXiv:1612.00796*.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436*.

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

Zhaojiang Lin, Andrea Madotto, and Pascale Fung. 2020. Exploring versatile generative language model via parameter-efficient transfer learning. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 441–459, Online. Association for Computational Linguistics.

Rajarshee Mitra. 2017. A generative approach to question answering. *arXiv preprint arXiv:1711.06238*.

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. 2021. Fetaqa: Free-form table question answering. *arXiv preprint arXiv:2104.00369*.

Kyosuke Nishida, Itsumi Saito, Kosuke Nishida, Kazutoshi Shinoda, Atsushi Otsuka, Hisako Asano, and Junji Tomita. 2019. Multi-style generative reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2273–2284, Florence, Italy. Association for Computational Linguistics.

Kate Pearce, Tiffany Zhan, Aneesh Komanduri, and Justin Zhan. 2021. A comparative study of transformer-based language models on extractive question answering. *arXiv preprint arXiv:2110.03142*.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. Adapterhub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP 2020): Systems Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online. Association for Computational Linguistics.

Jerin Philip, Alexandre Berard, Matthias Gallé, and Laurent Besacier. 2020. Monolingual adapters for zero-shot neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4465–4470, Online. Association for Computational Linguistics.

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

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.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Owen Rambow, Srinivas Bangalore, and Marilyn Walker. 2001. Natural language generation in dialog systems. In *Proceedings of the First International Conference on Human Language Technology Research*.

Adwait Ratnaparkhi. 2002. Trainable approaches to surface natural language generation and their application to conversational dialog systems. *Computer Speech & Language*, 16(3):435–455. Spoken Language Generation.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. CoQA: A conversational question answering challenge. *Transactions of the Association of Computational Linguistics (TACL)*.

Andreas Rucklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2020. Adapterdrop: On the efficiency of adapters in transformers. *arXiv preprint arXiv:2010.11918*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Svitlana Vakulenko, Kate Revoredo, Claudio Di Ciccio, and Maarten de Rijke. 2019. QRFA: A data-driven model of information-seeking dialogues. In *ECIR 2019: 41st European Conference on Information Retrieval*, pages 541–557. Springer.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically conditioned LSTM-based natural language generation for spoken dialogue systems. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721. Association for Computational Linguistics.

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. 2016. Neural generative question answering. *Proceedings of the Workshop on Human-Computer Question Answering*.

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

Shuo Zhang, Zhuyun Dai, Krisztian Balog, and Jamie Callan. 2020. Summarizing and exploring tabular data in conversational search. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

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

APPENDICES

We provide further details on training hyper-parameters for adapter-tuning (Appendix A), training hyper-parameters for fine-tuning (Appendix B), statistics of the datasets used (Appendix C), Rouge-2 scores for an encoder-decoder adapter layer ablation study (Appendix D), Bleu scores for an encode-decoder adapter layer ablation study (Appendix E), example outputs of adapter-tuned bart-large model on Tablesum dataset, FeTaQA dataset and NarrativeQA dataset. Appendix F includes examples of predictions on the 3 datasets.

A Training Hyper-Parameters for Adapter-Tuning

The hyper-parameters for adapter-tuning for Tablesum dataset, FeTaQA dataset and NarrativeQA dataset are listed as in Table 5.

Dataset	Hyper-parameters	Value
Tablesum	learning rate	6e-4
	scheduler	linear
	batch size	32
	sequence length	200
	seed	6
	training epochs	15
FeTaQA	learning rate	6e-4
	scheduler	linear
	batch size	32
	sequence length	100
	seed	6
	training epochs	15
NarrativeQA	learning rate	1e-4
	scheduler	linear
	batch size	32
	sequence length	50
	seed	6
	training epochs	15

Table 5: Hyper-parameters for adapter-tuning.

B Training Hyper-Parameter for Fine-Tuning

The hyper-parameters for fine-tuning on Tablesum dataset, FeTaQA dataset and NarrativeQA dataset are listed in Table 6.

Dataset	Hyper-parameters	Value
Tablesum	learning rate	4e-5
	scheduler	linear
	batch size	32
	sequence length	200
	seed	6
	training epochs	15
FeTaQA	learning rate	8e-4
	scheduler	linear
	batch size	32
	sequence length	100
	seed	6
	training epochs	15
NarrativeQA	learning rate	2e-5
	scheduler	linear
	batch size	32
	sequence length	50
	seed	6
	training epochs	15

Table 6: Hyper-parameters for fine-tuning.

C Dataset Statistics

Statistics of the 3 datasets, i.e., Tablesum, FeTaQA and NarrativeQA are listed in Table 7. Tablesum has the longest answer length. The answers are summary-like, often, describing aspects of the table contents. FeTaQA dataset contains answers of mostly single sentences and targeted towards specific facts asked in the question. NarrativeQA datasets focuses on questions from stories. The answer lengths vary from single words to long sentences. For the tabularQA dataset, Tablesum contains larger tables than FeTaQA dataset even though it is limited to 200 unique tables over which questions are asked. FeTaQA dataset's tables contain more columns on an average compared to Tablesum. Examples of samples can be found in Appendix F in Table 8 for Tablesum dataset, in Table 9 for FeTaQA dataset, and in Table 10 for NarrativeQA dataset.

Tablesum	
Domain	Open
Modality	Table
Table-type	Regular
Training samples	798
Validation samples	200
Test samples	–
Max question length	114
Max target length	$1,579$
Max table row	155
Max table column	8
FeTaQA	
Domain	Open
Modality	Table
Table-type	Hybrid
Training samples	$7,326$
Validation samples	$1,001$
Test samples	$2,003$
Train max question length	165
Train max target length	338
Train max table rows	34
Train max table columns	30
Val max question length	182
Val target length	325
Val max table rows	34
Val max table columns	22
Test max question length	193
Test max target length	295
Test max table lows	34
Test max table columns	22
NarrativeQA	
Domain	Stories
Modality	Text
Training samples	$65,494$
Validation samples	$6,922$
Test samples	$21,114$
Train max question length	175
Train max target length	171
Train max context length	$6,045$
Val max question length	158
Val target length	187
Val max context length	$6,033$
Test max question length	$1,220$
Test target length	224
Test max context length	$6,090$

Table 7: Dataset statistics.

D Encoder-Decoder Adapter layer Ablation Rouge-2 Scores

Ablation results (Rouge-2 F-scores) of 36 configurations of adapter layers deleted from the later half of the encoder and decoder. Deleting the last encoder adapter layers leads to massive drop in performance as observed in last 3 columns of Figures 7, 8 and 9. However, deleting the last decoder adapter layers results in better performance in comparison to the encoder layers at the same level as observed from the top 3 rows of Figures 7, 8 and 9.

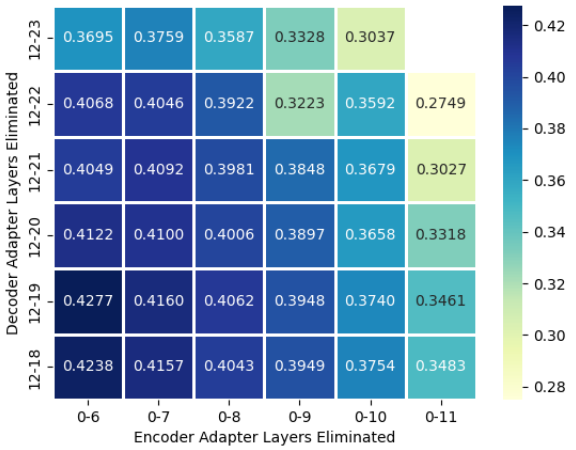


Figure 7: FeTaQA Rouge-2 scores. X-axis represents encoder adapter layers $(0-11)$ deleted, Y-axis represents decoder adapter layers $(12-23)$ deleted.

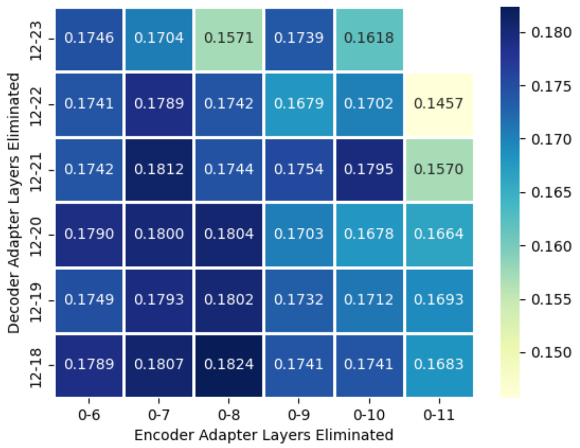


Figure 8: Tablesum Rouge-2 scores. X-axis represents encoder adapter layers $(0-11)$ deleted, Y-axis represents decoder adapter layers $(12-23)$ deleted.

E Encode-Decoder Adapter layer Ablation sacreBLEU Scores

Ablation results (sacreBLEU F-scores) of 36 configurations of adapter layers deleted from the later

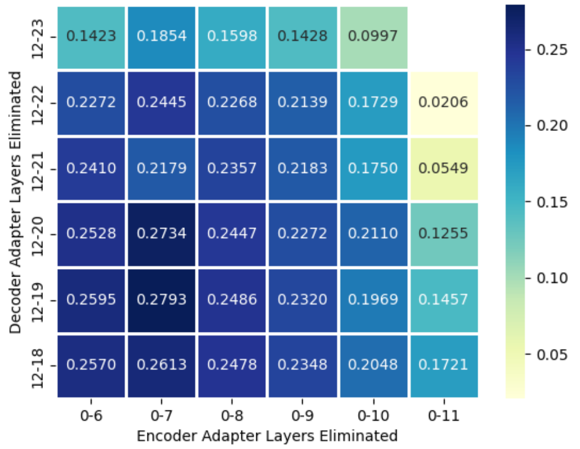


Figure 9: NarrativeQA Rouge-2 scores. X-axis represents encoder adapter layers(0-11) deleted, Y-axis represents decoder adapter layers(12-23) deleted.

half of the encoder and decoder. Deleting the last encoder adapter layers leads to massive drop in performance as observed in last 3 columns of Figures 10, 11 and 12. However, deleting the last decoder adapter layers results in less performance drop in comparison to the encoder layers at the same level as observed from the top 3 rows of Figures 7, 8 and 9.

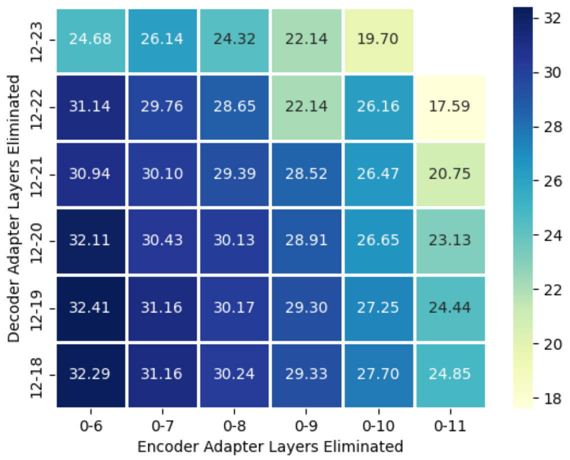


Figure 10: FeTaQA sacreBLEU-F scores. X-axis represents encoder adapter layers $(0-11)$ deleted, Y-axis represents decoder adapter layers $(12-23)$ deleted.

F Examples

We present a few examples of predictions from tabular QA datasets (FeTaQA and Tablesum) and text QA (NarrativeQA). Each row of Tables 8, 9 and 10 depicts the input sub-sequences to the system such as *Question*, *title*, context in the form of *Flattened table* for tabular QA or *Text summary* for text QA, the ground-truth *Target* abstractive answer and the *Prediction* by the adapter-tuned model.

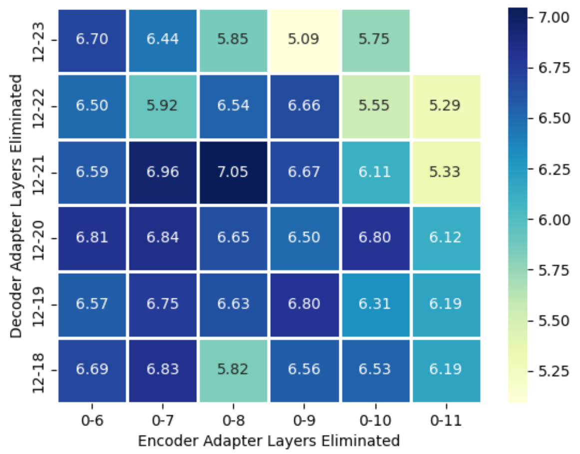


Figure 11: Tablesum sacreBLEU scores. X-axis represents encoder adapter layers $(0-11)$ deleted, Y-axis represents decoder adapter layers $(12-23)$ deleted.

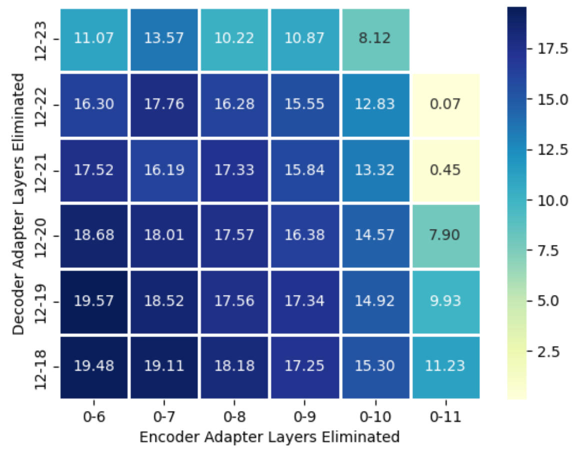


Figure 12: NarrativeQA sacreBLEU-F scores. X-axis represents encoder adapter layers $(0-11)$ deleted, Y-axis represents decoder adapter layers $(12-23)$ deleted.

For Tablesum samples, the abstractive answer is long summary type. The questions can vary from fact-based queries as depicted in the 1-st sample to summarization of table cells as depicted in the 2-nd and 3-rd samples of Table 8. The predictions of the adapter-tuned model are usually shorter than the target as observed by the lower BLEU scores due to brevity penalty for Tablesum dataset as listed in Table 2. The Rouge scores are also lower in comparison to the other 2 datasets. This indicates that the target contains n-grams not present in the predicted answer. This pattern also persists across the ablation studies as observed from similar trend in the Rouge scores in Figure 3, 4, 8, and 6b, and BLEU scores are in Figure 11.

The FeTaQA dataset mostly contains single sentence abstractive answers as ground-truth as observed from the *Target* of Table 9. Our model performs best over FeTaQA dataset generating similar length answers as the ground-truth. This is indicated in more stable Rouge and BLEU scores as depicted in Table 3. The Rouge scores can be observed in Figure 3, 4, 7, and 6a, and BLEU scores are in Figure 10.

NarrativeQA dataset also contains shortest abstractive answers compared to the other 2 datasets as observed in Table 7. The maximum answer length for the test set is 224 compared to 295 for test split of FeTaQA and $1,579$ for Tablesum. Many target answers of NarrativeQA are short phrases as depicted in the last row of Table 10. Many predictions of the adapter-tuned model change the order of words even though the facts are reproduced correctly as shown in the 1-st and 2-nd rows of Table 9.

Tablesum Examples

Question: What are the songs included in 21 & Over?
Title: Track listing
Flattened table: No.: 1. <,> Title: "Likwit" (featuring King Tee) <,> Writer(s): SmithBrooksRobinsonRoger McBride <,> Producer(s): E-SwiftTha Alkaholiks[a] <,> Length: 3:27 <,> No.: 2. <,> Title: "Only When I'm Drunk" <,> Writer(s): ...
Target: The table is about listing of tracks related to group "21 & Over" In this track listing one can find the title of the track, Producer name, Performer name and Length of the performance.
Prediction: I found a table of the track listing of 21 & Over. It is a compilation of songs by E-Swift and King Tee. The songs include "Likwit", "Only When I'm Drunk", "Last Call", "Soda Pop", "Make Room", "Bullshit", "Can't Tell Me Shit" and "Last Night". The songs are all written by SmithBrooksRobinsonRoger McBride.

Question: Tell me about the games that the Badgers lost in 1981
Title: 1981 Wisconsin Badgers football team schedule
Flattened table: Date: September 12 <,> Opponent: No. 1 Michigan <,> Rank: <,> Site: Camp Randall StadiumMadison, WI <,> Result: W 21–14 <,> Attendance: 68,733 <,> Date: September 19 <,> Opponent: No. 9 UCLA* <,> Rank: No. 20 <,> Site: Camp Randall StadiumMadison, WI <,> Result: L 13–31 ...
Target: In 1981 the Wisconsin Badgers lost to UCLA 13-31 on September 19th for the first loss of the season. October showed two losses, both away games, at Michigan State and again at Illinois with just a 2 point deficit. In November Iowa stomped them on the field winning with a ten point lead, and their final loss in December was to Tennessee.
Prediction: I found a table of the 1981 wisconsin badgers football team schedule. it shows the games that the badgers lost in 1981.

Question: Can you introduce to me Richard Strauss's operas?
Title: List of operas by Richard Strauss
Flattened table: Period[a]: 1876 <,> Title: Der Kampf mit dem Drachen <,> Genre: ? <,> Act(s): 1 act <,> Librettist: Körner <,> Premiere (Date): Unperformed <,> Premiere (Venue): Unperformed <,> Op.[1]: – <,> TrV[1]: 44 <,> Av: 206 <,> Period[a]: 1892–93 <,> Title: Guntram(revised 1940) <,> Genre: Opera[3] <,> Act(s): 3 acts <,> Librettist: Strauss <,> Premiere (Date): 10 May 1894;revised version: 29 October 1940 <,> Premiere (Venue): Weimar, Grossherzogliches Hoftheater (both versions)...
Target: I found a table called List of operas by Richard Strauss. 18 operas have been made by the composer Richard Strauss. Strauss has operas from 1876 all they way through 1964.
Prediction: I found a table of Richard Strauss's operas. It is a list of operas by Richard Strauss from 1876 to 1940. There are 12 operas in the list. The first one is Der Kampf mit dem Drachen (1876) and the last one is Elektra (1908).

Question: Tell me about the teams and their performance in the 2007 San Jose Grand Prix Race
Title: 2007 San Jose Grand Prix at Redback Raceway
Flattened table: 1: 2 <,> [United_Kingdom|]: [France|] <,> [Justin_Wilson_(racing_driver)|Justin Wilson]: [S%C3%A9bastien_Bourdais|Sébastien Bourdais] <,> [RSPORTS|RSPORTS]: [N/H/L_Racing|N/H/L Racing] <,> 49.747:...
Target: I see the top 17 results. Within the 17, the best and worst times ranged between 40.039 and 50.083. The top person was Justin Wilson from Great Brittain. The table shows the rank, name and country of person as well as their times.
Prediction: i found a table of 2007 san jose grand prix at redback raceway. there were 5 teams and their performance in the race. the first team was minardi team usa with sébastien bourdais. the second team was forsythe racing with neel jani.

Table 8: Predictions by adapter-tuned bart-large model on the Tablesum dataset.

FeTaQA Examples
Question: What roles did Christian Cooke play in 2009? **Title:** Christian Cooke Television **Flattened table:** Title: Wilmot Year: 1999 Role: Wilmot Tanner Notes: Main role Title: Where the Heart Is Year: 2000–06 Role: Luke Kirkwall Notes: 68 episodes... **Target:** During 2009, Christian Cooke played the lead role of Luke Rutherford in Demons & Lord Dorian Gaudain in Trinity. **Prediction:** In 2009, Christian Cooke played Lord Dorian Gaudain in Dark Relic and Luke Rutherford-Van Helsing in Trinity.
Question: Who did Grace Dove Play in The Revenant and How It Ends? **Title:** Grace DoveFilm **Flattened table:** Year: 2012 Title: These Walls Role: Mary Notes: - Year: 2014 Title: The Cut Role: Emma Notes: - Year: 2015 Title: The Revenant Role: Hugh Glass's wife Notes: - Year: 2017 Title: O for a Thousand Tongues Role: Yaahl-Ts'uu-Kuuyas Notes: - Year: 2017 Title: Mino Bimaadiziwin Role: Bangishimogikwe Notes: - Year: 2018 Title: How It Ends Role: Ricki Notes: - Year: 2019 Title: Monkey Beach Role: Lisa Notes: Filming ... **Target:** Grace Dove played as Hugh Glass's wife in the 2015 film The Revenant and Ricki in the 2018 film How It Ends. **Prediction:** Grace Dove starred as Hugh Glass's wife in The Revenant (2015) and as Ricki in How It Ends (2018).
Question: What was Yevgeniy Ektov's personal record in the 2007 Asian Indoor Games, and how did he place? **Title:** Yevgeniy Ektov Competition record **Flattened table:** Year: Representing Kazakhstan Competition: Representing Kazakhstan Venue: Representing Kazakhstan Position: Representing Kazakhstan Event: Representing Kazakhstan Notes: Representing Kazakhstan Year: 2005 Competition: Asian Championships Venue: Incheon, ... **Target:** At the 2007 Asian Indoor Games, Yevgeniy Ektov jumped 16.34 m, which brought him the bronze. **Prediction:** Yevgeniy Ektov won the bronze medal at the 2007 Asian Indoor Games with a personal record of 16.34 m.
Question: In the 86th United States Congress, What date was the end of Richard M. Simpson's service, and who was his successor? **Title:** 86th United States CongressHouse of Representatives **Flattened table:** District: Missouri 4th Vacator: George H. Christopher (D) Reason for change: Died January 23, 1959 Successor: William J. Randall (D) Date of successor's formal installation: March 3, 1959 District: New York 43rd Vacator: Daniel A. Reed (R) Reason for change: Died February 19, 1959 Successor: Charles Goodell (R) Date of successor's formal installation: ... **Target:** In the 86th United States Congress, Richard M. Simpson (R) served until January 7, 1960 and was succeeded by Douglas H. Elliott (R) on April 26, 1960, who served until June 19, 1960. **Prediction:** In the 86th United States Congress, Richard M. Simpson (R) served from January 7, 1960 to January 7th, 1960, when he was succeeded by Douglas H. Elliott (R).

Table 9: Predictions by adapter-tuned bart-large model on the FeTaQA dataset.

NarrativeQA Examples
Question: What are the four objects the three-headed giant gives Jack as a show of gratitude? **Title:** Jack the Giant Killer **Text summary:** This plot summary is based on a text published ca. 1760 by John Cotton and Joshua Eddowes, which in its turn was based on a chapbook ca. 1711, and reprinted in 'The Classic Fairy Tales' by Iona and Peter Opie in 1974. The tale is set during the reign of King Arthur and tells of a young Cornish farmer's son named Jack who is not only strong but so clever he easily confounds the learned with his penetrating... **Target:** cap of knowledge, shoes of swiftness, magic sword, and invisibility cloak. **Prediction:** a magic sword, a cap of knowledge, a cloak of invisibility, and shoes
Question: Why did Vivie's mother change her name to Mrs. Warren? **Title:** Mrs. Warren's Profession **Text summary:** This plot summary is based on a text published ca. 1760 by John Cotton and Joshua Eddowes, which in its turn was based on a chapbook ca. 1711, and reprinted in 'The Classic Fairy Tales' by Iona and Peter Opie in 1974. The tale is set during the reign of King Arthur and tells of a young Cornish farmer's son named Jack who is not only strong but so clever he easily confounds the learned with his penetrating wit. Jack encounters a cattle-eating giant called Cormoran (Cornish: 'The Giant of the Sea' SWF:Kowr-Mor-An) and lures him to his death in a pit trap. Jack is dubbed 'Jack the Giant-Killer' for this feat and receives not only the giant's wealth, but a sword and belt to commemorate the event. .... **Target:** To give the impression of being married and to hide her true identity. **Prediction:** to hide her identity and give the impression that she is married.
Question: What battle is Jim and Jack preparing for? **Title:** The Great Shadow **Text Summary:** Instead of Sherlock Holmes being the main character, the story follows Jack Colder, who claims his only notable childhood experience was when he accidentally prevented a burglary during an attempt to escape a boarding school. This event caught Jim Horscroft and the two become friends. Once Jim goes off to medical school, Jack reunites with his cousin Edie, who found herself in a lot of money due to her father's death. ... **Target:** The Battle of Waterloo **Prediction:** The Battle of Waterloo

Table 10: Predictions by adapter-tuned bart-large model on the NarrativeQA dataset.