StoryQA: Story Grounded Question Answering Dataset

Anonymous ACL submission

Abstract

The abundance of benchmark datasets supports the recent trend of increased attention given to Question Answering (QA) tasks. However, most of them lack a diverse selection of QA types and more challenging questions. In this work, we present StoryQA, a new task and dataset addressing diverse QA problems for both in-context and out-of-context questions. Additionally, we developed QA models based on large pretrained language models. Our experiments on the new dataset show our developed model achieves comparable performance to answers provided by humans. The resources in this work will be released to foster future research.

1 Introduction

Recent years have seen a lot of attention paid to QA systems. This trend is further supported by an abundance of benchmark datasets that are specifically designed to encourage research in this field. As Khashabi et al. (2020) summarized, current QA datasets can be categorized into four common types: Extractive QA (Rajpurkar et al., 2016; Kočiský et al., 2018), Abstractive QA (Kočiský et al., 2018; Nguyen et al., 2016), Yes/No QA (Clark et al., 2019) and Multiple-Choice QA (Lai et al., 2017). In this paper, we address the first three QA problems since they occur more frequently in real world use cases such as human conversations to ask and answer questions. In Extractive QA, the answer is always a span in the given document context; in Yes/No QA, the answer is always either "yes" or "no"; and in Abstractive QA the response is based on a given context but not restricted to the exact substrings of the given context.

The majority of existing datasets were collected specifically for a single research problem, therefore most of them only contain a single QA type (see Table 1). In addition, their data collection approach limits the scope to in-context questions only, and thus the datasets do not contain any out-of-context questions which occur in realistic QA use cases.

To address these weaknesses, we introduce a new dataset called StoryQA that includes multiple types of QAs on the same context, including Extractive QA, Yes/No QA and Abstractive QA. Our work addresses also the out-of-context questions which are still related, but not directly answerable just by the given context. Note that SQuAD2.0 (Rajpurkar et al., 2018) contains out-of-context unanswerable questions, but their goal is to just identify and filter out those, rather than answering them. During the creation of our dataset, we observed that many of the out-of-context questions, especially those asking for non-factual information in a fictional story, can still be answered by humans. One example is shown in Table 2 where the question is about what was in the boy’s mind. Although the story does not have explicit answer for this, humans can still provide a reasonable answer after reading the story. Most existing models are unable to respond such questions reasonably, due to the in-context limitation of the training datasets.

We summarize our contributions as follows:
1) We publish a new dataset called StoryQA

<table>
<thead>
<tr>
<th>Dataset</th>
<th># QAs</th>
<th>QA Type</th>
<th>OOC</th>
</tr>
</thead>
<tbody>
<tr>
<td>SQuAD1.1</td>
<td>~98.2k</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>SQuAD2.0</td>
<td>~142.2k</td>
<td>✓</td>
<td>33.38% no ans.</td>
</tr>
<tr>
<td>BoolQ</td>
<td>~900.8k</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>DROP</td>
<td>~12.7k</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>MS MARCO</td>
<td>~86.9k</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>NaturalQuestion</td>
<td>~93.5k</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>NarrativeQA</td>
<td>~98.8k</td>
<td>✓</td>
<td>-</td>
</tr>
<tr>
<td>StoryQA (ours)</td>
<td>~36k</td>
<td>✓ ✓ ✓</td>
<td>56.62%</td>
</tr>
</tbody>
</table>

Table 1: Dataset Comparison: EX = Extractive, YN = Yes/No, AB = Abstractive, OOC = Out-of-Context. StoryQA contains diverse QA types and more challenging out-of-context questions. For out-of-context questions, SQuAD2.0 only needs to detect without answering them, while StoryQA provides all answers.
Story (given context):
A Boy was given permission to put his hand into a pitcher to get some filberts. But he took such a great fistful that he could not draw his hand out again. There he stood, unwilling to give up a single filbert and yet unable to get them all out at once. Vexed and disappointed he began to cry. "My boy," said his mother, "be satisfied with half the nuts you have taken and you will easily get your hand out. Then perhaps you may have some more filberts some other time."

Table 2: Sample responses from models trained on various datasets (column 1) for an out-of-context question for the popular fable “The Boy and the Filberts”.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Human</th>
<th>StoryQA (ours)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Original QA</td>
<td>The boy was greedy because he really liked filberts</td>
<td>The boy was greedy because he wanted to get as many nuts as possible.</td>
</tr>
<tr>
<td>SQuAD 2.0</td>
<td>he took such a great fistful</td>
<td></td>
</tr>
<tr>
<td>NaturalQuestion</td>
<td>unwilling to give up a single filbert</td>
<td></td>
</tr>
<tr>
<td>DROP</td>
<td>vexed and disappointed he began to cry</td>
<td></td>
</tr>
</tbody>
</table>

That contains multiple types of in-context and out-of-context questions. It is collected based on Aesop’s Fables, because we found that as compared with questions in non-fictional contexts such as Wikipedia or news articles, fictional stories are better to collect more diverse questions. This dataset aims to tackle the following three QA problems: Extractive QA, Yes/No QA, and Abstractive QA. Among them, Abstractive QA is the most challenging problem with out-of-context questions that most existing models cannot answer properly.

2) We propose a unified QA model that handles all three QA types and demonstrate via both automatic and human evaluation that it performs consistently better than the fine-tuned models on just a single type of QAs. The results also show that our unified model achieves comparable performances to the human references.

2 Related Work

Most existing datasets were collected by asking crowd workers to provide questions and answers following specific guidelines designed for a particular research problem. As representatives of Extractive QA datasets, SQuAD 1.1 (Rajpurkar et al., 2016), SQuAD 2.0 (Rajpurkar et al., 2018) and NaturalQuestion (Kwiatkowski et al., 2019) were collected by asking each worker to write down a pair of question and answer together. Since every answer is always restricted to a span in the context, the datasets contain the in-context questions only.

This limitation also applies to the Multiple-choice and Yes/No QA datasets. For the Multiple-choice QAs, workers need to provide a list of answer candidates including a correct answer and other distractors. Since each correct answer needs to be explicitly validated by the given context, the collected data covers in-context questions only. In addition, workers are asked to provide both questions and answers, as in the RACE (Lai et al., 2017) data collection. As a result, workers are likely to provide questions that are easy to identify the correct answers. Similarly, the Yes/No QA datasets such as BoolQ (Clark et al., 2019) includes the in-context questions that can be clearly answered by either “yes” or “no” based on given contexts.

Abstractive QA datasets place less constraints, but still have narrow scopes due to the specific data collection requirements to guide the collected data towards particular research problems. For example, as part of the NarrativeQA (Kociský et al., 2018) data collection process, workers are instructed to avoid copying from the context, but provide specific and diverse QA pairs. In DROP (Dua et al., 2019), workers are encouraged to provide questions that need to be answered through discrete reasoning. MS MARCO (Nguyen et al., 2016) is based on search logs. As mentioned in Kociský et al. (2018), many answers are in fact verbatim copies of short spans from the context.

To the best of our knowledge, SQuAD 2.0 is the only dataset including out-of-context questions. However, SQuAD 2.0 aims to filter out out-of-context questions rather than providing the answer to them. We suppose that the lack of out-of-context QA data is because (1) most existing datasets require workers to provide paired questions and answers together, thus discouraging them from asking out-of-context questions, and (2) workers are instructed to ask questions mainly to test the reading comprehension skills rather than pretending to be inquisitive about the given context. To collect more realistic and challenging QAs, our data was collected in an alternative way that each question and its answer were collected by different workers from each other. We believe this results in more diverse data, because the workers can provide the questions with no consideration about how to answer them at the same time.

3 StoryQA Dataset

In this section, we introduce our new dataset, StoryQA, which addresses many of the above-mentioned limitations of the existing QA datasets.
3.1 Desiderata

From the limitations of other datasets discussed in Section 2, we define our desiderata as follows. First, we construct a dataset containing a large number of QA pairs collected by two groups of crowd workers for questions and answers separately, where the questioners can ask diverse questions regardless of whether and how they can be answered. Second, we set as few restrictions as possible to make the collected data plausible in real-world use cases. In the same line, we took the fictional children stories from Aesop’s Fables and asked crowd workers to pretend they were 5-8 years old, which aims to collect more flexible and creative questions which result in more challenges in QA research.

3.2 Data Collection Method

We collected three subsets each of which addresses Extractive, Yes/No, and Abstractive QA types.

**Extractive QA Subset**: Here, every answer must be a span in a given story context. We first automatically generated the answer candidates from each story in Aesop’s Fables. We revised the Extractive QA model (see Section 4.2), where story context and questions are fed into a base model and two pointers are learned to locate a single answer, and only fed the context in order to locate multiple spans for answer candidates. We used AlBERT-xxLarge (Lan et al., 2019) as the base model and trained it on the SQuAD 2.0 dataset. For each story-answer pair, we asked crowd workers to provide a question that can be answered by the span.

**Yes/No QA Subset**: The generated answer spans above were used also for collecting Yes/No QA subset. Here we provided each span as the additional information to guide annotators with their questions. Similar to Extractive QA, the crowd workers were shown the full story with highlighted span and asked to submit a Yes/No question for the given Yes/No answer.

**Abstractive QA Subset**: Different from the first two categories, the abstract QAs were collected by two crowdsourcing tasks, questions first followed by the answers. To collect diverse questions, we asked the crowd workers to provide free-form questions. We only require that the questions should be relevant to the given story context. Then, we had a subsequent task to collect the answer for each question. To categorize the answer sources for the questions, we first asked the crowd workers to specify whether it can be answered only with the given story context or requires any external knowledge beyond the story content. Then for the question they provided the answer in their own words that is grounded on either the story context or their background knowledge.

To ensure the answer quality, a pilot task was conducted first with a small amount of data followed by a manual evaluation task. Then, the full data collection was done only with the highly-scored workers in the pilot task. Table 3 shows the statistics of the collected data. These are collected using 148 Aesop’s Fables as the story context.

<table>
<thead>
<tr>
<th>Subset</th>
<th>#QAs</th>
<th>Priming</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extractive QA</td>
<td>14,766</td>
<td>story span</td>
<td>✓</td>
</tr>
<tr>
<td>Yes/No QA</td>
<td>11,779</td>
<td>story span</td>
<td>✓ ✓</td>
</tr>
<tr>
<td>Abstractive QA</td>
<td>12,148</td>
<td>1. none</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2. Q from step 1</td>
<td>✓</td>
</tr>
</tbody>
</table>

Table 3: Collection of StoryQA subsets. For Extractive QA and Yes/No QA datasets, crowd workers were shown story span extracted by an Extractive QA model along with full story. Abstractive QA dataset was collected in 2 steps where a free-form question was collected in step 1 from a given story, and later showed to an independent set of workers to get their answers.

3.3 Analysis of Abstractive QA Subset

We analyze the Abstractive QA subset in more detail since it differs from most existing datasets and introduces new research challenges.

**Question Format**: Table 4 shows a breakdown of the Abstractive QA subset by question format. We observed that 39.27% of the questions in the Abstractive QA subset can be answered by Yes/No, which is the most common category followed by What and Why questions. In addition, we notice that some questions belong to multiple question formats, which introduces more challenges to QA models.

**Knowledge Source for Answer**: As we mentioned earlier, there are many out-of-context questions in the Abstractive QA subset and thus it is important to understand the properties of such questions and how to develop models to answer them. During data collection, we explicitly asked crowd workers to identify the category of answer sources. As shown in Table 5, only 43.38% of the questions have the explicit answers within the story content. Within the rest out-of-context questions, only a small percentage (14.49%) requires external...
Table 4: Breakdown by Question Format: Questions are statistically analyzed by their “question” word. Questions such as “Ohh, why did the fox hurry? what were they sick with?”, “Who is the shepherd and why would he care?” will be counted twice, once for Why-based questions and once for What-based questions.

Table 5: Breakdown by Knowledge Source. Only questions grouped under “In-context” can be answered just from the given story context; the rest “out-of-context” questions require factual or commonsense knowledge outside the story contents.

Initially focus on fine tuning pretrained language models and adapting the knowledge in these large language models to generate reasonable answers on StoryQA.

4.1.2 Model Architecture
In this section, we show our model architecture for Abstractive QA. We followed the UnifiedQA architecture and employed Transformer-based Encoder-Decoder framework (Vaswani et al., 2017). As in Figure 1, we concatenate question and story into a single packed sequence. These are separated by the new line character “\n” and fed into Transformer Encoder to obtain the hidden states $T_{enc}$. Khashabi et al. (2020) explains how this ensures a human-like encoding while not making it overly-specific to a certain format. The Transformer Decoder models the probabilities of each word $w_i$ in the answer as $p(w_i|w_{i-1}, w_{i-2}, ..., T_{enc})$ in an auto-regressive manner. The sum of log-likelihoods of $w_i$ is used as the training objective.

4.1.3 Handling Out-of-Context Questions
Considering that our dataset includes many out-of-context questions that requires external knowledge sources, we attempt to retrieve additional relevant contexts and incorporate them into answering the question.

4. Model Development
In this section, we present the baseline models for each QA type as well as a unified model to address all QA types.

4.1 Model for Abstractive QA
As mentioned in Section 3.3, Abstractive QA poses the most challenges due to its diversity and hence we elaborate more.

4.1.1 Analysis of Existing Models
Table 2 shows a typical out-of-context question in our dataset. Although ExtractiveQA models fail for these questions, we observed in general that the UnifiedQA model (Khashabi et al., 2020) can generate the most reasonable answers. We will initially focus on fine tuning pretrained language models and adapting the knowledge in these large language models to generate reasonable answers on StoryQA.

4.1.2 Model Architecture
In this section, we show our model architecture for Abstractive QA. We followed the UnifiedQA architecture and employed Transformer-based Encoder-Decoder framework (Vaswani et al., 2017). As in Figure 1, we concatenate question and story into a single packed sequence. These are separated by the new line character “\n” and fed into Transformer Encoder to obtain the hidden states $T_{enc}$. Khashabi et al. (2020) explains how this ensures a human-like encoding while not making it overly-specific to a certain format. The Transformer Decoder models the probabilities of each word $w_i$ in the answer as $p(w_i|w_{i-1}, w_{i-2}, ..., T_{enc})$ in an auto-regressive manner. The sum of log-likelihoods of $w_i$ is used as the training objective.

4.1.3 Handling Out-of-Context Questions
Considering that our dataset includes many out-of-context questions that requires external knowledge sources, we attempt to retrieve additional relevant contexts and incorporate them into answering the question.
questions. We investigate two retrieval methods widely used in QA research communities, namely DPR (Karpukhin et al., 2020) on Wikipedia passages and ColBERT (Khattab and Zaharia, 2020; Lin et al., 2021) on MARCO (web pages). Both models are trained by minimizing the distance between the question and the relevant document in an information retrieval fashion. We used ColBERT and DPR to retrieve Wikipedia and web pages, respectively, and appended the retrieved document to the end of the story context, again using the newline character “\n” as a separator (Figure 1).

4.2 Model for Extractive QA

Extractive QA requires the answer to be a span in the story and has been widely studied in the QA research community. We followed the standard procedure to extract answers, where a pretrained language model is used as the basis to predict the start and end positions of the span for the answer in the given context. Specifically, we concatenate the input question and story, and fed them into a pretrained language model to obtain hidden states $T$. The probability of word $w_i$ being the start of the answer span is computed as the dot product between $T_i$ and $T_s$ where $s$ is the start position of the answer. The same thing is done for the end of answer span. The training objective is the sum of the log-likelihoods of the correct start and end positions.

4.3 Model for Yes/No QA

All of the answers in this subset are either “yes” or “no”, and can be answered from the given context. Due to limited resources in Yes/No QA research, we followed the UnifiedQA model (Khashabi et al., 2020) as described in Section 4.1.2, but with the constrained decoding only for either “yes” or “no”.

4.4 Our Unified StoryQA Model

In addition to above the QA problem-specific models, we propose a unified StoryQA model for all the question types. Inspired by the prefix constraint idea (Takeno et al., 2017; Liu et al., 2018; Zhao et al., 2019; Martin et al., 2019), we add the question-type prefix before the question, as shown in Figure 2. Specifically, we fine-tuned the UnifiedQA model on the entire StoryQA dataset containing all three QA types (Extractive, Yes/No) by using prefix tokens “abstractive”, “extractive” and “yesno” respectively. With such a design, we hope that different QA datasets will complement each other and improve the performances across all the subsets.

5 Experiments and Result Analysis

5.1 Experiment Setup

Our data is divided into five splits as shown in Table 6. For all of our experiments, we picked the best models based on the merged Dev split (Dev-seen + Dev-unseen) and reported the performance separately for Test-seen and Test-unseen. All models were trained on an 8 A100 GPU machine with LAMB optimizer (Khashabi et al., 2020) and learning rate warm-up technique. For larger models, such as T5-3B and T5-11B, we used ZeRO (Ren et al., 2021; Rajbhandari et al., 2021) to train our model.

5.2 Experiments on Abstractive QA Subset

In Abstractive QA, we compare transformer-based encoder-decoder frameworks, with a particular focus on different size of T5 (Raffel et al., 2019) and BART models (Lewis et al., 2019) as well as their fine-tuned versions, all based on the UnifiedQA model (Khashabi et al., 2020). In this section, we discuss the automatic evaluation results with the reference-based metrics including BLEU (Papineni et al., 2002) and Rouge (Lin, 2004). Human evaluation results will be presented later.

Effect of Model Size and Fine-tuning: Table 7 shows the performances of UnifiedQA on the Abstractive QA subset, with base models of different sizes following the procedure in Section 4.1.2 and Figure 1. We observe that T5-base performs significantly better than BART-Large (about twice the size) in all the metrics while it keeps improving as we increase the number of parameters, as we tried
Table 6: Data Splits. StoryQA contains five splits. Both Dev and Test splits contain seen and unseen story versions, which indicate whether the splits share the same stories with training or not, respectively.

<table>
<thead>
<tr>
<th>Split</th>
<th>Description</th>
<th>Extractive</th>
<th>Yes/No</th>
<th>Abstractive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>Training dataset with QAs sampled from 128 stories</td>
<td>8,652</td>
<td>8,397</td>
<td>10,772</td>
</tr>
<tr>
<td>Dev-seen</td>
<td>Sample 1000 QAs share the same stories in training set</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Dev-unseen</td>
<td>Sample 10 stories not in Training or Test sets</td>
<td>552</td>
<td>797</td>
<td>995</td>
</tr>
<tr>
<td>Test-seen</td>
<td>Sample 1000 QAs share the same stories in training set</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Test-unseen</td>
<td>Sample 10 stories not in Training or Dev sets</td>
<td>575</td>
<td>954</td>
<td>999</td>
</tr>
</tbody>
</table>

Table 7: Abstractive QA subset. “-FT” indicates fine-tuned version on the Abstractive QA subset. Base models are shown in bold in column 1. Best values are shown bold faced.

<table>
<thead>
<tr>
<th>Model Config</th>
<th>#Params</th>
<th>Test-seen</th>
<th>Test-unseen</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>BLEU</td>
<td>Rouge1</td>
</tr>
<tr>
<td>UnifiedQA-BART-Large</td>
<td>406M</td>
<td>0.10</td>
<td>11.30</td>
</tr>
<tr>
<td>UnifiedQA-T5-Base</td>
<td>220M</td>
<td>1.05</td>
<td>15.07</td>
</tr>
<tr>
<td>UnifiedQA-T5-Large</td>
<td>770M</td>
<td>1.07</td>
<td>15.56</td>
</tr>
<tr>
<td>UnifiedQA-T5-3B</td>
<td>3B</td>
<td>1.15</td>
<td>17.02</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B</td>
<td>11B</td>
<td>3.81</td>
<td>24.02</td>
</tr>
<tr>
<td>UnifiedQA-BART-Large-FT</td>
<td>406M</td>
<td>10.64</td>
<td>33.59</td>
</tr>
<tr>
<td>UnifiedQA-T5-Base-FT</td>
<td>220M</td>
<td>10.29</td>
<td>34.89</td>
</tr>
<tr>
<td>UnifiedQA-T5-Large-FT</td>
<td>770M</td>
<td>10.94</td>
<td>35.20</td>
</tr>
<tr>
<td>UnifiedQA-T5-3B-FT</td>
<td>3B</td>
<td>11.19</td>
<td>36.40</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B-FT</td>
<td>11B</td>
<td>11.38</td>
<td>36.81</td>
</tr>
</tbody>
</table>

5.3 Experiments on Extractive QA Subset

For the extractive QAs, we compared the performances of SQuAD (Rajpurkar et al., 2016) model variations with AlBERT (Lan et al., 2019) and DeBERTa (He et al., 2020) as base models, and also when fine-tuning them on our Extractive QA dataset. Following the SQuAD evaluation set-ups, we used the Exact Match and F1 as the evaluation metrics. Table 11 shows that the larger models perform the better in general (except AlBERT-xxLarge-FT); and the fine-tuning helps to improve the performances significantly, especially when using DeBERTa.

5.4 Experiments on Yes/No QA Subset

For the Yes/No QAs, we experiment with the Unified QA variations by changing the base models. All of these models were fine-tuned on our Yes/No QA subset and evaluated on accuracy for the binary predictions as in (Clark et al., 2019). Table 12 indicates that UnifiedQA models do not perform well, but when fine-tuned they improve significantly. This may be due to the limited amount of Yes/No QA datasets in training the UnifiedQA model.

<table>
<thead>
<tr>
<th>Split</th>
<th>Description</th>
<th>Extractive</th>
<th>Yes/No</th>
<th>Abstractive</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>Training dataset with QAs sampled from 128 stories</td>
<td>8,652</td>
<td>8,397</td>
<td>10,772</td>
</tr>
<tr>
<td>Dev-seen</td>
<td>Sample 1000 QAs share the same stories in training set</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Dev-unseen</td>
<td>Sample 10 stories not in Training or Test sets</td>
<td>552</td>
<td>797</td>
<td>995</td>
</tr>
<tr>
<td>Test-seen</td>
<td>Sample 1000 QAs share the same stories in training set</td>
<td>1,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Test-unseen</td>
<td>Sample 10 stories not in Training or Dev sets</td>
<td>575</td>
<td>954</td>
<td>999</td>
</tr>
<tr>
<td>Question Format</td>
<td>Test-seen</td>
<td>Test-unseen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>----------------</td>
<td>-----------</td>
<td>-------------</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(% samples)</td>
<td>BLEU</td>
<td>Rouge1</td>
<td>Rouge2</td>
<td>RougeL</td>
</tr>
<tr>
<td>Yes/No (39.3%)</td>
<td>9.61</td>
<td>34.96</td>
<td>17.94</td>
<td>32.31</td>
</tr>
<tr>
<td>What (20.3%)</td>
<td>15.34</td>
<td>40.76</td>
<td>23.43</td>
<td>38.31</td>
</tr>
<tr>
<td>Why (23.07%)</td>
<td>8.41</td>
<td>33.38</td>
<td>15.2</td>
<td>29.93</td>
</tr>
<tr>
<td>How (8.2%)</td>
<td>9.20</td>
<td>30.3</td>
<td>13.39</td>
<td>28.3</td>
</tr>
<tr>
<td>Where (2.9%)</td>
<td>21.5</td>
<td>50.35</td>
<td>32.24</td>
<td>48.39</td>
</tr>
<tr>
<td>Who (4.1%)</td>
<td>22.39</td>
<td>53.75</td>
<td>34.95</td>
<td>50.96</td>
</tr>
<tr>
<td>When (2.1%)</td>
<td>7.92</td>
<td>26.69</td>
<td>12.55</td>
<td>23.92</td>
</tr>
</tbody>
</table>

Table 8: Performance of UnifiedQA-T5-11B-FT on Abstractive QA subset based on the breakdown as in Table 4.

<table>
<thead>
<tr>
<th>Knowledge Source</th>
<th>Test-seen</th>
<th>Test-unseen</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>Rouge1</td>
</tr>
<tr>
<td>In-context</td>
<td>15.31</td>
<td>42.45</td>
</tr>
<tr>
<td>Factual Knowledge</td>
<td>11.97</td>
<td>36.96</td>
</tr>
</tbody>
</table>

Table 9: Performance of UnifiedQA-T5-11B-FT on Abstractive QA subset based on the breakdown as in Table 5.

<table>
<thead>
<tr>
<th>Model Config</th>
<th>Test-seen</th>
<th>Test-unseen</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>Rouge1</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B-FT</td>
<td>11.38</td>
<td>36.81</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B-FT + MARCO</td>
<td>11.59</td>
<td>37.27</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B-FT + Wiki</td>
<td>11.83</td>
<td>37.32</td>
</tr>
</tbody>
</table>

Table 10: Performance of UnifiedQA-T5-11B-FT (the best model from Table 7) on Abstractive QA subset, when augmented with relevant retrieved web pages from MS Marco (“+MARCO”) or Wikipedia passages (“+Wiki”). Best values are shown bold faced.

<table>
<thead>
<tr>
<th>Model Config</th>
<th>#Params</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EM</td>
<td>F1</td>
</tr>
<tr>
<td>AlBERT-Base</td>
<td>12M</td>
<td>51.1</td>
</tr>
<tr>
<td>AlBERT-Large</td>
<td>18M</td>
<td>54.3</td>
</tr>
<tr>
<td>AlBERT-xLarge</td>
<td>60M</td>
<td>56.5</td>
</tr>
<tr>
<td>AlBERT-xxLarge</td>
<td>235M</td>
<td>57.6</td>
</tr>
<tr>
<td>DeBERTa-Base</td>
<td>139M</td>
<td>54.3</td>
</tr>
<tr>
<td>DeBERTa-Large</td>
<td>405M</td>
<td>56.7</td>
</tr>
<tr>
<td><em>DeBERTa-Base-FT</em></td>
<td>5M</td>
<td>51.8</td>
</tr>
<tr>
<td><em>DeBERTa-Large-FT</em></td>
<td>18M</td>
<td>54.4</td>
</tr>
<tr>
<td><em>AlBERT-LL-FT</em></td>
<td>260M</td>
<td>58.0</td>
</tr>
<tr>
<td><em>AlBERT-xxL-F</em></td>
<td>235M</td>
<td>55.7</td>
</tr>
<tr>
<td>DeBERTa-Base-FT</td>
<td>139M</td>
<td>58.4</td>
</tr>
<tr>
<td>DeBERTa-Large-FT</td>
<td>405M</td>
<td>59.5</td>
</tr>
</tbody>
</table>

Table 11: Extractive QA subset: “-FT” indicates fine-tuned version on the Extractive QA subset. EM = Exact Match. Best values are shown bold faced.

<table>
<thead>
<tr>
<th>Model Config</th>
<th>Test-seen</th>
<th>Test-unseen</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BLEU</td>
<td>Rouge1</td>
</tr>
<tr>
<td>UnifiedQA-T5-Base</td>
<td>68.6</td>
<td>70.61</td>
</tr>
<tr>
<td>UnifiedQA-T5-Large</td>
<td>77.8</td>
<td>76.87</td>
</tr>
<tr>
<td>UnifiedQA-T5-3B</td>
<td>86.1</td>
<td>85.57</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B</td>
<td>54.6</td>
<td>53.57</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B-FT</td>
<td>88.0</td>
<td>86.78</td>
</tr>
<tr>
<td>UnifiedQA-T5-3B-FT</td>
<td>85.6</td>
<td>86.26</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B-FT</td>
<td>91.1</td>
<td>90.43</td>
</tr>
<tr>
<td>UnifiedQA-T5-11B-FT</td>
<td>92.4</td>
<td>91.13</td>
</tr>
</tbody>
</table>

Table 12: Yes/No QA subset: “-FT” indicates fine-tuned version on the Yes/No QA subset. Best values are shown bold faced.

5.5 Our Unified StoryQA Model

As mentioned in Section 4.4, our Unified StoryQA model is based on the UnifiedQA model with adaptations as in Figure 2. Our model is based on the best configuration on Abstractive QA subset, namely “UnifiedQA-T5-11B” (see Table 7), and fine-tuned on the entire StoryQA dataset (not one subset). We call this single model as “Unified StoryQA Model”. We compare this single model against the best models we presented for each of the three subsets. Note again that these competing models were fine-tuned on only the relevant subsets and not the entire StoryQA dataset. Comparisons are shown in Tables 13, 14, and 15 for Abstractive QA, Extractive QA and Yes/No QA subsets of StoryQA dataset, respectively. Our single Unified StoryQA Model achieves the best performance for all three subsets (except the Yes/No QA subset on Test-seen where it is still close), including the challenging Abstractive QA subset that has out-of-context questions. This also shows that the different subsets specialized in different QA types complement each other and can further improve performance when we combine them together.
Thus, answers produced by models finetuned on Table 15: Yes/No QA Model Performance: “-FT” indicates fine-tuned version. “-UFT” indicates fine-tuned provided answers. The crowd workers from Amazon our dataset are more expressive. For reference, it is 4.19 for NarrativeQA dataset. man) are 4.30, 10.28, 10.08 and 12.16 respectively.

Table 14: Extractive QA Model Performance: “-FT” indicates fine-tuned version. “-UFT” indicates the single unified model introduced in Section 4.4. EM = Exact Match.

Table 13: Abstractive QA Model Performance: “-FT” indicates fine-tuned version. “-UFT” indicates fine-tuned unified model.

Table 15: Yes/No QA Model Performance: “-FT” indicates fine-tuned version. “-UFT” indicates the single unified model introduced in Section 4.4.

5.6 Human Evaluation on Abstractive QA Subset

Since the automatic metrics are known to be limited in capturing the comprehensive model performances beyond the overlaps with the references, we conducted the human evaluation studies to further analyze the models. We used all the questions in the Test dataset and shuffled the predicted answers and ground truth to determine the qualitative gap between the model predictions and the human provided answers. The crowd workers from Amazon Mechanical Turk were asked to read a story and a question, and then rate each answer on a 5-point scale (1-5, where 5 is the best) for appropriateness.

Table 16 compares the average ratings for each fine-tuned model. Results are consistent with earlier findings from automatic evaluations, indicating that larger models have superior performance. We can also see that our best model performs very closely to ground truth answers. Note that answers are rated for how accurate they are for the given question, rather than how natural they are. The average number of whitespace-delimited tokens per answer from UnifiedQA-T5-11B, UnifiedQA-T5-11B-FT, Unified StoryQA and Ground Truth (Human) are 4.30, 10.28, 10.08 and 12.16 respectively. For reference, it is 4.19 for NarrativeQA dataset. Thus, answers produced by models finetuned on our dataset are more expressive.

We also conducted a human evaluation study to analyze the effect of Retrieving Relevant Context. We followed a similar setup above, but sampled 300 questions from the test dataset and shuffled the model predictions of all the models mentioned in Table 10 for evaluation. Table 17 shows consistent results.

Table 17: Human Evaluation For Handling Out-of-Context Questions based on a 5-point scale (1-5, where 5 is the best) for appropriateness of answer.

6 Conclusion

In this work, we introduce a new task and dataset, named StoryQA. Our dataset covers three types of QA problems: Extractive QA, Yes/No QA and Abstractive QA. In addition, it includes many challenging questions, especially those that are out of context. We have conducted extensive experiments showing the insights related to the size of the models, fine-tuning, source of knowledge and types of questions. We also propose a Unified StoryQA Model that performs better than the equivalent models fine-tuned on a single specific subset. We hope that our proposed StoryQA dataset, baseline models and the findings from the experiments will inspire future work in the QA and NLP communities, moving towards a QA system that addresses more open-ended and diverse questions. More contextual QAs across multiple turns is also a natural future extension from the current settings.
References


A Implementation Details

Our implementation is based on Huggingface (Wolf et al., 2020) and Pytorch (Paszke et al., 2019) libraries. PyTorch is a Python library used to build deep learning projects. Huggingface is built on top of Pytorch and provides pre-trained models. For large models like T5-3B and T5-11B, which will not fit in a single GPU, we used ZeRO (Ren et al., 2021; Rajbhandari et al., 2021), which implemented in DeepSpeed (Rasley et al., 2020) library.

B Data Collection Details

In this section, we present more details of our data collection, including user interface (UI) and Amazon Mechanic Turk (AMT) setup.

B.1 Extractive QA Subset

As shown in Figure 3, we simulate that our data is a part of the conversation between a child and storyteller. The storyteller starts by telling a story to warm up the conversation and then asks the child a question. As mentioned earlier, an answer span was extracted, which was highlighted in the story, and we asked crowd workers to write down questions for that answer span.

B.2 Yes/No QA Subset

For the Yes/No QAs, we employed a similar setup as the ExtractiveQA data collection by simulating a conversation for collecting each question (see Figure 4). However, the answer spans are only used as hints. We asked crowd workers to write down a question if the answer was "yes" or "no" and was relevant to the highlighted hints.

B.3 Abstractive QA Subset

As mentioned previously, data collection for Abstractive QA subset is divided into two tasks. In the first task, we aim to collect questions as diverse as possible. As shown in Figure 5, we asked the crowd worker to write down a question by pretending to be an enthusiastic child, but didn’t ask them to answer it. This is a key setting that allows our dataset to contain a variety of in-context and out-of-context questions. For the second task, it is more difficult to collect answers because there may be multiple correct answers, especially for out-of-context questions. As shown in Figures 6 and 7, before writing the answer, crowd workers were asked if the question requires any external knowledge beyond the story’s content, and what type of external knowledge is used. We found that this approach allowed us to obtain higher quality answers.

B.4 AMT Setup

We list the default qualifications for participants when launching the data collection tasks using Amazon Mechanical Turk:

• Location is in the US
• HIT Approval Rate for all Requesters’ HITs greater than 95%
• Number of HITs Approved greater than 5000

For AbstractiveQA answer collection, to ensure high quality answers, we split the task into two steps. First, we launched a small pilot task with 100 randomly sampled questions followed by human evaluation to get a 5-point rating (range of rating is 1-5, where 5 is the best score) for each collected answer. Then, we selected a total of 28 high quality crowd workers who provided the answers scored higher than 4.0 on average. Finally, the main data collection was done only with these selected workers.

The total crowd annotation cost to collect StoryQA dataset was $11,383.0 with a breakdown as follows:

• Extractive QA Subset: We paid $0.2 for each question. So, the total cost for collecting our Extractive QA subset was $2953.2 ($0.2 * 14,766).
• Yes/No QA Subset: Similar to Extractive QA Subset, we paid $0.2 for each question, for a total of $2355.8 ($0.2 * 11,779) to collect our Yes/No QA subset.
• Abstractive QA Subset: Each worker was paid $0.2 to create a question and $0.3 for each answer, for a total cost of $6074 ($0.3 * 12,148 + $0.2 * 12,148) to collect our Abstractive QA subset.
Figure 3: Data Collection UI for Extractive QA subset. The story span for answer was extracted by a model and highlighted in red. The worker is expected to create a question for which the highlighted span is a good fit as an answer.

Figure 4: Data Collection UI for Yes/No QA subset. Similar to data collection procedure for our Extractive QA subset, a story span was extracted by a model and highlighted in red. The worker is expected to use this span as hint to generate a Yes/No question based on the given Yes/No answer.

Figure 5: Data Collection UI for Abstractive QA subset - Question Collection. In order to get diverse questions and increase the chances of out-of-context questions, we first collect relevant questions from crowd workers, given the story text as context. These story-question pairs will later be shown to an independent set of workers to get corresponding answers as in Figures 6 and 7.
Step 1

Read carefully the following story.

One moonlight evening as Master Fox was taking his usual stroll in the woods, he saw a number of Pheasants perched quite out of his reach on a limb of a tall old tree. The sly Fox soon found a bright patch of moonlight, where the Pheasants could see him clearly; there he raised himself up on his hind legs, and began a wild dance. First he whirled 'round and 'round like a top, then he hopped up and down, cutting all sorts of strange capers.

The Pheasants stared giddily. They hardly dared blink for fear of losing him out of their sight a single instant. Now the Fox made as if to climb a tree, now he fell over and lay still, playing dead, and the next instant he was hopping on all fours, his back in the air, and his bushy tail shaking so that it seemed to throw out silver sparks in the moonlight.

By this time the poor birds' heads were in a whirl. And when the Fox began his performance all over again, so dazed did they become, that they lost their hold on the limb, and fell down one by one to the Fox.

Step 2

Read carefully the following question about the story.

Question: Why did the Pheasants stare giddily?

Step 3

Do you think you can answer the question based on the given story contents only?

- Yes, it can be answered with the story contents only.
- No, it requires additional background or common-sense knowledge to be answered.

Step 4

Write down your answer to the question in your own words.

Notes:
- Your answer must be a complete sentence with more than 5 words.
- Make sure if your answer clearly and directly addresses the question.
- Make sure if your answer is grammatically correct.

Answer: [Please write down your answer to the question]

Figure 6: Data Collection UI for Abstractive QA - Answer Collection - Example 1. The free-form question related to the story content as generated by an independent worker is displayed along with the full story. The user is expected to answer the question and identify the source of knowledge for the answer.
**Step 1**

Read carefully the following story.

A little hungry Mouse found his way one day into a basket of corn. He had to squeeze himself a good deal to get through the narrow opening between the strips of the basket. But the corn was tempting and the Mouse was determined to get in. When at last he had succeeded, he gorged himself to bursting. Indeed he became about three times as big around the middle as he was when he went in. At last he felt satisfied and dragged himself to the opening to get out again. But the best he could do was to get his head out. So there he sat groaning and moaning, both from the discomfort inside him and his anxiety to escape from the basket. Just then a Weasel came by. He understood the situation quickly. "My friend," he said, "I know what you've been doing. You've been stuffing. That's what you get. You will have to stay there till you feel just like you did when you went in. Good night, and good enough for you." And that was all the sympathy the poor Mouse got.

**Step 2**

Read carefully the following question about the story.

**Question:** Was monkey very hungry

**Step 3**

Do you think you can answer the question based on the given story contents only?

- [ ] Yes, it can be answered with the story contents only.
- [x] No, it requires additional background or common-sense knowledge to be answered.

**Step 4**

How would you find the evidence to answer this question?

- [ ] I need to google or look up the Wikipedia to find the relevant facts.
- [ ] I already know the fact to answer the question.
- [ ] I don't think this question is related to any fact. I would try to provide an answer based on my commonsense.

---

Figure 7: Data Collection UI for Abstractive QA - Answer Collection - Example 2. Also see Figure 6.
C Experiment Details

As mentioned in the main paper, all models were trained on an 8 A100 GPU machine. We applied the early stopping technique and evaluated on the development dataset (merge of Dev-seen and Dev-unseen splits) at each epoch. We stop training when the evaluation does not improve within 5 epochs. Most of the smaller models are trained within 1 day. Large models like T5-11b take 1-3 days, depending on the learning rate. We also used the development dataset to choose the best hyperparameters, such as the learning rate. Considering the budget, we trained the model only once for each setting. Figure 8 is a screen shot of our human evaluation UI.

D Human Evaluation Setup

To ensure reliable results, we hired the same crowd workers who participated in answer collection (Section B.4), for all human evaluation tasks. As shown in Figure 8, each task includes a triple of story, question and a list of responses from either the model outputs or the ground-truth human reference. To avoid any position bias, all responses were provided in a random order. We hired three crowd workers for each task and asked them to score the accuracy of each answer for the question in 5-point rating. We paid $0.2 for each task.

E Sample Outputs

Out-of-context questions from a seen story: “The Ant and the Grasshopper” is a story in our dataset that is in the training set. Figure 18 shows the entire story along with 3 out-of-context questions. These questions require common sense, and general knowledge to answer them correctly. Our Unified StoryQA model produces reasonable answers in complete sentence for every question.

Out-of-context questions from an unseen story: Figure 19 contains “Belling the Cat” that is not part of the training set but in StoryQA dataset. There are 4 out-of-context questions that require common sense in order to answer them correctly. Once again, our Unified StoryQA model are complete sentences.

Questions for a story outside of our dataset: Figure 20 shows sample outputs to 4 questions for a story not part of our dataset (“Three Little Pigs”). These questions can be answered completely from the given story or require common sense or general knowledge. Our Unified StoryQA model is able to respond correctly even though this story is not from our dataset or from Aesop Fables.

In all cases, note that models fine-tuned on our dataset produce longer and complete answers.

F Limitations of Dataset and Model

1. StoryQA dataset is relatively small, although diverse. It only covers short stories from 148 Aesop Fables.

2. Current best-performing model is big and latency is high for practical applications.

3. We discuss multiple times that we are focusing on conversations but QA is single turn conversation. Still far away from real use case.
Instructions
This HIT asks you to read a story, a question and score the appropriateness of different answers on a scale of 1 - 5. Appropriateness means how well the response is naturally connected to the question. A score of 1 means that the response is very inappropriate and it is "not" naturally connected to the question. A score of 5 means that the response is very appropriate, and it is very naturally connected to the question.

Please complete the task by taking the following steps:
1. Read the story.
2. Read the question.
3. Read the responses.
4. Examine how appropriate is each response to the given conversation.
5. Select the appropriateness score for each response.
6. Click the submit button.

Story
A Lion, an Ass, and a Fox were hunting in company, and caught a large quantity of game.

The Ass was asked to divide the spoil.

This he did very fairly, giving each an equal share.

The Fox was well satisfied, but the Lion flew into a great rage over it, and with one stroke of his huge paw, he added the Ass to the pile of slain.

Then he turned to the Fox.

"You divide it," he roared angrily.

The Fox wasted no time in talking.

He quickly piled all the game into one great heap.

From this he took a very small portion for himself, such undesirable bits as the horns and hoofs of a mountain goat, and the end of an ox tail.

The Lion now recovered his good humor entirely.

"Who taught you to divide so fairly?" he asked pleasantly.

"I learned a lesson from the Ass," replied the Fox, carefully edging away.

Question
Is the Fox scared of the Lion?

Responses

<table>
<thead>
<tr>
<th>Answer</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>The fox is not scared of the lion.</td>
<td>1: very inappropriate 5: very appropriate</td>
</tr>
<tr>
<td>The story doesn't say exactly but it is implied he wasn't.</td>
<td>1: very inappropriate 5: very appropriate</td>
</tr>
<tr>
<td>The fox is not scared of the lion.</td>
<td>1: very inappropriate 5: very appropriate</td>
</tr>
<tr>
<td>Because the Fox is &quot;carefully&quot; moving away from the Lion, and gave him most of his portion, one could infer that he is afraid of the Lion.</td>
<td>1: very inappropriate 5: very appropriate</td>
</tr>
<tr>
<td>Yes, the fox is scared of the lion.</td>
<td>1: very inappropriate 5: very appropriate</td>
</tr>
<tr>
<td>Yes, the Fox is scared of the Lion.</td>
<td>1: very inappropriate 5: very appropriate</td>
</tr>
</tbody>
</table>

Submit

Figure 8: Human Evaluation UI for Abstractive QA. Crowd workers were asked to rate answers after reading story and questions.
The Ant and the Grasshopper

One bright day in late autumn a family of Ants were bustling about in the warm sunshine, drying out the grain they had stored up during the summer, when a starving Grasshopper, his fiddle under his arm, came up and humbly begged for a bite to eat.

“What!” cried the Ants in surprise, “haven’t you stored anything away for the winter? What in the world were you doing all last summer?”

“I didn’t have time to store up any food,” whined the Grasshopper; “I was so busy making music that before I knew it the summer was gone.”

The Ants shrugged their shoulders in disgust.

“Making music, were you?” they cried. “Very well; now dance!” And they turned their backs on the Grasshopper and went on with their work.

Question 1:
Don’t the Ants like music?

Answers:

Ground Truth (Human): The story doesn’t say but it isn’t implied that they don’t.
UnifiedQA-T5-11B: they don’t like music.
UnifiedQA-T5-11B-FT: The ants don’t like music because it takes time away from their work.
Unified StoryQA: The ants don’t like music because they think it’s a waste of time.

Question 2:
Is the Grasshopper full of food?

Answers:

Ground Truth (Human): No, the Grasshopper is starving and is begging for food.
UnifiedQA-T5-11B: the Grasshopper is full of food.
UnifiedQA-T5-11B-FT: No, the Grasshopper is starving.
Unified StoryQA: No, the Grasshopper is starving.

Question 3:
Is “bustling about”, a dance move?

Answers:

Ground Truth (Human): Bustling about means to move around doing things, it is not a way to dance.
UnifiedQA-T5-11B: yes
UnifiedQA-T5-11B-FT: No, bustling about is not a dance move.
Unified StoryQA: No, bustling about is not a dance move.

Table 18: Example 1 - Model response to out-of-context questions from seen story. Title of Aesop Fables from our dataset is “The Ant and the Grasshopper”. Note that models fine-tuned on our dataset produce longer and complete answers.
Story:

Belling the Cat

The Mice once called a meeting to decide on a plan to free themselves of their enemy, the Cat. At least they wished to find some way of knowing when she was coming, so they might have time to run away. Indeed, something had to be done, for they lived in such constant fear of her claws that they hardly dared stir from their dens by night or day. Many plans were discussed, but none of them was thought good enough.

At last a very young Mouse got up and said: "I have a plan that seems very simple, but I know it will be successful. All we have to do is to hang a bell about the Cat’s neck. When we hear the bell ringing we will know immediately that our enemy is coming." All the Mice were much surprised that they had not thought of such a plan before.

But in the midst of the rejoicing over their good fortune, an old Mouse arose and said: "I will say that the plan of the young Mouse is very good. But let me ask one question: Who will bell the Cat?"

Question 1:

which animal was the enemy of the cat?

Answers:

Ground Truth (Human): The mice were the animals that were the enemy of the cat.
UnifiedQA-T5-11B: the mouse.
UnifiedQA-T5-11B-FT: The mice were the enemy of the cat.
Unified StoryQA: The mice were the enemy of the Cat.

Question 2:

How old was the mouse?

Answers:

Ground Truth (Human): The mouse was a year old.
UnifiedQA-T5-11B: the old mouse was a hundred years old.
UnifiedQA-T5-11B-FT: The story doesn’t say exactly but it is implied he was young.
Unified StoryQA: The mouse was very young.

Question 3:

What happened to the cat?

Answers:

Ground Truth (Human): Nothing was done to the cat because no mouse would bell it.
UnifiedQA-T5-11B: the cat was killed by the mice.
UnifiedQA-T5-11B-FT: The cat was not able to be rung because the mice were not able to get
Unified StoryQA: The cat was not hung up on a bell.

Question 4:

Who had to bell the Cat?

Answers:

Ground Truth (Human): One brave mouse might be selected to hang a bell on the cat’s neck.
UnifiedQA-T5-11B: the Cat.
UnifiedQA-T5-11B-FT: The Cat would have to be rung by someone.
Unified StoryQA: The story doesn’t say who would bell the cat.

Table 19: Example 2 - Model response to out-of-context questions from unseen story. Title of Aesop Fables from our dataset is “Belling the Cat”. Note that models fine-tuned on our dataset produce longer and complete answers.
Story:
Three Little Pigs
Once upon a time three little pigs, who were brothers left their mother’s home to make a living for themselves. The pigs set out to build their own houses, and the first pig was lazy and built his house with straw. He exclaimed “Brother, I am done and ready to play.”
The second pig was also lazy and built his house with sticks that he found nearby. He said “Okay Brother, let’s play now. I am done building my house.”
The eldest pig decided to spend time to build a strong house with bricks. He said “I don’t have time to play, as I have to build a strong house of bricks.”
The wolf spotted the first two pigs while they played. He thought “I will have these juicy pigs for dinner.” He followed the first pig home and said “Little pig, let me in.” The pig said “No, not by the hair on my chin chinny chin.”
The wolf said “Then I’ll huff, and I’ll puff, and I’ll blow your house in,” and blew the house in.
The pig ran inside his brother’s house of sticks for safety. The wolf followed him to the second pig’s house and said “Little pigs, let me in.” The pigs said “No, not by the hair on our chin chinny chins.”
The wolf said “Then I’ll huff, and I’ll puff, and I’ll blow your house in,” and blew the house in.
The pigs ran to their elder brother’s house made of bricks. The wolf followed them to the third pig’s house and said “Little pigs, let me in.” The pigs said “No, not by the hair on our chin chinny chins.”
The wolf said “Then I’ll huff, and I’ll puff, and I’ll blow your house in.” The wolf tried but could not blow the bricks and the pigs survived.

Question 1:
Which pig is the laziest?
Answers:
Ground Truth (Human): The first pig was the laziest.
UnifiedQA-T5-11B: pig 1
UnifiedQA-T5-11B-FT: The first pig was the laziest.
Unified StoryQA: The first pig was the laziest.

Question 2:
Which pig is the most industrious?
Answers:
Ground Truth (Human): The third pig is the most industrious.
UnifiedQA-T5-11B: the third pig.
UnifiedQA-T5-11B-FT: The eldest pig was the most industrious.
Unified StoryQA: The eldest pig was the most industrious.

Question 3:
What materials should be used to build a house?
Answers:
Ground Truth (Human): It depends. Bricks, straw, sticks are all good to build a house.
UnifiedQA-T5-11B: bricks
UnifiedQA-T5-11B-FT: Bricks are a good material to use to build a house.
Unified StoryQA: The story doesn’t say exactly but it is implied bricks were the best choice.

Question 4:
What is the relationship between pigs?
Answers:
Ground Truth (Human): They are brothers.
UnifiedQA-T5-11B: brothers
UnifiedQA-T5-11B-FT: The pigs were brothers.
Unified StoryQA: The pigs were brothers.

Table 20: Example 3 - Model response to questions for story outside of our dataset. Title of the story is “Three Little Pigs”. This is a popular story that is not part of our dataset. This example contains both in- and out-of-context questions. Note that models fine-tuned on our dataset produce longer and complete answers.