

A Additional Use Cases

In this appendix we describe additional use cases of CROWDAQ. We mainly describe the task definitions and what types of annotation UIs they require. For more information, please refer to the section for examples in our documentation at <https://www.crowdaq.com/>.

A.1 DROP

DROP¹⁴ (Dua et al., 2019) is a reading comprehension dataset which focuses on performing discrete reasoning/operation over multiple spans in the context to get the final answer. The input contexts were sampled from Wikipedia with high frequency of numbers. Annotators from MTurk were asked to write 12 questions per context. The answers can be a number (free-form input), a date (free-form input) or a set of spans (from the context).

Following the original DROP dataset collection guidelines we create a similar annotation task on CROWDAQ. We pose EXAM questions like:

- Which of the following is (not) a good question that you will write for this task?,

where correct options are those questions that require discrete reasoning and incorrect ones are those that do not require such type of reasoning (e.g., questions that require predicate-argument structure look-ups).

On CROWDAQ, we define an `annotation group` that require at least 12 repetitions (`min=12`) of the following: an input box for writing a question, a multiple-choice question for selecting the type of answer (i.e., a number that does not appear in text, a date, a set of spans from the context), and then depending on the type (fulfilled by using `conditions`), we will show an input box, a datetime collector, or a span selector, all from the built-in `annotation types` in CROWDAQ.

The original DROP interface had a lot of complex constraints to ensure the quality of collected data, which can easily be implemented in CROWDAQ using the customized constraints API described at the end of Sec. 3.3.

- A constraint over a set of annotation objects (number, date or spans) to ensure that the worker has provided an answer in at least one of the annotation objects.
- A constraint on question annotation object to allow only how-many type of questions when the answer is a number.

- A task-level constraint to ensure that a worker does not repeat a previously written question within the same task.

A.2 MATRES

MATRES¹⁵ (Ning et al., 2018) is a dataset for temporal relation extraction. The task is to determine the temporal order of two events, i.e., whether some event happened *before*, *after*, or *simultaneously with* another event. MATRES took the articles and all the events annotated in the TempEval3 dataset (UzZaman et al., 2013), and then used CrowdFlower to label relations between these events. Specifically, crowd workers were first asked to label the axis of each event according to the multi-axis linguistic formalism developed in Ning et al. (2018), and then provide a label for every pair of events that are on the same axis.

Similar to Ning et al. (2018), we split the task into two steps: axis annotation and relation annotation. The original UI for axis annotation would show a sentence with only one event highlighted (we can use an `html context` in CROWDAQ), and ask for a label for the axis (we can use the `multiple-choice annotation type`). As for relation annotation, the original UI would show two events on the same axis (we can use an `html context`) and ask for a label for the temporal ordering (`multiple-choice annotation`). Both steps are readily supported by CROWDAQ.

Moreover, CROWDAQ has more advanced features to even improve the original UIs designed in Ning et al. (2018). For instance, when showing two events on the same axis, we can ask if the annotator agrees with the same-axis claim, and only if the annotator agrees with it, we ask for a relation between them. This will reduce axis errors propagated from the axis annotation step.

A.3 TORQUE

TORQUE (Ning et al., 2020) is a reading comprehension dataset composed of questions specifically on temporal relations. For instance, given a passage, “*Rescuers searching for a woman trapped in a landslide said they had found a body,*” typical questions in TORQUE are:

- What happened before a woman was trapped?
- what happened while a woman was trapped?

Annotators of TORQUE are required to identify events from a passage, ask questions that query

¹⁴<https://allennlp.org/drop>

¹⁵<https://github.com/qiangning/MATRES>

temporal relations, and answer those questions correctly. The original instruction and qualification exam for TORQUE are publicly available at <https://qatmr-qualification.github.io/>. Since its qualification exam is already in the format of multiple-choice questions, we can easily transfer it to CROWDAQ.

The original UI for the main annotation process of TORQUE is also available at <https://qatmr.github.io/>. On each passage, it has the following steps (corresponding components of CROWDAQ in parentheses): (1) label all the events in the passage (span selector); (2) answer 3 pre-defined, warm-up questions (span selector); (3) write a new question that queries temporal relations (input box); (4) answer the question (span selector) using those events labeled in Step 1 (customized constraints); (5) repeat Step 3 & 4 for at least 12 times (min=12 in the annotation group). CROWDAQ supports all these features.

A.4 VQA-E

VQA-E¹⁶ (Visual Question Answering with Explanation; Li et al., 2018) is a dataset for training and evaluating models that generate explanations that justify answers to questions about given images. Besides constructing such a dataset, CROWDAQ can also be used to evaluate the plausibility of an explanation (i.e., whether a generated explanation supports the answer in the context of the image), and its visual fidelity (i.e., whether the explanation is grammatical, but mentions content unrelated to the image—anything that is not directly visible and is unlikely to be present in the scene in the image).

We use CROWDAQ’s `contexts` with the type `image` to display a given image, the `html` context to display a question and answer, and CROWDAQ’s annotation of the type `multiple-choice` to ask the following prompts:

- Does the explanation support the answer?
- Is the explanation grammatically correct?,

We then use CROWDAQ’s `condition` to add another multiple-choice question:

- Does the explanation mention a person, object, location, action that is unrelated to the image?, that is shown only if the annotator has judged the explanation to be grammatical in the previous step. Conditioning helps us differentiate between ungrammaticality and visual “infidelity.” Finally, as an alternative way of measuring fidelity, we use the

annotation with the type `multi-label` to display the following prompt:

- Select nouns that are unrelated to the image, where nouns are extracted from the explanation, and the difference between `multiple-choice` and `multi-label` is that the latter allows for more than one options to be selected. However, in the EXAM, we teach annotators to select such nouns with the multiple-choice questions:

- Is the noun `<insert_noun>` related to the image? Because judging explanation plausibility and fidelity is difficult and subjective, CROWDAQ’s TUTORIAL and EXAM are of a great value.

A.5 Answering Information-Seeking Questions about NLP Papers

This is an ongoing dataset creation project aiming to collect a question answering dataset about NLP papers, where the questions written by real readers of NLP papers with domain expertise who have read only the titles and the abstracts of research papers and want to obtain information from the full text of the paper. In this case study, we focused on the more challenging task of obtaining answers, assuming that the questions are available. We used CROWDAQ to design a TUTORIAL for instructing workers, and an EXAM for qualifying them.

The task involves two steps: The first is identifying *evidence* for the question, which can be a passage of text, a figure, or a table in the paper that is sufficient to answer the question. The second step is providing an *answer*, which can be text that is either selected from the paper or written out free form, or a boolean (Yes/No). Some questions may be identified as being unanswerable, and do not require evidence and answers. We used the TUTORIAL and EXAM features in CROWDAQ to teach the workers and evaluate them on the following aspects of the task:

- **Identifying sufficient evidence** Quite often papers have several passages that provide information relevant to the question being asked, but they do not always provide all the information needed to answer them. We identified such passages that are relevant to real questions, and made TUTORIAL and EXAM questions of the form, “Is this evidence sufficient to answer the question?”
- **Preference of text over figures or tables** The task requires selecting figures or tables in NLP papers as evidence only if sufficient

¹⁶<https://github.com/liqing-ustc/VQA-E>

information is not provided by the text in the paper. To teach the workers this aspect of the task, we made multiple-choice questions showing a figure or a table from a paper, and some text referring it, and asked the workers questions of the form, “Given this chunk of text, and this figure from the same paper, what would be good evidence for the question? (A) Just the figure (B) Just the text (C) Both (D) Neither”.

- **Answer type** Since the task has multiple answer types, including extractive (span selection) and abstractive (free form) answers, it is important to teach the workers when to choose each type. We made multiple-choice questions with examples showing potential span selections, comparing them with potential free-form answers, asking the workers to choose the correct option for each case.

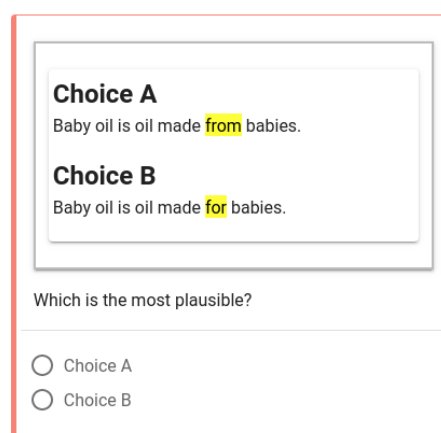
A.6 Acceptability Judgement Tasks

This is another one of the ongoing projects on CROWDAQ. *Acceptability judgements* are a common tool in linguistics for evaluating whether a given piece of text is grammatical or semantically meaningful (Chomsky and Lightfoot, 2002; T Schütze, 2016). Popular approaches for performing acceptability judgements include having annotators to make a Boolean evaluation of whether or not the text is acceptable, as well as forcing annotators to pick from a pair of sentences which is the most acceptable. Both of these approaches can be easily formulated as a sequence of multiple choice questions. Although multiple choice surveys are simple to design and deploy on many crowdsourcing platforms, CROWDAQ users have found some features particularly useful.

First, the ability to easily design EXAMS to qualify users. Due to their simplicity, multiple-choice questions are easily gamed by crowd workers using bots or by answering questions randomly. In a pilot study, one requester on CROWDAQ found that over 66% of participants in their acceptability judgement survey were bad actors if they directly launch the task on MTurk. By setting a performance threshold on the EXAM these actors were automatically disqualified from participating in the TASK without the user needing to setup and manage custom MTurk Qualifications. Although commercial crowd sourcing platforms provide similar qualification control, they have paywalls, are not

open-sourced, and are not as flexible as CROWDAQ (for instance, one can use only the qualification feature of CROWDAQ, while commercial platforms require using and paying the entire pipeline).

Second, the flexibility CROWDAQ allows when specifying contexts. In one case, a user received multiple requests from crowd workers to visualize which tokens differed between pieces of texts in order to increase the speed at which they were able to annotate longer pieces of text. Since contexts allow the insertion of arbitrary HTML, this user was able to easily accommodate this request by inserting `<mark>` tags around the relevant tokens. An illustration of one of their questions is provided in Fig. 2.



The screenshot shows a user interface for an acceptability judgement task. It features two choices, Choice A and Choice B, each containing the sentence "Baby oil is oil made from babies." The word "from" in both sentences is highlighted in yellow. Below the choices, the question asks "Which is the most plausible?" and provides two radio button options: "Choice A" and "Choice B".

Figure 2: Illustration of an acceptability judgement task deployed on CROWDAQ. Because contexts can contain html, this CROWDAQ user was easily able to highlight relevant spans of text for crowd workers using `<mark>` tags.

B Screenshots

In this appendix we show screenshots corresponding to the JSON specifications described in Sec. 2 and Sec. 3. We also provide an overview of all these figures in Table 1.

Reference	Description
Figure 1	An illustration of data collection using CROWDAQ and MTurk (this figure is in the main text)
Figure 2	An illustration of an acceptability judgement task deployed on CROWDAQ
Figure 3	A specification and visualization of TUTORIALS
Figure 4	A specification and visualization of EXAMS
Figure 5	A visualization of how CROWDAQ renders the <code>context</code> specified in Section 3
Figure 6	An illustration of selecting a span from a text and answering a question about it on CROWDAQ
Figure 7	An example of repeated annotations on CROWDAQ
Figure 8	An example of how CROWDAQ's <code>condition</code> works
Figure 9	An illustration of a violated <code>constraint</code> on CROWDAQ
Figure 10	A visualization of how to create an EXAM on CROWDAQ
Figure 11	An visualization of how to launch an exam to MTurk in the client package that comes with CROWDAQ
Figure 12	Feature visualizations: (a) Distribution of participants' scores. (b) Individual scores of each participant. (c) Participants' performance on each question with quick preview of individual questions.
Figure 13	Feature visualizations: (a) Average time workers spend on the task. (b) Progress monitoring. (c) Quick preview of individual annotations.

Table 1: An overview of figures.

```

"question_set": [
  {
    "type": "multiple-choice",
    "question_id": ...,
    "context": [{
      "type": "text",
      "text": "As of Tuesday, 144 of the state's
then-294 deaths involved nursing
homes or longterm care facilities."
    }],
    "question": {
      "question_text": "In \"294 deaths\", what
should you label as the quantity?",
      "options": {"A": "294", "B": "294 deaths"}
    },
    "answer": "A",
    "explanation": {
      "A": "Correct",
      "B": "In our definition, the quantity
should be \"294\"."
    }
  }
],
...
]

```

Figure 3: Specification and visualization of TUTORIALS. In this particular TUTORIAL, there are eight questions and the example participant has only made choice on one of them. Please see <https://beta.crowdaq.com/w/tutorial/qiang/CrowdAQ-demo> for this interface.

```

"question_set": [
  {
    "type": "multiple-choice",
    "question_id": "...",
    "context": [
      {
        "type": "text",
        "text": "As of Tuesday, 144 of the state's then-294 deaths involved nursing homes or longterm care facilities."
      }
    ],
    "question": {
      "question_text": "In \"294 deaths\", what should you label as the quantity?",
      "options": {"A": "294", "B": "294 deaths"}
    },
    "answer": "A"
  },
  ...
]

```

Figure 4: Specification and visualization of EXAMS. In this particular EXAM, the requester has specified that every time a participant will see five questions randomly sampled from the pool, and every participant only has two opportunities to pass it. Please see <https://beta.crowdaq.com/w/exam/qiang/CrowdAQ-demo> for this interface.

```

"contexts": [
  {
    "label": "Note",
    "type": "html",
    "html": "<p>Remember to ...</p>",
    "id": "note"
  },
  {
    "type": "text",
    "label": "The snippet was from an article published on 2020-05-20 10:30:00",
    "text": "As of Tuesday, 144 of the state's then-294 deaths involved nursing homes or longterm care facilities.",
    "id": "snippet"
  }
]

```

Figure 5: How CROWDAQ renders the context specification in Sec. 3. A major difference from TUTORIALS is that the participant will not see the answers. Please see https://beta.crowdaq.com/w/task/qiang/CrowdAQ-demo/quantity_extraction_typing for this interface.

```

"annotations": [
  {
    "type": "span-from-text",
    "from_context": "snippet",
    "prompt": "Select one quantity from below.",
    "id": "quantity",
  },
  {
    "type": "multiple-choice",
    "prompt": "Is this quantity related to COVID-19?",
    "options": {
      "A": "Relevant",
      "B": "Not relevant"
    },
    "id": "relevance"
  }
]

```

Figure 6: In this UI the annotator is asked to select a valid quantity and then choose whether it is relevant to COVID-19.

```

"annotation_groups": [
  {
    "annotations": [
      {"id": "quantity", ...},
      {"id": "relevance", ...}
    ],
    "id": "quantity_extraction_typing",
    "title": "COVID-19 Quantities",
    "repeated": true, "min": 1, "max": 3
  }
],

```

You need to provide 1-3 responses. You have saved 2 responses so far.

[+ ADD A NEW RESPONSE](#)

[DELETE](#)

Select one quantity from below.

As of Tuesday, 144 of the state's then-294 deaths involved nursing homes or longterm care facilities.

x (Saved) Selected span is: 144

Is this quantity related to COVID-19?

Relevant

Not relevant

[DELETE](#)

Select one quantity from below.

As of Tuesday, 144 of the state's then-294 deaths involved nursing homes or longterm care facilities.

x (Saved) Selected span is: 294

Is this quantity related to COVID-19?

Relevant

Not relevant

Figure 7: How to collect a group of annotations repeatedly. Note the annotation is valid only when one provides 1-3 responses because the requester specifies so. In the above we have added 2 responses.

```

"annotations": [
  {"id": "quantity", ...},
  {"id": "relevance", ...},
  {
    "id": "typing",
    "type": "multiple-choice",
    "prompt": "What type is it?",
    "options": {
      "A": "Number of Deaths",
      "B": "Number of confirmed cases",
      "C": "Number of hospitalized",
      ...
    },
    "conditions": [
      {
        "id": "relevance",
        "op": "eq",
        "value": "A"
      }
    ]
  }
],

```

The image shows two panels of an annotation interface. The top panel, labeled 'Enabled', shows a text box with the sentence 'As of Tuesday, 144 of the state's then-294 deaths involved nursing homes or longterm care facilities.' Below it, a 'Selected span is: 144' field is shown. A question 'Is this quantity related to COVID-19?' has 'Relevant' selected. Below that, 'What type is it?' has 'Number of Deaths' selected. The bottom panel, labeled 'Disabled', shows the same text box and span field, but 'Not relevant' is selected for the COVID-19 question, and the 'What type is it?' question is not visible.

Figure 8: A new question about the type of the quantity is enabled only if we select “Relevant” in the previous one. Above: Enabled. Below: Disabled.

The image shows a text box with the sentence 'As of Tuesday, 144 of the state's then-294 deaths involved nursing homes or longterm care facilities.' Below it, the 'Selected span is: 144_' field is shown. An orange error message reads: 'The quantity should only end with digits, letters, or %.'

Figure 9: When any constraint is violated, the annotator will receive an error message (the text in orange) and also prohibited from proceeding (not shown here). In this screenshot, the annotator wrongly included an extra space.

New Exam

id

[0-9a-zA-Z]*

Instruction Id

Tutorial Id

Question Set ID

Max Allowed Attempts

Number of assignments

Number of question per exam

Pass Grade

[CREATE EXAM](#)

Figure 10: How to create an exam on CROWDAQ.

```
config_editor = HitCreatorNotebook()
```

AWS Account Configurations:

AWS Profile:

Use Mturk Sandbox?

HIT Configurations:

Title of your HIT

Description of your HIT

Keyword of your HIT

Reward of your HIT \$

Lifetime (in minutes) of your HIT

Session length (in minutes) of your HIT

Auto Approve delay time (in minutes) of your HIT

Require Master Workers?

Require US Workers?

```
EXAM_ID="quantity_extraction_exam"
assert EXAM_ID is not None and EXAM_ID != ""

config_editor.create_hit(sess.urls.exam(EXAM_ID), 500) # we allow 500 HITS in the exam
```

Figure 11: Launch an exam to MTurk in the client package that comes with CROWDAQ.



Figure 12: Some handy features that CROWDAQ provides on EXAMS. (a) Distribution of participants' scores. (b) Individual scores of each participant. (c) Participants' performance on each question with quick preview of individual questions.

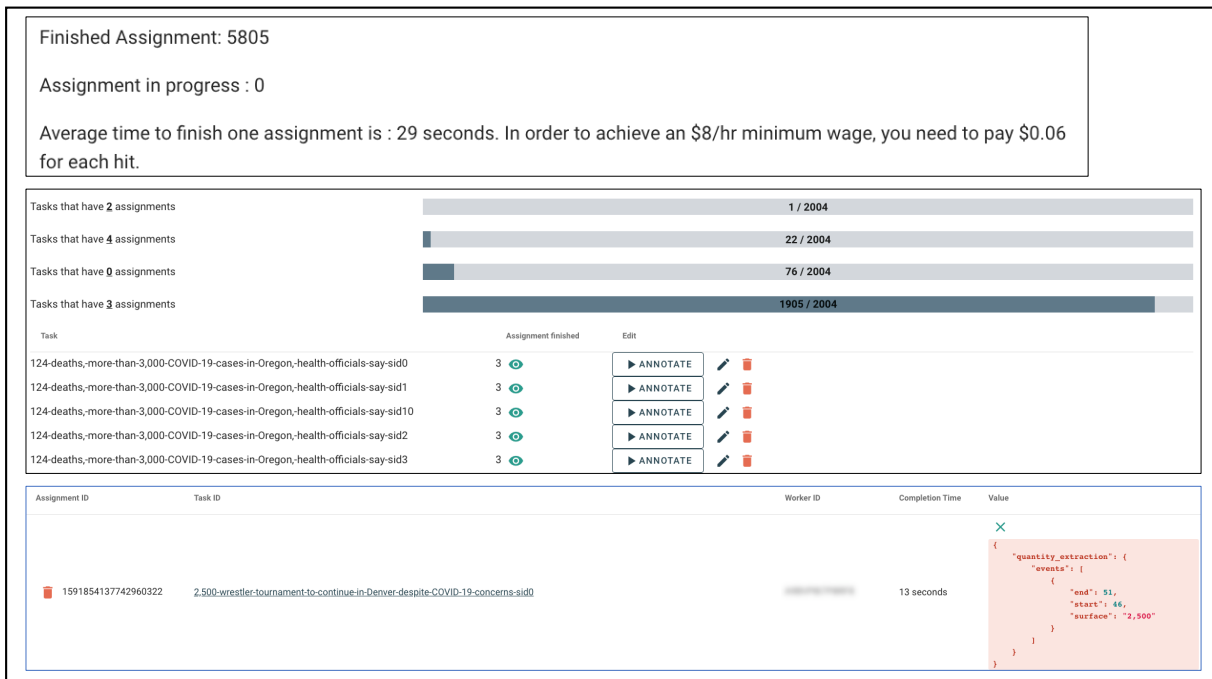


Figure 13: Some handy features that CROWDAQ provides on TASK SETS. Above: average time people spend on the task. Middle: progress monitoring. Below: quick preview of individual annotations.