

AI Engineering

6.S978 / Fall 2026

Homework 1 Public overview

Redaction for a Public Help Center

These are selected excerpts from Homework 1, presented as a public overview to share some of our intuitions around evaluating student learning for AI engineering.

The current release does not include the development dataset, the starter code, the leaderboard functionality and hidden evaluation aspects, or the client Q&A bot, which supplies the last-mile information for the students.

If you are interested in these materials, please feel free to get in touch with Omar Khattab at okhattab@mit.edu.

You are an AI engineering consultant. You are getting hired by Bespoke Support, a company that helps businesses troubleshoot software. They are hiring you to build a tool, which will process their customer support conversations for public release on its help center.

Currently, the message dump that Bespoke Support want to share contains lots of private customer information. Publishing those details unredacted could violate a customer's privacy or even give someone access to their systems! You are here to help them.

For each case, Bespoke gives you a **source** conversation or set of notes, plus **context** from related messages and documentation. Your tool must replace private values with asterisks and leave every other character unchanged.

Source

"The health check fails on my laptop, cedar-17," Sarah Miller wrote. "It started after I installed 2.4.0. Restarting the service did not help."

Alex Chen replied: "This was a bug in 2.4.0. Upgrade to 2.4.1, then run curl localhost:8080/health. Please keep the response in the thread."

"That fixed it," Sarah wrote. "The endpoint returns 200 now. You can reach me at sarah.miller@example.org if you need another test."

Redacted publication

"The health check fails on my laptop, ********," ********** wrote. "It started after I installed 2.4.0. Restarting the service did not help."

Alex Chen replied: "This was a bug in 2.4.0. Upgrade to 2.4.1, then run curl localhost:8080/health. Please keep the response in the thread."

"That fixed it," ***** wrote. "The endpoint returns 200 now. You can reach me at ********************** if you need another test."

The context identifies (1) Sarah as a customer, (2) cedar-17 as her laptop, and (3) Alex as the product maintainer. Notice how in the redacted copy Sarah's details are hidden, but Alex's name and the fix are kept unredacted.

Your program

Implement `redact(packet, llm)` using Qwen3.5-9B, Qwen3.7 Flash, or both. In your function, decide which judgments to delegate to a model and which properties to enforce in code.

The package from Bespoke Support grants you the following resources:

36 examples demonstrating their redaction policy

$35 in API credit for development and leaderboard evaluations

7 submissions to get a program evaluated by their data compliance team

30 questions to clarify Bespoke's needs, at no cost to your API credit

Bespoke's brief

Bespoke Support

Public help center / Project brief

Hi there! Thank you for helping our team at Bespoke.

We want to publish these troubleshooting conversations so other engineers can learn from them. Remove private information without losing the explanation of the problem and its fix. Return an exact copy with private text replaced by asterisks and every other character unchanged. We've provided examples and related context, and you can ask us to clarify our publication requirements.

The tool we'd like you to build

Your job is to implement the function `redact(packet, llm)`.

Each call receives a `packet` dictionary, consisting of: a `source` text to redact, related texts in `context`, and the `publication_agreement`.

We also supply `llm`, an object for calling the two allowed language models. Use `llm.complete(prompt, model="qwen37f")` to get a response as text, or choose `model="qwen35"` for the 9B model. It records token usage automatically.

For example:

```
packet = {
    "source": """Sarah Miller: My laptop, cedar-17, fails its health check.
Alex Chen: Upgrade to 2.4.1, then restart the service.
Sarah: That fixed it. The endpoint returns 200 now.""",
    "context": [
        "Sarah Miller is the customer; cedar-17 is her laptop.",
        "Alex Chen maintains the software product.",
    ],
    "publication_agreement": "standard",
}
```

`publication_agreement` is either `standard` or `volume-confidential`. You can ask us what these agreements require through Client Q&A.

Return the redacted text

`redacted_text`	The complete `source` but redacted, with exactly the same number of Unicode characters. At each position, either keep the original character or replace it with an ASCII asterisk `*`.

Examples & client requirements

Context can arrive in any order. Use its contents, not the positions of texts in the list, to establish any necessary chronology. Your system should support up to **120,000 characters** across source and context, and up to **160 context strings**.

Examples to work with

In `data/development-v1.json`, only each row's `case` reaches your function. Other metadata like IDs, titles, groups, and labels will not be available for your system to use.

Related examples from the same troubleshooting case share a `source_cluster`. If you set aside examples for testing, keep each case together so you test on new cases, not another excerpt of one you've already used.

For your development, we share some expert labels with you on a small amount of troubleshooting records. The labels come in two lists: `redact` contains values we want concealed, whereas `keep` contains values you might think to conceal but which we actually want published!

Each item gives its start and end positions and the text. Keep labels are examples, not an inventory of everything that matters. All other text should remain untouched.

Feedback on your program

Our data compliance team can evaluate up to seven versions of your program on conversations you haven't seen and return a leaderboard score. You can experiment with the supplied examples as often as you like without using these seven opportunities.

From Bespoke Support

Our team works with these conversations every day. Some of our conventions have never been written down, and we may not realize which decisions need explaining. If something is unclear, please ask.

We're still handling support requests, so please bring one specific question at a time. To keep our guidance consistent, our liaison can only relay the team's approved answers, word for word.

We've set aside time for 30 messages. Greetings, clarification requests, and repeated questions count too, so use these opportunities to resolve decisions about your tool.

Thank you,
The Bespoke Support team

Starter package

1. Start with one example

Open `explore.ipynb` in VS Code or Jupyter. It loads one example, shows its source and context, runs your program, and highlights the mistakes alongside the model usage.

You can iterate on the assignment in two ways. Either edit `solution.py` in your editor, or write your program in a notebook cell beginning with `%%writefile solution.py`. Running that cell writes your imports, helpers, and `redact` function to the file. In both workflows, the notebook tests that Python file. In general, you submit that file, not the notebook.

We strongly recommend working on one example at a time: skim it, run your program, understand some of its failures if any, and try some tweaks. Repeatedly changing the program in ad-hoc ways and rerunning the entire dataset can quickly consume your credit without teaching you much.

Once a change makes sense, check it on a few other examples. The development set is all yours; you can do whatever you want on it, but your system must generalize to new unseen instances and, ideally, to a different capable LLM.

Running experiments from the terminal

```
python3 -m hw1_utils run --limit 1 --out runs/first
python3 -m hw1_utils check runs/first --details
```

Omit `--limit` to run all examples. Use `--workers 4` to run four examples at a time in parallel, and provide a new output directory for each experiment. Predictions, labels, and usage are saved together.

2. Build your program

The baseline in `solution.py` uses one plain model call. You are free to change it as you see fit. Edit that file as you work in the notebook; each run reloads it!

You can define helper functions in `solution.py` and call them from `redact`. You can also import helpers from your own Python files; include those files when you package your submission.

Grading runs Python **3.13** in an isolated environment with the standard library and no direct network access. Include your Python helper files, since third-party packages, threads, subprocesses, and runtime installation are not supported.

Submit a version

```
python3 -m hw1_utils package
```

This saves a new dated ZIP in `submissions/`, containing a copy of your current `solution.py`. Functions defined in that file are already included. If your code imports your own `helpers.py` and `formatting.py` files, package them too: `python3 -m hw1_utils package --include helpers.py formatting.py`

Choose final versions

Choose up to two saved versions for final evaluation. If you do not choose, we use your two highest leaderboard scores, with earlier submissions breaking ties. We grade each version separately and keep the better total.

Evaluation & grades

Your leaderboard results contribute to your grade. After the deadline, we evaluate your selected versions on a separate, private set of examples that represents deployment.

Cost of the whole program

Model	Units / input token	Units / output token
Qwen3.5-9B	1	3
Qwen3.7 Flash	4	12

The supplied `llm` records usage automatically. Evaluation counts every call, including retries and calls that produce no valid publication. Output tokens include reasoning tokens once; cached input tokens count at the full input rate.

```
Net score = max(0, mean quality − mean token units / 25,000)
```

Quality 60 and 100,000 mean token units yield net 56. These units create an engineering trade-off; they are not dollars. Your development calls are billed separately by OpenRouter.

Your grade

Component	Points
Report	50
Public leaderboard score (revealed via any-time submissions)	35
Private held-out score (revealed after deadline)	35

The homework is out of 100 with up to 20 bonus points.

On the public leaderboard, credit increases proportionally from zero to **35/35 points at a net score of 20**. Higher scores earn no additional public-set credit. Beyond that target, focus on whether your improvements generalize: the private evaluation rewards performance on new examples.

On the private held-out set, your points are **35 × your net score / 100**. For example, a net score of 60 earns 21/35 points.

The report

You can submit to the leaderboard before your report is ready. A report is required for your final submission, and detailed instructions will follow.

As you work, keep notes on your hypotheses, what you tested, and the failures you observed. Record what changed your mind, including experiments that did not help. These notes will give you the evidence to write your report later, without having to reconstruct the process from memory.

Leaderboard

Staff systems only. Student submissions are omitted from this overview.

Publication quality

Quality measures how well your publication matches Bespoke's requirements. The local checker reports missing redactions, unnecessary redactions, output validity, and token use. The client might be able to tell you more about how they weigh different errors.

Name	Net score	Quality	Cost
STAFF **GPT-6 Sol via engineered pipeline** Out of competition. Careful system, with full access to all client answers.	**N/A** way too expensive	87.29	72x starter
STAFF **stronger baseline**	**57.11**	66.31	9.20
STAFF **GPT-6 Sol via basic prompt** Out of competition. Single prompt, without access to client Q&A.	**N/A** way too expensive	29.68	48x starter
STAFF **simple baseline**	**11.67**	12.01	0.33
STAFF **starter**	**0.00**	0.00	0.44

Sol is here for comparison, ranked by quality. Its cost is shown relative to the starter.

We included three simple baselines to give you a feel for the scores different systems can achieve. There is room to do better, and we encourage you to try. But be careful about submitting repeatedly to try to game the public leaderboard. The real challenge is to build a system that generalizes to unseen and potentially domain-shifted data.