CS3317: Artificial Intelligence

Final Project

Multimodal Classification Challenge

Instructor: Dr. Tao Huang | Worth 40% of the course grade | Teams of 1–3

TL;DR

You will receive multimodal features and raw text from an **undisclosed source domain** and predict **18 anonymized labels** per example. This is a **multi-label** classification problem (each example can have any subset of the 18 labels). Score is **F1-macro** on a held-out test set of 2,000 examples. The leaderboard ranks teams; your position on it determines part of your grade.

The methodological twist: **30% of test examples have one entire modality zeroed out** at inference time (image features all zero, *or* both text feature vectors zero and the raw text empty). Training data is clean. Your design must handle this train/test distribution gap. *How you do this is up to you.* That is the project.

What graders are looking for, in one sentence: a thoughtful, well-justified *method choice* that addresses both the missing-modality problem and the few-shot regime, supported by clean empirical analysis. The leaderboard score alone is only 15 of 100 points.

Key dates (Beijing time).

- **Group registration: 23:59, Friday May 29, 2026.** Form your team (1–3 members) and register on the leaderboard webpage by this date. After registration, the instructor creates your Canvas team accordingly; only then can you upload the final zip on Canvas. Late leaderboard registrations are technically possible but may miss the Canvas-team creation window — email the instructor if you register late.
- **Leaderboard closes: 20:00, Friday June 12, 2026.** No submissions accepted after this time. Your best score by then is your final leaderboard rank.
- **Code, report, and presentation slides due: 23:59, Sunday June 14, 2026** on Canvas. You have two days after the leaderboard closes to write the report and prepare slides.
- **Presentations:** on the last class meeting. Each team gives an 8-minute talk followed by open Q&A from the instructor and peers.

1 Task

You are given four splits of the same underlying dataset (Table 1). Every example has the same per-modality content: an image-feature vector, two text-feature vectors (CLIP- and BERT-encoded), and the raw text the features were computed from. Labels are multi-hot vectors of length 18; classes are anonymized integers `label_0` through `label_17`.

Table 1. Splits provided in the bundle.

Split	Size	Labels visible?	Test-time modality dropout?
`train`	2,000	yes	no (clean)
`val`	300	yes	no (clean)
`unlabeled`	5,000	no	no (clean)
`test`	2,000	no (graded online)	yes (30% have one modality zeroed)

2 Data: what you receive, file by file

Download: https://taohuang.info/cs3317/coursework_final/assets/student_bundle.tar.gz (∼80 MB tar.gz). After extracting, you should see a top-level `student_bundle/` directory with a `data/` subdirectory containing all CSV and NPZ files, plus `starter/`, `tests/`, and `docs/` containing code and templates.

For every split `S` ∈ {`train, val, unlabeled, test`} the bundle's `data/` subdirectory contains four files:

- `data/S_image.npz` array '`features`' of shape $(N, 768)$ float32
- `data/S_text_CLIP.npz` array '`features`' of shape $(N, 768)$ float32
- `data/S_text_BERT.npz` array '`features`' of shape $(N, 768)$ float32
- `data/S.csv` columns `example_id, text_raw, [labels or mask]` (see below).

Row i of every file for a split corresponds to the same example. The CSV's `example_id` column is the canonical mapping. The CSV schema varies by split:

`data/train.csv`, `data/val.csv`	`example_id, text_raw, label_0, ..., label_17`
`data/unlabeled.csv`	`example_id, text_raw`
`data/test.csv`	`example_id, text_raw, has_image, has_text`

The end-to-end inference pipeline implied by these files is shown in Figure 1.

3 The methodological twist: test-time modality dropout

This is the most important paragraph in the handout.

> At test time, ∼**30% of examples have one randomly-selected modality replaced with zeros**, indicated by the mask. Training data is clean — consider how to bridge this train/test distribution gap.

For test examples, the `has_image` and `has_text` columns in `data/test.csv` encode the mask. Figure 2 illustrates the three scenarios.

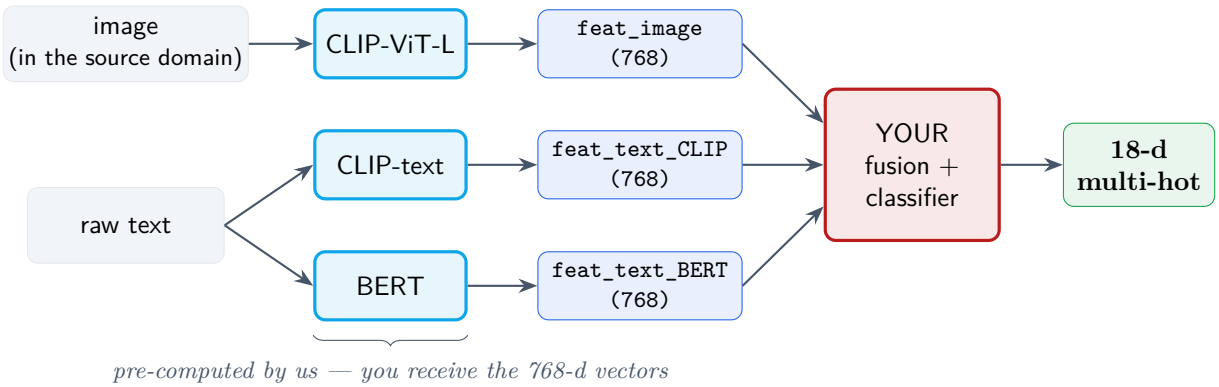

Figure 1. One possible pipeline from inputs to a multi-label output. The encoder column is pre-computed by us — you receive the 768-d feature vectors directly. Everything to the right of the feature column is your design choice. Nothing here is required: you may ignore the pooled text features and re-encode `text_raw` with your own model, take per-token features for cross-attention, route hard examples through an LLM, or design an entirely different architecture, as long as the rules in §8 are respected.

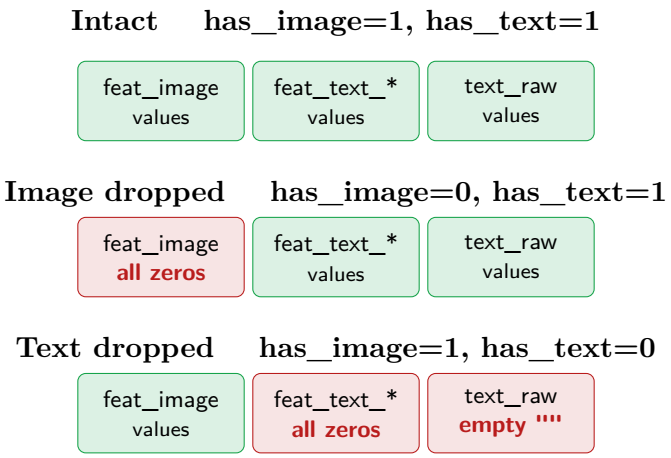

Figure 2. Three scenarios on the test set, pre-applied. When `has_text=0`, BOTH text feature files *and* `text_raw` are zeroed/emptied simultaneously — they go together. The mask is provided so you can identify these examples, not so you have to apply the zeroing yourself.

4 Setup & running the baseline

```
pip install numpy torch tqdm
python -m starter.baseline_concat_mlp     # ~1 minute, writes submission.csv
python -m tests.sanity_checks --submission submission.csv
```

The baseline trains a small MLP on concatenated features (image + CLIP-text + BERT-text = 2304-d) with BCE loss and 0.5 threshold, then writes `submission.csv`. Upload it to the leaderboard to see your first score.

5 Compute: SJTU HPC

A teaching HPC account on SJTU's HPC platform (*Jiao Wo Suan*) has been requested for every student. Using it is **optional**, not required.

- **Login portal:** https://my.hpc.sjtu.edu.cn/
- **Documentation:** https://docs.hpc.sjtu.edu.cn/

- **Your account:** the instructor has sent each student a personal username and password via **Canvas direct message**. If you did not receive the message, email the instructor at t.huang@sjtu.edu.cn.

Teaching-account quotas & rules (from the HPC operations team):

- Each account is capped at **1 concurrent job**, **1 CPU node**, **1 GPU card**, and a **12-hour wall-clock limit** per job.
- Prefer the π**2.0 cluster**.
- The login node is `pilogin`. **Do NOT run jobs on the login node, or your account will be banned.**
- Use the `cpu` queue for CPU work and the `dgx2` queue (32 GB per card) for GPU work.
- GPU resources are currently tight; plan your job submission times accordingly.

The original notice in Chinese is reproduced verbatim in the README and on the course web page.

6 Submission format

A CSV with exactly this header and 2,000 data rows (one per test example, any order):

```
example_id,label_0,label_1,label_2,...,label_17
ex_00123,0.81,0.04,0.12,...,0.03
...
```

Values may be probabilities in $[0, 1]$ or binaries in $\{0, 1\}$. The leaderboard thresholds probabilities at 0.5 by default. F1-macro is the leaderboard metric; F1-micro is reported but does not affect rank.

7 Leaderboard mechanics

- Each team is anonymized on the public leaderboard — only your group nickname is shown.
- **5 submissions per day** per team, **50 submissions total** over the 4 weeks.
- Scoring happens immediately after upload. Your rank is by best F1-macro, with ties broken by who reached the score first.
- Tier-graded points are assigned at the end of the project from your final rank.

8 Constraints

- **Total model parameters $\leq$ 100M** at inference time, including any pretrained weights you load. This rules out fine-tuning large pretrained backbones; it does *not* rule out using them via API (see below).
- **No external datasets** beyond what we provide.
- **No attempts to deanonymize the source domain.** Detected attempts are an automatic fail. The features are pre-computed and you do not need to know the source.
- **LLM API calls are allowed** (paid or free), but must be disclosed in your report: which provider/model, where in your pipeline, and approximate cost in CNY. Pure prompting alone will be marked low on Method Design.

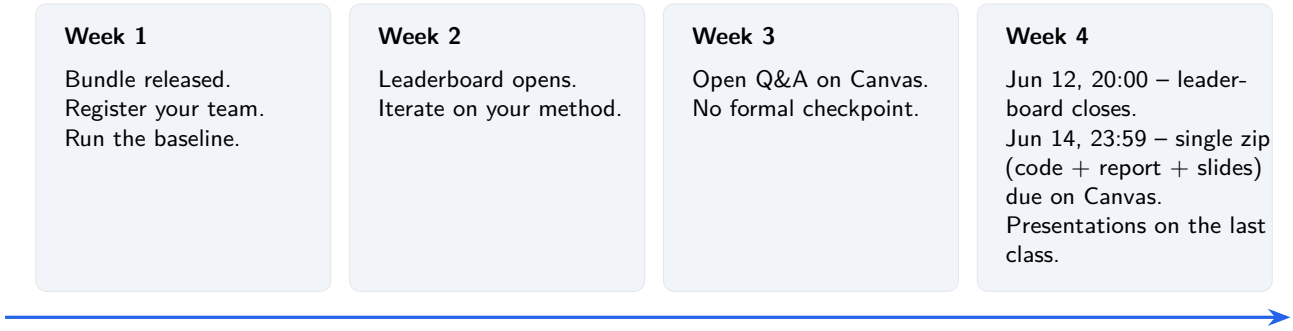

Figure 3. 4-week timeline. Exact dates are listed on the Canvas course page.

Component	Pts
Method design & implementation	25
Empirical analysis (ablations, per-modality, missing-modality robustness)	25
Method Choice Rationale (in the report)	10
Report quality (clarity, figures, scientific reasoning)	20
Final leaderboard tier — top 10%=15, top 30%=12, top 50%=9, rest with a valid scoring submission=6, no valid submission=0	15
Reproducibility (code runs, seeds fixed, results match report)	5
Total	**100**

9 Timeline

10 Grading rubric (100 pts)

The 8-minute presentation + Q&A on the last class is graded separately as the 10% course-level presentation component; it is NOT part of these 100 project points.

11 Required deliverables

All Beijing time. Late submissions are not accepted.

1. **Final leaderboard submission** (CSV via the leaderboard webpage), uploaded **by 20:00, Jun 12, 2026**.
2. **One single zip on Canvas, by 23:59, Jun 14, 2026**, containing *all three* of:
   - **Code** (all source needed to reproduce your final submission, plus a `README` describing how to run it).
   - **Report** as a PDF: max 8 pages, NeurIPS-style — use `docs/report_template.tex` or `report_template.docx`. Required sections: "Method Choice Rationale", "Missing-Modality Robustness Analysis", "Resources Used", and "AI usage" (see §12).
   - **Presentation slides** as a PDF.
3. **Presentation**: 8-minute talk + open Q&A from the instructor and peers on the last class. Graded separately as the course's 10% presentation component; not part of the project's 100 points.

12 Academic integrity

- Discussing high-level ideas with other teams is allowed. Sharing code, submissions, model checkpoints, or labels is not.

- Your final submission must come from your own implementation.
- Any attempt to deanonymize the source domain or the test labels (e.g. reverse-searching the raw text on the web for the purpose of recovering identity) is a violation and an automatic fail for the project.

AI usage disclosure (required)

AI tools — LLM APIs in your pipeline, coding assistants (Cursor, Claude Code, GitHub Copilot, ...), report-writing assistants, autonomous coding agents — are *permitted*. They are not a free pass on accountability.

In your report, include an "AI usage" subsection that itemizes **every tool you used and exactly what each accomplished**. Be specific. Examples of adequate disclosure:

- "Cursor wrote the data-loading boilerplate in `load.py` and the argparse plumbing in `train.py`; we wrote the model and training loop."
- "Claude API: one chat session to debug a numpy shape mismatch in our training loop. ∼3 turns, ∼CNY 0.20."
- "GPT-5 helped revise wording in the abstract and the discussion; we wrote the methodology and analysis sections."

Examples of *inadequate* disclosure: "I used ChatGPT", "some parts were AI-assisted". If a Q&A question reveals that you cannot explain a component you claimed as your own work, that is a violation.

The deliverable rule: **the design decisions must be yours, and you must be able to defend every part of your submission in person.**

> **One more time, because it is the project's spine:** the test set has ∼30% modality dropout, the training set does not. If your method does nothing to bridge that gap, you will leave a lot of score on the table.