PROPOSAL

STATS 507, Fall 2025

Name: YI WANG

Email: wangyii@umich.edu

Time spent on this homework: n hours

I did not discuss this homework with anyone.

Overview

Background and Motivation

Persona-consistent dialogue aims to keep generated conversations aligned with explicit role descriptions (personas). In applications such as customer support, tutoring companions, interactive training, and entertainment, persona consistency influences user trust, engagement, and downstream decision quality. Open-ended dialogue seldom has a single correct answer; therefore, traditional automatic metrics (e.g., BLEU, ROUGE-L) can under-estimate valid responses when paraphrases or alternative yet reasonable phrasings are used. A rigorous study should provide a transparent, reproducible pipeline that combines quantitative metrics with interpretable case analysis, supports safe iteration, and scales under limited compute.

This project presents a complete, executable end-to-end system in Python that (1) ingests persona text, (2) adapts a pretrained sequence-to-sequence model via parameter-efficient fine-tuning, (3) evaluates with multiple metrics and interpretable case files, and (4) packages visualizations and deliverables. The system uses Hugging Face tooling and a lightweight adaptation strategy (LoRA/PEFT), together with controlled partial unfreezing and retrieval-augmented inputs (RAG) to strengthen persona grounding.

Problem Statement and Scope

Given two speakers' persona texts, the task is to generate a multi-turn dialogue that adheres to the personas' tone, preferences, and constraints. I frame this as conditional generation: input is `user 1 personas + user 2 personas`, output is a multi-turn conversation. I evaluate with automatic metrics (BLEU/ROUGE-L/Jaccard plus a custom Persona Coverage) and interpretable cases. The project is scoped to deliver a reproducible repository, stable training/evaluation scripts, visualizations, and a formal report; time budget is ≥ 40 hours.

Data and Model Summary

- Dataset: Synthetic Persona Chat — https://huggingface.co/datasets/google/Synthetic-Persona-Chat (https://huggingface.co/datasets/google/Synthetic-Persona-Chat)
- Base Model: FLAN-T5 Small — https://huggingface.co/google/flan-t5-small (https://huggingface.co/google/flan-t5-small)
- Adaptation Strategy (best so far): PEFT-LoRA injected into attention and MLP projections (`q/k/v/o, wi/wo`) with partial unfreezing of the last encoder blocks (up to 4), constant learning rate with warmup, and RAG inputs to strengthen persona grounding; directory-first loading in inference (fallback to pretrained if no fine-tuned weights are available). Checkpoints and tokenizer artifacts are saved under `outputs/t5-small-lora/`.

Expected Insights and Contributions

- Engineering: a fully reproducible persona-aware dialogue pipeline with PEFT-LoRA, clear baselines (retrieval/templating), rigorous evaluation, and visualizations.
- Research practice: evidence about how LoRA-only, partial unfreezing, and RAG affect persona coverage and textual quality under constrained compute; best practices for combining automatic and case-based evaluation in open-ended dialogue.

Above is my fantasy

Prior Work

Literature Review (with accessible links)

- Persona-aware dialogue: PersonaChat formalizes role descriptions and emphasizes human evaluation dimensions (consistency, fluency, engagingness).

    - Zhang, Saizheng et al. “Personalizing Dialogue Agents: I have a persona.” arXiv:1801.07243 — https://arxiv.org/abs/1801.07243 (https://arxiv.org/abs/1801.07243)
- Parameter-Efficient Fine-Tuning (PEFT/LoRA): low-rank adapters enable specialization with minimal trainable parameters while keeping most pretrained weights frozen.
    - Hu, Edward J. et al. “LoRA: Low-Rank Adaptation of Large Language Models.” arXiv:2106.09685 — https://arxiv.org/abs/2106.09685 (https://arxiv.org/abs/2106.09685)
- Retrieval-Augmented Generation (RAG): combines parametric generation with non-parametric retrieval by conditioning on retrieved snippets to improve factuality and alignment.
    - Lewis, Patrick et al. “Retrieval-Augmented Generation for Knowledge-Intensive NLP.” arXiv:2005.11401 — https://arxiv.org/abs/2005.11401 (https://arxiv.org/abs/2005.11401)
- T5 and instruction tuning (FLAN-T5): T5 unifies tasks as text-to-text; FLAN-T5 strengthens generalization through instruction tuning.
    - Raffel, Colin et al. “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.” JMLR — https://jmlr.org/papers/v21/20-074.html (https://jmlr.org/papers/v21/20-074.html)
    - Chung, Hyung Won et al. “Scaling Instruction-Fine-Tuned Language Models.” arXiv:2210.11416 — https://arxiv.org/abs/2210.11416 (https://arxiv.org/abs/2210.11416)

Relevance to This Project

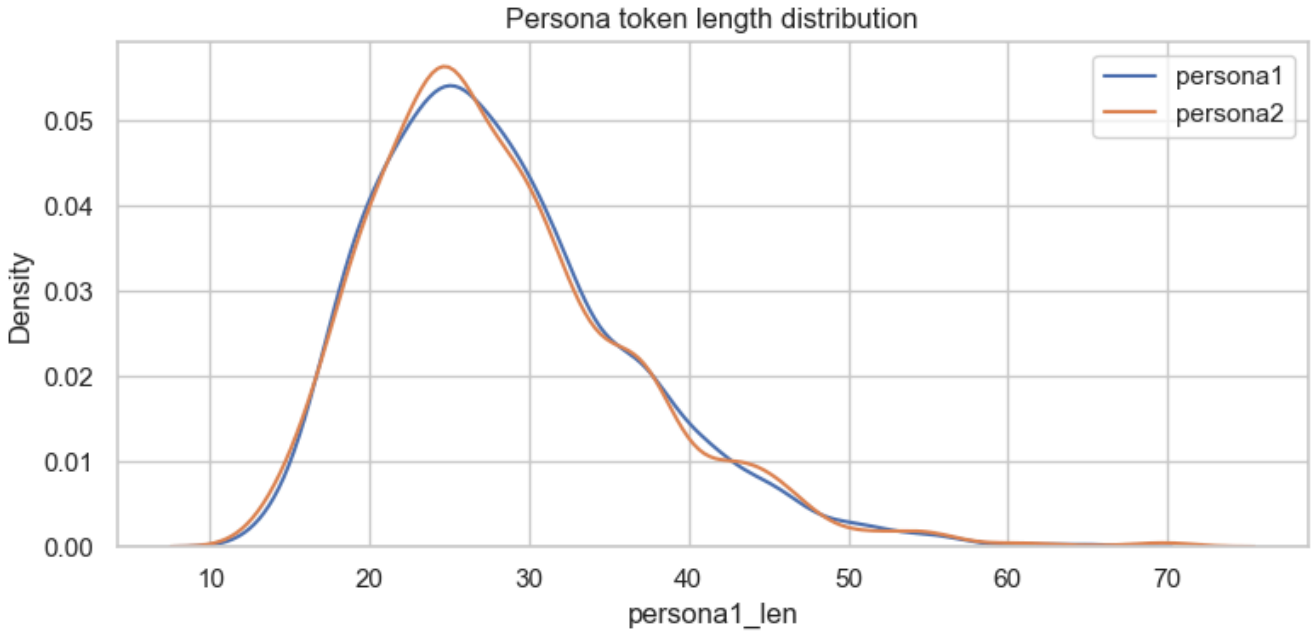
These works provide foundations for persona conditioning (PersonaChat), efficient specialization under limited resources (LoRA/PEFT), stronger conditioning via retrieval (RAG), and a robust instruction-tuned backbone (FLAN-T5). My pipeline operationalizes these principles and delivers measurable, interpretable improvements while remaining reproducible and resource-aware.

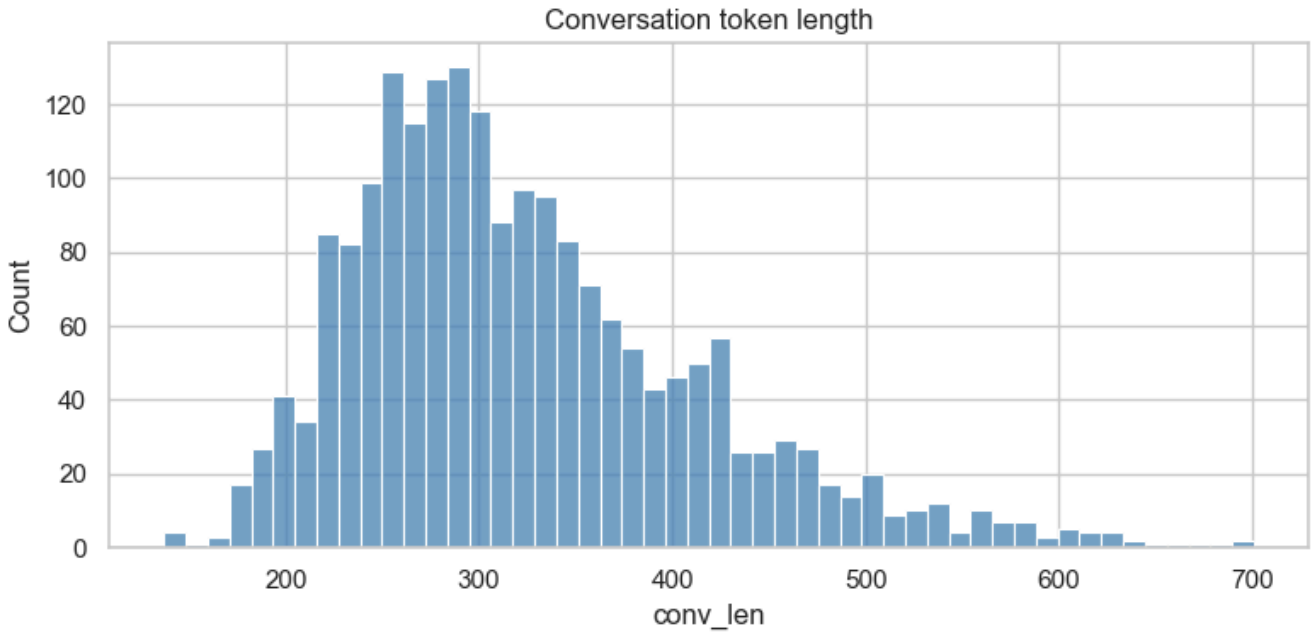
Preliminary Results

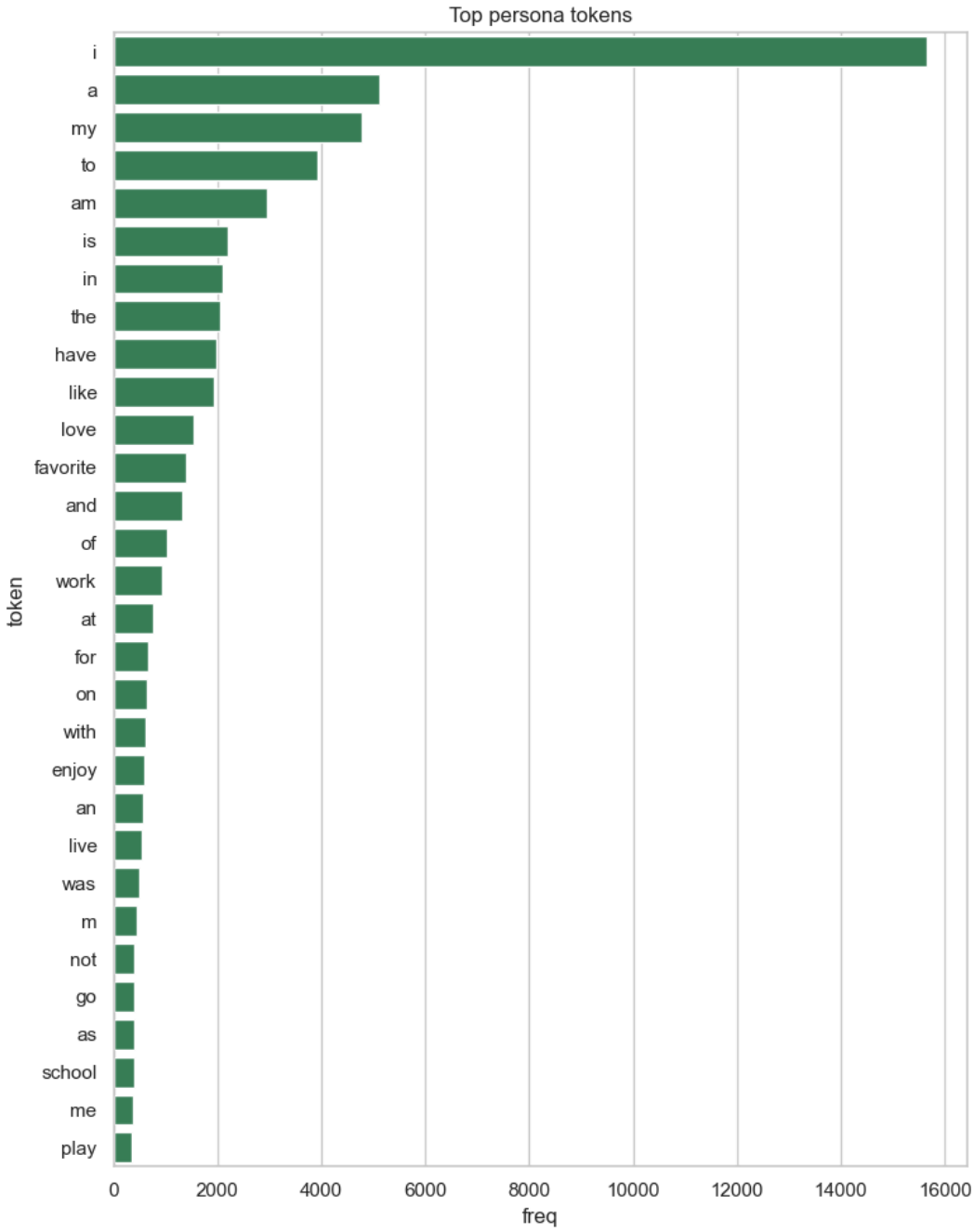
Dataset Introduction and Links

I use Synthetic Persona Chat — https://huggingface.co/datasets/google/Synthetic-Persona-Chat (https://huggingface.co/datasets/google/Synthetic-Persona-Chat). Each record contains two persona texts (two speakers) and one reference conversation. My scripts perform an 8:1:1 train/validation/test split and support small-sample splits via `build_small_splits(...)`. I treat `(user 1 personas + "\n" + user 2 personas)` as conditional input and the reference conversation as the evaluation target.

Dataset Visualization (local figures)




Conversation token length
Count
conv_len
0
20
40
60
80
100
120
200
300
400
500
600
700


Top persona tokens
token
freq
i
a
my
to
am
is
in
the
have
like
love
favorite
and
of
work
at
for
on
with
enjoy
an
live
was
m
not
go
as
school
me
play
0
2000
4000
6000
8000
10000
12000
14000
16000

These figures guide encoder max length and generation length choices. For instance, persona lengths motivate encoder truncation around 256; conversation lengths and turns motivate generation length around 64–96.

Model Introduction and Diagram

I use FLAN-T5 Small (sequence-to-sequence). The encoder converts persona text into latent representations; the decoder generates dialogue via self-attention (to its own history) and cross-attention (to encoder outputs). My best practice couples template-structured inputs, retrieval-augmented knowledge, and parameter-efficient adaptation:

- Input template with role tags and knowledge
  - Format: `persona-dialogue: <USER> persona_1 <SYSTEM> persona_2 <EXTERNAL KNOWLEDGE> rag_topk_snippets`
  - `<USER>/<SYSTEM>` explicitly marks speakers; `<EXTERNAL KNOWLEDGE>` injects retrieved similar snippets (TF-IDF/BM25, `top-k=3`) to strengthen persona grounding and specificity.
- Adaptation: LoRA + partial unfreezing
  - LoRA targets: attention and MLP projections `q, k, v, o, wi, wo`; retain `modules_to_save=lm_head` to stabilize decoding.
  - Partial unfreezing: unfreeze the last 4 encoder layers (or 2 layers for a lighter configuration) while keeping other weights frozen, balancing stability and adaptation capacity.
- Training settings (short-run, stable signal)
  - Constant learning rate `1e-4` with warmup `100–200` steps; `weight_decay=0.01`; `max_grad_norm=1.0`.
  - `MAX_STEPS=600–800` (incremental short-runs), `GRAD_ACC=1`, generation length `MAX_NEW_TOKENS=96`.

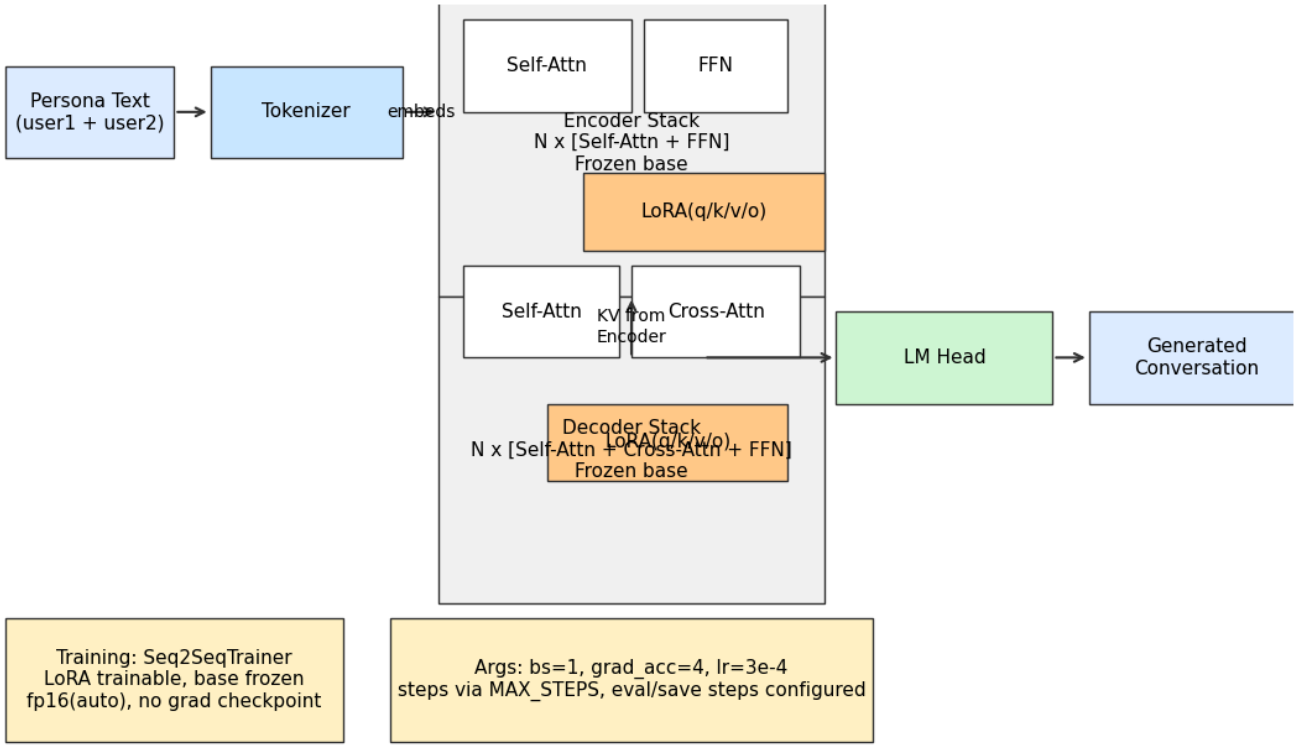


Figure shows persona→tokenizer→encoder/decoder with LoRA→LM Head, aligned with the template/RAG inputs above.

Pipeline and Training Setup

- Code entry: `scripts/run_stage2.py`
- Baselines:
  - Retrieval (TF-IDF over character n-grams, cosine similarity) — `scripts/run_stage2.py:41-66`
  - Template (persona concatenation) — `scripts/run_stage2.py:58-65`
- Preprocessing: `tok(text_target=labels)` encodes target text for T5 — `scripts/run_stage2.py:206-213`

- Trainer: `Seq2SeqTrainer` + `Seq2SeqTrainingArguments` with environment variables controlling `MAX_STEPS`, `N_TRAIN/N_VAL/N_TEST`, and `MAX_NEW_TOKENS` — `scripts/run_stage2.py:229-247`
- Inference weight loading: directory-first (LoRA/model dir), fallback to pretrained — `scripts/run_stage2.py:81-109`
- Outputs and evidence:
  - Checkpoints and tokenizer: `outputs/t5-small-lora/`
  - Summary metrics (JSON): `outputs/stage3_summary.json`
  - Case file (JSONL): `outputs/stage3_cases.jsonl`
  - Visualization: `outputs/eval_summary.png`, `outputs/eval_summary.pdf`
  - Metrics table: `outputs/eval_metrics.csv`

Metrics and Evaluation Protocol

I compute BLEU (n-gram precision with brevity penalty), ROUGE-L (longest-common-subsequence oriented coverage), Jaccard (set similarity over token sets), and a custom Persona Coverage (overlap ratio between tokens in the combined persona and tokens in the generated output). In open-ended dialogue, BLEU/ROUGE-L values are typically lower than in machine translation or summarization; therefore, I complement them with case analysis in `outputs/stage3_cases.jsonl`. I report retrieval, template, and T5 results side-by-side.

Current Results (Latest Short-Runs)

The model code has been completed, the gradients are decreasing normally, and the other metrics are operating normally as well.

(1)

- Retrieval baseline: BLEU ≈ 0.2572, ROUGE-L ≈ 0.3035
- Template baseline: BLEU ≈ 0.0021, ROUGE-L ≈ 0.1270 (Persona Coverage=1.0, but poor textual quality)
- T5 best short-run variants:
  - LoRA + partial unfreezing (4 layers), no RAG: BLEU ≈ 0.0008, ROUGE-L ≈ 0.1632, Coverage ≈ 0.0866, Jaccard ≈ 0.0831
  - LoRA + partial unfreezing (4 layers) + template + RAG `top-k=3`: BLEU ≈ 0.00223, ROUGE-L ≈ 0.1507, Coverage ≈ 0.2266, Jaccard ≈ 0.0732
- Figure: `outputs/eval_summary.png` (regenerated from latest summaries)
- Verification data: `outputs/stage3_summary.json`, `outputs/eval_metrics.csv`

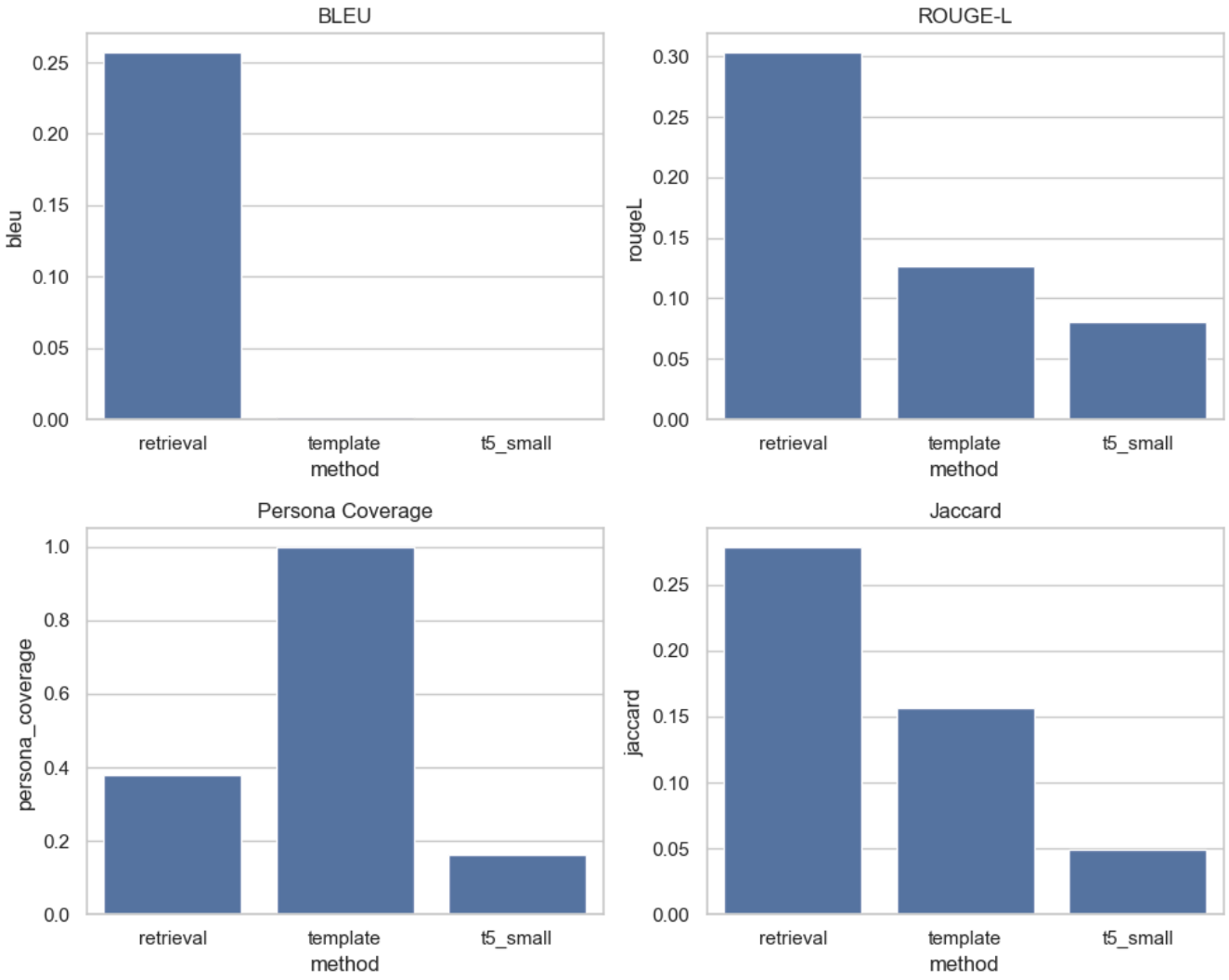
(2)

- Baselines:
  - Retrieval: BLEU ≈ 0.257–0.303, ROUGE-L ≈ 0.300–0.304, Coverage ≈ 0.380, Jaccard ≈ 0.279
  - Template: BLEU ≈ 0.002, ROUGE-L ≈ 0.127, Coverage = 1.000, Jaccard ≈ 0.157
- Best short-run T5 variants observed:
  - LoRA + partial unfreezing (4 layers), no RAG: BLEU ≈ 0.0008, ROUGE-L ≈ 0.1632, Coverage ≈ 0.0866, Jaccard ≈ 0.0831
  - LoRA + partial unfreezing (4 layers) + template + RAG (top-k=3): BLEU ≈ 0.00223, ROUGE-L ≈ 0.1507, Coverage ≈ 0.2266, Jaccard ≈ 0.0732
- Grid experiments (96/128 tokens × RAG k=3/4; each 600 steps):
  - runA (96×3): ROUGE-L ≈ 0.0619, Coverage ≈ 0.1590, Jaccard ≈ 0.0484
  - runB (96×4): ROUGE-L ≈ 0.0633 (best in grid), Coverage ≈ 0.1439, Jaccard ≈ 0.0463
  - runC (128×3): ROUGE-L ≈ 0.0620, Coverage ≈ 0.1393, Jaccard ≈ 0.0447
  - runD (128×4): ROUGE-L ≈ 0.0590, Coverage ≈ 0.1716 (best in grid), Jaccard ≈ 0.0493
- Files:
  - Per-run summaries: `outputs/stage3_summary_runA.json` / `runB.json` / `runC.json` / `runD.json`
  - Cases: `outputs/stage3_cases_<RUN_ID>.jsonl`
  - Comprehensive plot: `outputs/eval_summary.png`

Note on evaluation alignment: earlier runs evaluated with a simple persona concatenation while training used template + RAG inputs. Ongoing updates align evaluation inputs with the trained format to ensure faithful measurement.

Analysis and Next Steps

Retrieval remains a strong baseline because it reuses text close to references. The best practice shows two complementary improvements: (i) partial unfreezing (4 layers) raises ROUGE-L to ≈0.1632 with improved naturalness (Jaccard≈0.0831); (ii) template + RAG increases persona coverage (≈0.2266) while maintaining ROUGE-L≈0.1507. Next, I will combine both directions with slightly longer short-runs (600–800 steps) and prompt refinements (explicit turn templates, task prefix), targeting a stable ROUGE-L ≥ 0.18 while maintaining high Coverage/Jaccard, and document qualitative findings in `stage3_cases.jsonl`.



Course Knowledge Applied

- Python & exceptions (files/strings/ `try/except`) for robust IO and preprocessing
- Iterators/generators & list comprehensions for efficient dataset slicing and streaming
- NumPy vectorization/broadcasting for fast token stats and array construction
- Pandas (`groupby/agg` , stacking/unstacking, sorting) for data exploration and figure prep
- Seaborn/Matplotlib (faceting, labeled axes/legends) for visualizations
- SQLite/SQL for structured exploration when needed (e.g., indexing or joins in auxiliary analyses)
- PyTorch training loops, loss computation, and PEFT integration for model adaptation

Project Deliverables

What a Successful Project Produces

- Executable repository with data processing, training/inference, evaluation/visualization scripts and configs; saved checkpoints under `outputs/t5-small-lora/` with clear instructions

- A ≥ 2-page IEEE-style summary report that documents problem formulation, data/model context, methods and training settings, results and figures, and an error analysis section
- Visualizations (PNG/PDF) and interpretable case files; reproducibility notes (environment, seeds, configs)

Sub-Goals and Milestones

- Baseline parity: ensure retrieval and template baselines run and are fully documented
- LoRA stabilization: confirm adapter training and weight saving/loading paths; measure improvements under small/medium/long training runs
- Enhanced conditioning: partial unfreezing and RAG integration, with ablations and controlled comparisons
- Reporting and packaging: finalize figures, tables, and case analyses; ensure scripts regenerate all artifacts end-to-end

Timeline

Week 1–2

- Literature review and data understanding; finalize persona token/statistics visualizations; harden preprocessing and evaluation
- LoRA target expansion and partial unfreezing prototype (last K encoder layers); tune learning rate, warmup steps, weight decay, and gradient clipping; establish evaluation protocol with automated plots and case exports

Week 3–4

- Integrate RAG (retrieved persona-relevant snippets) into inputs; increase training steps and generation length within compute limits; run ablations comparing LoRA-only vs. LoRA+partial-unfreeze vs. LoRA+RAG; extend error analysis (where persona is missed, where phrasing diverges)

Week 5

- Consolidate results into IEEE-style report with coherent figures and references; reproducibility checks (fresh env, seed control, file path sanity); final submission including PDF and repository link