Aidan Sibley
Monica Stef CISC 473

Midterm Check-In Report

Introducing and Tracking the S1-S7 Pipeline

Prelude:

Repository Access

- Clone the project repository using the following commands:
```
git clone https://github.com/ajsib/CISC-473-Project.git
cd face-restoration-bench
```
- Create and activate the project environment:
```
conda env create -f env.yml
conda activate face-restoration
```
- Download and place the CelebA aligned dataset into:
```
./data/img_align_celeba/
```
- Execute the sequential pipeline:
```
bash scripts/make_synth_testset.sh   # S3 degradation
bash scripts/eval_all.sh             # S4A and S4B inference
bash scripts/make_figures.sh         # S6 visualization
```
- Results are automatically organized as follows:
```
./results/outputs/   # model outputs
./results/tables/    # metrics
./results/figures/   # panels and plots
./results/logs/      # config and run logs
```
- The pipeline ensures deterministic behaviour. Seeds are logged in each JSONL, `config.json` specified presets, weights, and paths, and every run generates a unique timestamped log for full traceability.

1. Introduction and Methodological Context (S1 - S7 Pipeline Overview)

1.1 Purpose of Pipeline Framework:

The S1-S7 pipeline serves as a structured, reproducible framework for conducting a comparative study of two pretrained face restoration models (GFPGAN and CodeFormer). Each stage of the pipeline represents a discrete operational phase from dataset ingestion to visualization and logging. The pipeline ensures deterministic behaviour through fixed seeds, a centralized configuration file (`config.json`), and standardized output directories under `./results/`. It is essentially the methodological backbone of the project and serves as both a workflow and a way to track progress.

1.2 Overview of Stages S1 - S7:

- **S1 Data:** Ingest pre-aligned CelebA face images and metadata to use as immutable ground-truth (GT) data.

- **S2 Align:** Verify or reapply geometric normalization using five-point facial landmarks to ensure consistent image framing.
- **S3 Degrade:** Apply controlled degradation (Gaussian blur and JPEG compression) to create low quality (LQ) versions for model input.
- **S4A GFPGAN Inference:** Restore degraded images using the pretrained GFPGAN v1.4 model.
- **S4B CodeFormer Inference:** Restore the same LQ images using CodeFormer (sczhou) at varying fidelity factors (between 0 and 1 inclusive).
- S5 Metrics: evaluate quantitative and perceptual differences using PSNR. SSIM, LPIPS, and ArcFace cosine similarity.
- **S6 Figures:** Generate visual comparison panels and performance plots.
- **S7 Logging:** Consolidate configuration details, seeds, weights, and timestamps for full reproducibility.

1.3 Methodology as Progress Tracking Mechanism:

The pipeline doubles as a progress-tracking framework, where each stage serves as a checkpoint. By structuring progress by S1 - S7, we maintais transparency, reproducibility, and traceability from the raw data all the way through inference and evaluation. Each completed stage outputs reproducible artifacts (datasets, logs, metrics) that show cumulative progress.

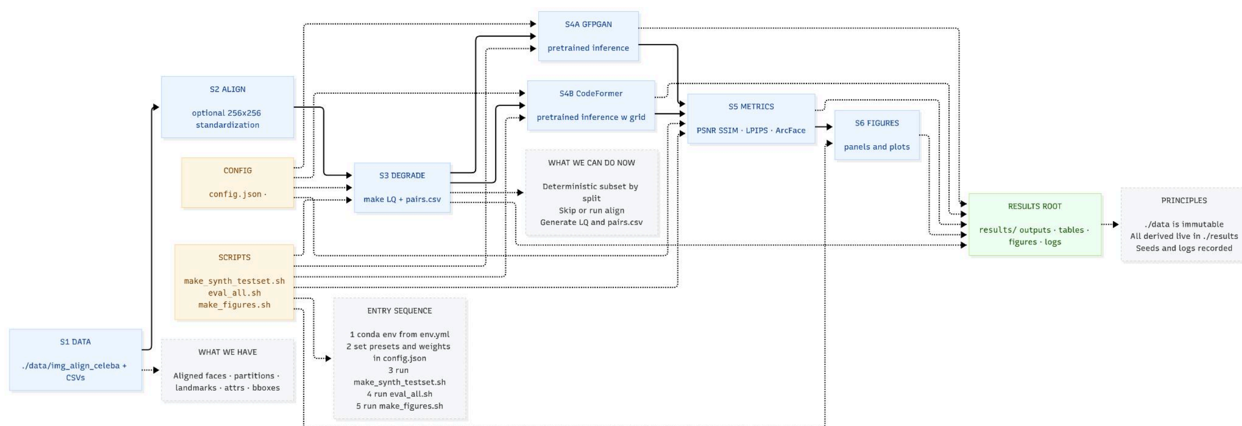

Figure 1. Overall pipeline flow connecting S1-S7, showing data flow, configuration files, and output directories.

2. Current Status: Pipeline Progress

2.1 Active Stage Identification:

The project has successfully completed Stages S1 - S4A and is currently progressing through S4B - S7.Preliminary runs of GFPGAN and CodeFormer have produced measurable baseline outputs, and we are finalizing the integration of LPIPS and ArcFace metrics.

2.2 Summary of Overall Progress Across S1 - S7

Stage	Description	Status
S1	Data ingestion and verification	Complete
S2	Alignment and normalization	Complete
S3	Synthetic degradation (blur + JPEG)	Complete
S4A	GFPGAN inference (v1.4 pretrained)	Complete
S4B	CodeFormer inference (sczhou, w sweep)	In Progress
S5	Metrics Evaluation (PSNR, LPIPS, ArcFace)	In Progress
S6	Visualization and figure generation	In Progress
S7	Logging and reproducibility	In Progress

3. Completed Work (Since Last Check-In)

3.1 Summary of Completed Stages:

- S1 Data: Ingested and validated CelebA aligned dataset 202,599 images). Metadata verified for integrity and unique IDs.
- S2 Align: Landmark-driven alignment test. Determined CelebA alignment sufficient.
- S3 Degrade: Implemented reproducible degradation pipeline with Gaussian blur and JPEG compression. Random seeds logged for repeatability
- S4A GFPGAN: Executed inference using pretrained GFPGAN v1.4 (from Hugging Face). Outputs generated for select images.

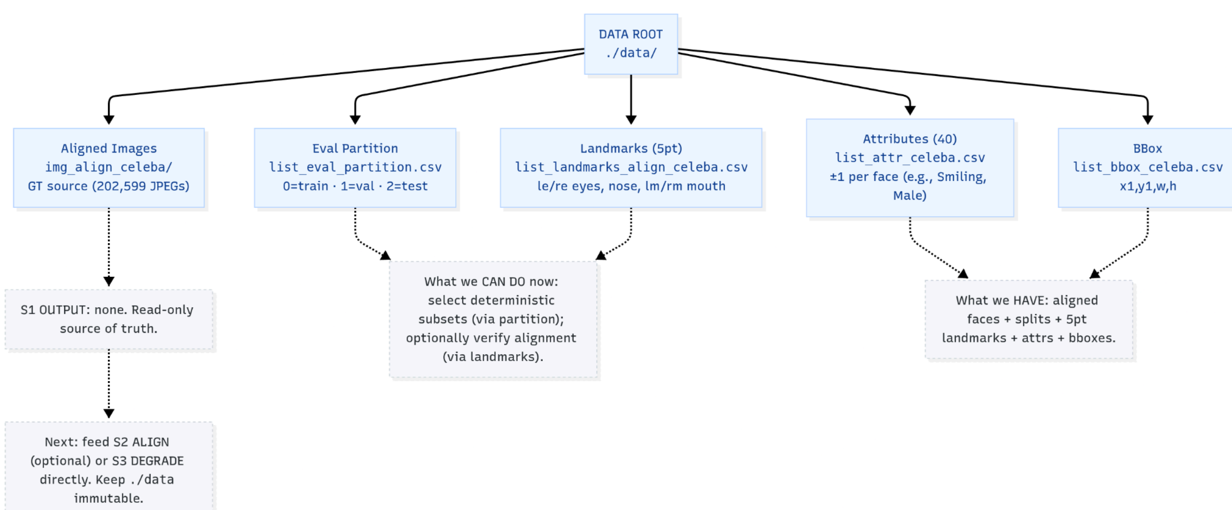

Figure 2. Data root and metadata structure for S1, including aligned images, partitions, landmarks, attributes, and bounding boxes.

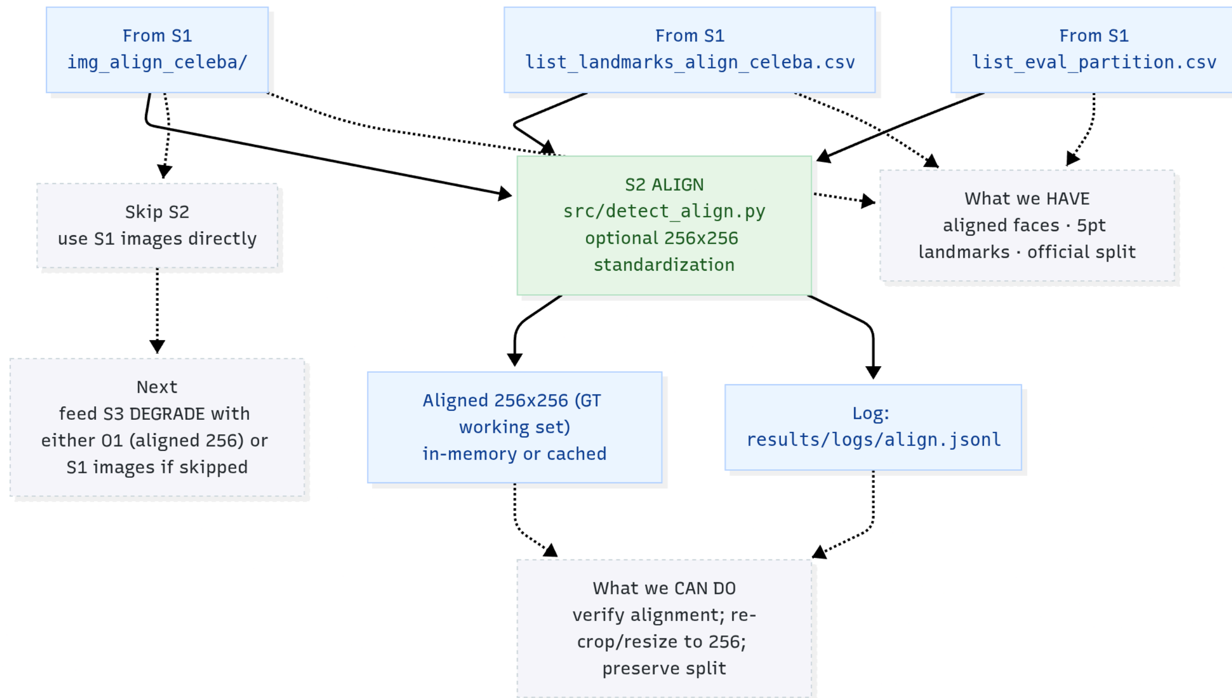

Figure 3. Optional alignment workflow for geometric normalization using five-point landmarks.

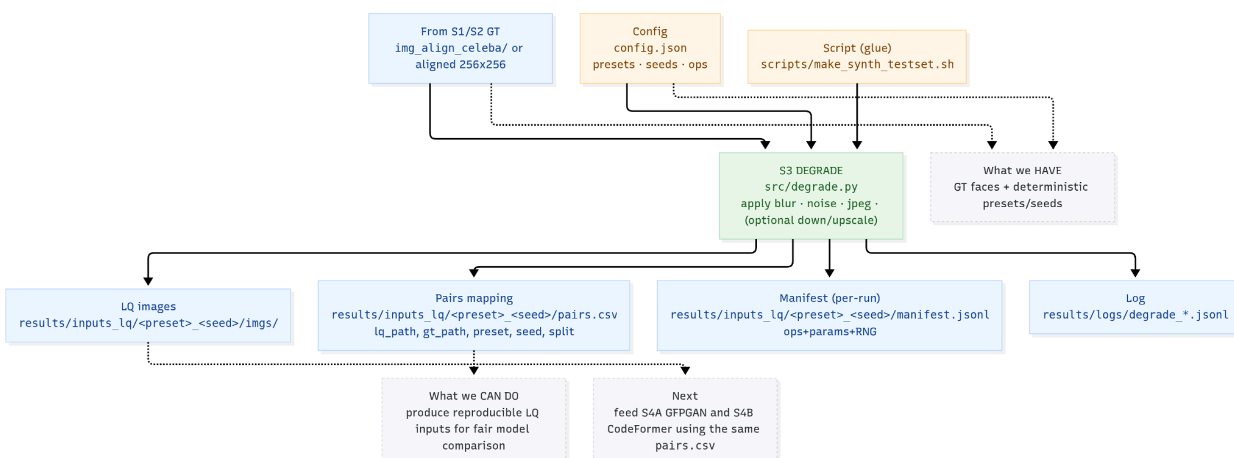

Figure 4. Degradation process generating low quality inputs (LQ) and corresponding `pairs.csv` mappings.

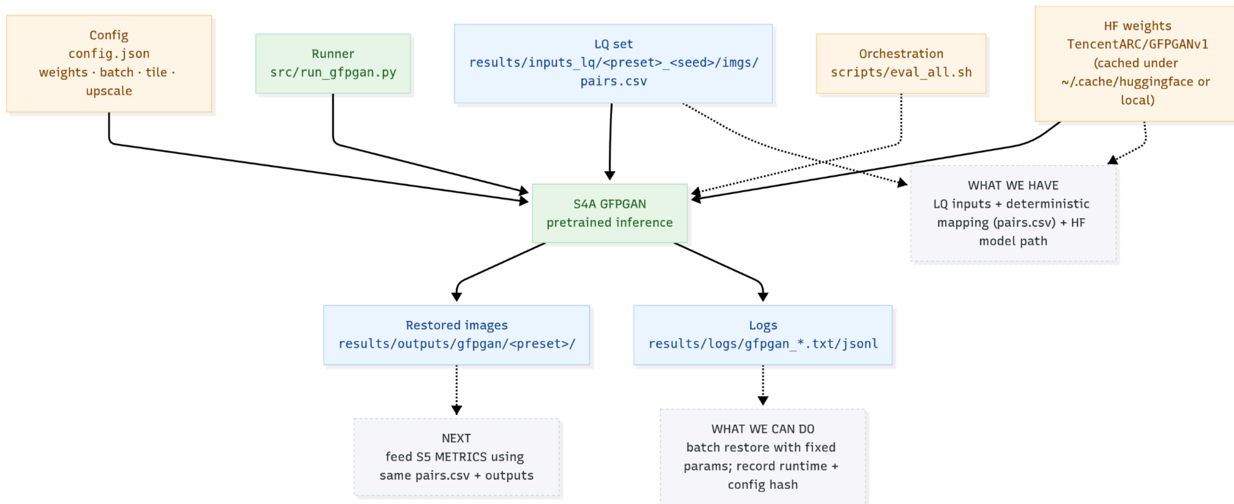

Figure 5. GFPGAN inference workflow showing configuration, Hugging Face weights, and output directories.

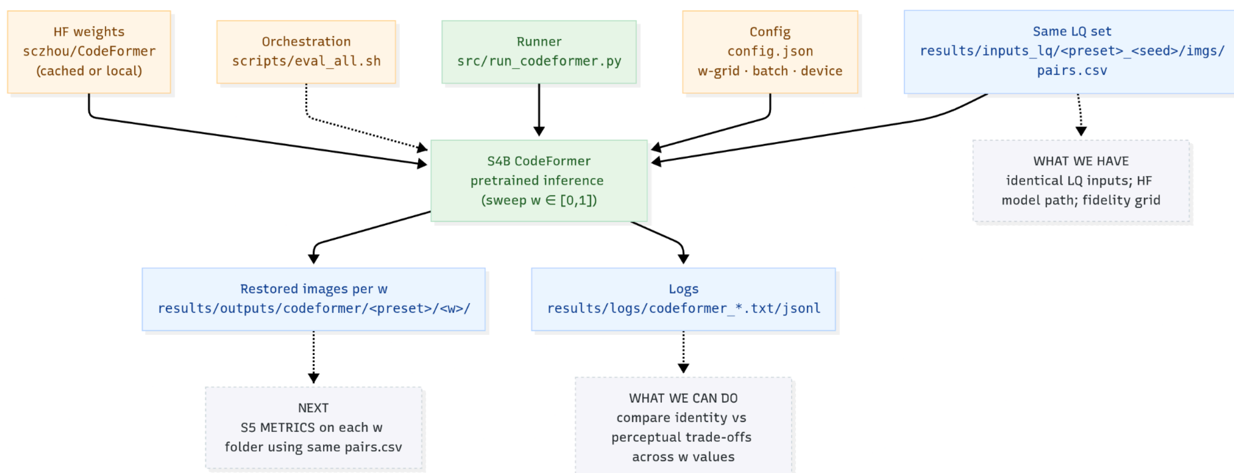

Figure 6. CodeFormer inference workflow showing fidelity parameter sweep and output mapping.

4. Current Work (In Progress)

4.1 Active Stage Objectives:

- S4B CodeFormer. The goal is to complete an inference for the CodeFormer model `sczhou` that can yields the best balance between realism and identity
- S5 Metrics Evaluation. The goal is to generate quantitative comparisons between GFPGAN and CodeFormer across all metrics.

4.2 Tasks and Interim Results:

- CodeFormer implementation
- LPIPS implementation

- ArcFace identity embeddings initialized using pretrained model (batch evaluation underway)
- Initial results show GFPGAN scoring higher in identity retention, while CodeFormer achieves lower perceptual loss (LPIPS)
- In parallel, figure generation scripts for S6 are being drafted to visualize trade-offs

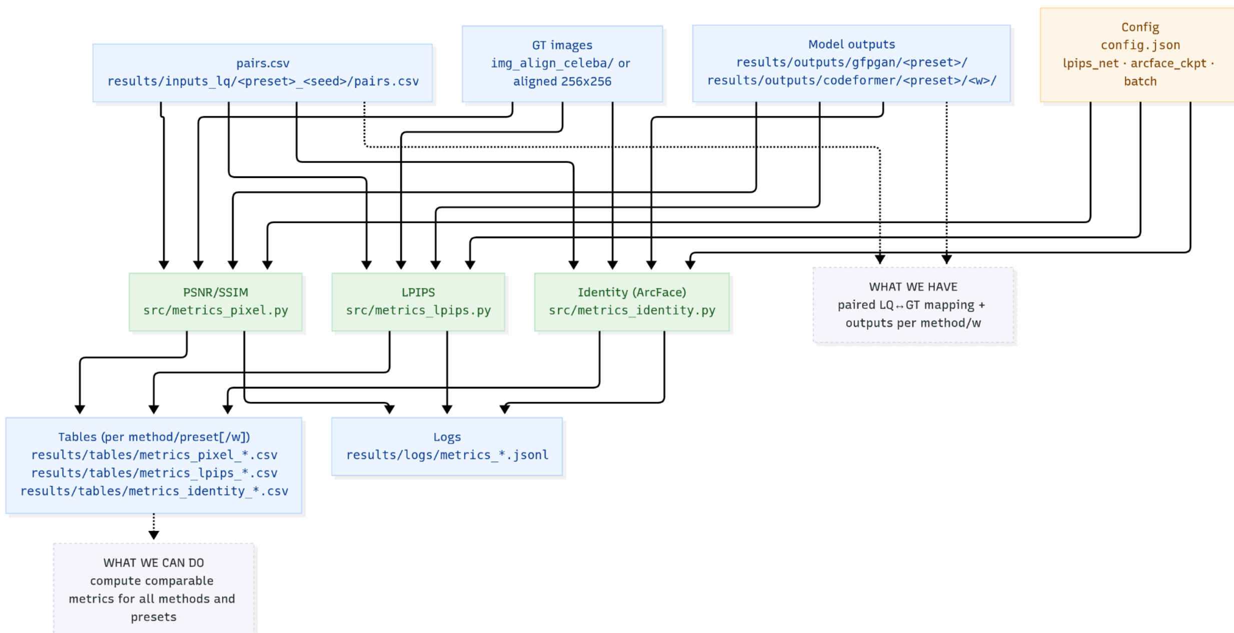

Figure 7. Metrics evaluation diagram detailing how paired inputs feed into pixel, perceptual, and identity metrics.

5. Blockers and Challenges

5.1 Identified Issues:

- NumPy 2.x compatibility issue: PyTorch and basicsr failed to import correctly under NumPy 2.x
- Compute constraints: local CPU only inference proved to be time intensive
- RealESRGAN dependency management: background upsampler occasionally failed to initialize on non-GPU machines

5.2 Mitigation Actions and Status

- Downgraded to NumPy 1.26.4 to restore PyTorch stability
- Migrate heavy inference and metric computations to Google Colab
- Added a fallback flag to disable background upsampler when unavailable

6. Next Steps and Timeline

6.1 Planned Stages and Transition Points

- S4A → S4B: Complete GFPGAN and CodeFormer inference
- S5 → S6: Complete ArcFace Evaluation and aggregate results into tables by Week 9

- S6 → S7: Generate visual comparison panels and bar charts for PSNR, LPIPS, and ArcFace metrics by Week 10.

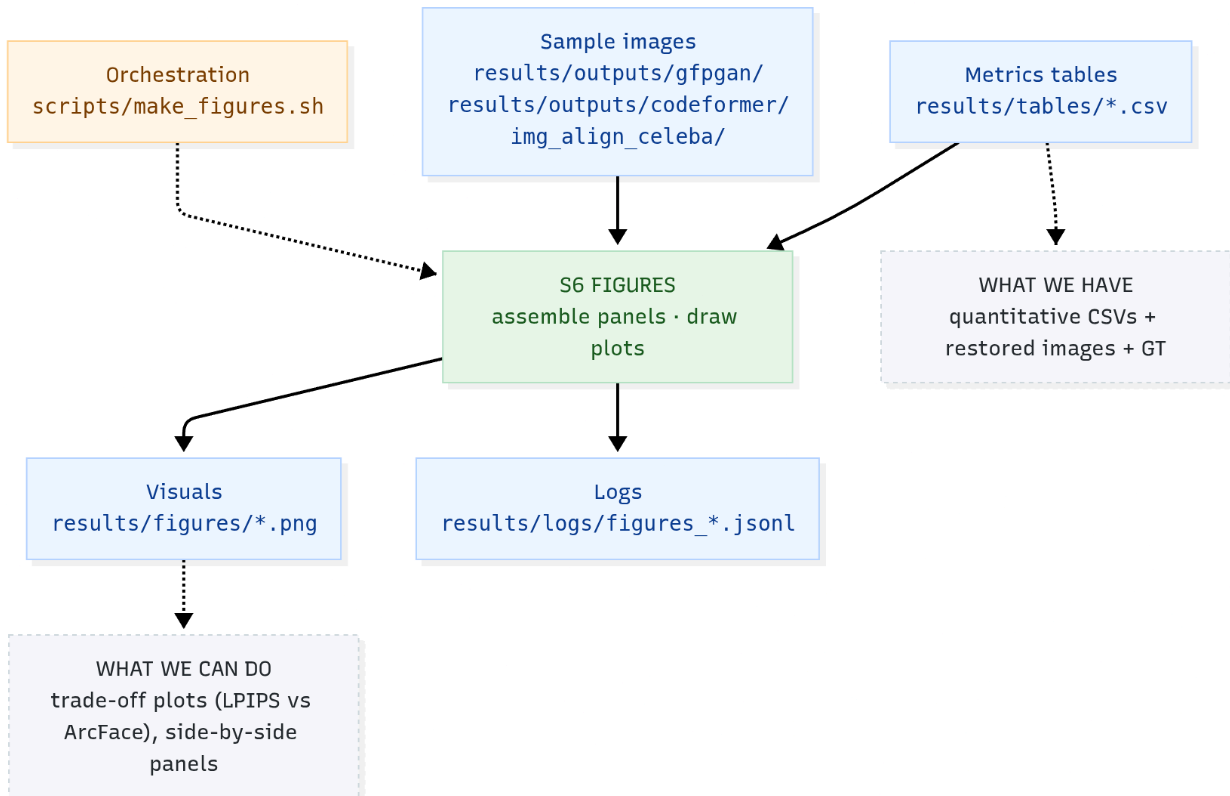

Figure 8. Visualization pipeline assembling figure panels and trade-off plots from metric outputs.

6.2 Anticipated Milestones and Deadlines

- Week 8: LPIPS and ArcFace full dataset evaluation complete
- Week 9: Visualization scripts finalized (S6)
- Week 10: Final report submission and presentation

7. Overall Assessment

7.1 Project Status and Readiness

The project is on track with all core pipeline stages (S1 - S4A) completed and successful baselines established. The methodology remains reproducible and robust, with stable model performance across the controlled dataset. The migration to Colab has mitigated hardware constraints.

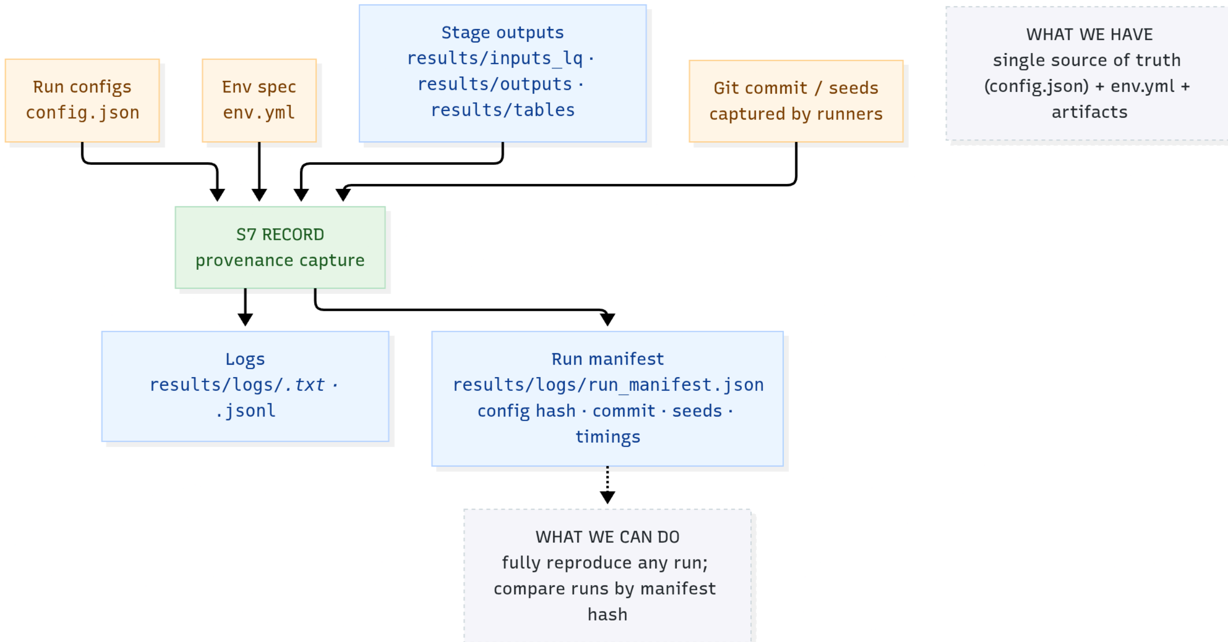

Figure 9. S7 logging and provenance architecture capturing environment configuration, seeds, and run manifests.

7.2 Forward Outlook

The remaining stages (S4B - S7) focus on implementation, evaluation, visualization, and documentation. The expected outcome is a comprehensive comparison that highlights the trade-off between GFPGAN's generative realism and CodeFormer's fidelity control, positioning the project for a strong final report and presentation.