Design decisions: temporal encoding pipeline

Architecture, component choices, and test strategy

Tymo van Rijn
CMI – Informatica (1057297)

March 2026

Abstract

This document records the key architectural decisions made when designing the temporal deep-learning pipeline for fluorescein angiography (FA) grading. It covers the integration of the existing RETFound-Green backbone with the new temporal components, the rationale for a recurrent sequence core (specified here as an LSTM for the design review and prototype; the downstream implementation uses a GRU after supervisor guidance), the binning algorithm design, and the test and validation strategy. Where multiple design options existed, alternatives are stated and the reasons for the chosen direction are given explicitly.

1 Overall pipeline architecture

The design goal was to extend the existing single-frame pipeline—which passes one FA frame through RETFound-Green and classifies the resulting 384-dimensional embedding directly—into a pipeline that processes an entire examination as an ordered sequence. The chosen architecture keeps the existing backbone frozen and adds three new components on top of it:

1. **Temporal binning layer.** Frames of an examination are assigned to B temporal bins, and one representative embedding per bin is selected or aggregated. This produces a fixed-length sequence of B vectors regardless of how many frames an examination contains, solving the variable-length problem without padding or masking.
2. **Sequence model (recurrent).** The B-length sequence of bin embeddings is passed through a two-layer recurrent core. This document and the diagram use an LSTM as the design-time specification; the shipped training code uses a GRU with the same role and tensor contract. The final hidden state summarises the temporal progression of the examination into a single vector, which is then passed to the classification head.
3. **Classification head.** A single linear layer maps the recurrent hidden state to class logits, followed by softmax for probability estimation. This mirrors the head used in the existing single-frame baseline, ensuring that any performance difference is attributable to the temporal component rather than to changes in the classifier.

A schematic of this pipeline is provided in `04_Design/architecture-temporal-pipeline.svg`, and a runnable prototype that validates the tensor shapes and data flow is provided in `04_Design/prototype-temporal-pipeline-shapes.py`.

2 Design decisions and justification

Recurrent core (LSTM in design, GRU in implementation) over Transformer

A full self-attention Transformer would be architecturally more general, but requires substantially more data and is harder to debug when the sequence length is small (in the range of 2–10 bins). A gated recurrent block (LSTM or GRU) is better suited here because: (1) it is inductive in its treatment of sequence order, so it does not need positional encodings to understand that bin 1 precedes bin 2; (2) it scales gracefully with short sequences and small datasets; (3) its hidden state has a direct interpretation as a running summary of the examination so far, which is clinically intuitive. The design review and `prototype-temporal-pipeline-shapes.py` use an LSTM; the downstream trainer uses a GRU after supervisor recommendation, without changing the data layout. A Transformer variant remains a valid future direction once the bin-level temporal signal is validated.

Two recurrent layers, hidden size 20 (prototype)

Two layers are a common and well-justified starting point for sequence modelling: the first layer captures low-level temporal transitions between adjacent bins, while the second layer can capture higher-order patterns across the full sequence. A hidden size of 20 is intentionally small relative to the 384-dimensional input, acting as an information bottleneck that forces the model to compress the temporal signal rather than memorise per-exam noise. These values were selected for the LSTM prototype and are treated as tunable hyperparameters in the benchmark; full downstream runs document their own hidden size and dropout in the training script.

Frame selection within a bin

Two aggregation strategies were considered: mean-pooling all frames within a bin, or selecting a single representative frame (e.g. the centre or a random frame). Mean-pooling is computationally simple but may dilute the discriminative signal of individual frames. Selecting the centre frame is more principled because it represents the temporal midpoint of the bin, which is more stable than a random choice and less likely to include transitional frames at the bin boundary. Both strategies are evaluated in the Temporal Separability Benchmark.

Binning strategy: uniform vs. quantile

Two binning strategies are included in the benchmark design. Uniform binning divides the elapsed-time axis into equal-width windows; quantile binning divides it into windows containing an equal number of frames. Uniform binning aligns better with the physical FA phase structure (early, mid, late) but is sensitive to outlier exam durations. Quantile binning is more robust to duration variability and always produces populated bins, but may compress or stretch phases in ways that do not reflect clinical boundaries. Both strategies are tested across bin counts $B \in \{2, 3, 5, 8, 10\}$ and evaluated with the LDA probe and downstream recurrent-model metrics.

3 Test and validation strategy

The model is evaluated against the single-frame baseline using a fixed train/val/test split defined by the `hyperftype.json` file kept on Idiap with the experiment code (not redistributed in the graduation folder). The logical format and a redacted schema example are documented in `06_Appendices/hyperftype-split-readme.txt`. The held-out test set contains $n = 211$ examinations and is used identically for all configurations, so performance rankings are not confounded by evaluation variance. Primary metrics are ROC-AUC and Average Precision (AP), consistent with the single-frame baseline, allowing a direct comparison on the same task and split.

The specific bin count and binning strategy to use as the primary configuration are informed by the analysis work documented in `02_Analysis/`; the pipeline is parameterised so that these values can be updated without architectural changes.

Unit test strategy for temporal components

To reduce regression risk during temporal-model refactoring, a focused set of unit tests was defined for the highest-risk logic paths. The selected scope is intentionally small and stable:

- `apply_backbone_unfreeze_last_n`: freeze/unfreeze policy and invalid-input handling.
- `ViTGRU` forward pass: temporal tensor contract validation with a mocked backbone.
- `GroupedJSONSplit`, `_uniform_pick_indices`, and `collate_sequence`: split parsing and variable-length sequence collation/padding.

These tests are intentionally isolated from heavy dataset and training dependencies so they can run quickly and deterministically in a local development loop. This test strategy provides evidence for maintainability and reproducibility: implementation changes in temporal data handling or model wiring can be validated before running expensive downstream experiments.

The test implementation and execution evidence are documented in `05_Realisation/unit-test-realisation.tex`.

4 Quality characteristics addressed

Characteristic	**How the design addresses it**
Reproducibility	Fixed random seeds; precomputed embeddings stored once; split file frozen before any experiment.
Scalability	Binning reduces variable-length sequences to fixed-length inputs; compute cost scales with B, not with the number of frames per exam.
Interpretability	Recurrent hidden state and bin-level Fisher scores are human-readable; backbone is frozen so there are no unexplained representation shifts.
Compatibility	No changes to the RETFound-Green backbone or the classification head interface; new components are inserted as a drop-in layer.
Debuggability	Prototype (`04_Design/prototype-temporal-pipeline-shapes.py`) validates tensor shapes end-to-end before integration into the training pipeline.