Tymo's progress

Week 3 & 4

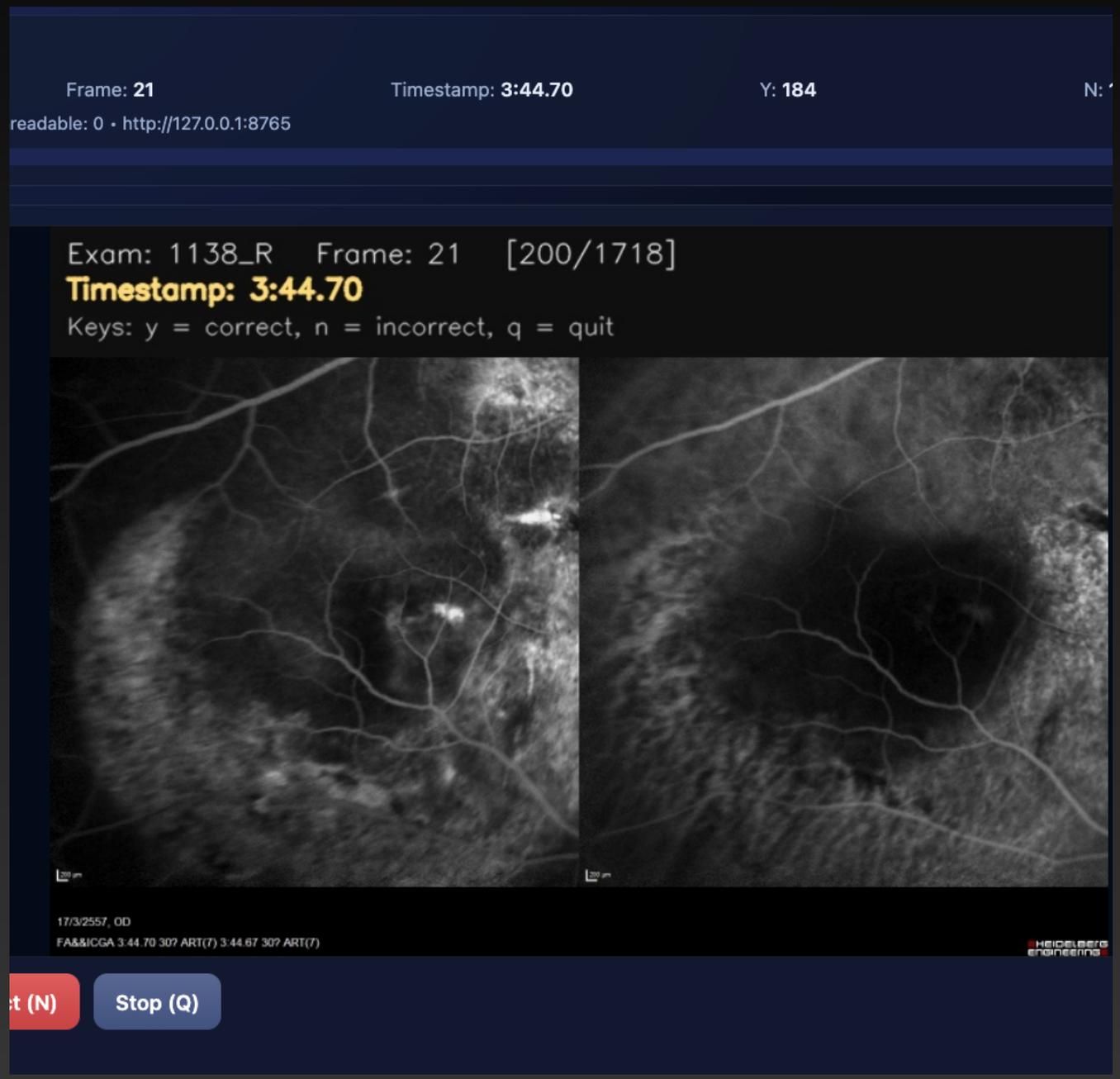
Week 3

Timestamp Validation

Validate if the OCR extractions are correct by “checking” the frames manually

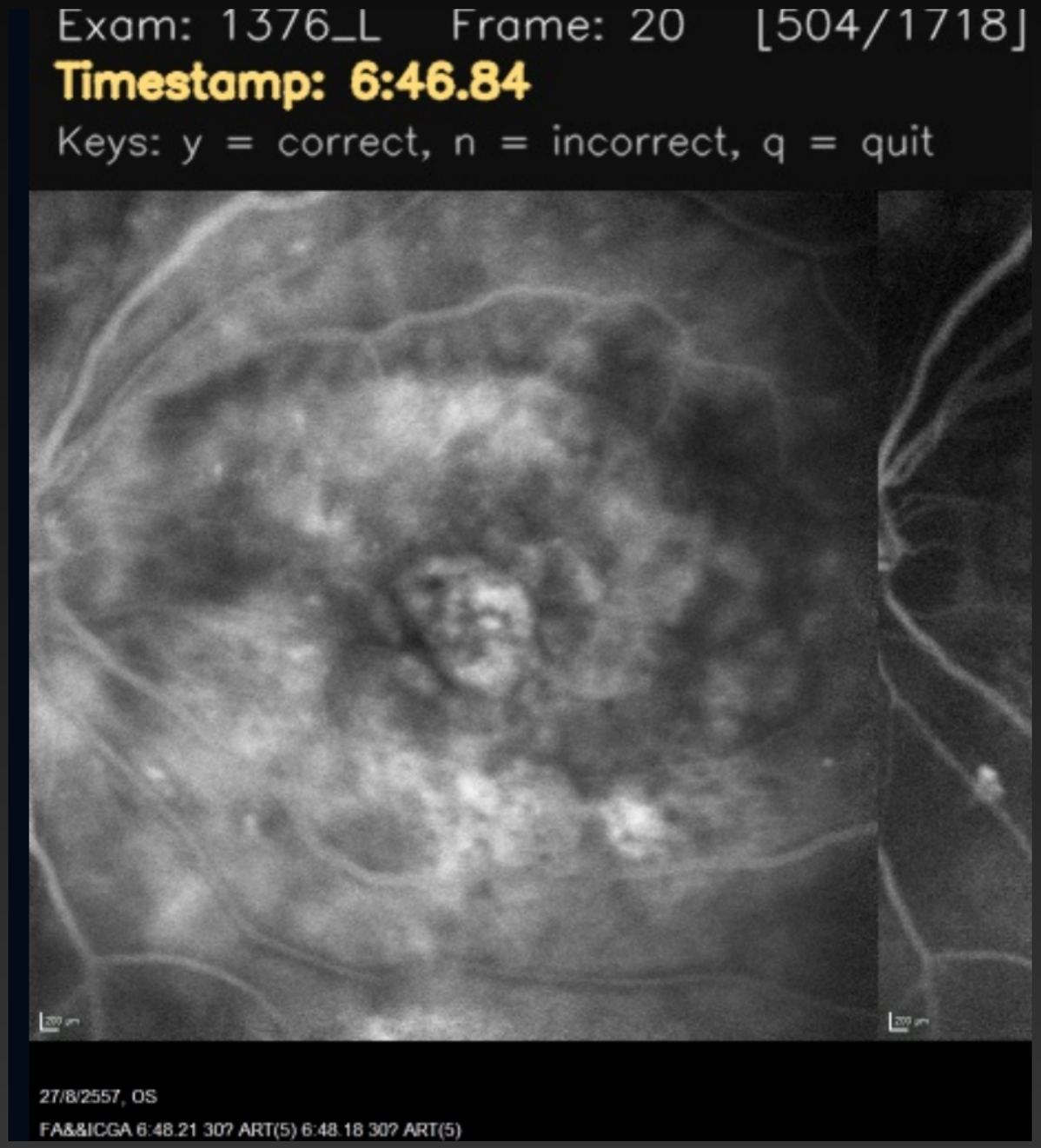
Timestamp Validation

- I used this GUI to validate the timestamps.
- Results were instantly sent to a .csv that kept track of the scores.
- (Not my own product)



Timestamp Validation

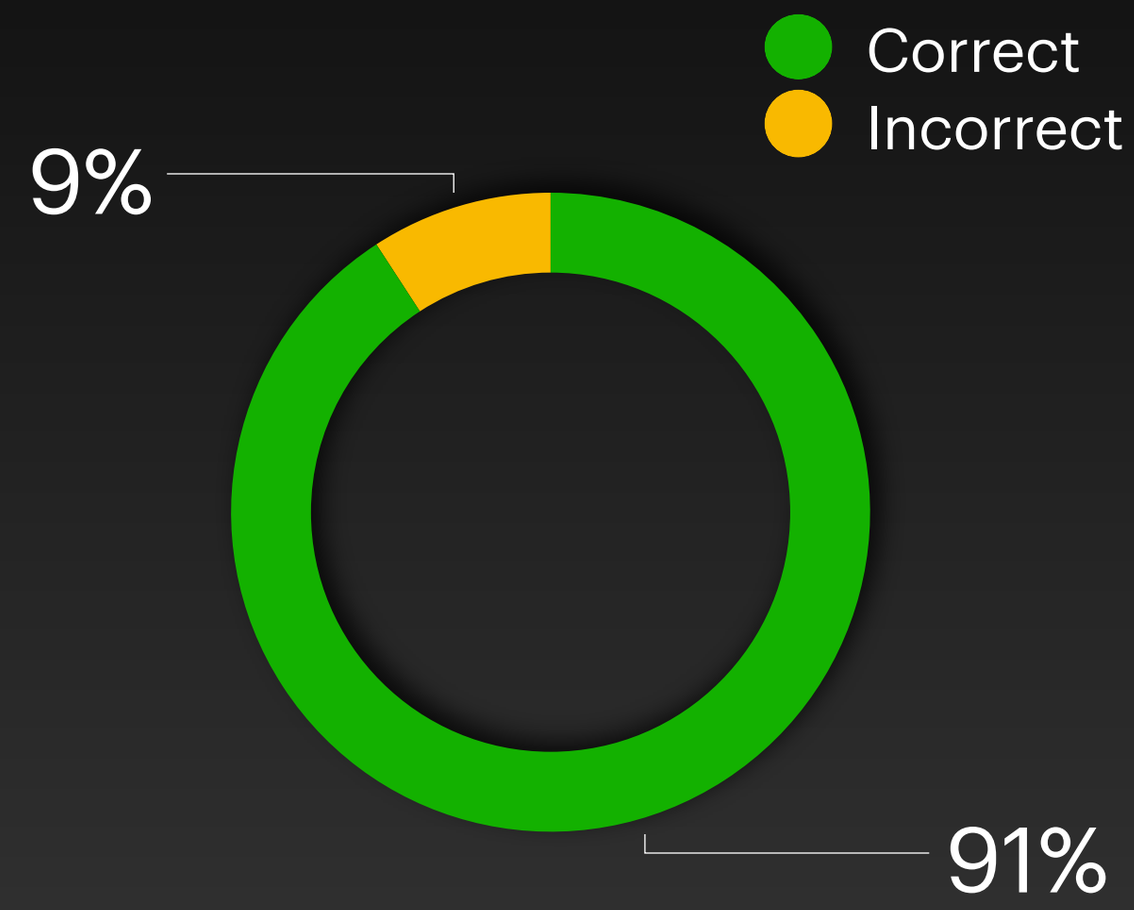
- Noticed most of the mistakes are made between ‘6’ and ‘8’
- 16 is often seen as 18 etc.
- But 18 is not often seen as 16.



Timestamp Validation

Results

- 455/501 frames had correct timestamps **90.82%**
- 46/501 frames had incorrect timestamps **9.18%**



Timestamp Validation

Conclusion

Based on these validation results

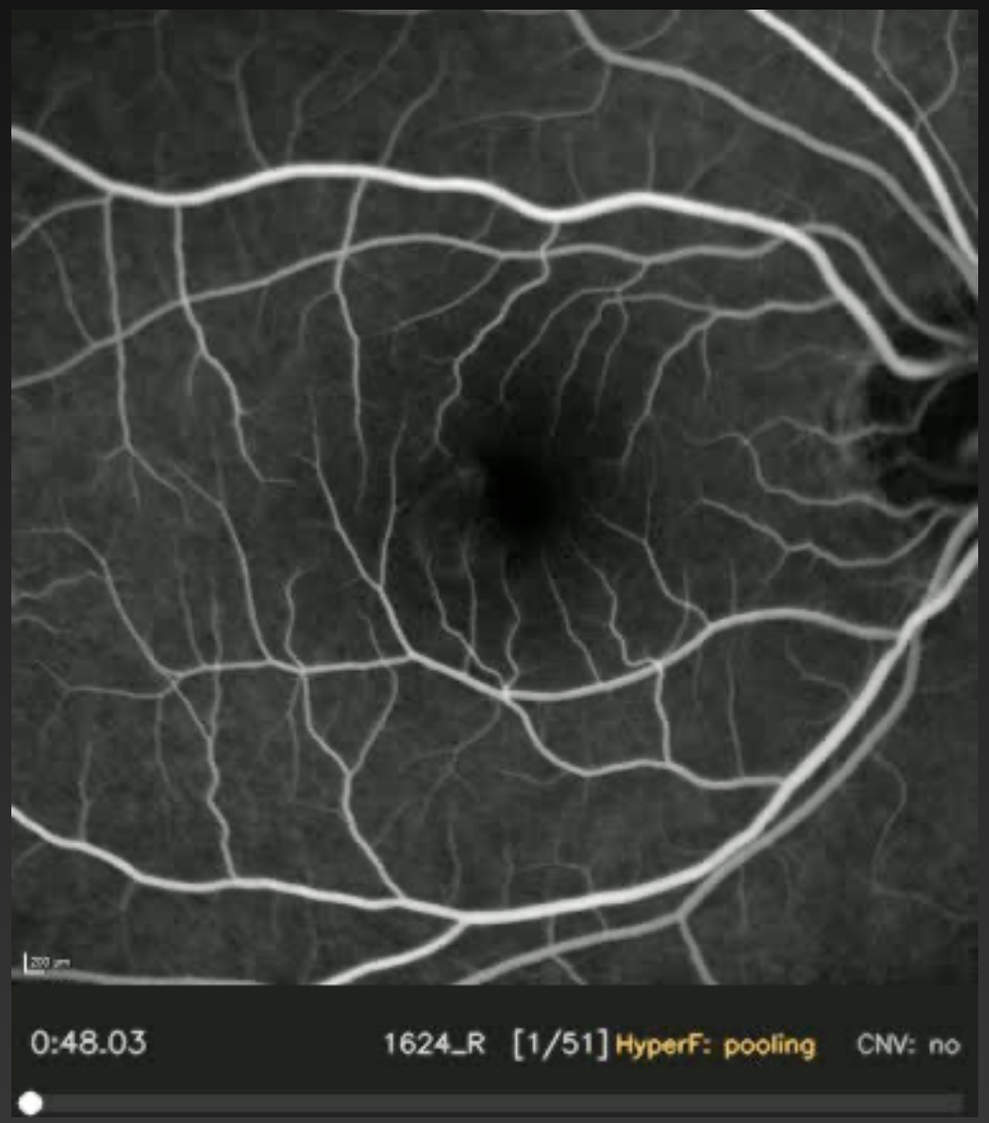
- The charts from last week show a good representation of the actual time distribution
- It is something that can be used later on for various occasions (like basing the binning on elapsed time)

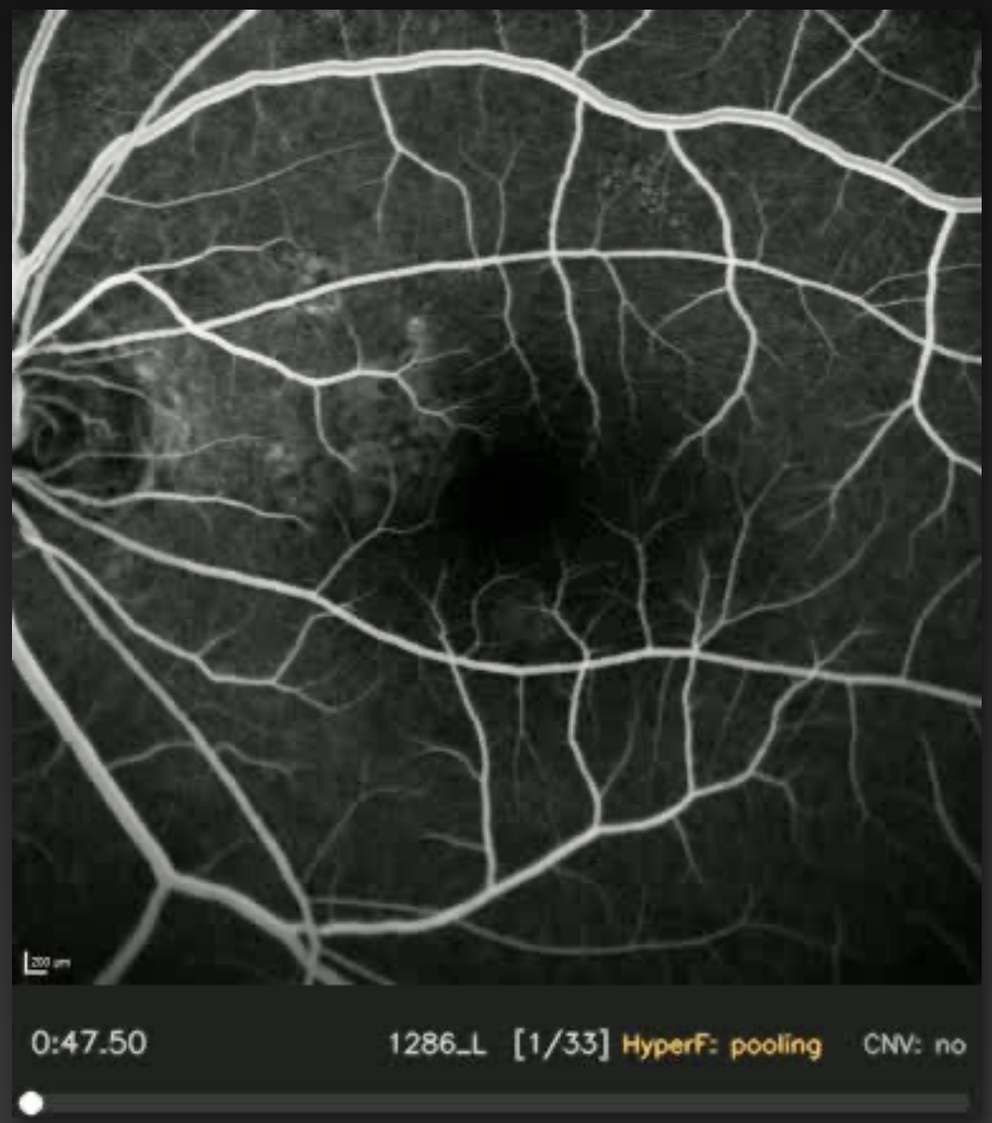
FA F2V (Frames2Video)

Goal is to produce useful videos out of the frames captured during an FA exam

FA F2V

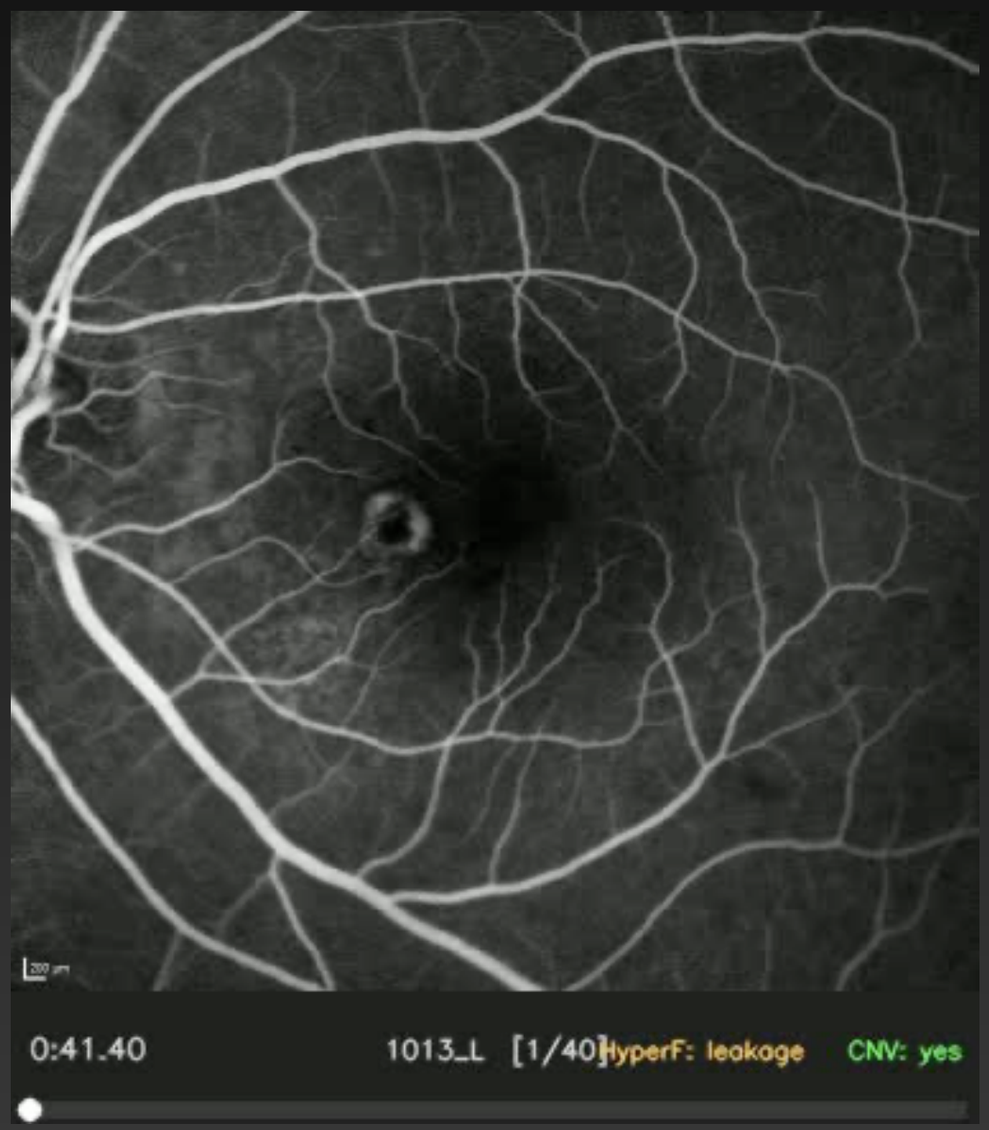
Pooling





FA F2V

Leakage



FA F2V

Staining

FA F2V

Window Defect

FA F2V

How it works

ORB (*Orientated FAST and Rotated BRIEF)*

✳ **FAST** for key point detection

- *Searches for points that differ from the environment*

✳ **BRIEF** for descriptor generation

- *Creates a compact binary string (descriptor) that describes the pixel density around each key point*

✳ ***Orientation & Rotation***

- *Adds orientation to each key point and rotates the brief descriptor. So they remain the same*

FA F2V

How it works

- The use of the key points is to make sure they key points align through following frames, so the video becomes more stable
- Based on the matched key points, you can estimate the rotation needed to align frame A with frame B
- ORB is a functionality within de python opencv library, which makes it very easy to implement
- Used the code snippets from **LearnopenCV.com**

FA F2V

Conclusion

The generated video can give an idea of how each of the HyperF_Type's evolve over time during the FA exam.

It is still possible to make better representative videos, using better exams, this would require manual searching for the right exams.

Overall, I had a fun and educational time exploring this and generating useful videos.

Study

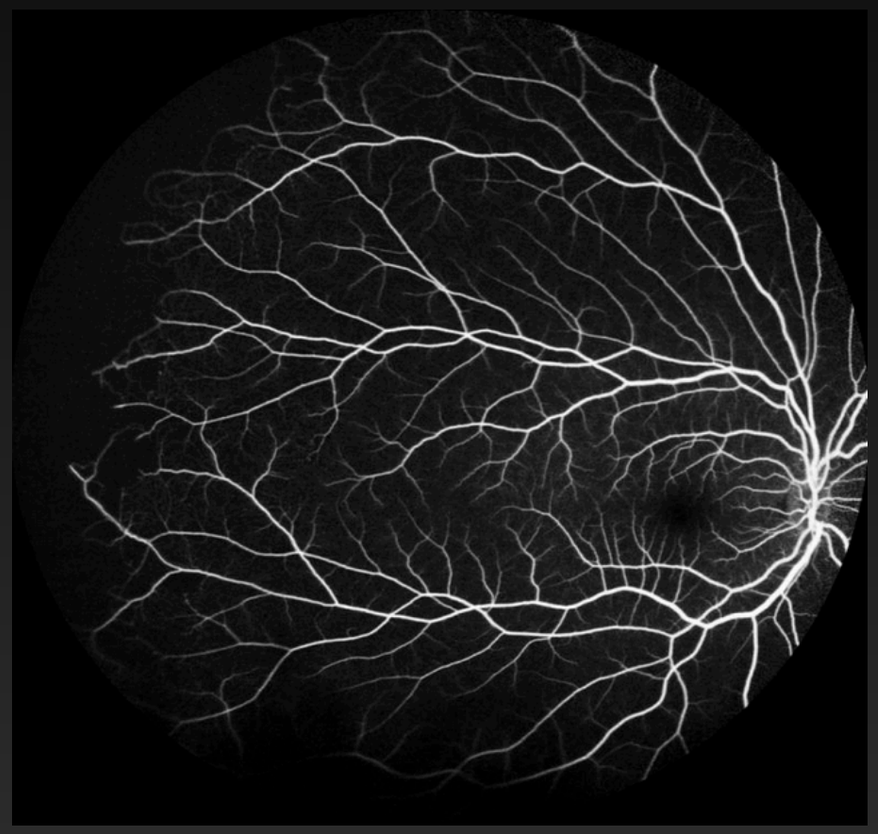
Dynamic versus static ultra-widefield fluorescein in eyes with diabetic retinopathy: a pilot prospective cross-sectional study

(Found by Oscar)

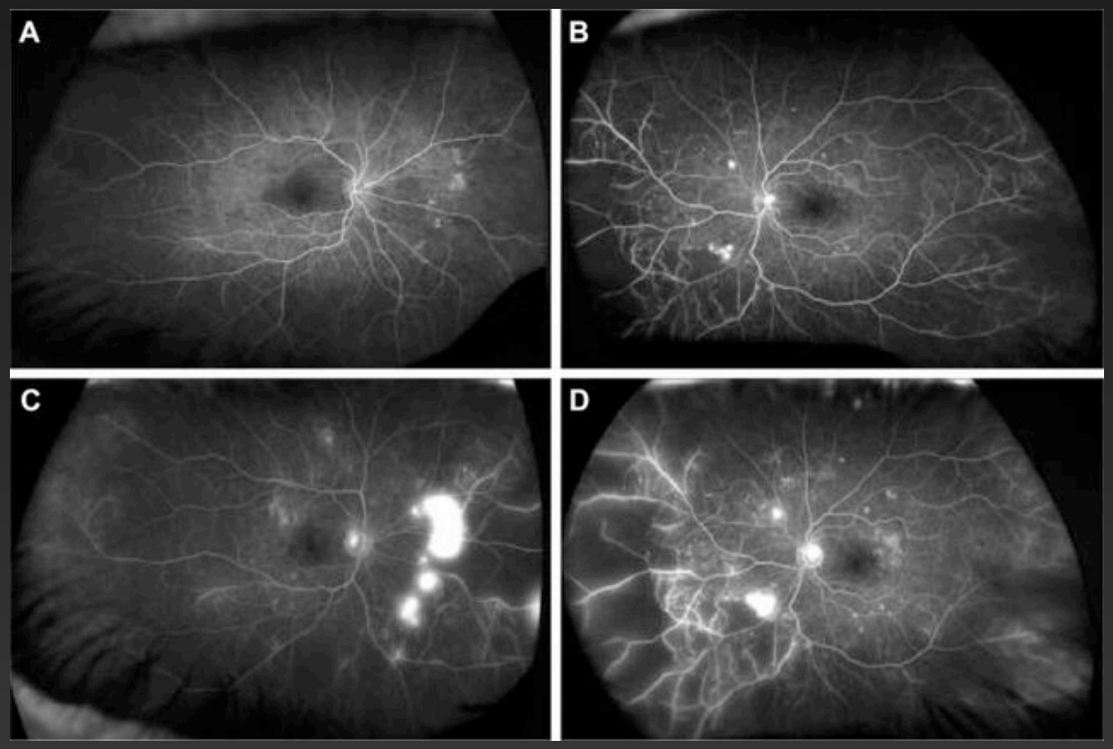
Study

Timelapse vs. Static images

UWFA



UWFA (Diabetic Retinopathy)



Study

Timelapse vs. Static images

- Timelapse helped distinguish noise from actual medical abnormalities

Temporality Experiment

Temporality Experiment

Research Question

Do visual embeddings contain temporal information?

Method: **PCA** analysis on **384D embeddings**

Data: **807** examinations, **2.500** frames analysed

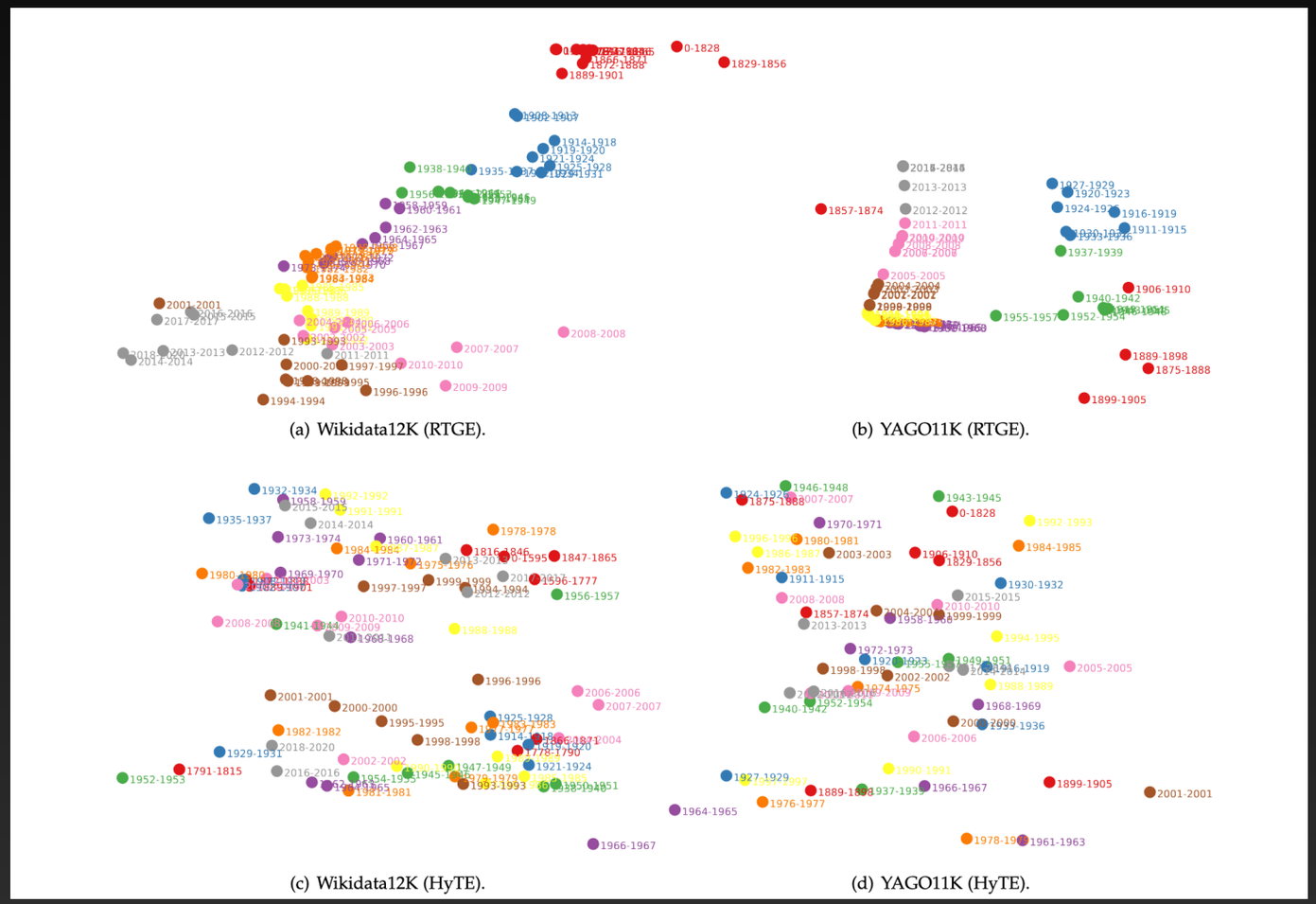
Temporality Experiment

Motivation

Studies

Time-aware Graph Embedding: A temporal smoothness and task-oriented approach

Learning Time Embedding for Temporal Knowledge Graph Completion



(a) Wikidata12K (RTGE). (b) YAGO11K (RTGE).

(c) Wikidata12K (HyTE). (d) YAGO11K (HyTE).

+ Talk with Roberto about tSNE plot

Temporality Experiment

Analysis overview

807 examinations

22,742 total frames

5 temporal bins (*equal sample size*)

384D (RETFound-Green) -> **2D PCA** reduction

1 Second -> 19 minutes Time range

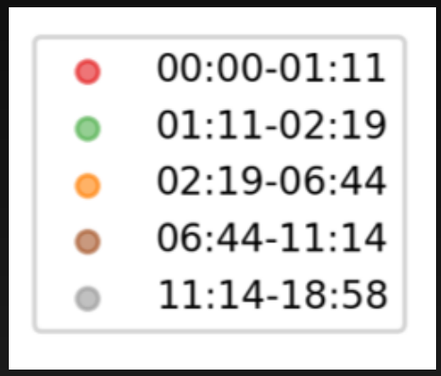
25.5% *variance* explained

Temporality Experiment

Methodology

Time binning

5 equal-sized temporal bins per pathology



Dimensionality Reduction

PCA: 384D -> 2D visualisation

Quantitative metric

ANOVA F-statistic (temporal separation)

Temporality Experiment

Methodology

Quantitative metric

ANOVA F-statistic (temporal separation)

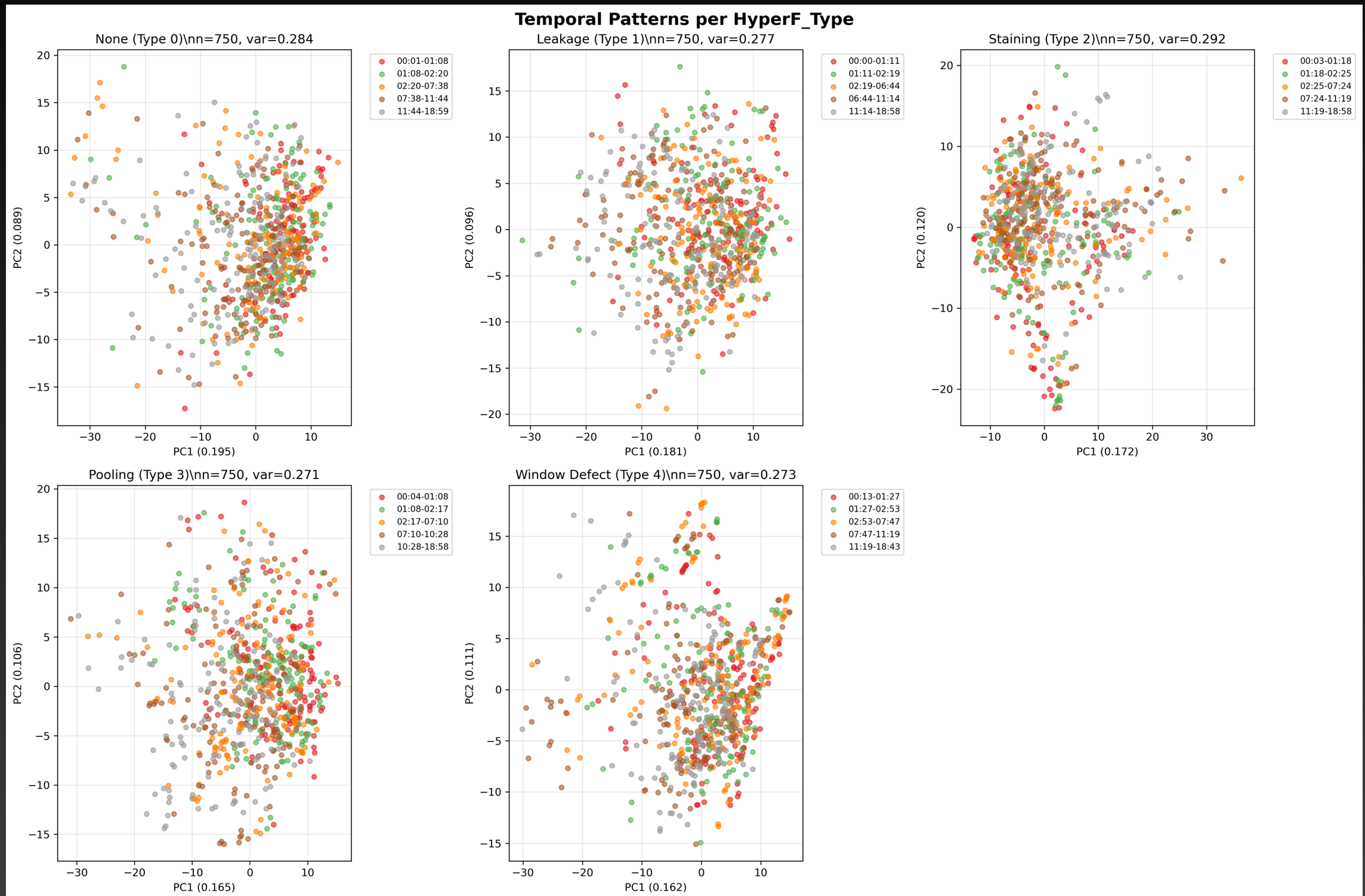
$$F = \frac{MSB}{MSW}$$

Mean Square Between groups (variance between time bins)

Mean Square Within groups (variance within each time bin)

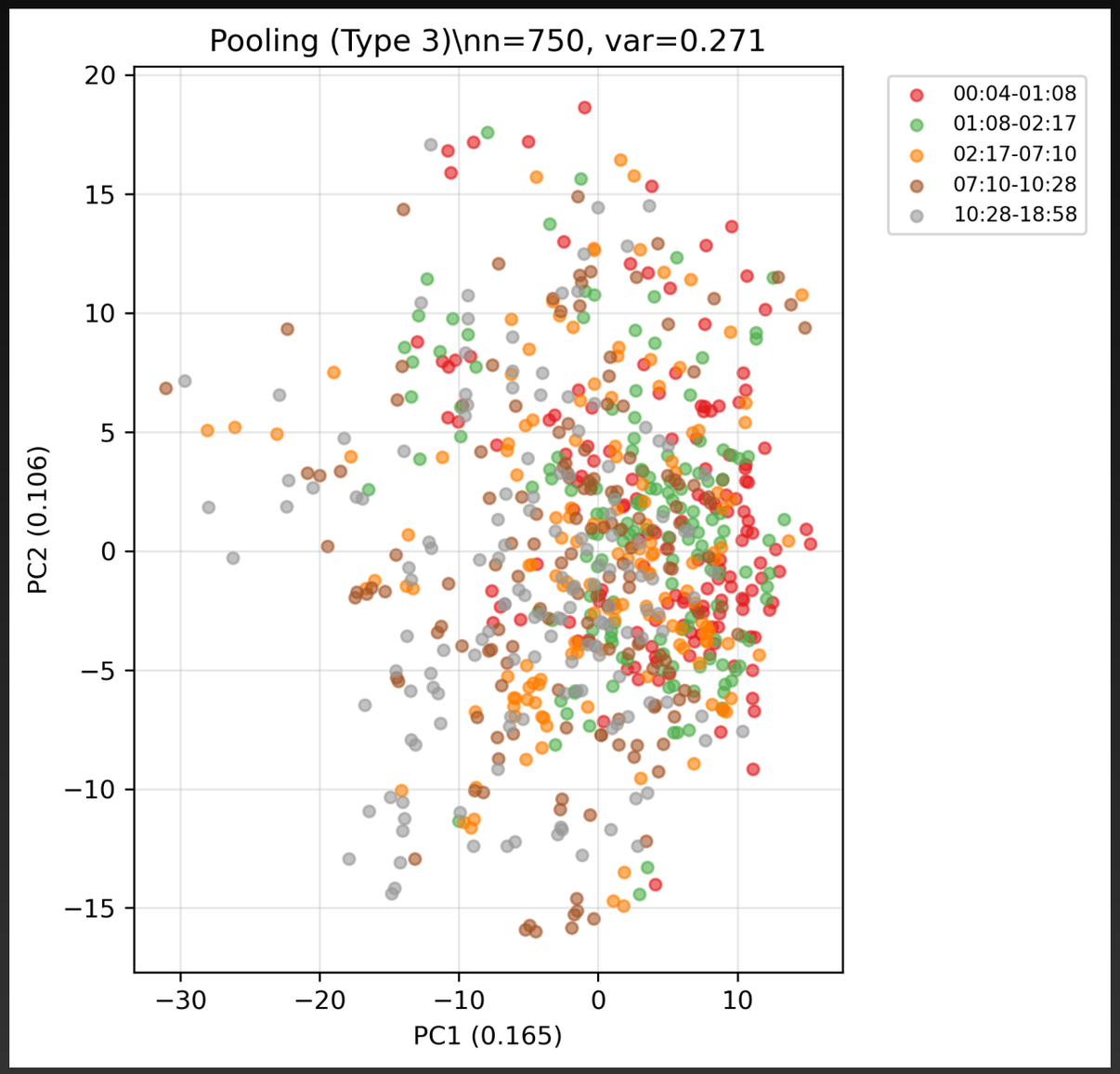
Temporality Experiment

Key Result



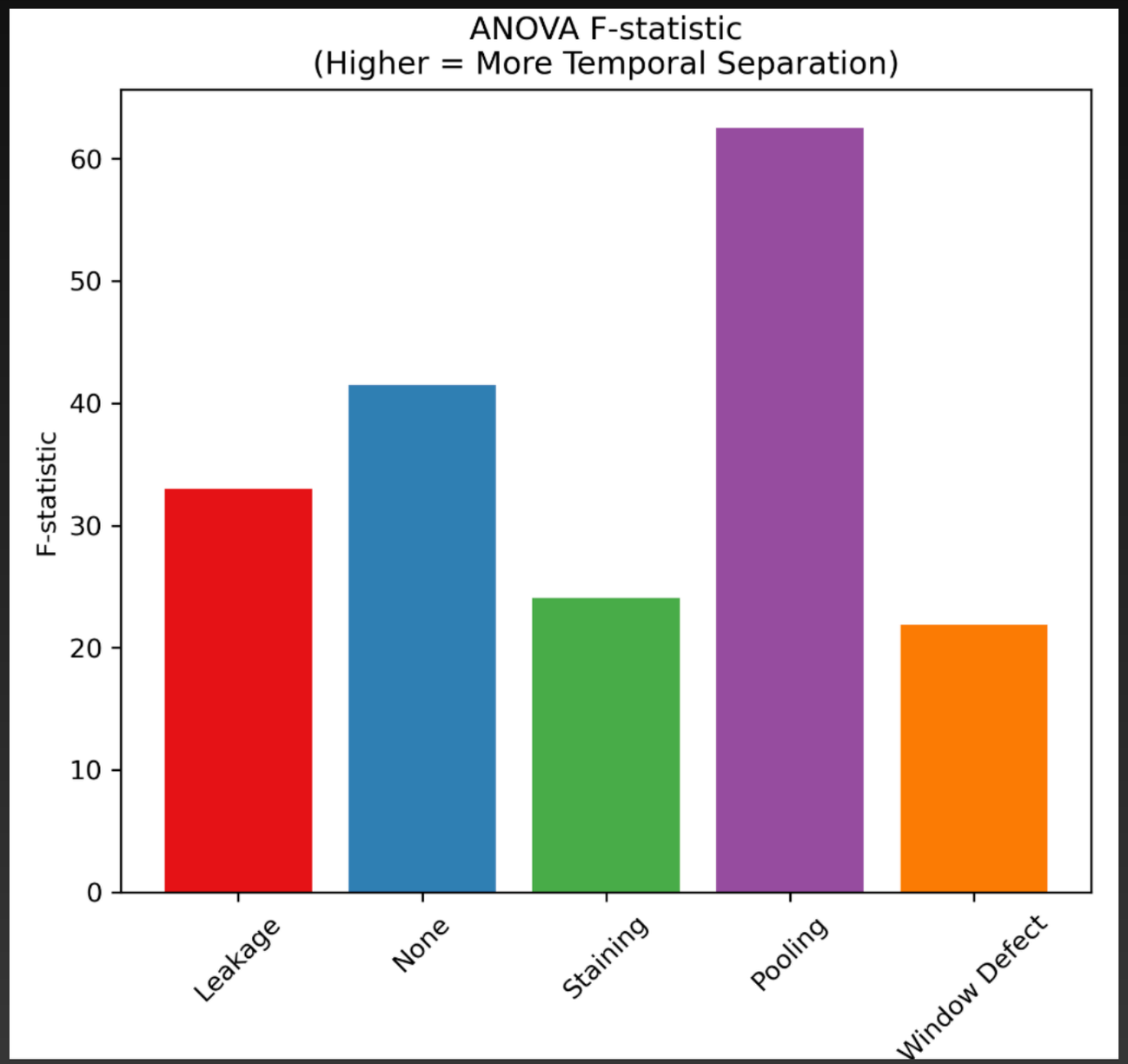
Temporality Experiment

Key Result



Temporality Experiment

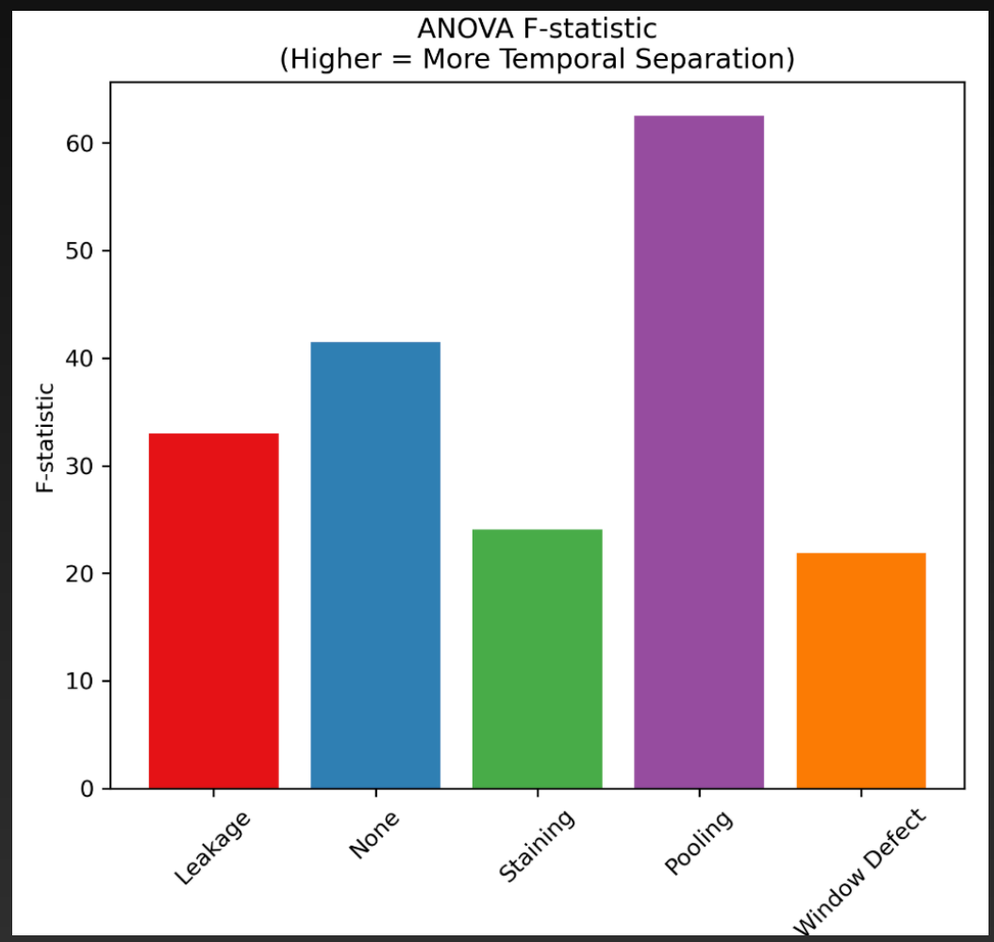
Key Result



Temporality Experiment

Key Result

ANOVA confirms structure in embeddings, with strongest separation in **pooling** and weakest in **window defect**



Temporality Experiment

Limitations

- 2D PCA compression loss
- ANOVA on PC1 only
- Potential confounding factors
- Class/sample imbalance effects

Temporality Experiment

Key Result

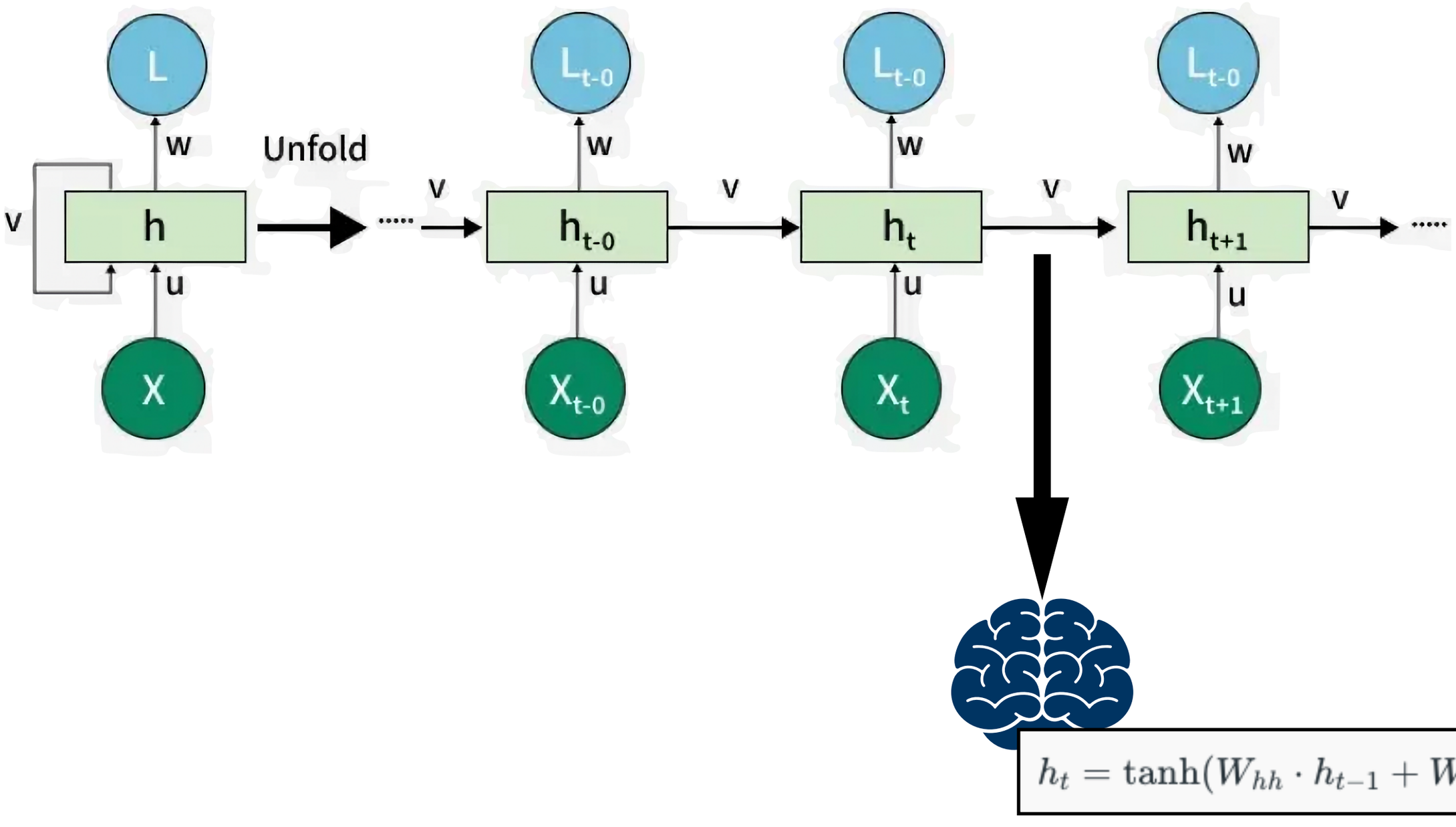
Could be useful to use these **time_bins** after all

LTSM / GRU

Applying LSTM and/or GRU to the Angioreport sequential data

LSTM / GRU

Recurrent Neural Network (RNN)



$$h_t = \tanh(W_{hh} \cdot h_{t-1} + W_{xh} \cdot x_t)$$

LSTM / GRU

Recurrent Neural Network (RNN)

$$h_{t-1} = \begin{bmatrix} 0.1 \\ -0.4 \end{bmatrix}$$

$$x_1 = 0.5$$

$$W_{hh} = \begin{bmatrix} 0.7 & -0.2 \\ 0.1 & 0.4 \end{bmatrix}$$

$$W_{xh} = \begin{bmatrix} 0.6 \\ -0.3 \end{bmatrix}$$

Linear step

$$a_t = W_{hh}h_{t-1} + W_{xh}x_t = \begin{bmatrix} 0.45 \\ -0.30 \end{bmatrix}$$

Non-Linear step

$$h_t = tanh(a_t) \approx \begin{bmatrix} 0.422 \\ -0.291 \end{bmatrix}$$

LSTM / GRU

Recurrent Neural Network (RNN)

Linear step

$$a_t = W_{hh}h_{t-1} + W_{xh}x_t = \begin{bmatrix} 0.45 \\ -0.30 \end{bmatrix}$$

Non-Linear step

$$h_t = tanh(a_t) \approx \begin{bmatrix} 0.422 \\ -0.291 \end{bmatrix}$$

$t + 1$

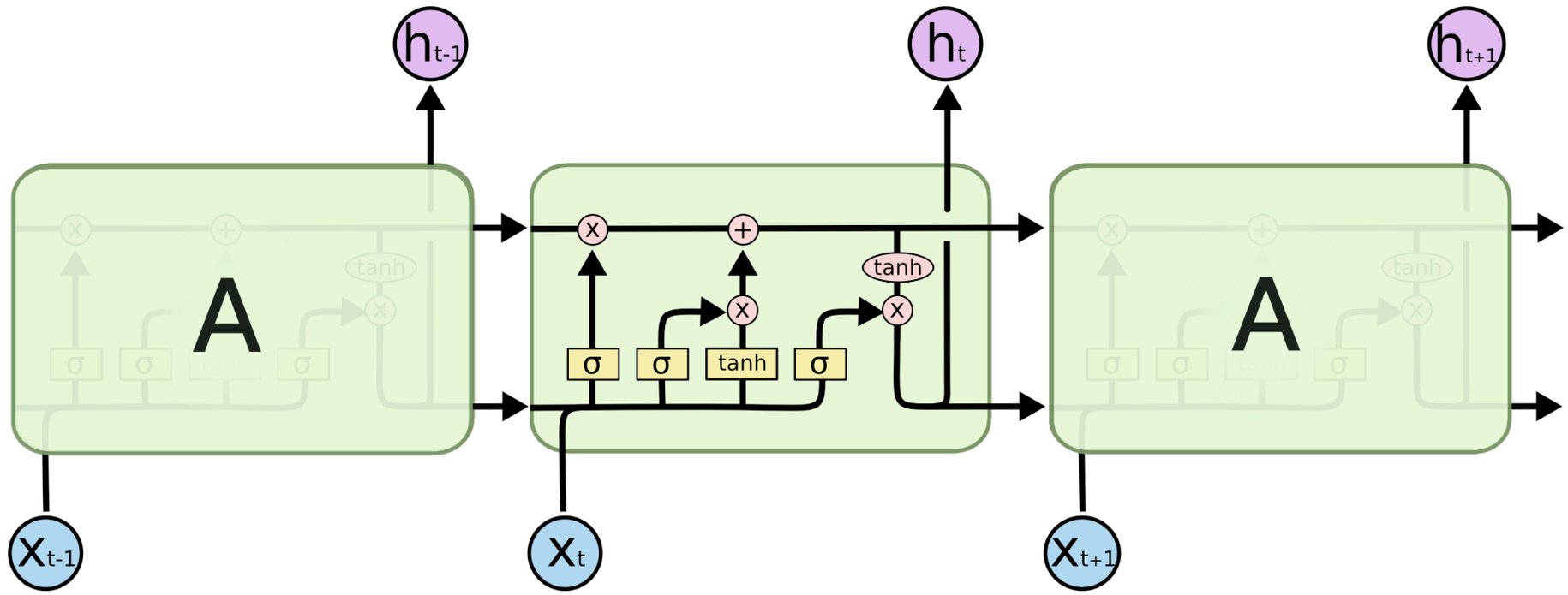
$$h_{t-1} \approx \begin{bmatrix} 0.422 \\ -0.291 \end{bmatrix}$$

LSTM / GRU

LSTM

Discovered in 1997

<u>Literature</u>

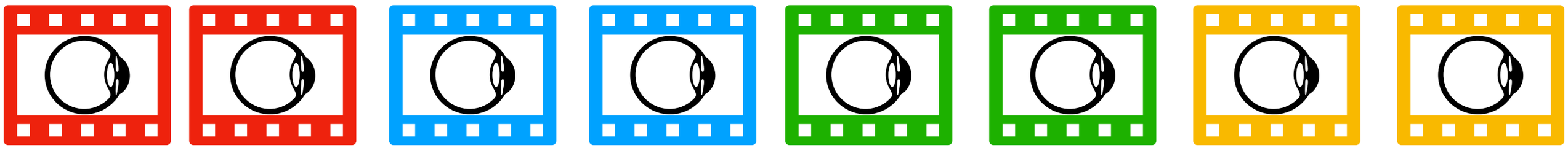


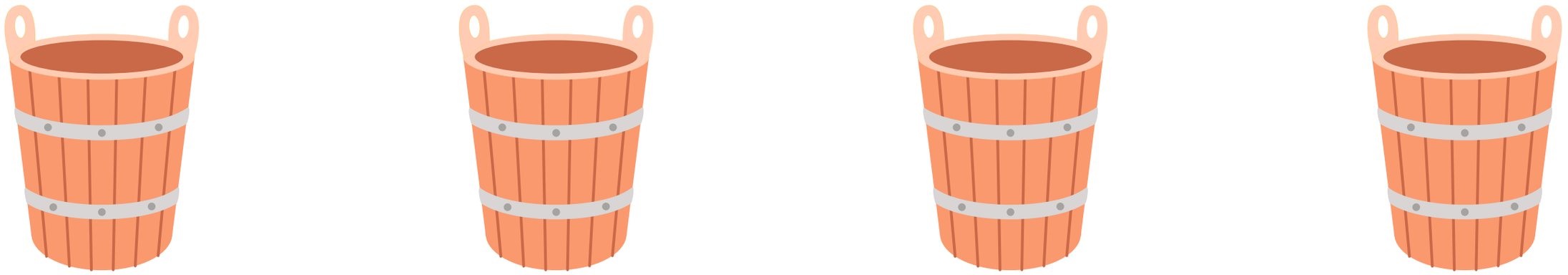
LSTM / GRU

LSTM

LSTM / GRU

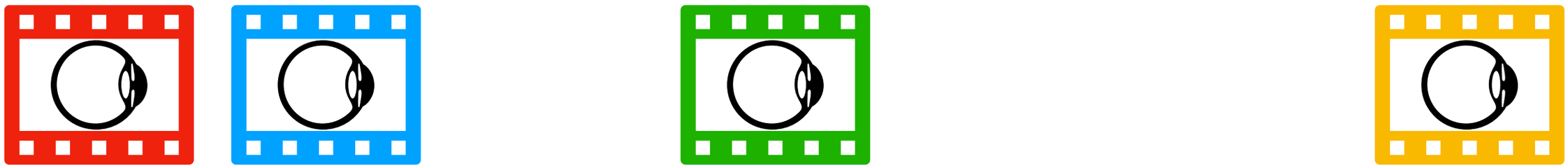
LSTM

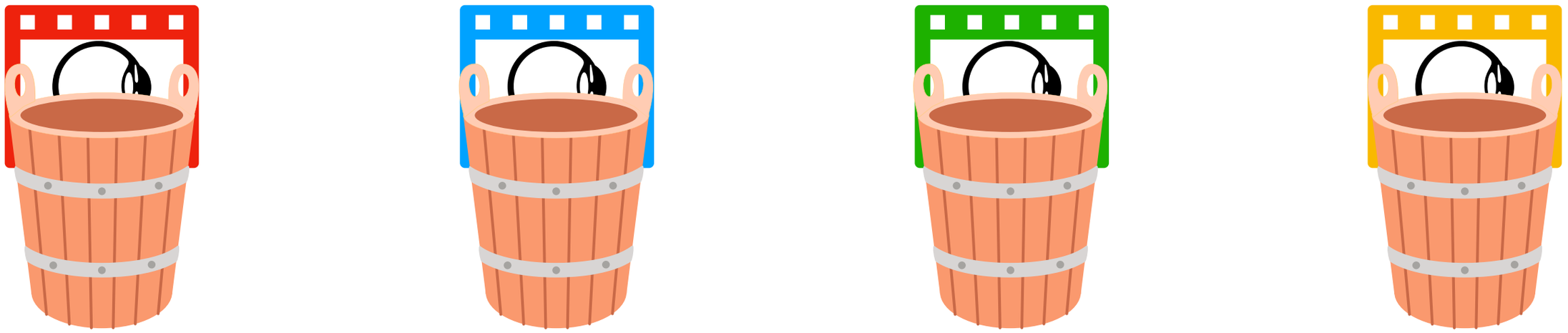




LSTM / GRU

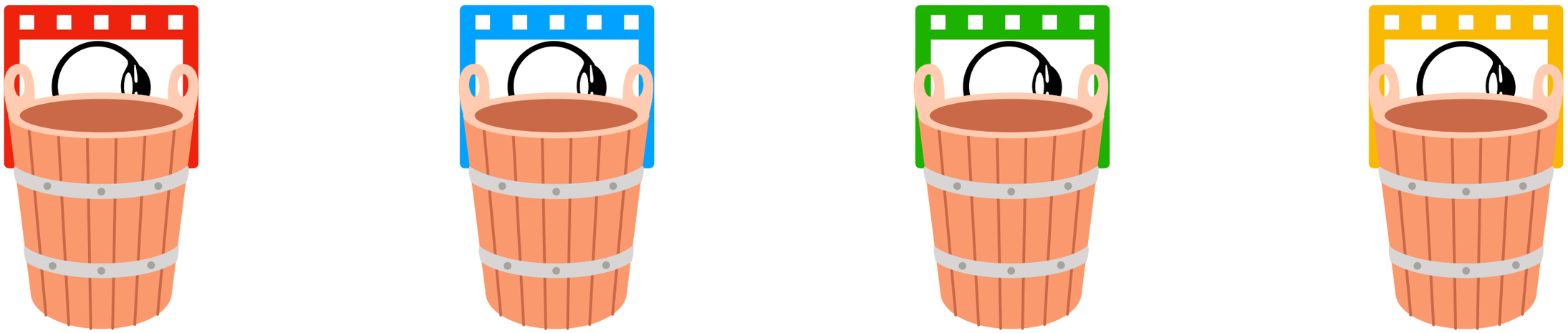
LSTM





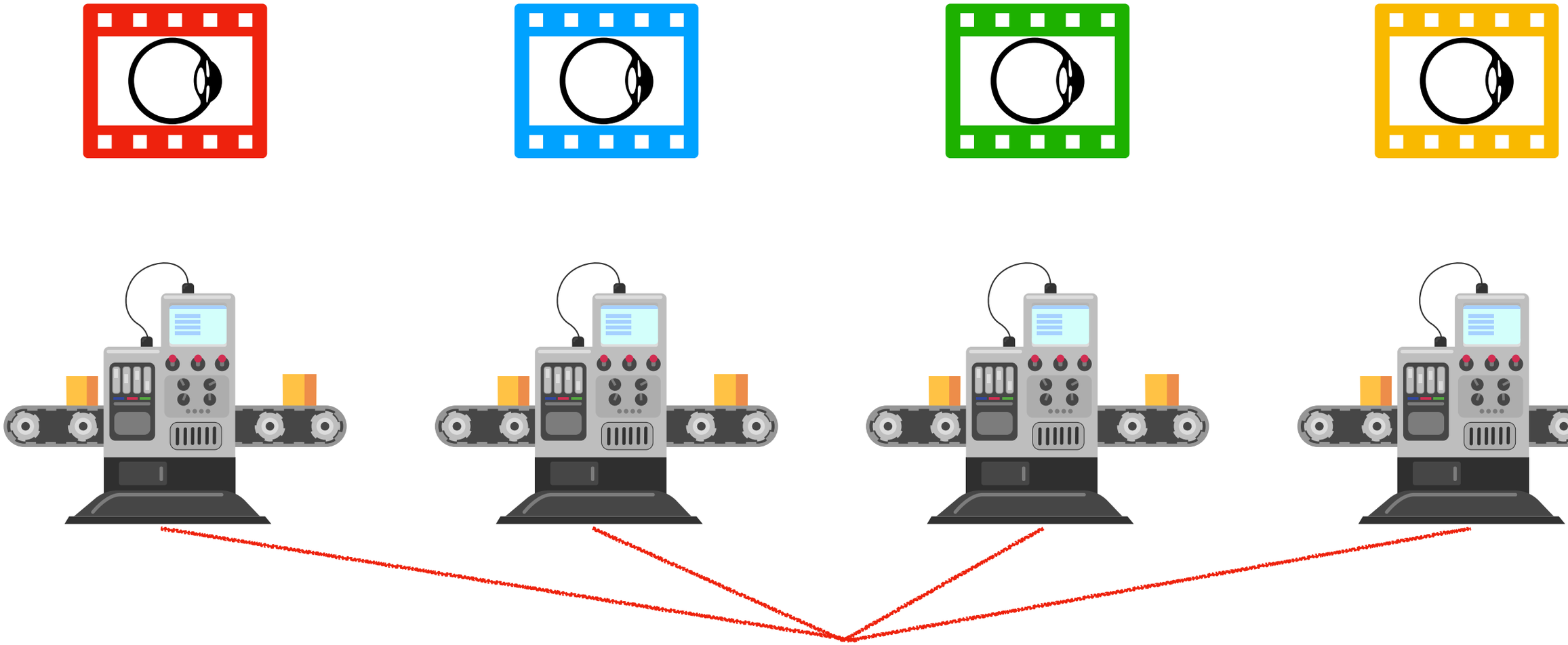
LSTM / GRU

LSTM



LSTM / GRU

LSTM

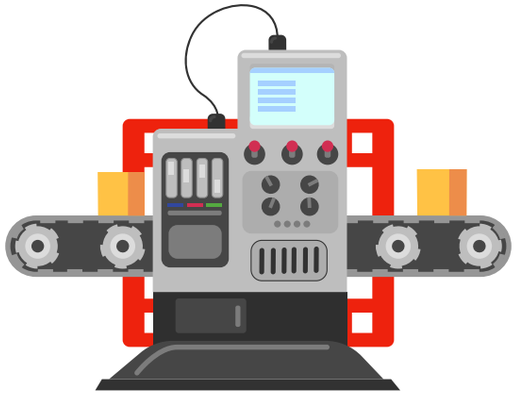
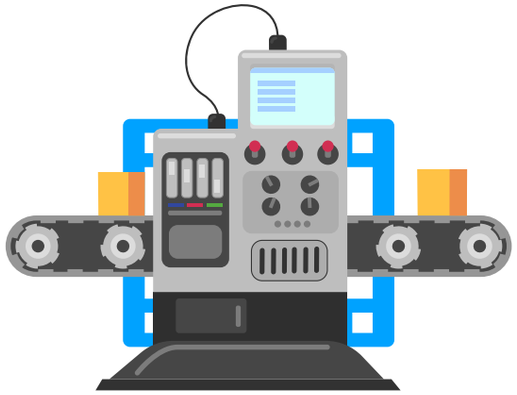
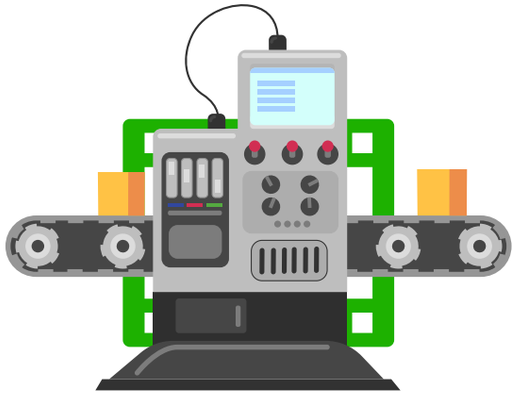
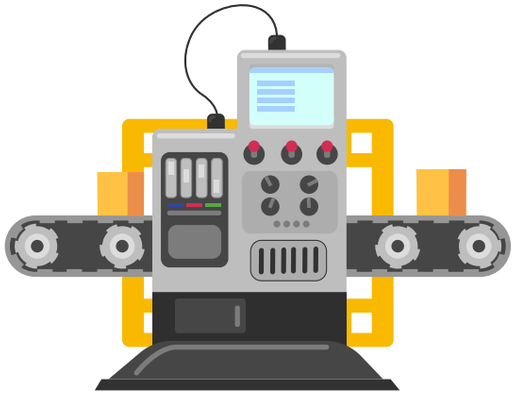


RETFound-Green

Vision backbone

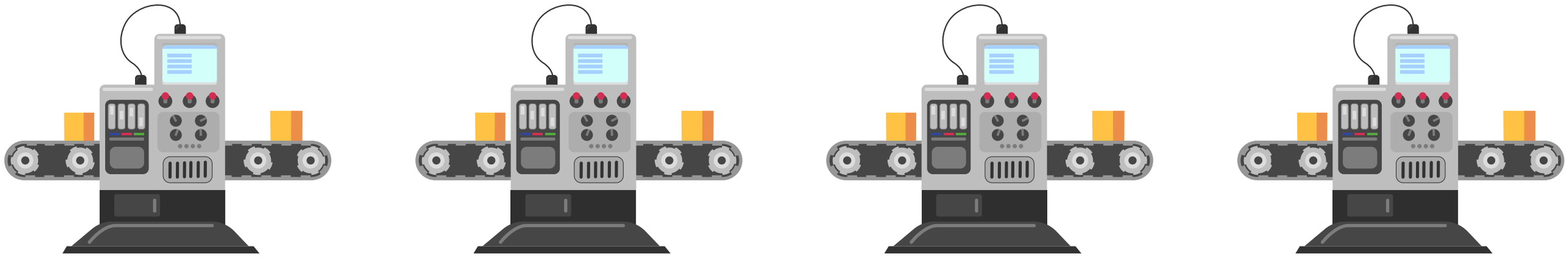
LSTM / GRU

LSTM

LSTM / GRU

LSTM



Embedding

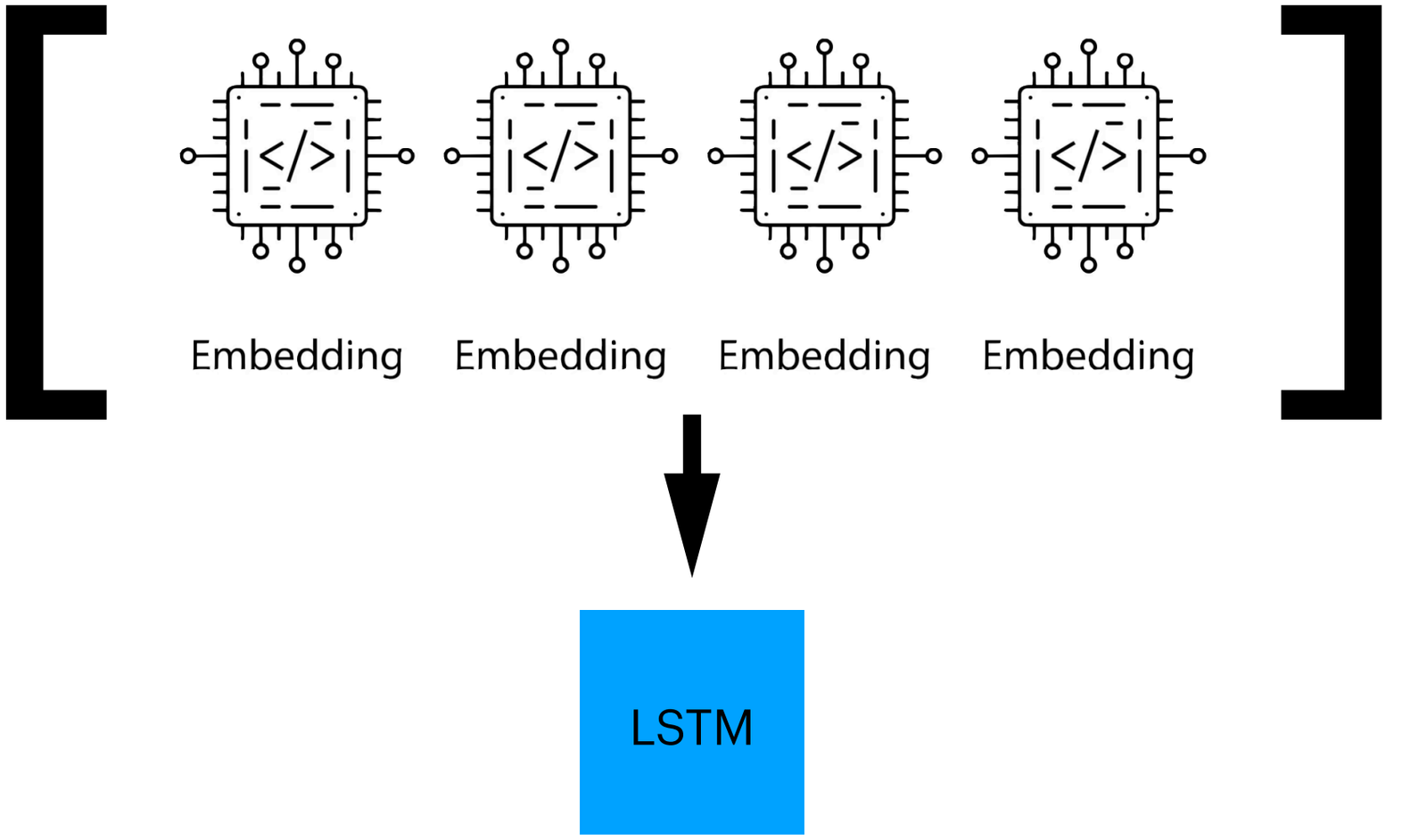
Embedding

Embedding

Embedding

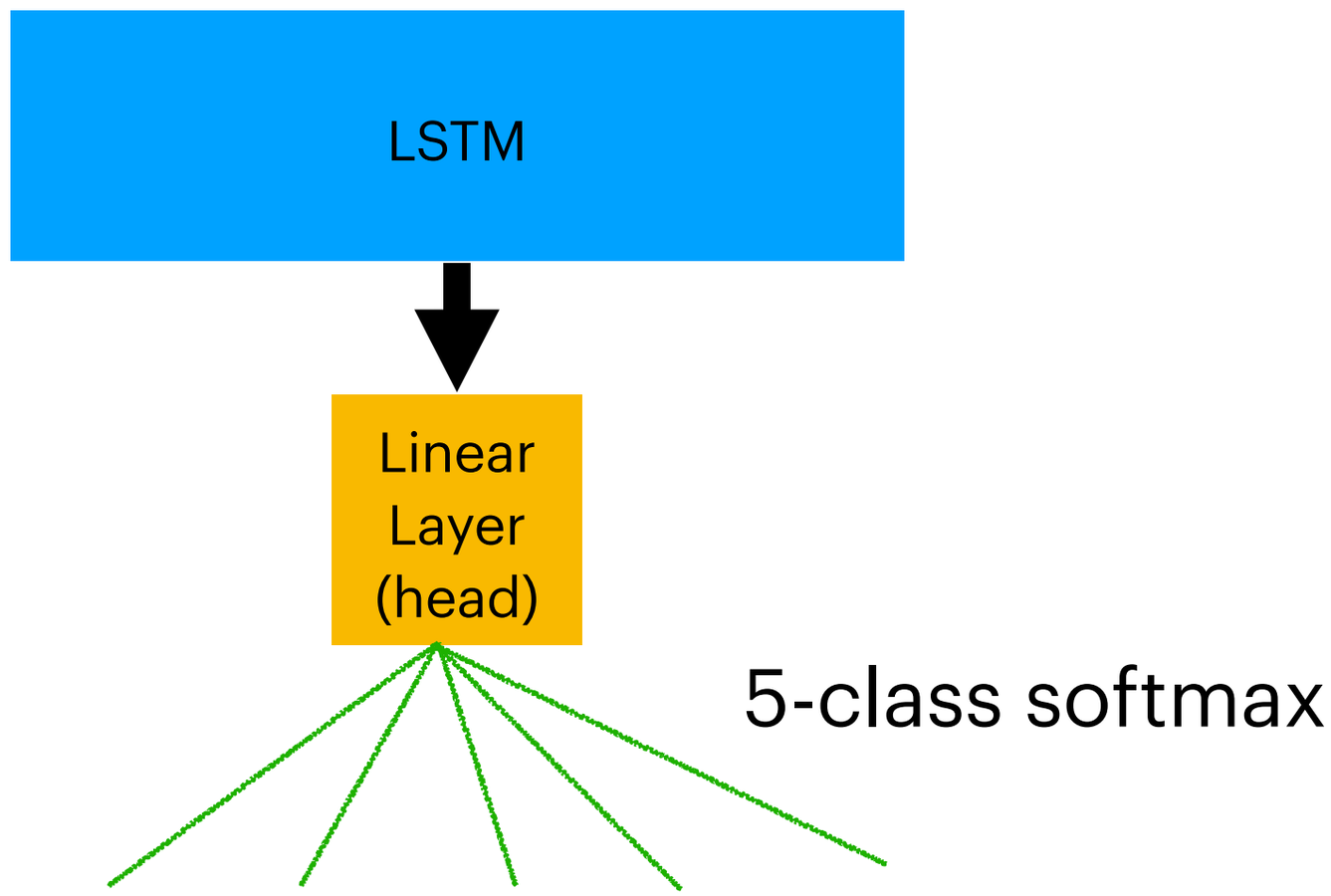
LSTM / GRU

LSTM



LSTM / GRU

LSTM



Param distribution

```
  | Name            | Type               | Params | Mode
-------------------------------------------------------------
0 | train_loss      | CrossEntropyLoss   | 0      | train
1 | validation_loss | CrossEntropyLoss   | 0      | train
2 | model           | VisionTransformer  | 21.6 M | train
3 | _normalizer     | Normalize          | 0      | train
4 | auc             | MulticlassAUROC    | 0      | train
5 | temporal_lstm   | LSTM               | 1.2 M  | train
-------------------------------------------------------------
22.8 M    Trainable params
0         Non-trainable params
22.8 M    Total params
91.258    Total estimated model params size (MB)
281       Modules in train mode
0         Modules in eval mode
```

Percentages showing the distribution of HyperF_Type exams

*After 30 epochs FFT

LSTM / GRU

*1 LSTM layer

Results

n_bins = 3 (random frame picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.84	0.70
Staining *(18.9%)*	0.70	0.40
None *(24.8%)*	0.95	0.86
Pooling *(10.4%)*	0.73	0.19
Window Defect *(6.7%)*	0.79	0.22

Last frame baseline

Class	AUC	AP
Leakage *(39.3%)*	0.87	0.76
Staining *(18.9%)*	0.79	0.56
None *(24.8%)*	0.95	0.89
Pooling *(10.4%)*	0.77	0.33
Window Defect *(6.7%)*	0.79	0.23

*After 30 epochs FFT

LSTM / GRU

*1 LSTM layer

Results

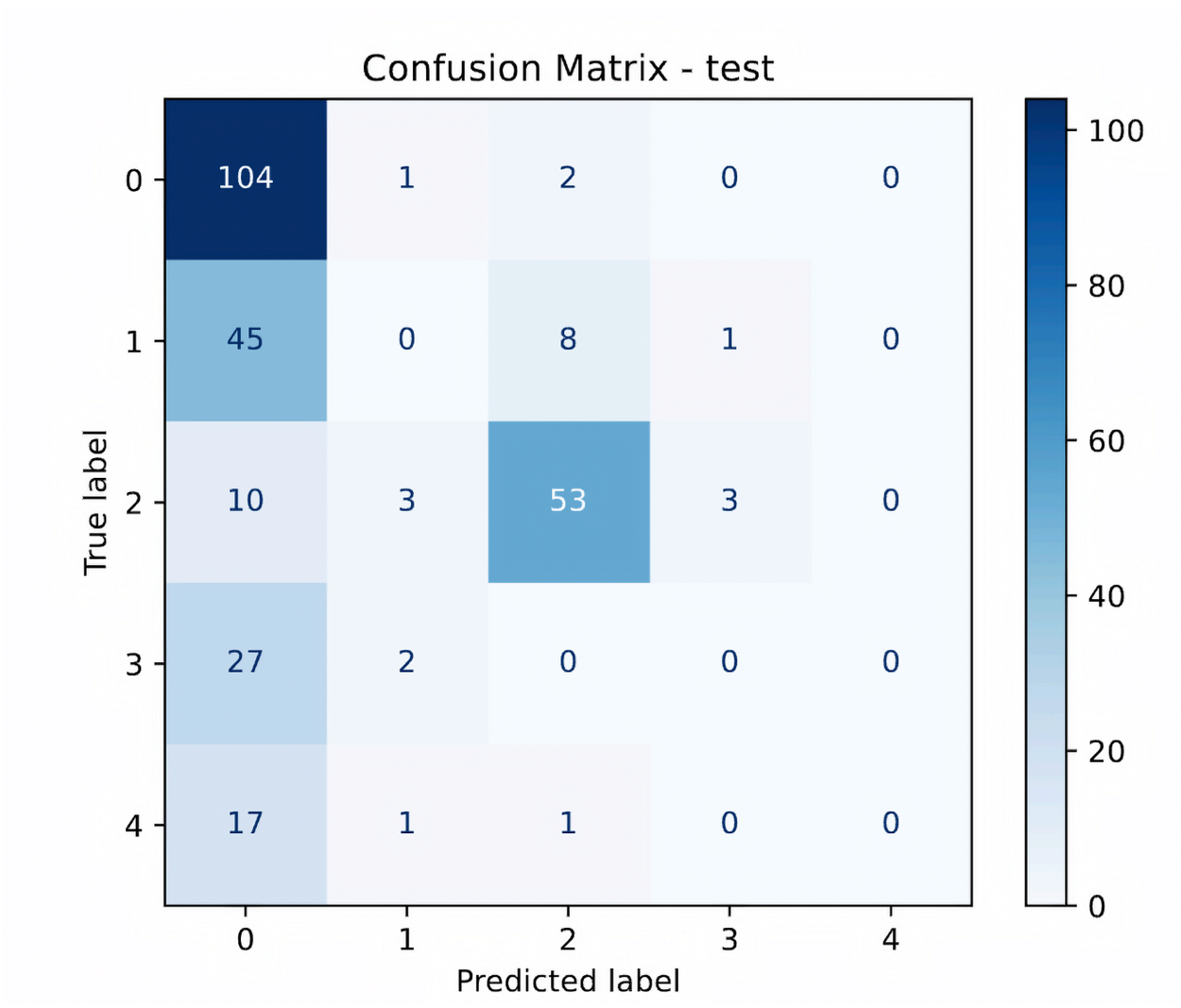
Last frame baseline

Class	AUC	AP
Leakage *(39.3%)*	0.87	0.76
Staining *(18.9%)*	0.79	0.56
None *(24.8%)*	0.95	0.89
Pooling *(10.4%)*	0.77	0.33
Window Defect *(6.7%)*	0.79	0.23

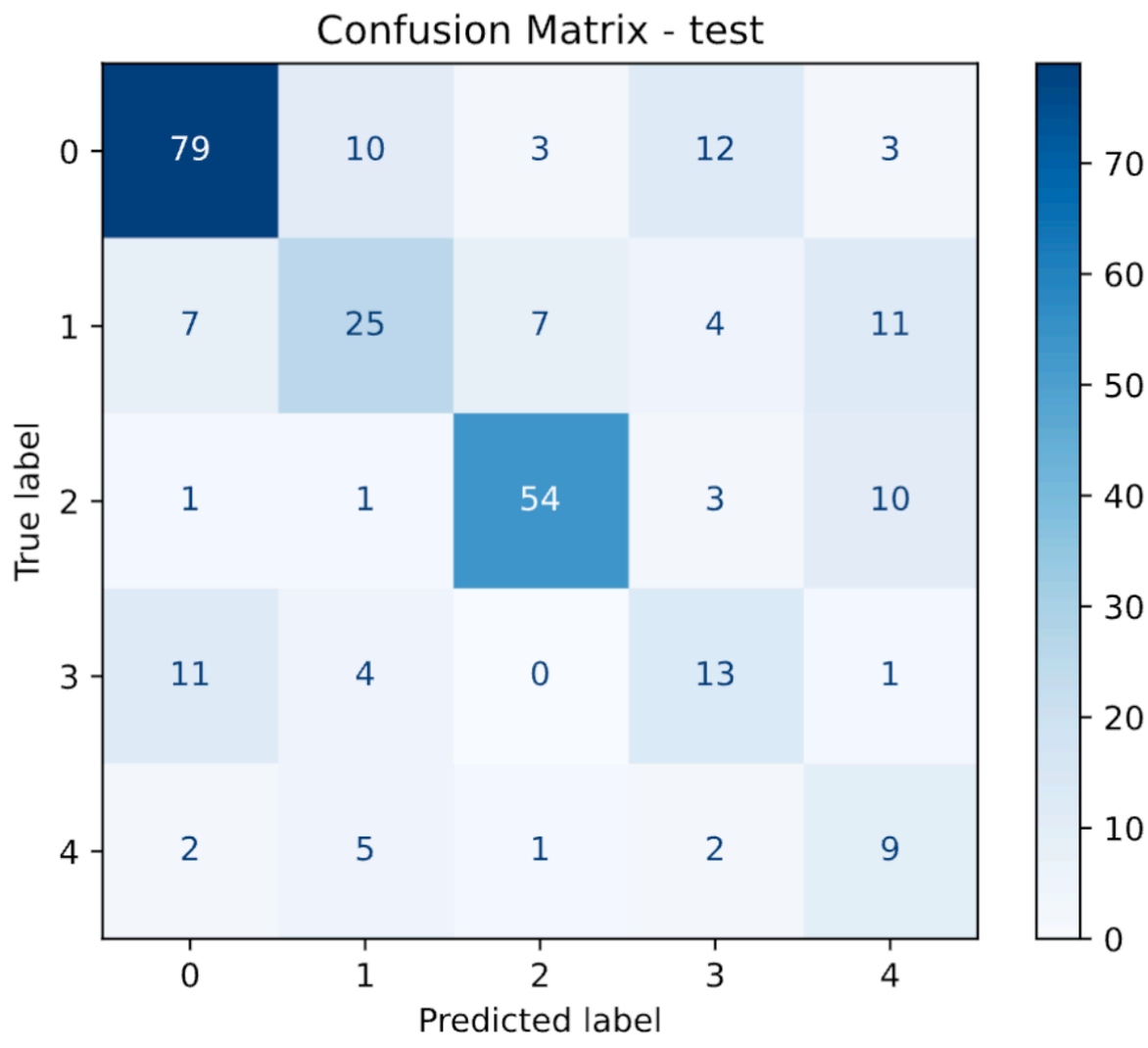
LSTM / GRU

Results

n_bins = 3 (random frame picking)



Last frame baseline



*After 30 epochs FFT

LSTM / GRU

*1 LSTM layer

Results

n_bins = 3 (random frame picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.84	0.70
Staining *(18.9%)*	0.70	0.40
None *(24.8%)*	0.95	0.86
Pooling *(10.4%)*	0.73	0.19
Window Defect *(6.7%)*	0.79	0.22

Last 2 frames (avg)

Class	AUC	AP
Leakage *(39.3%)*	0.85	0.73
Staining *(18.9%)*	0.75	0.40
None *(24.8%)*	0.95	0.83
Pooling *(10.4%)*	0.78	0.38
Window Defect *(6.7%)*	0.90	0.46

LSTM / GRU

Results

Last 2 frames (avg)

Class	AUC	AP
Leakage *(39.3%)*	0.85	0.73
Staining *(18.9%)*	0.75	0.40
None *(24.8%)*	0.95	0.83
Pooling *(10.4%)*	0.78	0.38
Window Defect *(6.7%)*	0.90	0.46

**After 28 epochs FFT

*After 30 epochs FFT

**1 LSTM layer

LSTM / GRU

*1 LSTM layer

Results

n_bins = 3 (random frame picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.79	0.63
Staining *(18.9%)*	0.58	0.23
None *(24.8%)*	0.96	0.84
Pooling *(10.4%)*	0.65	0.19
Window Defect *(6.7%)*	0.73	0.14

n_bins = 5 (random frame picking)**

Class	AUC	AP
Leakage *(39.3%)*	0.85	0.69
Staining *(18.9%)*	0.70	0.37
None *(24.8%)*	0.96	0.85
Pooling *(10.4%)*	0.71	0.20
Window Defect *(6.7%)*	0.77	0.21

**After 28 epochs FFT

**1 LSTM layer

LSTM / GRU

Results

n_bins = 5 (random frame picking)**

Class	AUC	AP
Leakage *(39.3%)*	0.85	0.69
Staining *(18.9%)*	0.70	0.37
None *(24.8%)*	0.96	0.85
Pooling *(10.4%)*	0.71	0.20
Window Defect *(6.7%)*	0.77	0.21

*After 28 epochs FFT

*1 LSTM layer

LSTM / GRU

*After 28 epochs FFT

*1 LSTM layer

Results

n_bins = 5 (random frame picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.85	0.69
Staining *(18.9%)*	0.70	0.37
None *(24.8%)*	0.96	0.85
Pooling *(10.4%)*	0.71	0.20
Window Defect *(6.7%)*	0.77	0.21

n_bins = 10 (random frame picking)**

Class	AUC	AP
Leakage *(39.3%)*	0.87	0.76
Staining *(18.9%)*	0.74	0.46
None *(24.8%)*	0.96	0.81
Pooling *(10.4%)*	0.71	0.21
Window Defect *(6.7%)*	0.82	0.28

*After 28 epochs FFT

LSTM / GRU

*1 LSTM layer

Results

n_bins = 10 (random frame picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.87	0.76
Staining *(18.9%)*	0.74	0.46
None *(24.8%)*	0.96	0.81
Pooling *(10.4%)*	0.71	0.21
Window Defect *(6.7%)*	0.82	0.28

Winner of random picking!

*After 28 epochs FFT

LSTM / GRU

*1 LSTM layer

Results

Rand_Pick

Class	AUC	AP
Leakage	0.87	0.76
Staining	0.74	0.46
None	0.96	0.81
Pooling	0.71	0.21
Window	0.82	0.28

Class	AUC	AP
Leakage		
Staining		
None		
Pooling		
Window		

Class	AUC	AP
Leakage		
Staining		
None		
Pooling		
Window		

“Life is a random experience”

~ Anthony T. Hincks

But why should the bins be random?

“Life is a random experience”
~ Anthony T. Hincks

They shouldn’t have to be

```python
for i in range(actual_bins):
    start = i * bin_size + min(i, remainder)
    end = (i + 1) * bin_size + min(i + 1, remainder)
    bin_frames = frame_files[start:end]
    rand_fram = random.randint(0, (len(bin_frames)-1))

    #This is made for always using the first frame and the last (seemed logical)
    if i == 0:
        chosen = frame_files[0]
    elif i == actual_bins - 1:
        chosen = frame_files[-1]
    else:
        mid = len(bin_frames) // 2
        chosen = bin_frames[mid]

    #RThis is made for picking random frames for each bin
    #chosen = bin_frames[rand_fram]
```

*After 17 epochs FFT

*After 30 epochs FFT

*1 LSTM layer

LSTM / GRU

*1 LSTM layer

Results

n_bins = 3 (random frame picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.84	0.70
Staining *(18.9%)*	0.70	0.40
None *(24.8%)*	0.95	0.86
Pooling *(10.4%)*	0.73	0.19
Window Defect *(6.7%)*	0.79	0.22

n_bins = 3 (non_random picking)**

Class	AUC	AP
Leakage *(39.3%)*	0.87	0.75
Staining *(18.9%)*	0.77	0.51
None *(24.8%)*	0.97	0.90
Pooling *(10.4%)*	0.78	0.34
Window Defect *(6.7%)*	0.83	0.28

*After 18 epochs FFT

LSTM / GRU

*1 LSTM layer

Results

n_bins = 3 (non_random picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.87	0.75
Staining *(18.9%)*	0.77	0.51
None *(24.8%)*	0.97	0.90
Pooling *(10.4%)*	0.78	0.34
Window Defect *(6.7%)*	0.83	0.28

**After 17 epochs FFT

**1 LSTM layer

LSTM / GRU

Results

*After 18 epochs FFT

*1 LSTM layer

n_bins = 3 (non_random picking)*

Class	AUC	AP
Leakage *(39.3%)*	0.87	0.75
Staining *(18.9%)*	0.77	0.51
None *(24.8%)*	0.97	0.90
Pooling *(10.4%)*	0.78	0.34
Window Defect *(6.7%)*	0.83	0.28

n_bins = 10 (non_random picking)**

Class	AUC	AP
Leakage *(39.3%)*	0.90	0.81
Staining *(18.9%)*	0.80	0.56
None *(24.8%)*	0.97	0.88
Pooling *(10.4%)*	0.79	0.34
Window Defect *(6.7%)*	0.89	0.41

**After 17 epochs FFT

**1 LSTM layer

LSTM / GRU

Results

n_bins = 10 (non_random picking)**

Class	AUC	AP
Leakage *(39.3%)*	0.90	0.81
Staining *(18.9%)*	0.80	0.56
None *(24.8%)*	0.97	0.88
Pooling *(10.4%)*	0.79	0.34
Window Defect *(6.7%)*	0.89	0.41

*After 28 epochs FFT

LSTM / GRU

*1 LSTM layer

Results

Rand_Pick

Class	AUC	AP
Leakage	0.87	0.76
Staining	0.74	0.46
None	0.96	0.81
Pooling	0.71	0.21
Window	0.82	0.28

Non_Rand_Pick

Class	AUC	AP
Leakage	0.90	0.81
Staining	0.80	0.56
None	0.97	0.88
Pooling	0.79	0.34
Window	0.89	0.41

Class	AUC	AP
Leakage		
Staining		
None		
Pooling		
Window		

Still no logic.

Still no logic.

```
    best_path = bin_frames[len(bin_frames) // 2]
    best_score = -1.0
    for idx in range(start, end):
        cur_path = frame_files[idx]
        prev_idx = max(0, idx - 1)
        score = float(
            numpy.abs(_thumb(cur_path) - _thumb(frame_files[prev_idx])).mean()
        )
        if score > best_score:
            best_score = score
            best_path = cur_path
    return best_path
```

**After 18 epochs FFT

**1 LSTM layer

*After 17 epochs FFT

*1 LSTM layer

LSTM / GRU

Results

n_bins = 8 (logic)*

Class	AUC	AP
Leakage *(39.3%)*	0.85	0.74
Staining *(18.9%)*	0.75	0.43
None *(24.8%)*	0.96	0.88
Pooling *(10.4%)*	0.74	0.21
Window Defect *(6.7%)*	0.80	0.30

n_bins = 10 (logic)**

Class	AUC	AP
Leakage *(39.3%)*	0.90	0.80
Staining *(18.9%)*	0.79	0.56
None *(24.8%)*	0.97	0.90
Pooling *(10.4%)*	0.80	0.40
Window Defect *(6.7%)*	0.85	0.49

**After 18 epochs FFT

**1 LSTM layer

LSTM / GRU

Results

n_bins = 10 (logic)**

Class	AUC	AP
Leakage *(39.3%)*	0.90	0.80
Staining *(18.9%)*	0.79	0.56
None *(24.8%)*	0.97	0.90
Pooling *(10.4%)*	0.80	0.40
Window Defect *(6.7%)*	0.85	0.49

LSTM / GRU

Results

Rand_Pick

Class	AUC	AP
Leakage	0.87	0.76
Staining	0.74	0.46
None	0.96	0.81
Pooling	0.71	0.21
Window	0.82	0.28

Non_Rand_Pick

Class	AUC	AP
Leakage	0.90	0.81
Staining	0.80	0.56
None	0.97	0.88
Pooling	0.79	0.34
Window	0.89	0.41

Logic_Pick

Class	AUC	AP
Leakage	0.90	0.80
Staining	0.79	0.56
None	0.97	0.90
Pooling	0.80	0.40
Window	0.85	0.49

LSTM / GRU

Results

Best

Middle

Worst

Rand_Pick

Class	AUC	AP
Leakage	0.87	0.76
Staining	0.74	0.46
None	0.96	0.81
Pooling	0.71	0.21
Window	0.82	0.28

Non_Rand_Pick

Class	AUC	AP
Leakage	0.90	0.81
Staining	0.80	0.56
None	0.97	0.88
Pooling	0.79	0.34
Window	0.89	0.41

Logic_Pick

Class	AUC	AP
Leakage	0.90	0.80
Staining	0.79	0.56
None	0.97	0.90
Pooling	0.80	0.40
Window	0.85	0.49

LSTM / GRU

Results

Logic_Pick

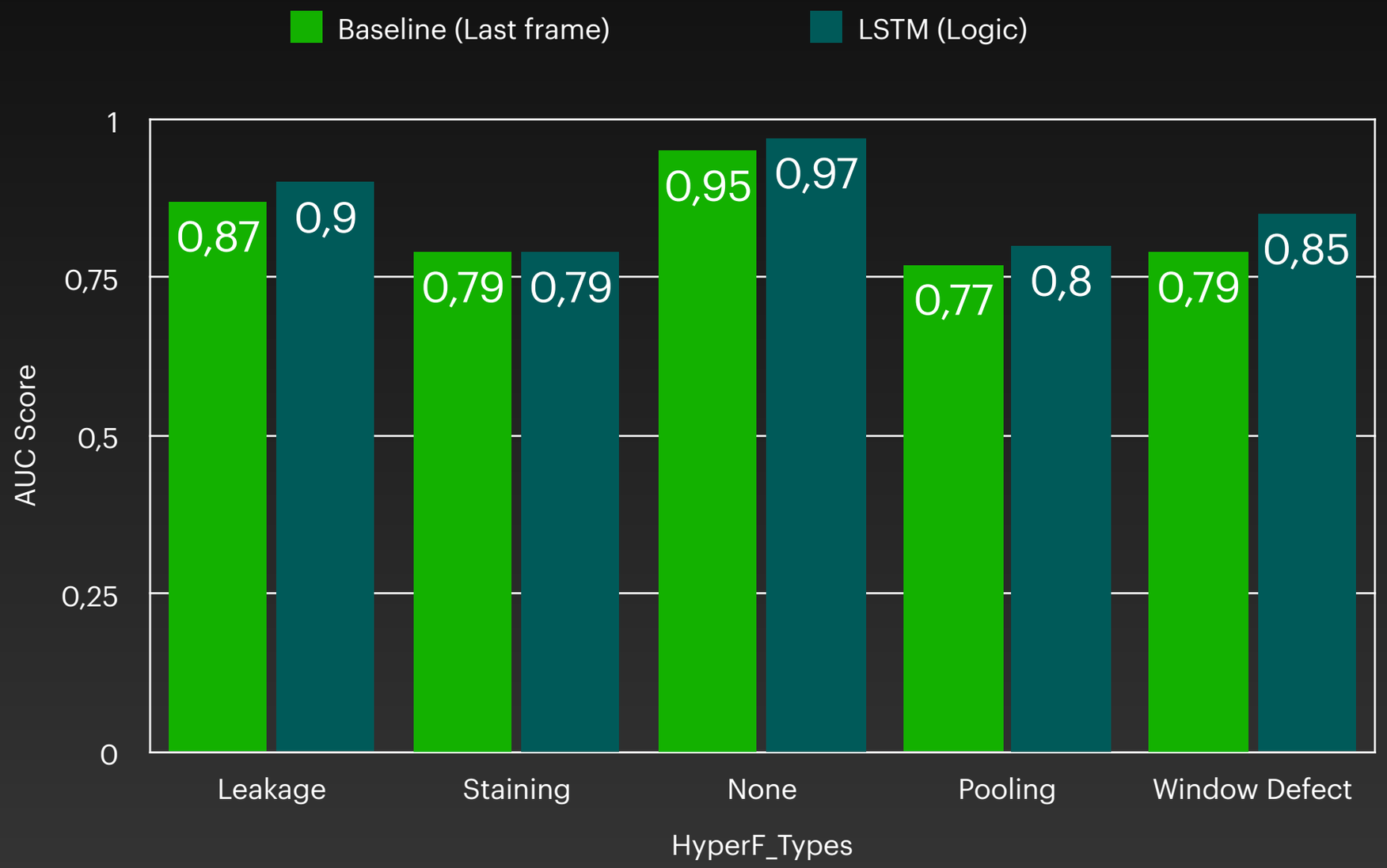
Class	AUC	AP
Leakage *(39.3%)*	0.89 ± 0.012	0.77 ± 0.022
Staining *(18.9%)*	0.79 ± 0.030	0.51 ± 0.036
None *(24.8%)*	0.97 ± 0.006	0.90 ± 0.041
Pooling *(10.4%)*	0.78 ± 0.017	0.34 ± 0.055
Window Defect *(6.7%)*	0.84 ± 0.019	0.42± 0.061

5 Runs with different seeds

Baseline Comparison

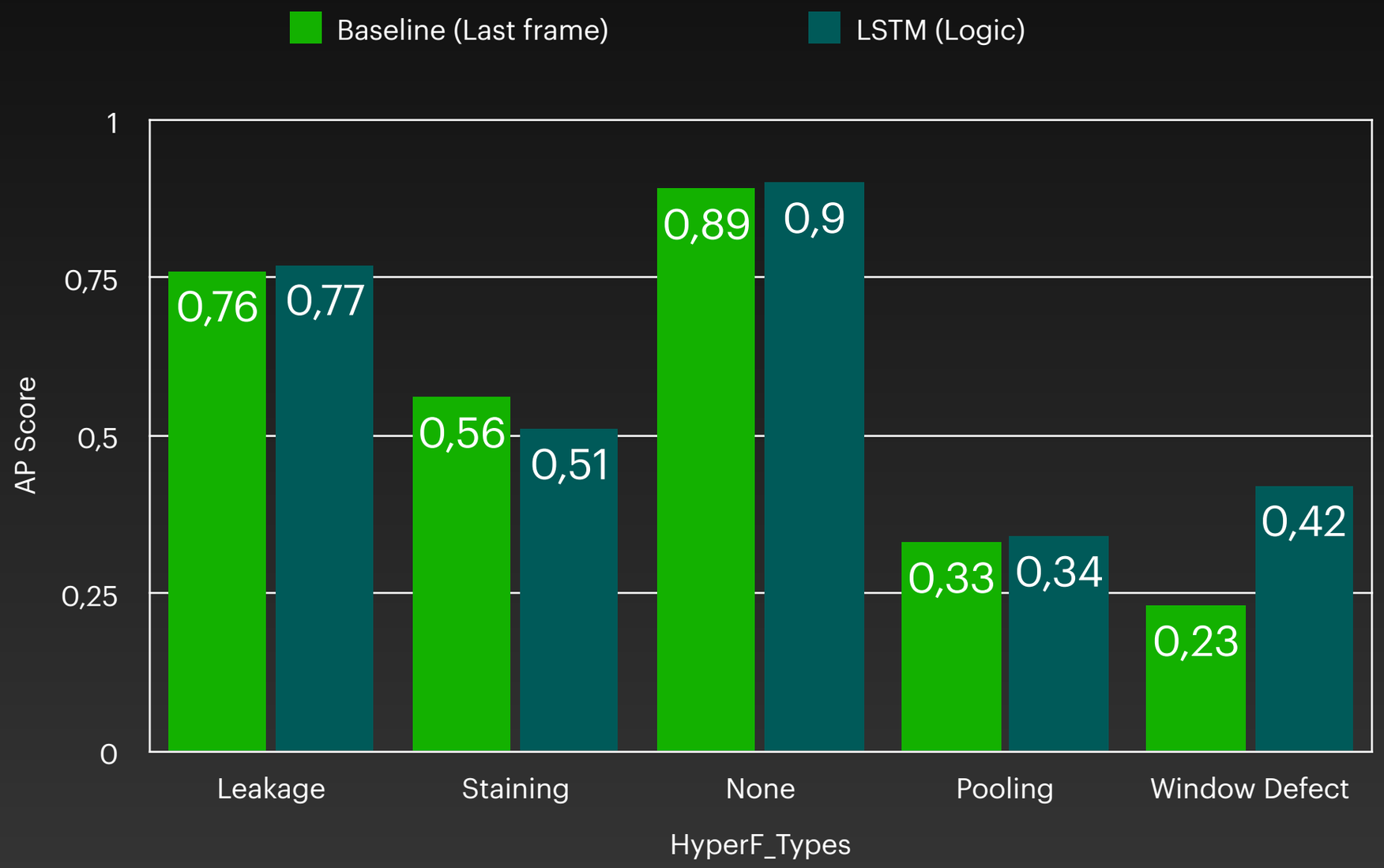
Baseline Comparison

AUC



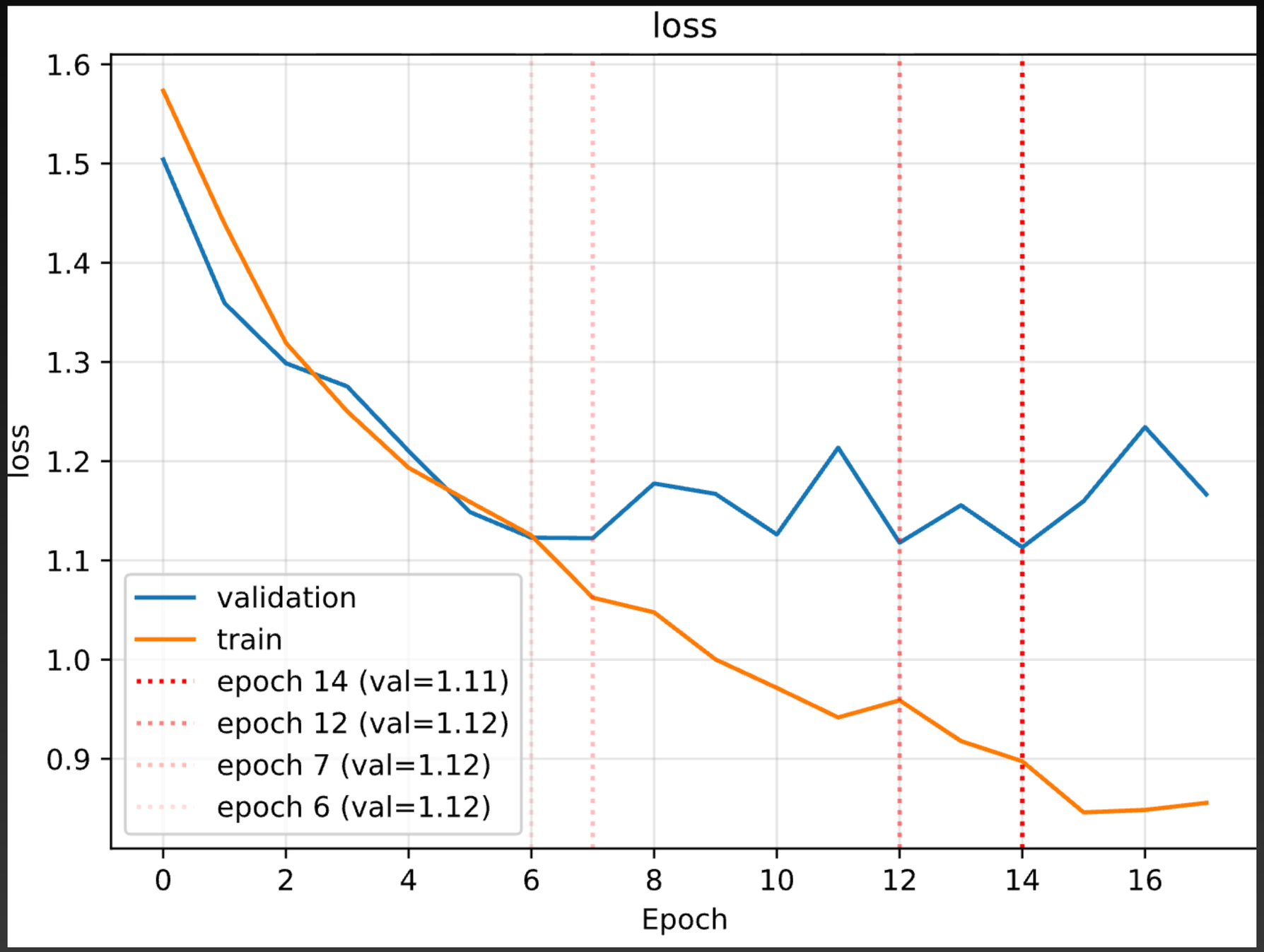
Baseline Comparison

AP



Future steps

Training loss (LSTM logic)



Future steps

Training loss (LSTM logic)

(Partially) Freeze backbone

1. Training only LSTM
2. Unfreeze last 1-2 ViT blocks (low lr)
3. Unfreeze more blocks if necessary