Online Learning for Indoor Asset Detection

Adith Balamurugan & Avideh Zakhor

University of California, Berkeley

Video and Image Processing Lab

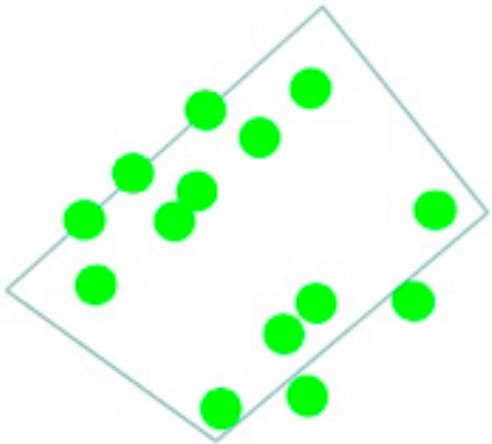
Motivation

- **Problem:**
    - Goal: record asset type and location within building
    - Uses: Climate control, safety, security, maintenance, etc…
    - Example assets: Router, fire sprinkler, fire alarm, fire alarm handle, EXIT sign, cardkey reader, light switch, emergency lights, fire extinguisher, outlet, etc.
- **Challenges:**
    - Existing methods manual, slow, & error prone
    - ML solutions have long far too long training time
    - Training on acquired imagery, not existing databases
- **Advantages:**
    - Semi-automated
    - Online learning learning on the fly with limited data
    - Exploit asset similarities within buildings
        - Instance rather than category recognition

(a) (b)

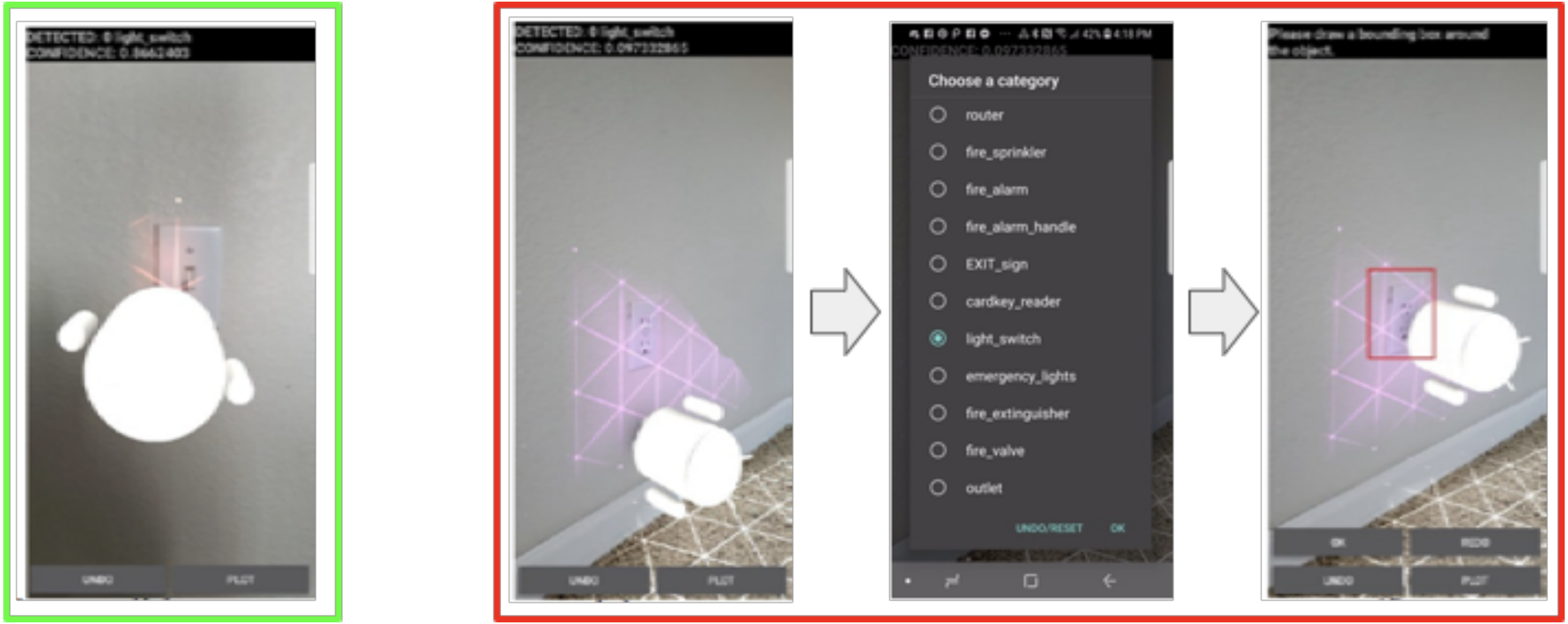
Smartphone App

- Human in the loop
- Detect/Identify assets
- Localize assets



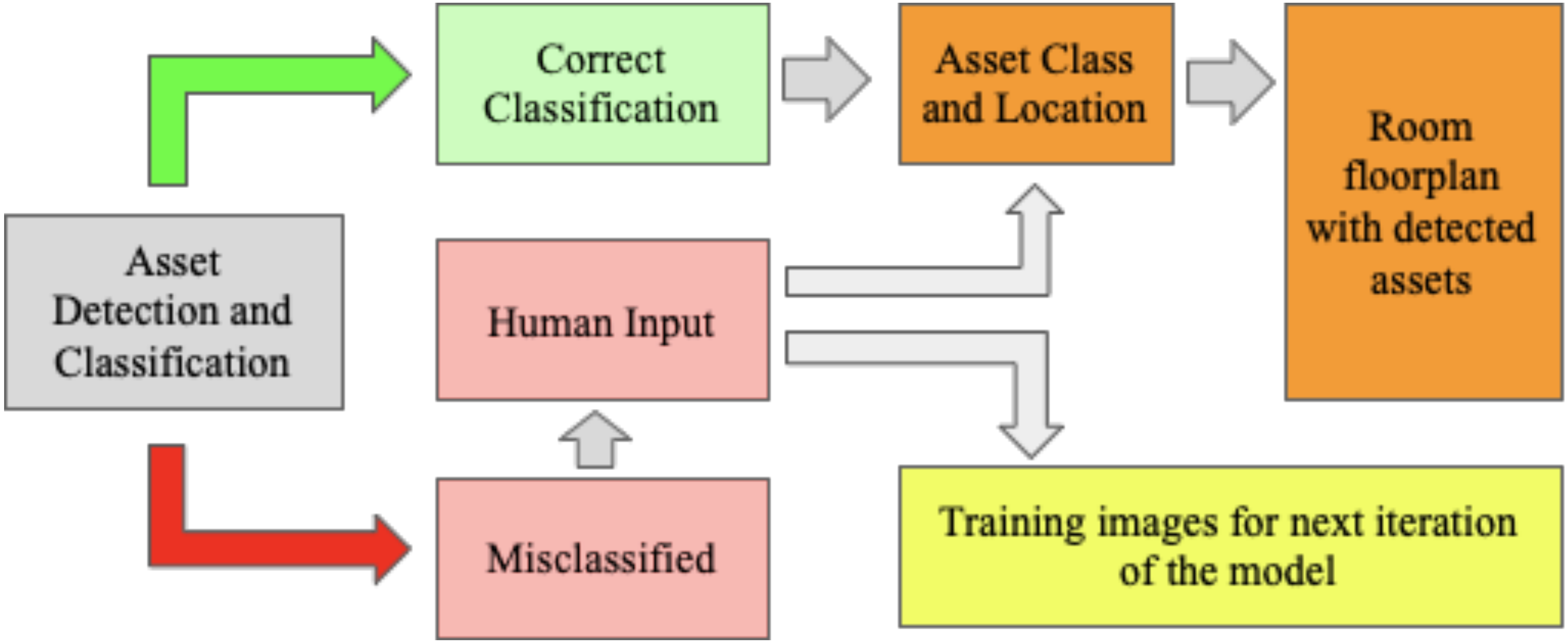
[1] Kostoeva et. al. "Indoor 3D Interactive Asset Detection Using a Smartphone", SPIE Electronic Imaging 2018

Operation of the App

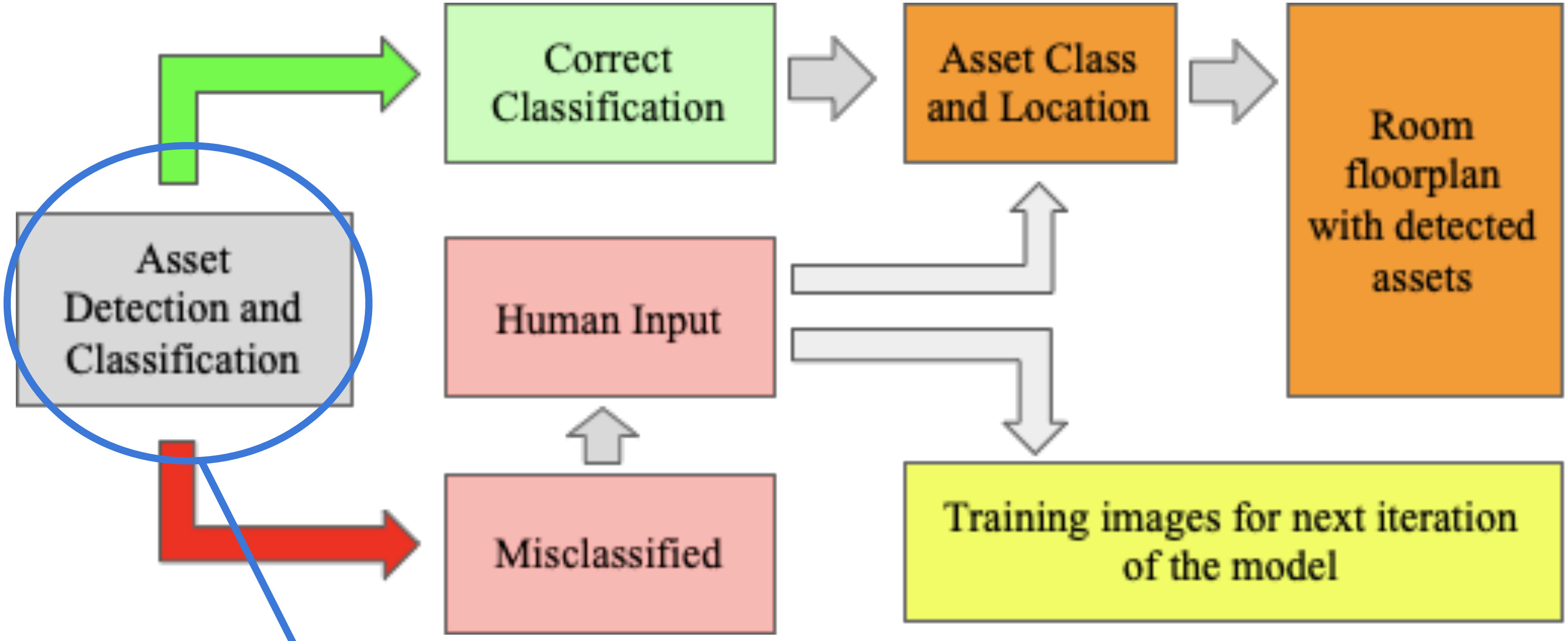


[1] Kostoeva et. al. “Indoor 3D Interactive Asset Detection Using a Smartphone”, SPIE Electronic Imaging 2018

Workflow Within a Session



Workflow Within a Session



How do we accomplish this?

Previous Approach: Transfer Learning

- Single Shot Detector (SSD) model [1]:
    - Mobile net V1 pre-trained on the MSCOCO dataset [2]
    - Data Augmentations
- Convert trained model to TFLite Model to reside on smartphone

[1] Kostoeva et. al. “Indoor 3D Interactive Asset Detection Using a Smartphone”, SPIE Electronic Imaging 2018
[2] Lin et. al. “Miscrosoft COCO: Common Objects in Context“ , 2015

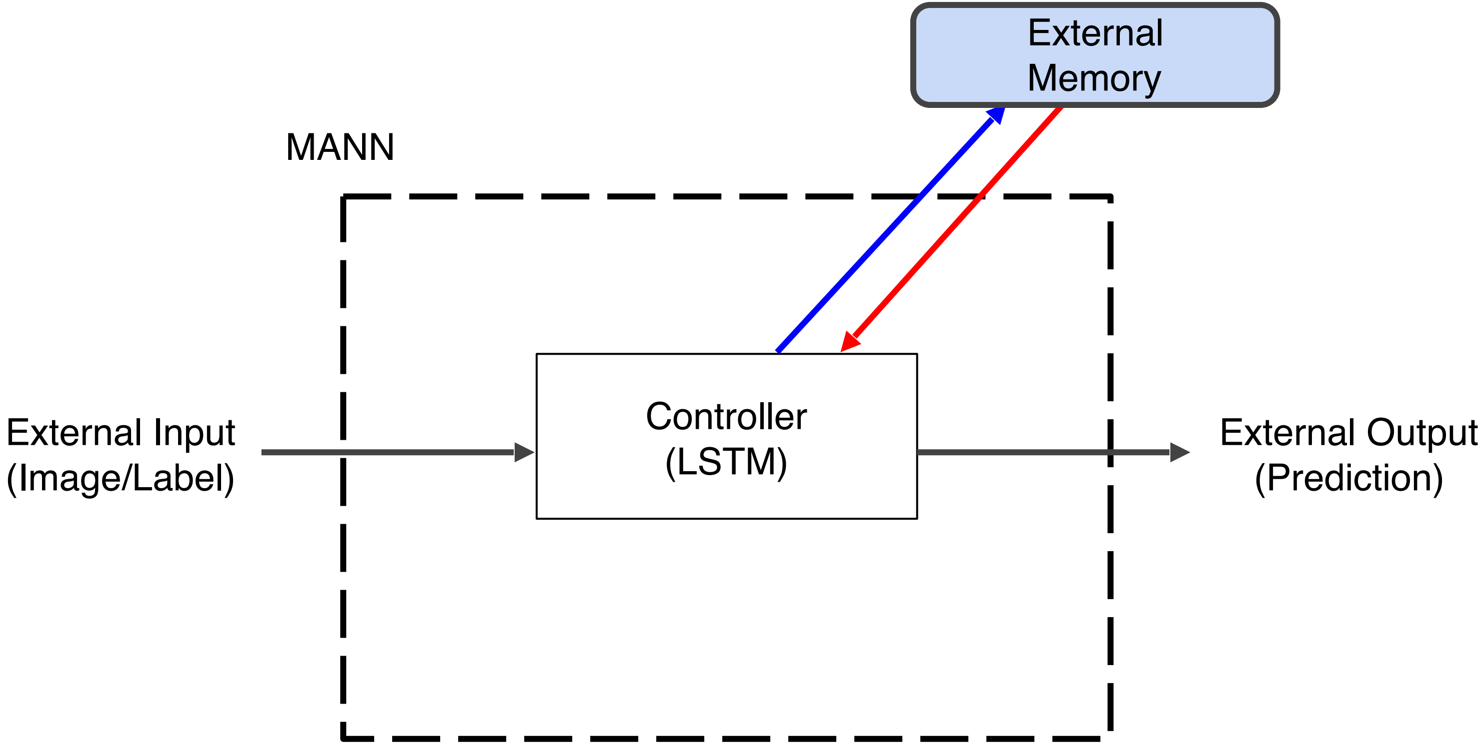
Shortcomings of previous approach:

- No real-time adaptation to seen examples
  - Model frozen during each session; updated only in between sessions
  - Lower accuracy for new buildings
- Retrain on all previous examples
  - Training times of more than 18 hours!

[1] Kostoeva et. al. "Indoor 3D Interactive Asset Detection Using a Smartphone", SPIE Electronic Imaging 2018

Proposed Approach: Online Learning

MANN (Memory Augmented NN)



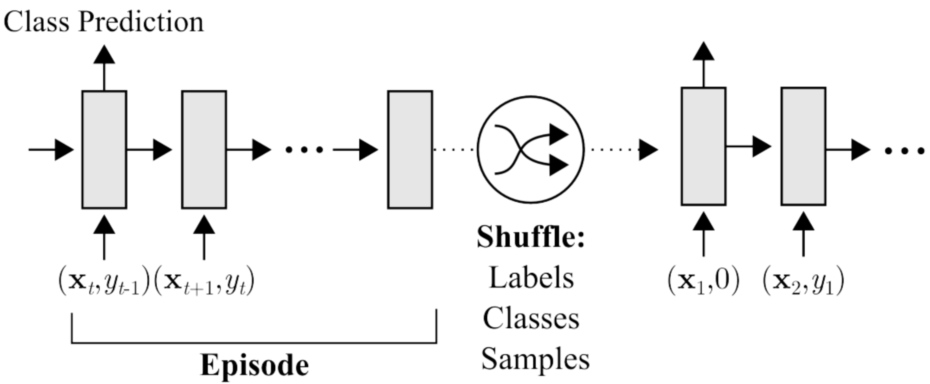
[3] Graves et. al. "Neural Turing Machines", 2014
[4] Santoro et. al. "One-Shot Learning with Memory-Augmented Neural Networks", 2016

Why Use a MANN?

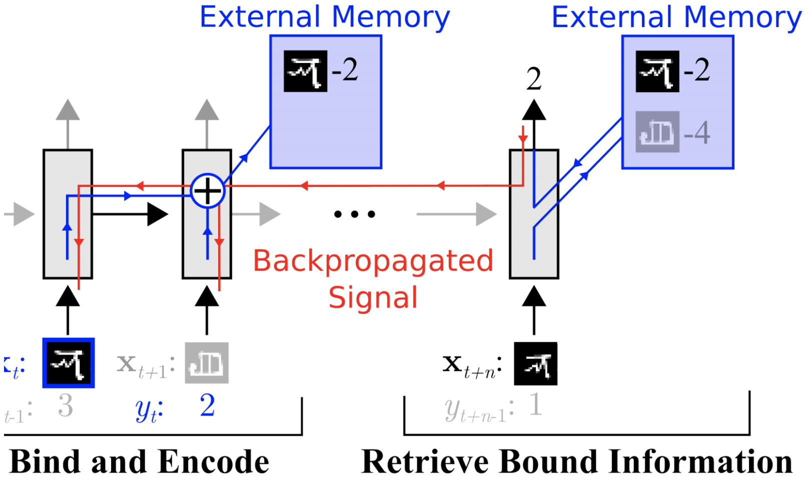
- Meta-learning in tasks that carry significant short- and long-term memory demands.
- Allows successful classification of never-before-seen instances
- **Long Term Memory - Update Weights of the Model:**
  - Used to maintain high level class differentiation
- **Short Term Memory - Utilization of External Memory:**
  - Adapt quickly to the new instances

[4] Santoro et. al. "One-Shot Learning with Memory-Augmented Neural Networks", 2016

Data Preparation

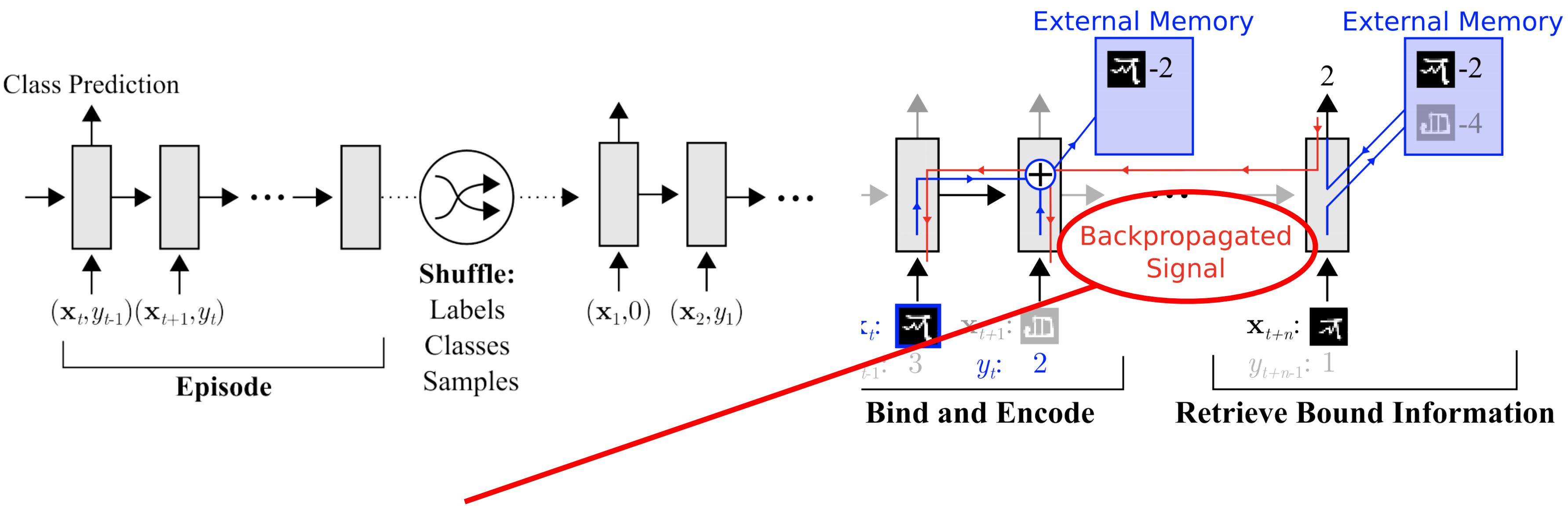


Training



Data Preparation

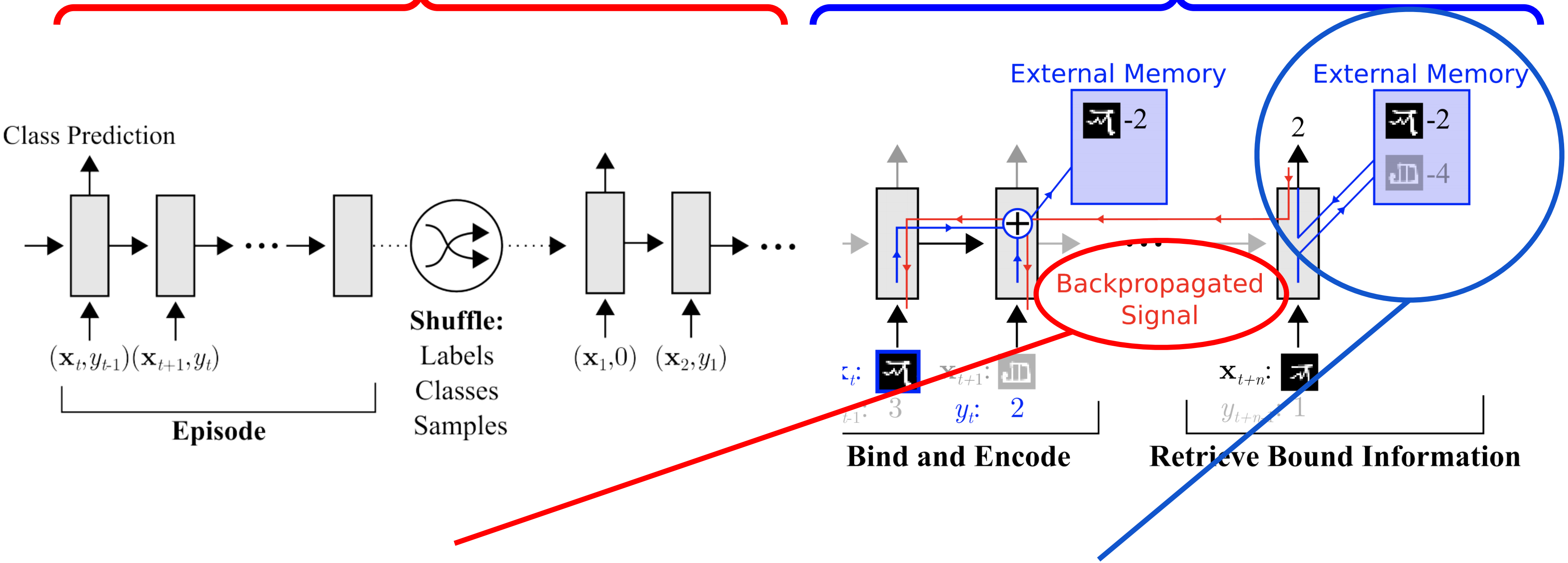
Training



Long Term Memory Updates

Data Preparation

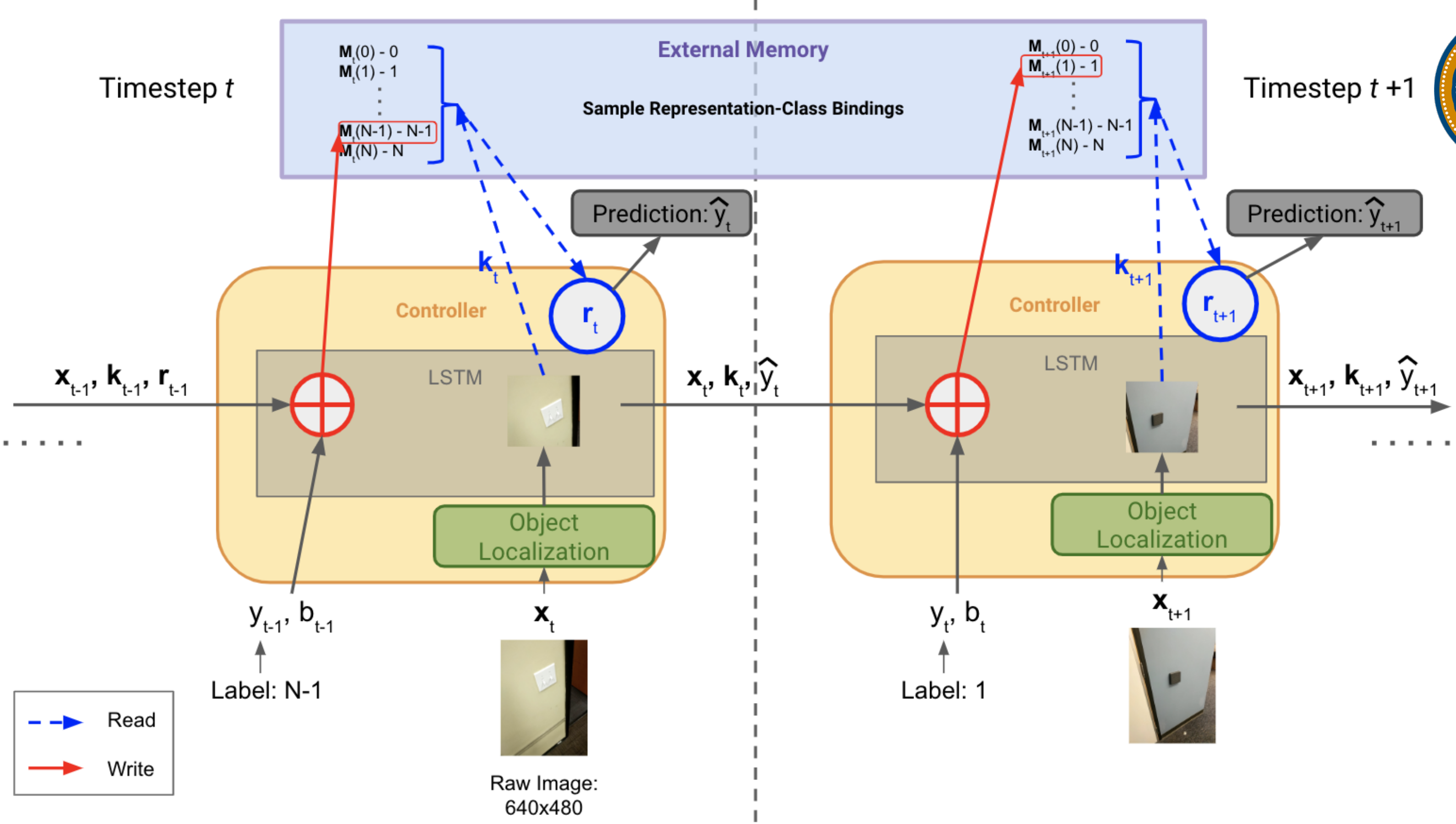
Training



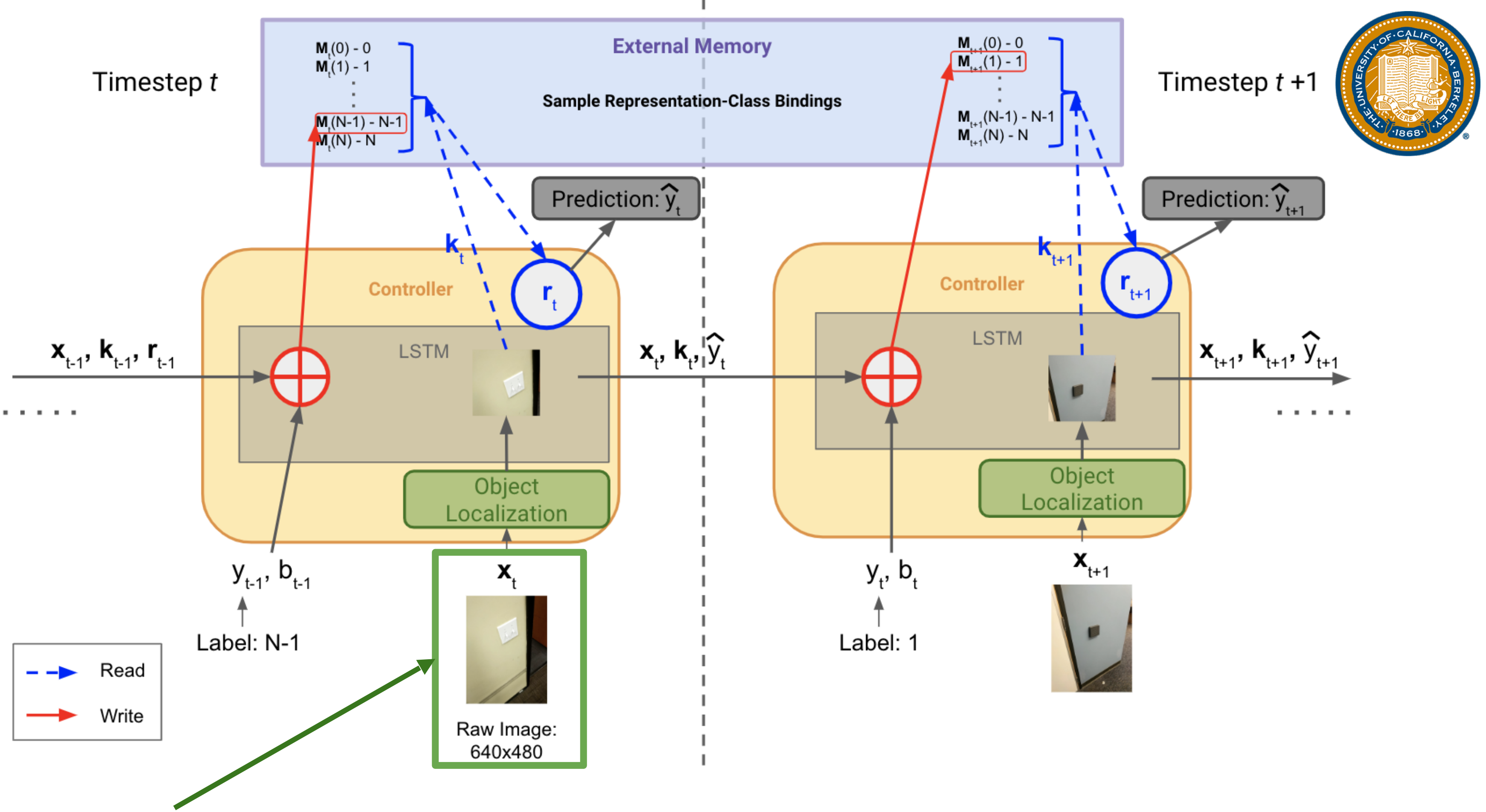
Long Term Memory Updates

Short Term Memory Updates

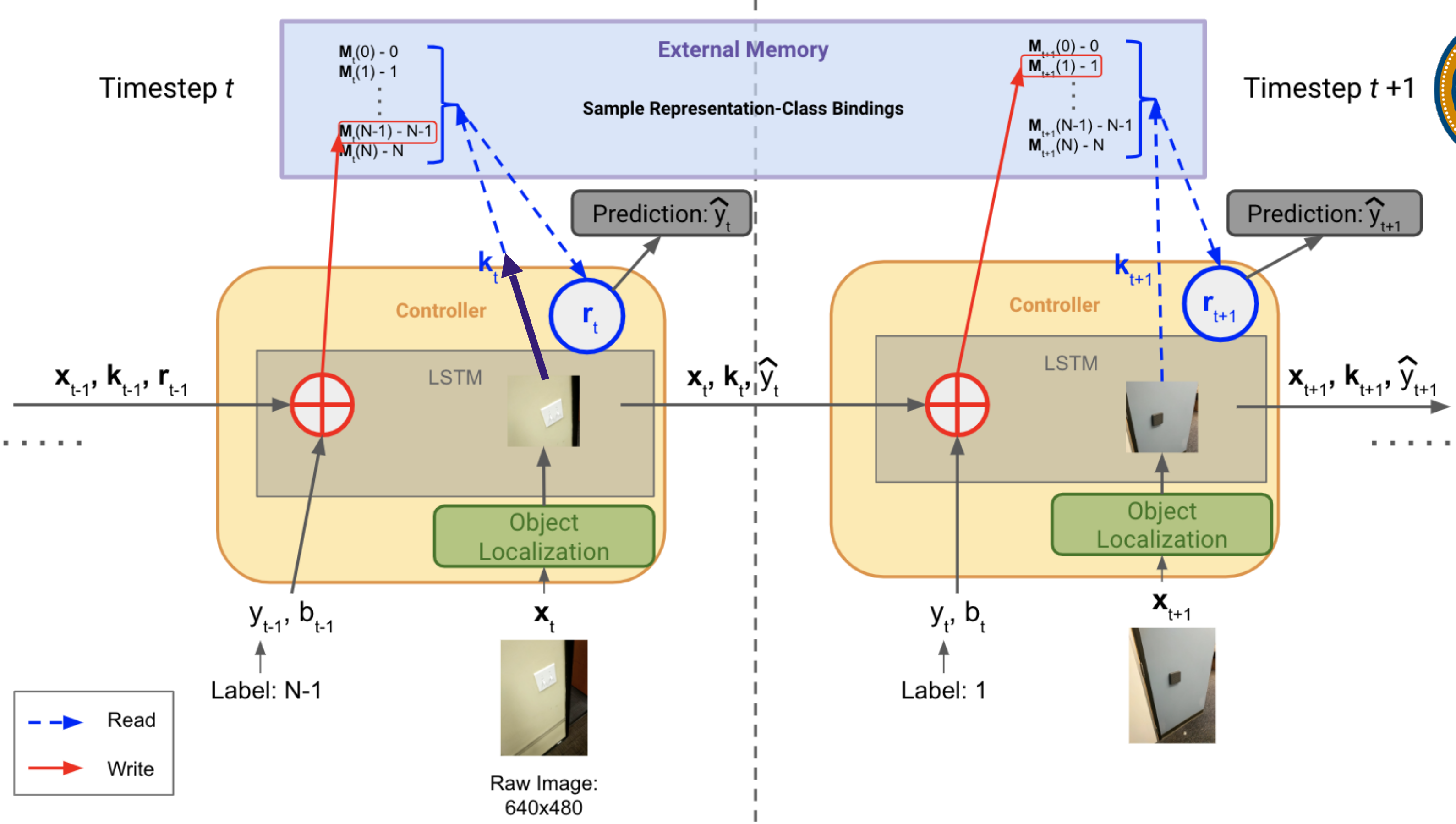
Define the moving parts of the system



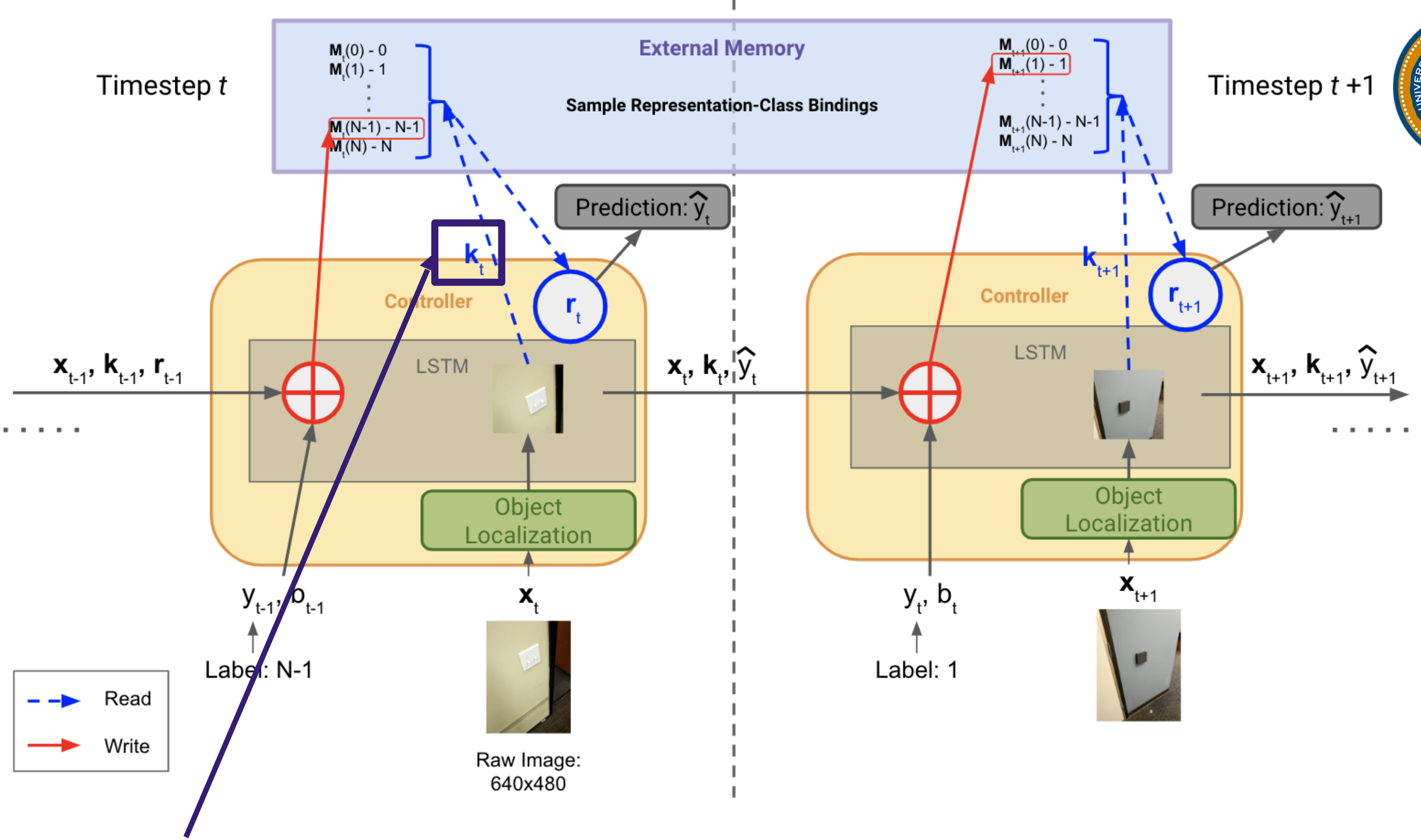
Timestep t
External Memory
Sample Representation-Class Bindings
Timestep t +1
Prediction: ŷt
Prediction: ŷt+1
Controller
LSTM
Object Localization
Controller
LSTM
Object Localization
Label: N-1
Label: 1
Raw Image: 640x480
Read
Write



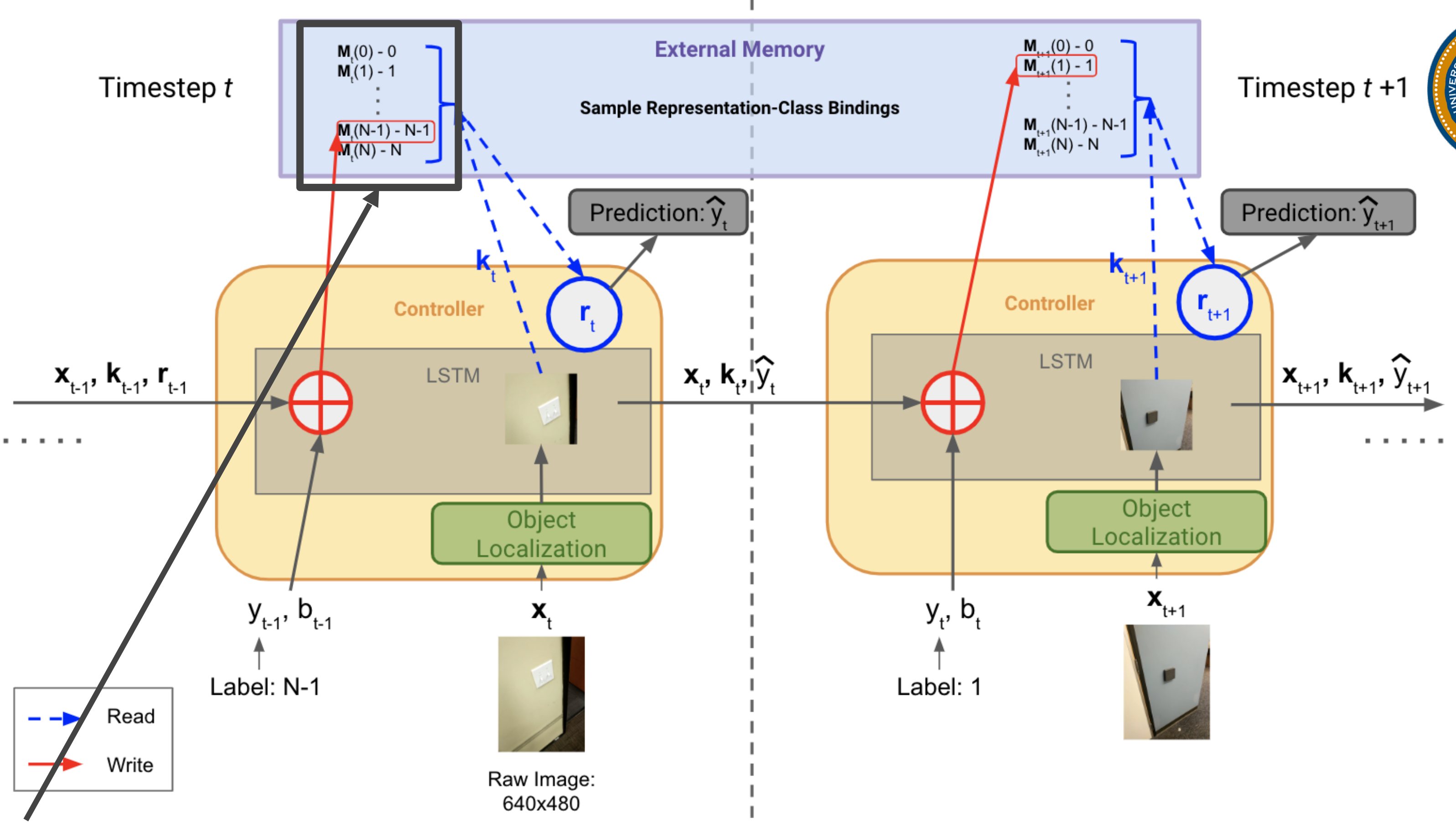
- $\mathbf{x}_t$ - vectorized raw image provided to model at time *t*



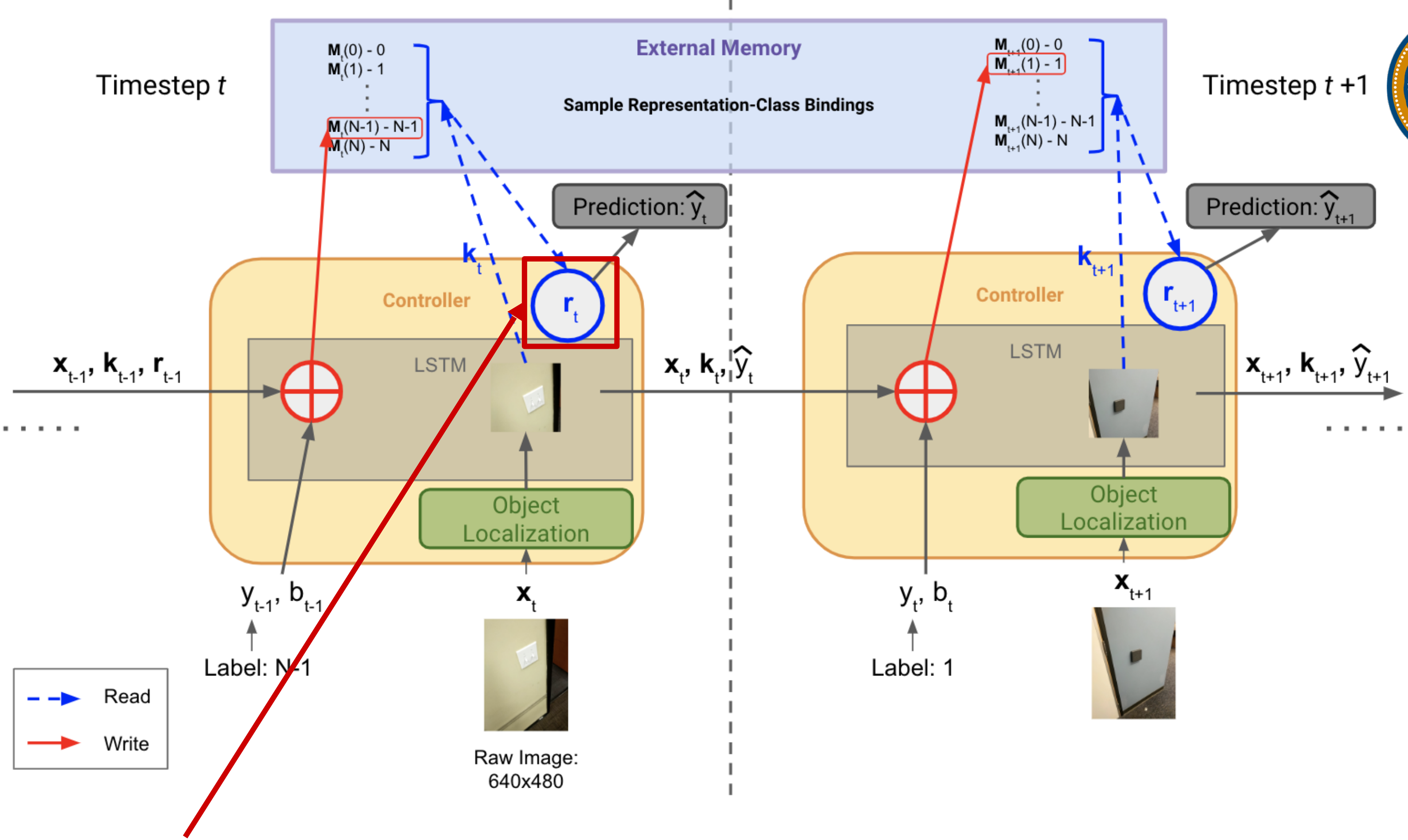
Timestep t
Timestep t +1
External Memory
Sample Representation-Class Bindings
M_t(0) - 0
M_t(1) - 1
M_t(N-1) - N-1
M_t(N) - N
M_t+1(0) - 0
M_t+1(1) - 1
M_t+1(N-1) - N-1
M_t+1(N) - N
Prediction: ŷ_t
Prediction: ŷ_t+1
k_t
r_t
k_t+1
r_t+1
Controller
LSTM
Controller
LSTM
x_t-1, k_t-1, r_t-1
x_t, k_t, ŷ_t
x_t+1, k_t+1, ŷ_t+1
Object Localization
Object Localization
y_t-1, b_t-1
Label: N-1
x_t
Raw Image: 640x480
y_t, b_t
Label: 1
x_t+1
Read
Write



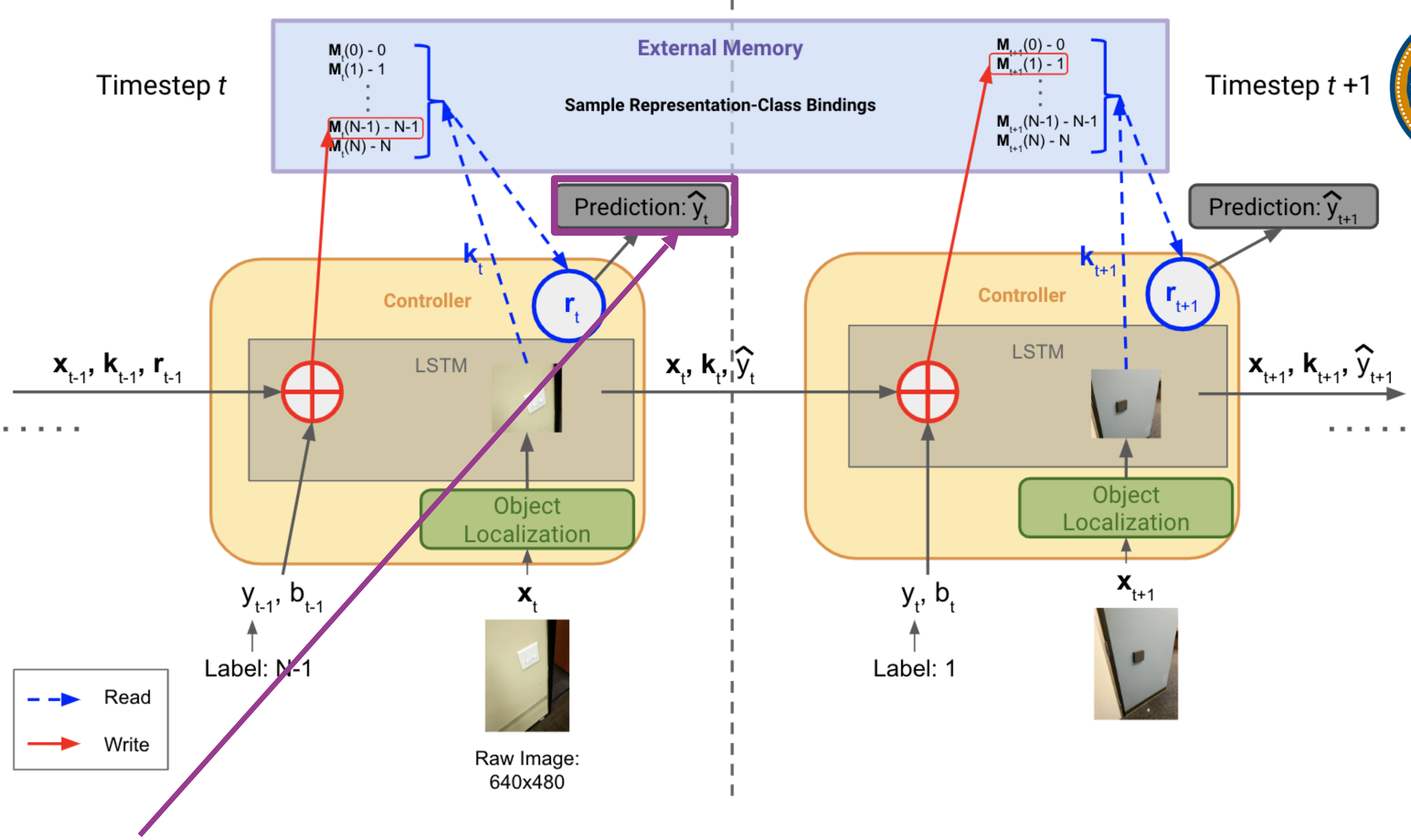
- $\mathbf{k}_t$ - *M*-dimensional vector “key” representation outputted by LSTM



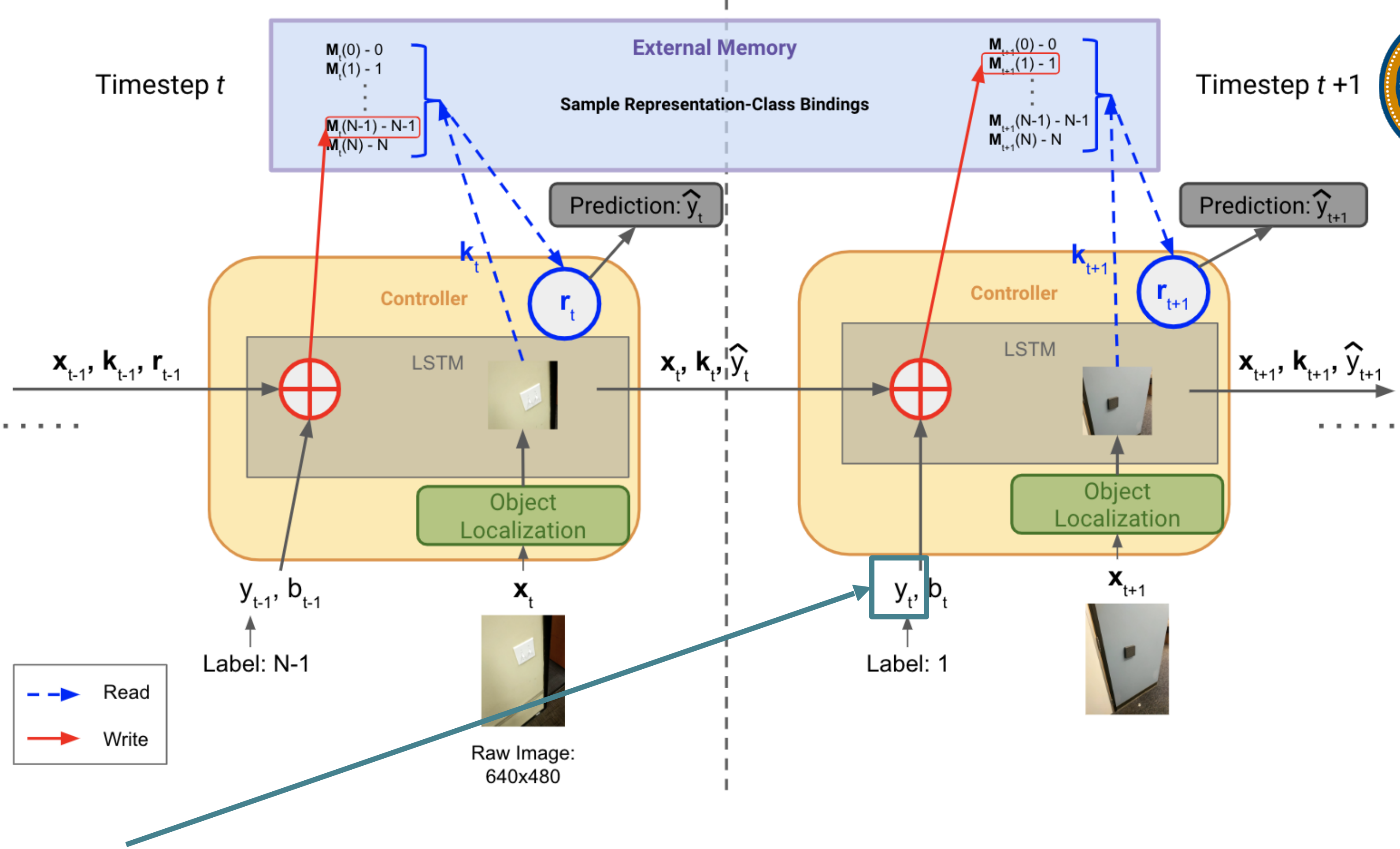
- $\mathbf{M_t}$ - the $N \times M$ memory matrix at time t
  - N - the number of classes in our system, 10, for the number of assets
  - M - dimension of the condensed representation, generated by the LSTM, for an input image



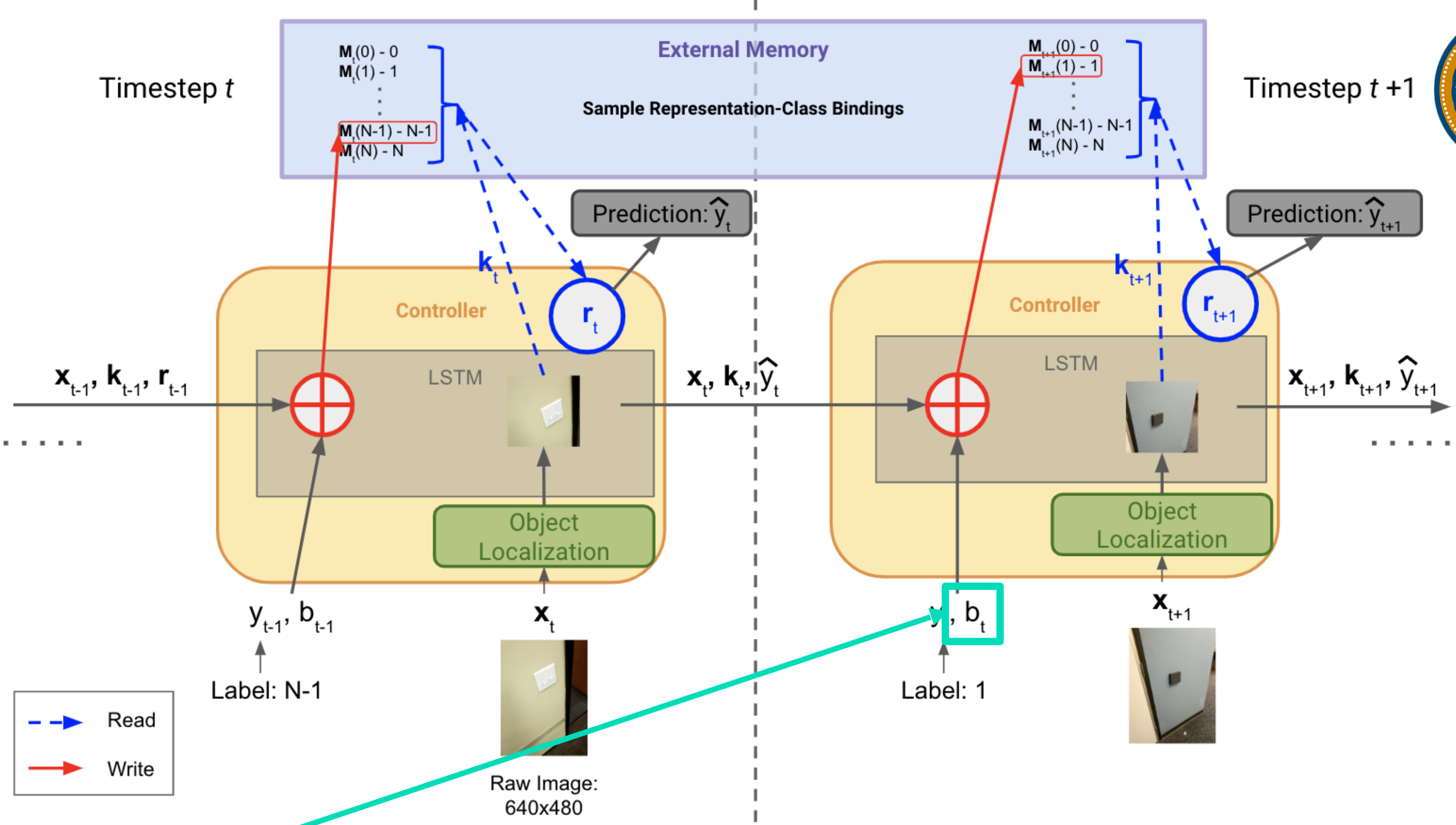
- $\mathbf{r_t}$ - “memory” read from the memory matrix read at time *t*



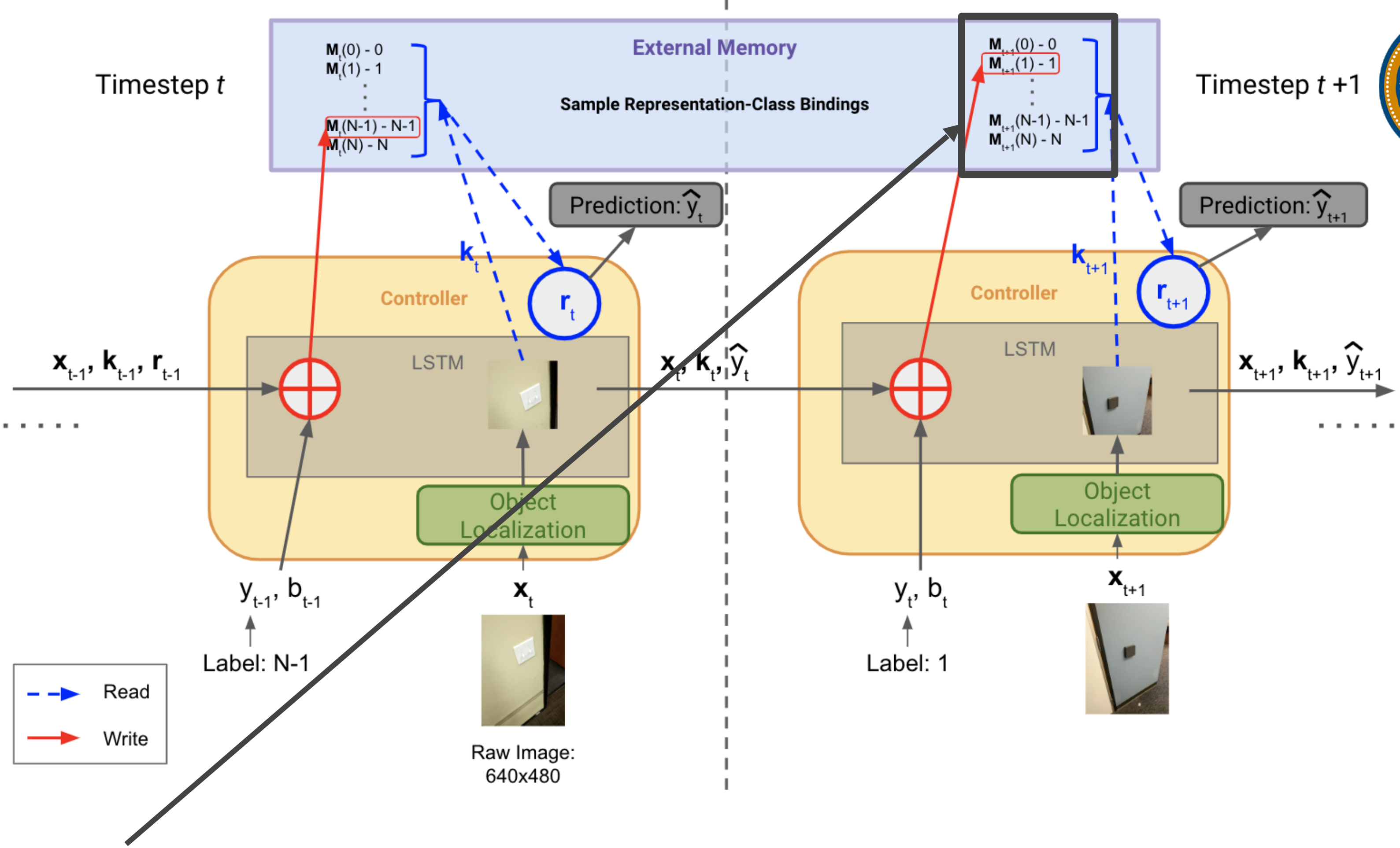
- $\hat{y}_t$ - predicted label for image at time *t* and corresponds to input image $\mathbf{x}_t$



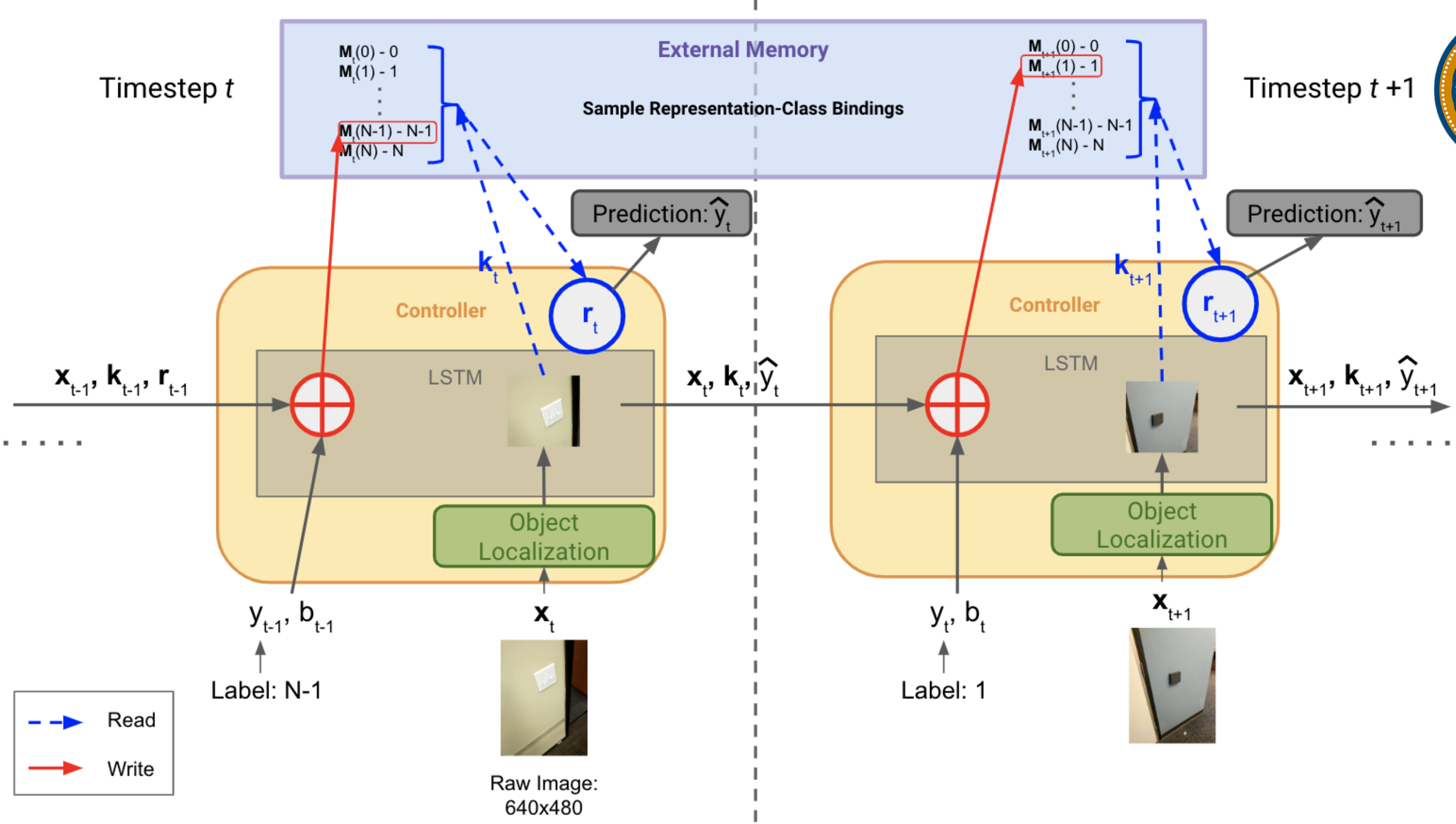
- y_t - ground truth label passed in to model at time t+1 and corresponds to input image $\mathbf{x}_t$



- b_t – bounding box provided at time *t+1* and corresponds to input image $\mathbf{x}_t$



- $\mathbf{M_{t+1}}$ - the $N \times M$ memory matrix at time $t + 1$



Timestep t
Timestep t +1
External Memory
Sample Representation-Class Bindings
M_t(0) - 0
M_t(1) - 1
M_t(N-1) - N-1
M_t(N) - N
M_t+1(0) - 0
M_t+1(1) - 1
M_t+1(N-1) - N-1
M_t+1(N) - N
Prediction: ŷ_t
Prediction: ŷ_t+1
k_t
k_t+1
r_t
r_t+1
Controller
LSTM
Object Localization
x_t-1, k_t-1, r_t-1
x_t, k_t, ŷ_t
x_t+1, k_t+1, ŷ_t+1
y_t-1, b_t-1
y_t, b_t
x_t
x_t+1
Label: N-1
Label: 1
Raw Image: 640x480
Read
Write

Model Update Flow

Model Update Flow

1. Load model weights onto phone for a new session

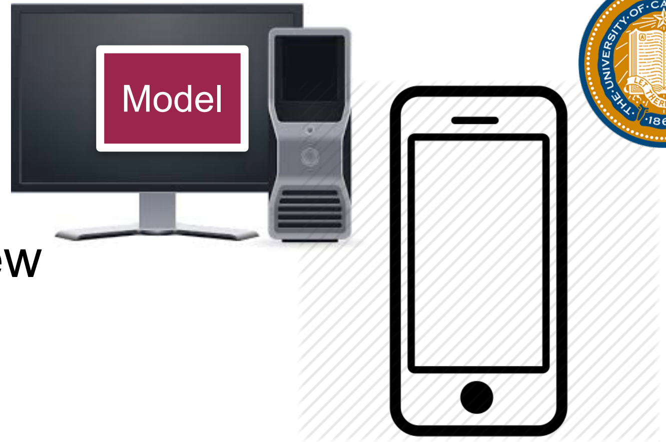
Model Update Flow

1. Load model weights onto phone for a new session
2. During data collection session:
   - Model makes predictions
   - Updates external memory: few ms

Model Update Flow

1. Load model weights onto phone for a new session
2. During data collection session:
   - Model makes predictions
   - Updates external memory: few ms
3. With each added asset, performance improves

Model Update Flow



1. Load model weights onto phone for a new session
2. During data collection session:
    - Model makes predictions
    - Updates external memory: few ms
3. With each added asset, performance improves
4. Use newly collected data in the session to train long-term memory weights

Model Update Flow

1. Load model weights onto phone for a new session
2. During data collection session:
    - Model makes predictions
    - Updates external memory: few ms
3. With each added asset, performance improves
4. Use newly collected data in the session to train long-term memory weights

Back to step 1

Choices in Offline Training Method

- Should we train on only the negative, or incorrectly classified, examples?
- Should we retain short-term memory across sessions?

	Clear External Memory	Retain External Memory
(-) Examples	Mode 1	Mode 2
(-/+) Examples	Mode 3	**Mode 4 w/ decaying memory retention**

Results compared to previous method

Model		Day 1 (CH)	Day 2 (CH)	Day 3 (SDH)	Day 3 (EH)	Day 4 (CH)
		Evaluation Set				
SSD [1]		67.2	81.7	74.3	54.7	73.6
Ours						
Mode (1)	Acc.(%)	58.6	60.9	63.5	62.9	69.78
	Std Dev.(%)	0.11	0.08	0.05	0.14	0.03
	Min Acc.(%)	54.1	55.7	60.6	56.2	60.1
	Max Acc.(%)	62.0	64.1	65.8	67.5	70.3
Mode (2)	Acc.(%)	66.7	63.9	67.6	64.33	76.2
	Std Dev.(%)	0.21	0.11	0.08	0.13	0.02
	Min Acc.(%)	59.9	61.3	63.4	60.1	74.0
	Max Acc.(%)	70.0	66.2	69.3	66.7	77.3
Mode (3)	Acc.(%)	73.1	74.3	**76.2**	69.3	77.1
	Std Dev.(%)	0.12	0.13	0.01	0.10	0.008
	Min Acc.(%)	68.8	70.2	75.1	68.8	76.0
	Max Acc.(%)	75.2	75.1	77.1	70.1	77.8
Mode (4)	Acc.(%)	**73.1**	**82.3**	74.7	**69.8**	**79.0**
	Std Dev.(%)	0.07	0.01	0.04	0.08	0.10
	Min Acc.(%)	71.8	77.8	70.9	67.1	77.2
	Max Acc.(%)	76.2	84.1	76.3	71.5	83.6

[1] Kostoeva et. al. "Indoor 3D Interactive Asset Detection Using a Smartphone", SPIE Electronic Imaging 2018

TFLite Model Results for Mode 4:

Tensorflow Model (150 MB) converted to TFLite Model (80 MB) to port onto phone

	Evaluation Set				
Model	Day 1 (CH)	Day 2 (CH)	Day 3 (SDH)	Day 3 (EH)	Day 4 (CH)
SSD [1]	67.2	81.7	74.3	54.7	73.6
Ours (TFLite)	71.8	81.8	74.0	68.8	77.9
Ours (Full TF)	**73.1**	**82.3**	**74.7**	**69.8**	**79.0**
Δ Ours (Full TF) - Ours (TFLite)	+1.3	+0.5	+0.7	+1.0	+1.1
Δ Ours (TFLite) - SSD	+4.6	+0.1	-0.3	+14.1	+4.3

[1] Kostoeva et. al. "Indoor 3D Interactive Asset Detection Using a Smartphone", SPIE Electronic Imaging 2018

TFLite Model Results for Mode 4:

Tensorflow Model (150 MB) converted to TFLite Model (80 MB) to port onto phone

	Evaluation Set				
Model	Day 1 (CH)	Day 2 (CH)	Day 3 (SDH)	Day 3 (EH)	Day 4 (CH)
SSD [1]	67.2	81.7	74.3	54.7	73.6
Ours (TFLite)	71.8	81.8	74.0	68.8	77.9
Ours (Full TF)	**73.1**	**82.3**	**74.7**	**69.8**	**79.0**
Δ Ours (Full TF) - Ours (TFLite)	+1.3	+0.5	+0.7	+1.0	+1.1
Δ Ours (TFLite) - SSD	+4.6	+0.1	-0.3	+14.1	+4.3

- Little loss from TF model to TFLite from conversion

[1] Kostoeva et. al. “Indoor 3D Interactive Asset Detection Using a Smartphone”, SPIE Electronic Imaging 2018

TFLite Model Results for Mode 4:

Tensorflow Model (150 MB) converted to TFLite Model (80 MB) to port onto phone

	Evaluation Set				
Model	Day 1 (CH)	Day 2 (CH)	Day 3 (SDH)	Day 3 (EH)	Day 4 (CH)
SSD [1]	67.2	81.7	74.3	54.7	73.6
Ours (TFLite)	71.8	81.8	74.0	68.8	77.9
Ours (Full TF)	**73.1**	**82.3**	**74.7**	**69.8**	**79.0**
Δ Ours (Full TF) - Ours (TFLite)	+1.3	+0.5	+0.7	+1.0	+1.1
Δ Ours (TFLite) - SSD	+4.6	+0.1	-0.3	+14.1	+4.3

- Little loss from TF model to TFLite from conversion
- TFLite model outperforms SSD by up to 14 points

[1] Kostoeva et. al. "Indoor 3D Interactive Asset Detection Using a Smartphone", SPIE Electronic Imaging 2018

Comparison of Offline Training Times

	Offline Training		Online Training		Online Inference	
Data Set	SSD (Kostoeva et al.)	Ours (Mode 4)	SSD [1]	Ours	SSD [1]	Ours
Day 0	18hr 29m	5hr 27m	–	–	–	–
Day 1	20hr 11m	4hr 17m	–	1.13 s	0.084 s	0.064 s
Day 2	20hr 56m	3hr 4m	–	1.06 s	0.082 s	0.041 s
Day 3	22hr 40m	4hr 13m	–	1.00 s	0.081 s	0.050 s
Day 4	–	4hr 9m	–	1.21 s	0.084 s	0.034 s

- Up to 7x speedup in offline training times
- Up to 2x speedup in online inference times

[1] Kostoeva et. al. "Indoor 3D Interactive Asset Detection Using a Smartphone", SPIE Electronic Imaging 2018

Conclusions & Future Work

- **Proposed Online Learning Method for Asset Detection:**
  - Better Accuracy
  - Improved Latencies
  - Real-time Learning
  - Shorter Offline Training time

- **Future Work:**
  - Extend to hundreds of asset categories → scalability
  - Adapt to previously unseen asset classes in real-time