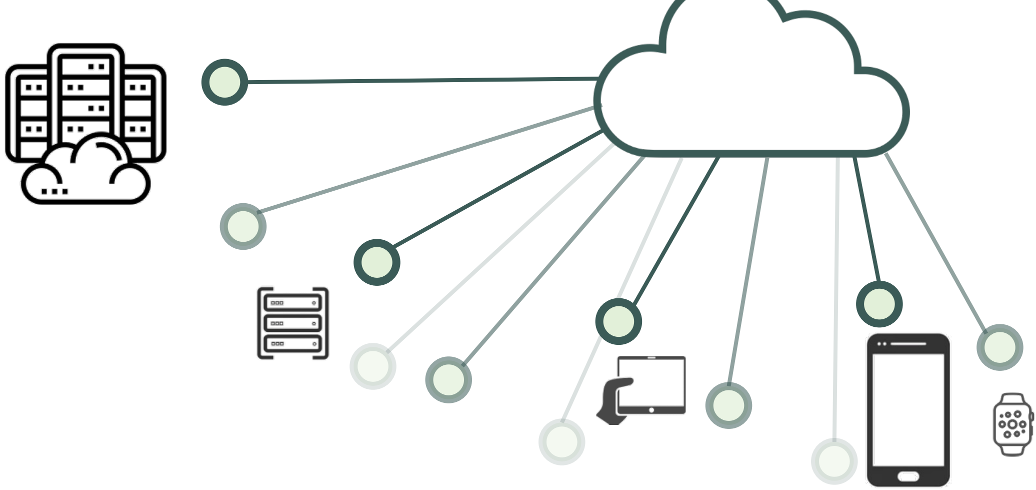
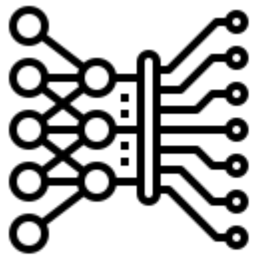


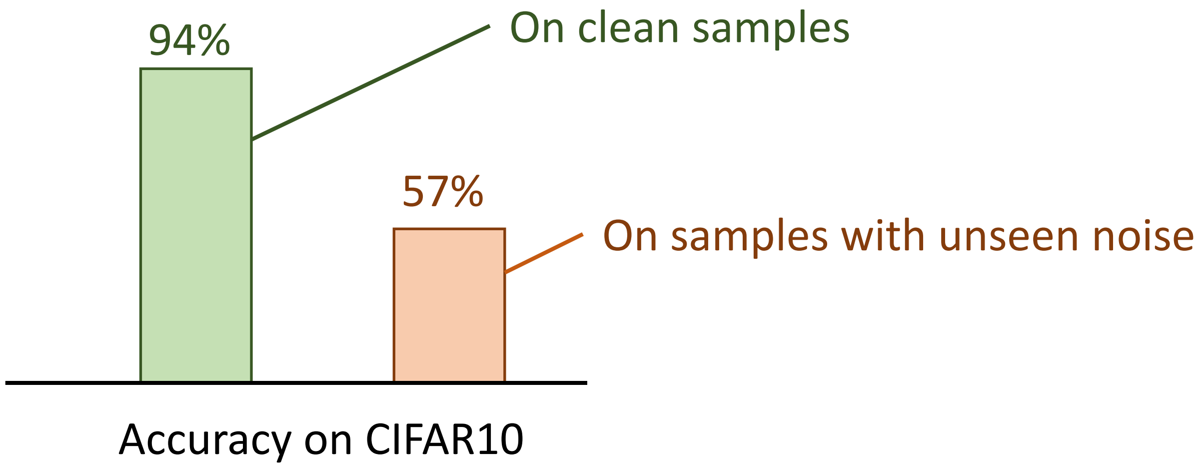
# MECTA: Memory-Economic Continual Test-Time Model Adaptation

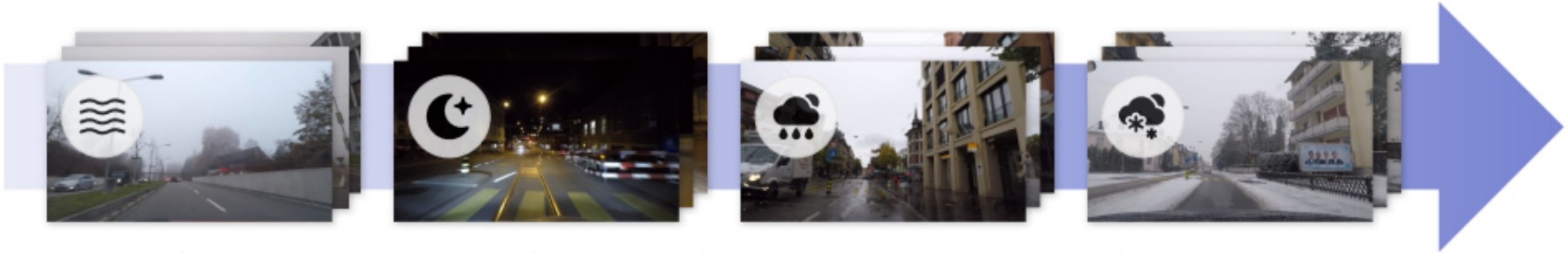
**Junyuan Hong**[1*], Lingjuan Lyu[2], Jiayu Zhou[1], Michael Spranger[2]
[1]Michigan State University, [2]Sony AI
*Work done during internship at Sony AI

# Continually Changing Environments

Of-the-shelf pre-trained model



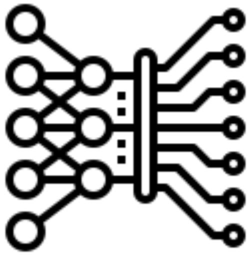


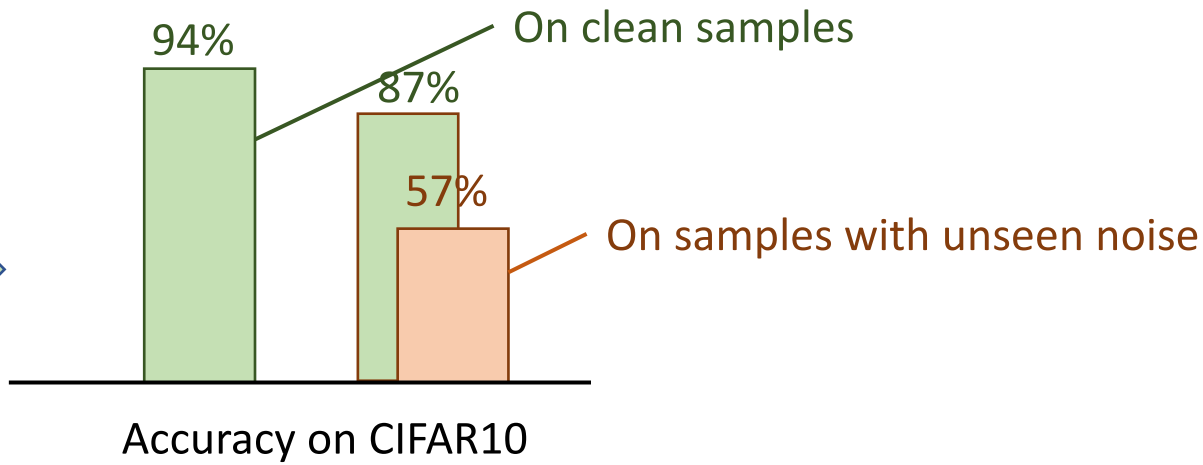
Wang, Q., Fink, O., Van Gool, L., & Dai, D. (2022). Continual test-time domain adaptation. CVPR.

# Continual Test-time Adaptation (CTA)

Of-the-shelf pre-trained model



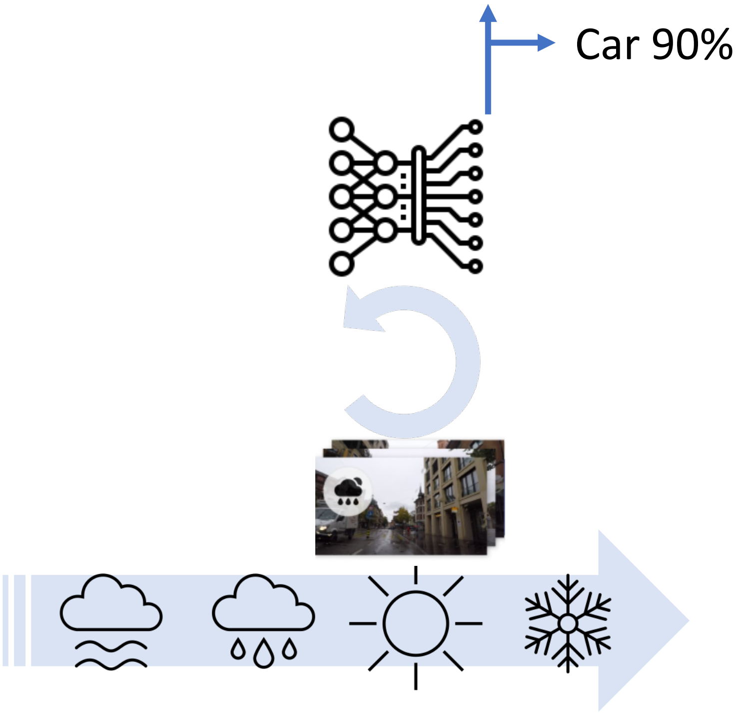
Fine-tune




A batch of unlabeled test samples

Wang, Q., Fink, O., Van Gool, L., & Dai, D. (2022). Continual test-time domain adaptation. CVPR.

# Continual Test-time Adaptation (CTA)



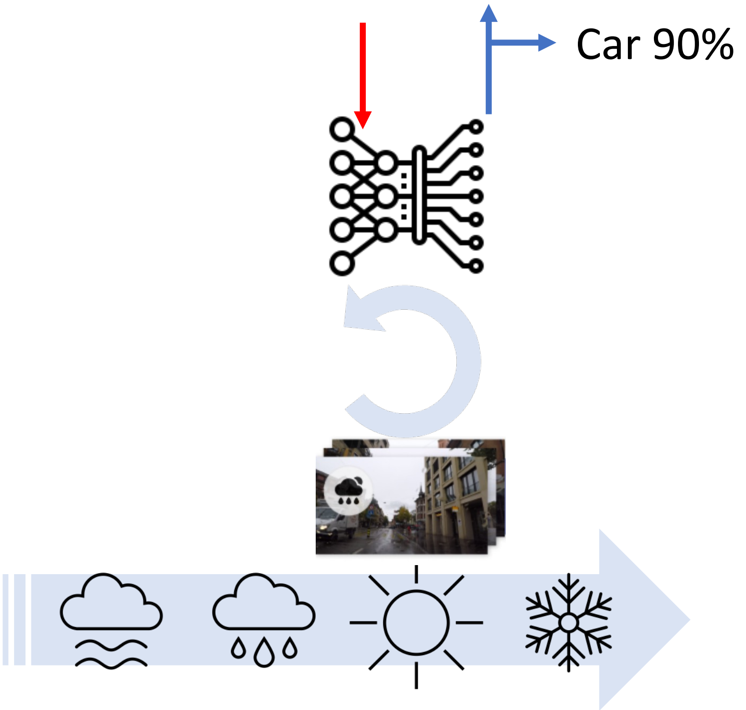

Continual adaptation via Tent/EATA
of ResNet50 w/ 64-sized batches

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., & Darrell, T. (2021). Tent: Fully test-time adaptation by entropy minimization. ICLR.

# Continual Test-time Adaptation (CTA)

Entropy Minimization (Tent)

$$\theta_t = \text{Optimize}_{\theta \in \Theta_t}(\mathbb{E}_{x \sim P_t(x)}[H(f_\theta(x))], \theta_{t-1}),$$

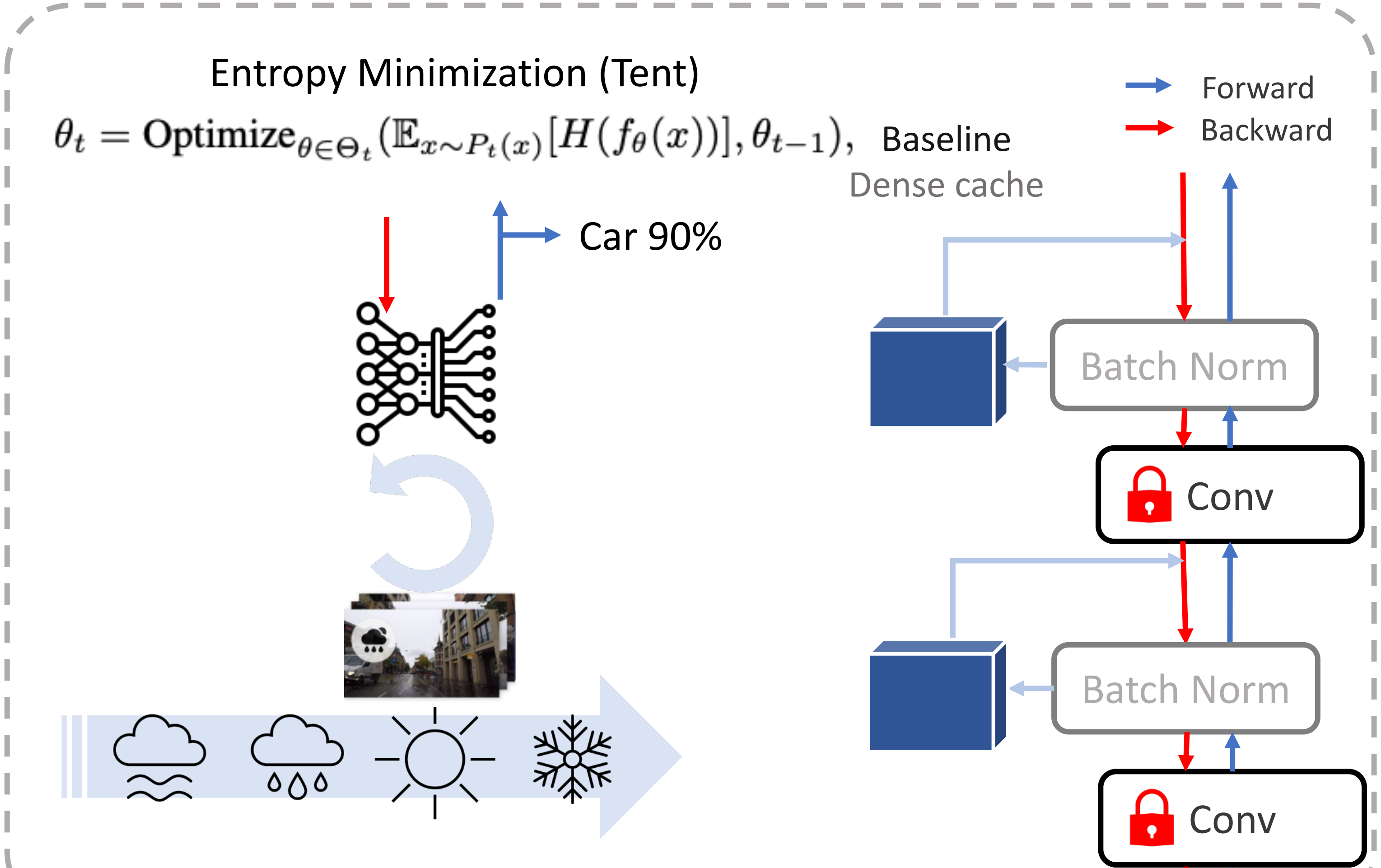



Continual adaptation via Tent/EATA of ResNet50 w/ 64-sized batches

CTA:
- Unsupervised finetuning.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., & Darrell, T. (2021). Tent: Fully test-time adaptation by entropy minimization. ICLR.

# Continual Test-time Adaptation (CTA)




Continual adaptation via Tent/EATA
of ResNet50 w/ 64-sized batches

CTA:
- Unsupervised finetuning.
- Parameter efficient: Update BN only.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., & Darrell, T. (2021). Tent: Fully test-time adaptation by entropy minimization. ICLR.

# Continual Test-time Adaptation (CTA)

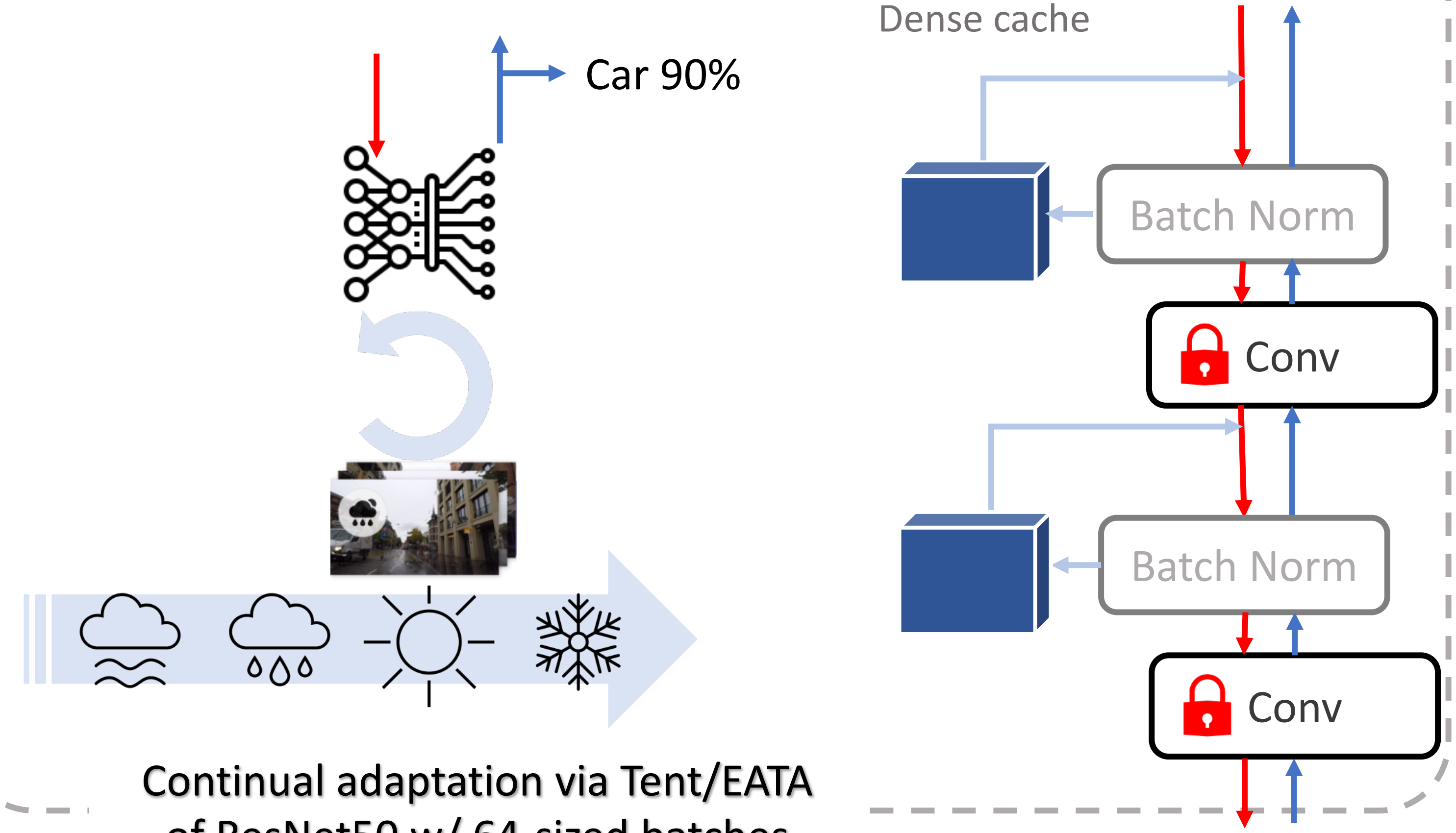
Entropy Minimization (Tent)

$$\theta_t = \text{Optimize}_{\theta\in\Theta_t}(\mathbb{E}_{x\sim P_t(x)}[H(f_\theta(x))], \theta_{t-1}),$$

Baseline

Forward

Backward




Continual adaptation via Tent/EATA of ResNet50 w/ 64-sized batches

Batch Norm (BN)

$$z^l_{n,i,j,k} = \frac{x^l_{n,i,j,k} - \boxed{\mu^l_i}}{\sqrt{\sigma_i^2 + \epsilon_0}} \qquad a^l_{n,i,j,k} = \gamma^l_i z^l_{n,i,j,k} + b^l_i$$

CTA:

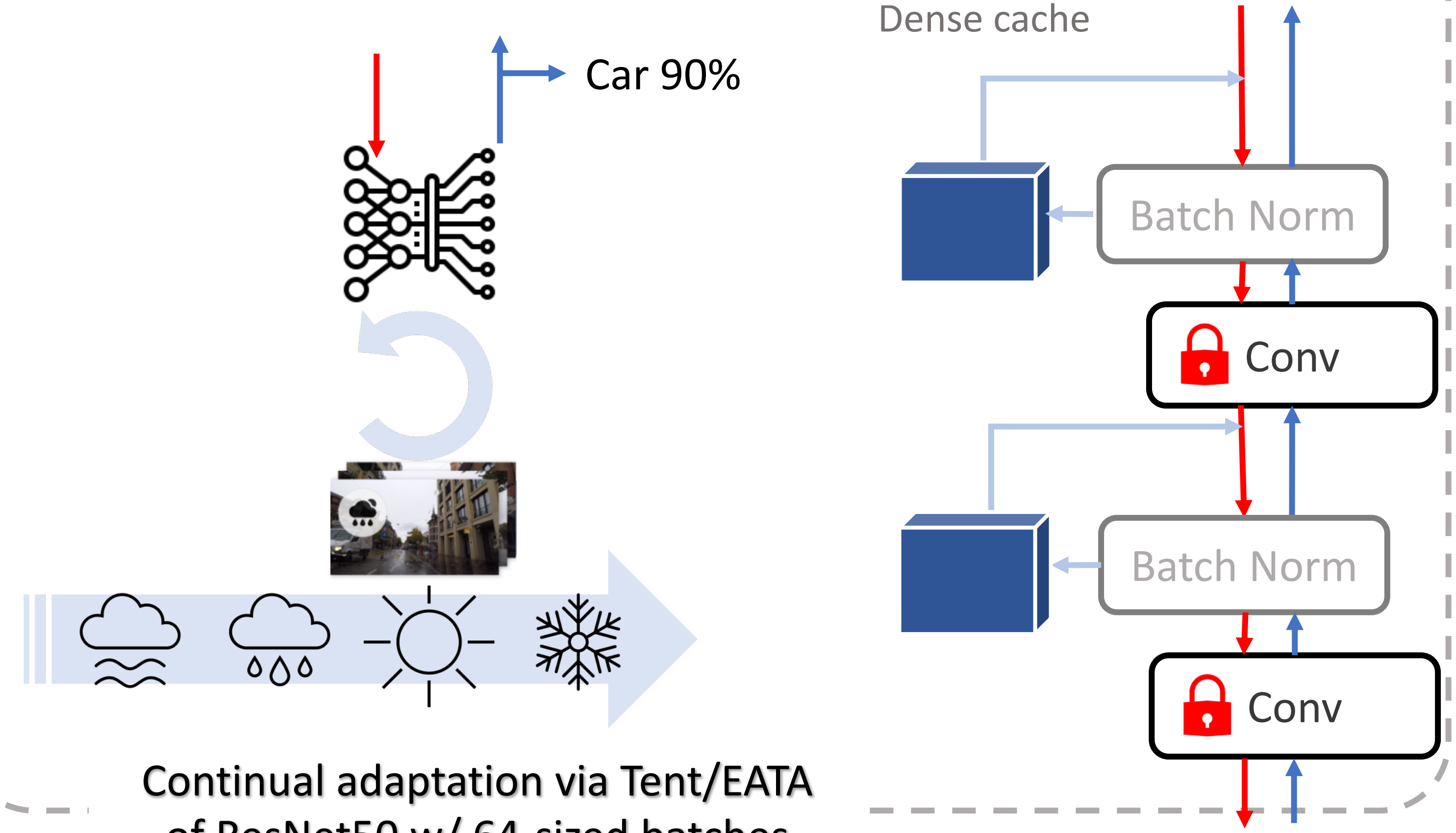
- Unsupervised finetuning.
- Parameter efficient: Update BN only.
- Batch-estimated BN statistics $(\mu, \sigma)$ for capturing new environment.

Wang, D., Shelhamer, E., Liu, S., Olshausen, B., & Darrell, T. (2021). Tent: Fully test-time adaptation by entropy minimization. ICLR.

# Continual Test-time Adaptation (CTA)

Entropy Minimization (Tent)

$$\theta_t = \textbf{Optimize}_{\theta\in\Theta_t}(\mathbb{E}_{x\sim P_t(x)}[H(f_\theta(x))], \theta_{t-1}),$$




Continual adaptation via Tent/EATA of ResNet50 w/ 64-sized batches

Batch Norm (BN)

$$z^l_{n,i,j,k} = \frac{x^l_{n,i,j,k} - \boxed{\mu^l_i}}{\sqrt{\sigma_i^2+\epsilon_0}} \qquad a^l_{n,i,j,k} = \gamma^l_i z^l_{n,i,j,k} + b^l_i$$

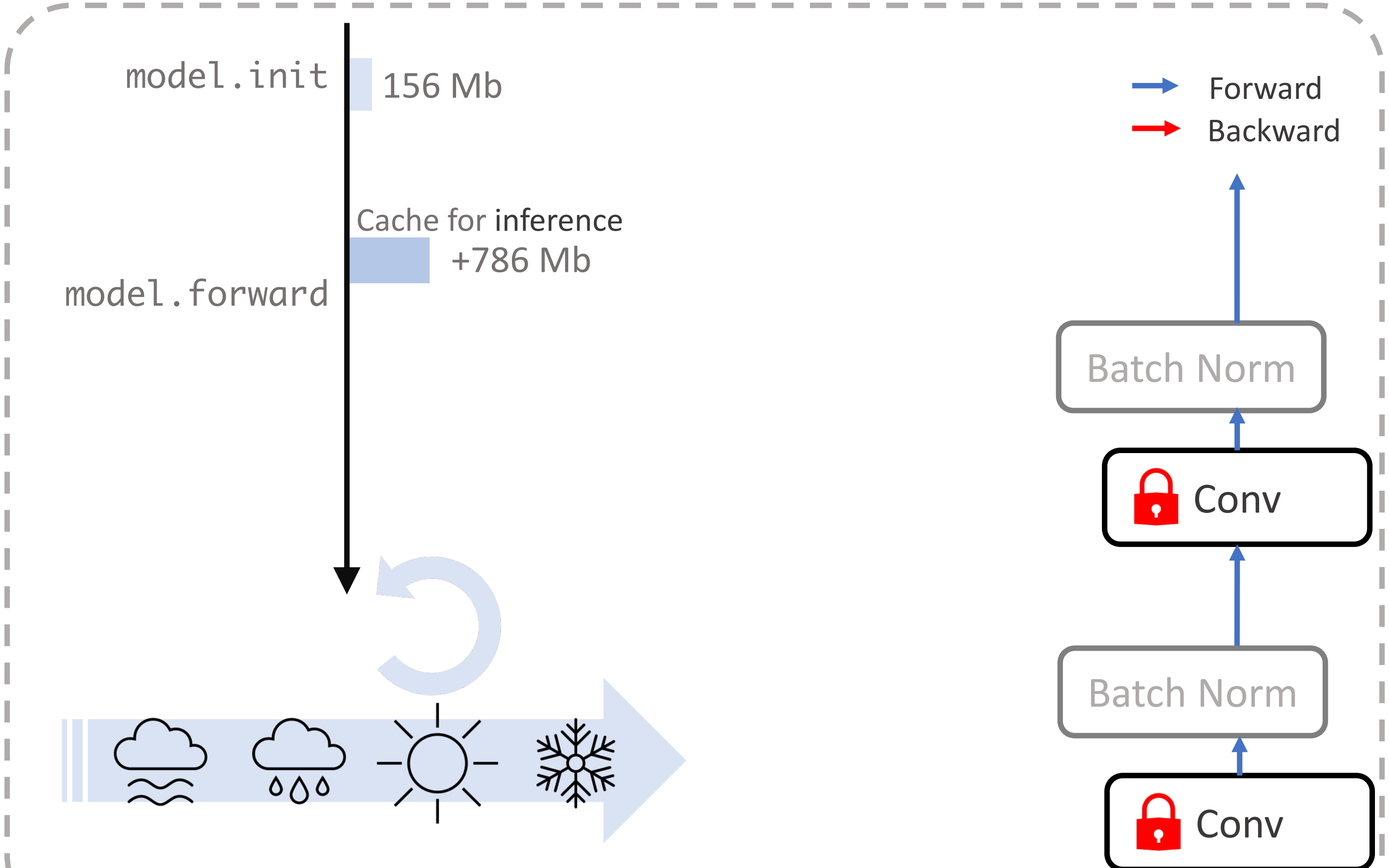
CTA:

- Unsupervised finetuning.
- Parameter efficient: Update BN only.
- Batch-estimated BN statistics $(\mu, \sigma)$ for capturing new environment.
- (EATA by Niu et al., 2022) Computation efficient.


Wang, D., Shelhamer, E., Liu, S., Olshausen, B., & Darrell, T. (2021). Tent: Fully test-time adaptation by entropy minimization. ICLR.

Niu, S., Wu, J., Zhang, Y., Chen, Y., Zheng, S., Zhao, P., & Tan, M. (2022). Efficient test-time model adaptation without forgetting. ICML.

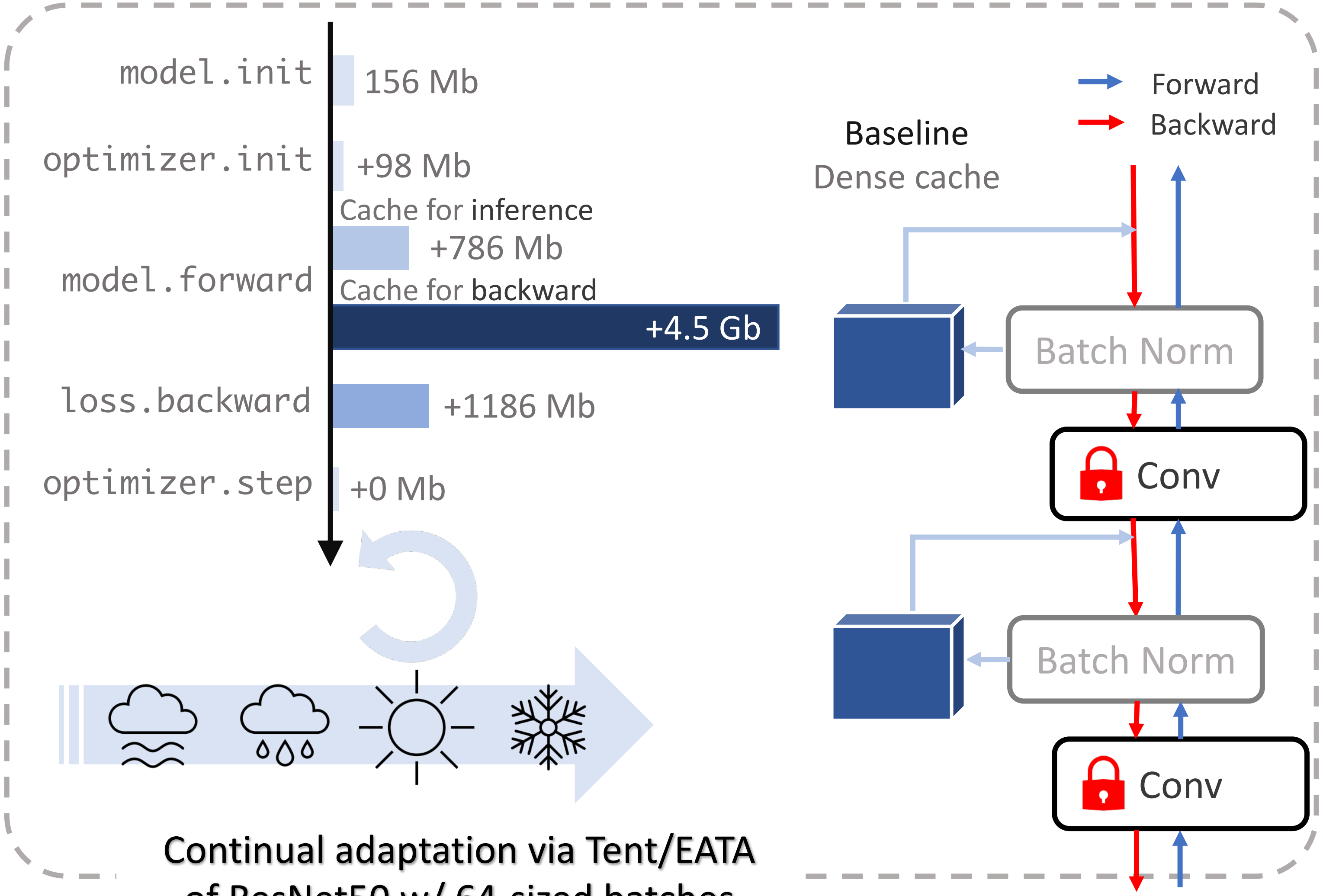
# High memory load for CTA on edge



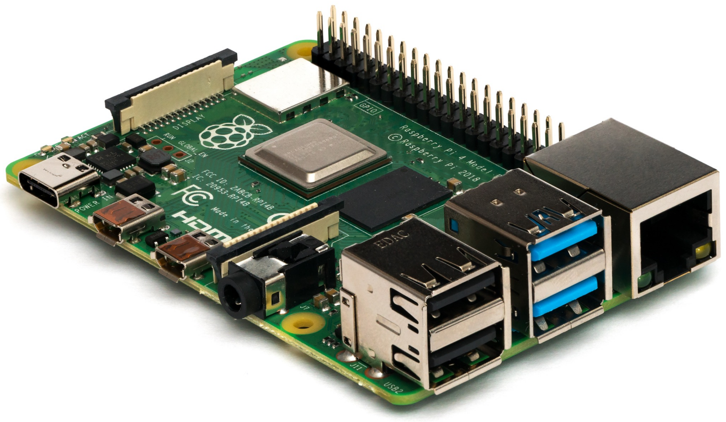

Continual adaptation via Tent/EATA
of ResNet50 w/ 64-sized batches

Raspberry Pi: 1-4Gb RAM
* Enough for inference

# High memory load for CTA on edge

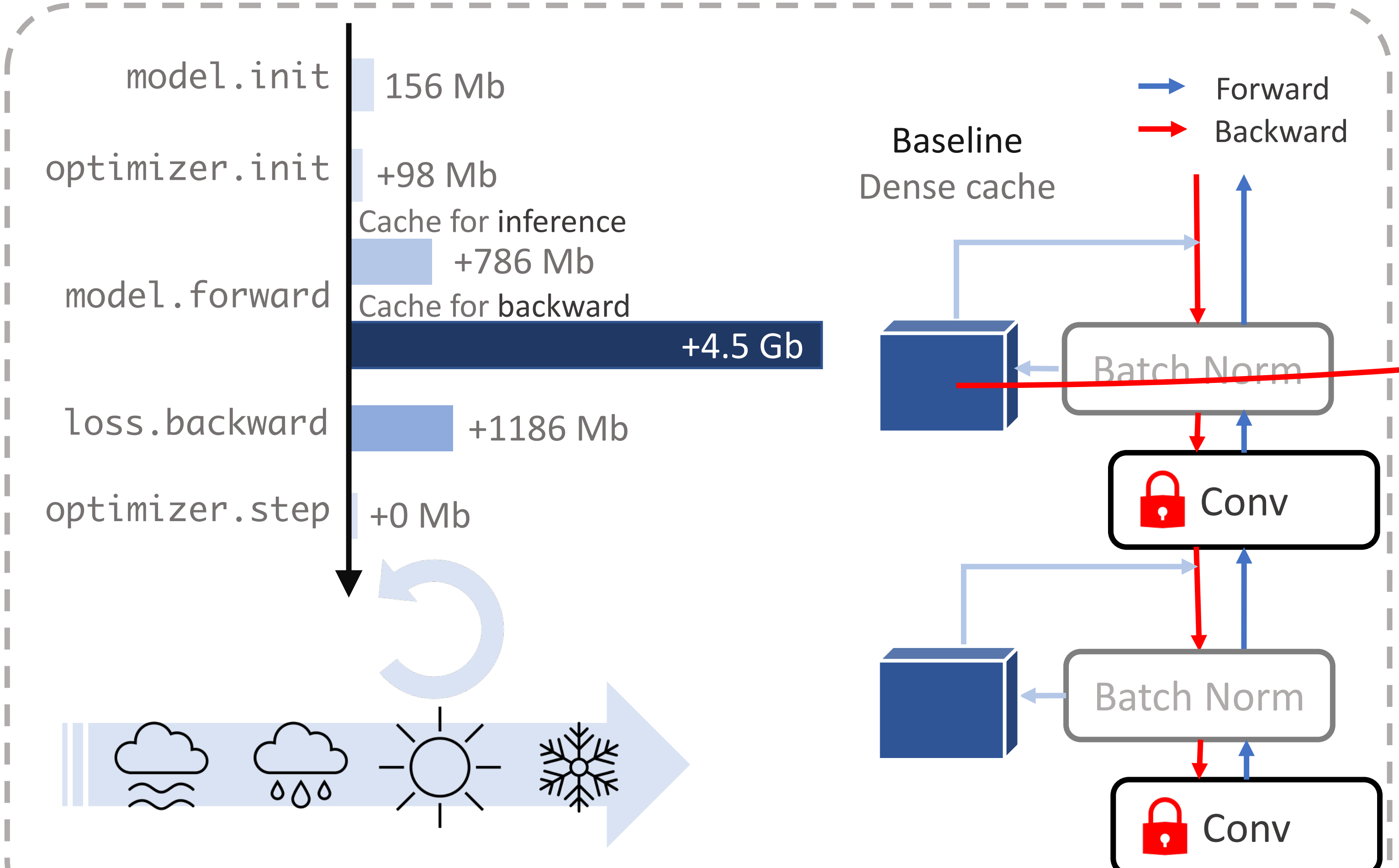



Continual adaptation via Tent/EATA of ResNet50 w/ 64-sized batches



Raspberry Pi: 1-4Gb RAM
* Enough for inference **but not for adaptation**

# High cache memory for back-propagation




Continual adaptation via Tent/EATA of ResNet50 w/ 64-sized batches

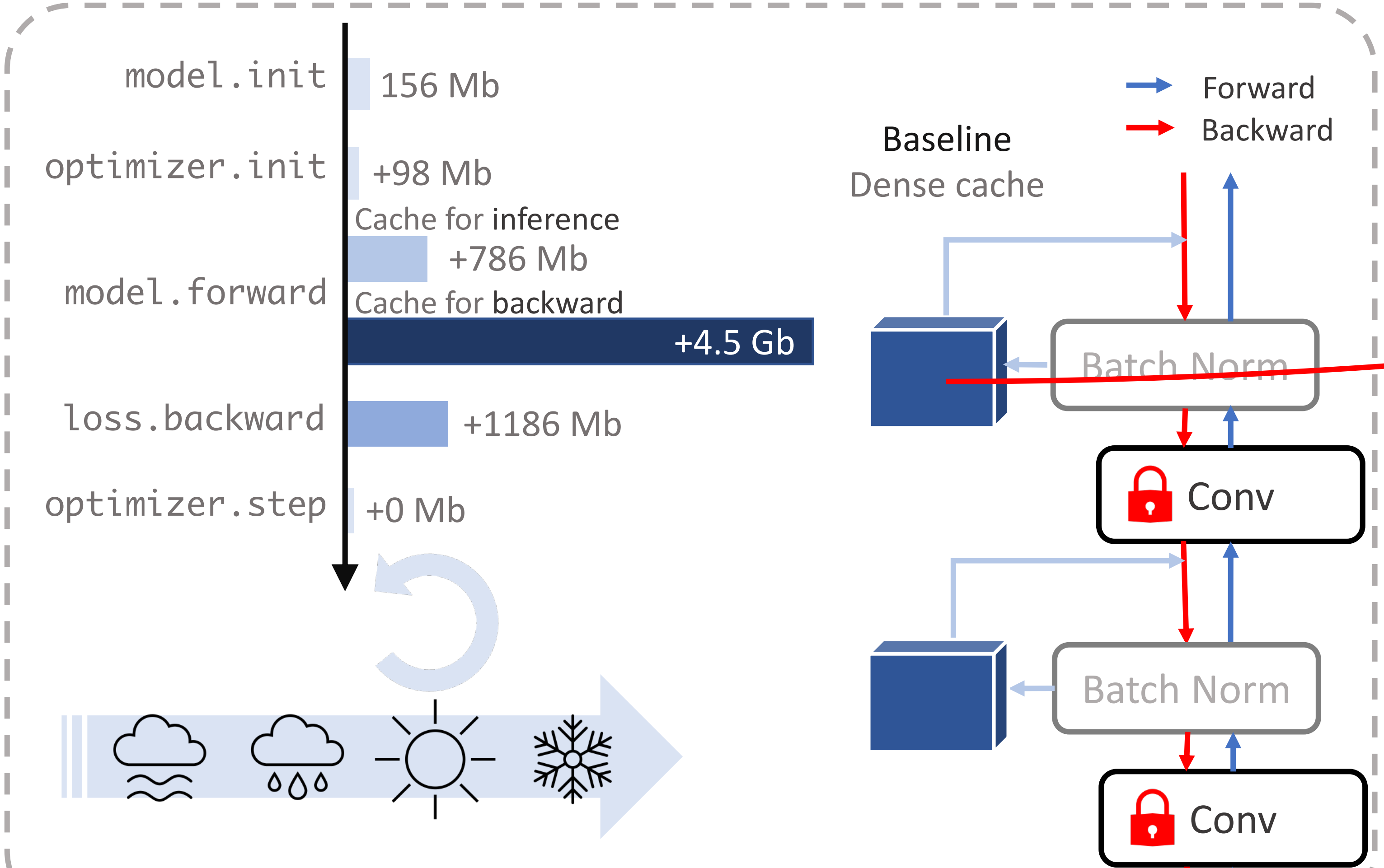
Batch-norm

$$z^l_{n,i,j,k} = \frac{x^l_{n,i,j,k} - \mu^l_i}{\sqrt{\sigma^2_i + \epsilon_0}} \qquad a^l_{n,i,j,k} = \boxed{\gamma^l_i z^l_{n,i,j,k}} + b^l_i$$

cache

$$\sum_{n=1}^{B} \frac{\partial \ell_n}{\partial \gamma^l_i} = \sum_{n=1}^{B} \sum_{j=1}^{W} \sum_{k=1}^{H} \frac{\partial \ell_n}{\partial a^l_{i,j,k}} \boxed{z^l_{n,i,j,k}}.$$

# High cache memory for back-propagation




Continual adaptation via Tent/EATA of ResNet50 w/ 64-sized batches

Batch-norm

$$z^l_{n,i,j,k} = \frac{x^l_{n,i,j,k} - \mu^l_i}{\sqrt{\sigma^2_i + \epsilon_0}} \qquad a^l_{n,i,j,k} = \gamma^l_i z^l_{n,i,j,k} + b^l_i$$

$$\sum_{n=1}^{B} \frac{\partial \ell_n}{\partial \gamma^l_i} = \sum_{n=1}^{B} \sum_{j=1}^{W} \sum_{k=1}^{H} \frac{\partial \ell_n}{\partial a^l_{i,j,k}} \boxed{z^l_{n,i,j,k}.}$$

Cache size

$$R_{\text{fwd}} = \max_{l \in \{1,\cdots,L\}} B \times C^l \times W^l \times H^l.$$

$$R_{\text{bwd}} = \boxed{\sum_{l=1}^{L} B \times C^l \times W^l \times H^l \geq R_{\text{fwd}}}$$

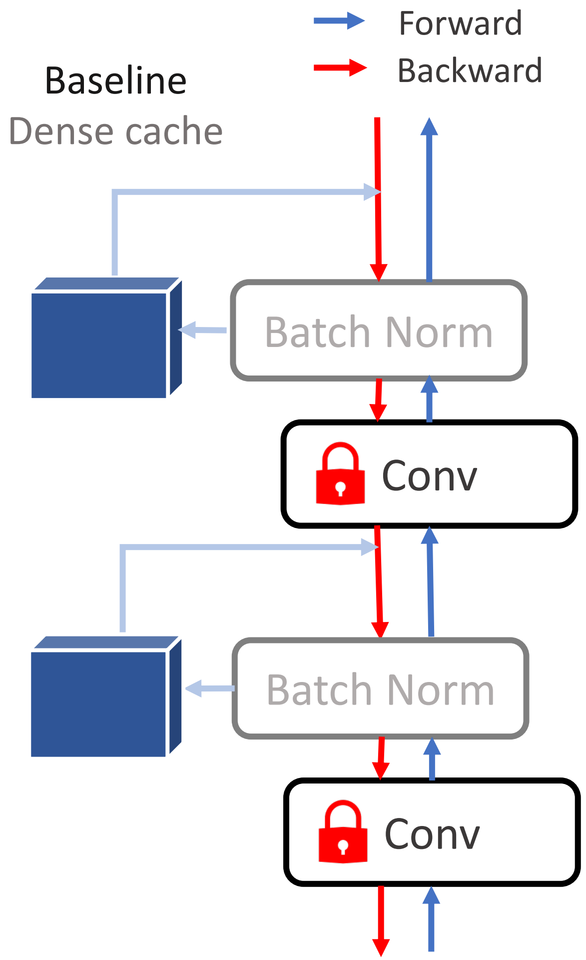
Traditional CTA cannot fit into low-memory devices.

- **(B)** Require large <u>batch</u> size for statistic estimation.
- **(L&C)** The cache tensor $z$ scales by number of <u>layers</u> and <u>channels</u>.

# Memory-Efficient Adaptation by MECTA

$$z_{n,i,j,k}^{l} = \frac{x_{n,i,j,k}^{l} - \mu_i^l}{\sqrt{\sigma_i^2 + \epsilon_0}}$$

$$R_{\text{bwd}} = \sum_{l=1}^{L} B \times C^l \times W^l \times H^l \geq R_{\text{fwd}}$$



# Memory-Efficient Adaptation by MECTA

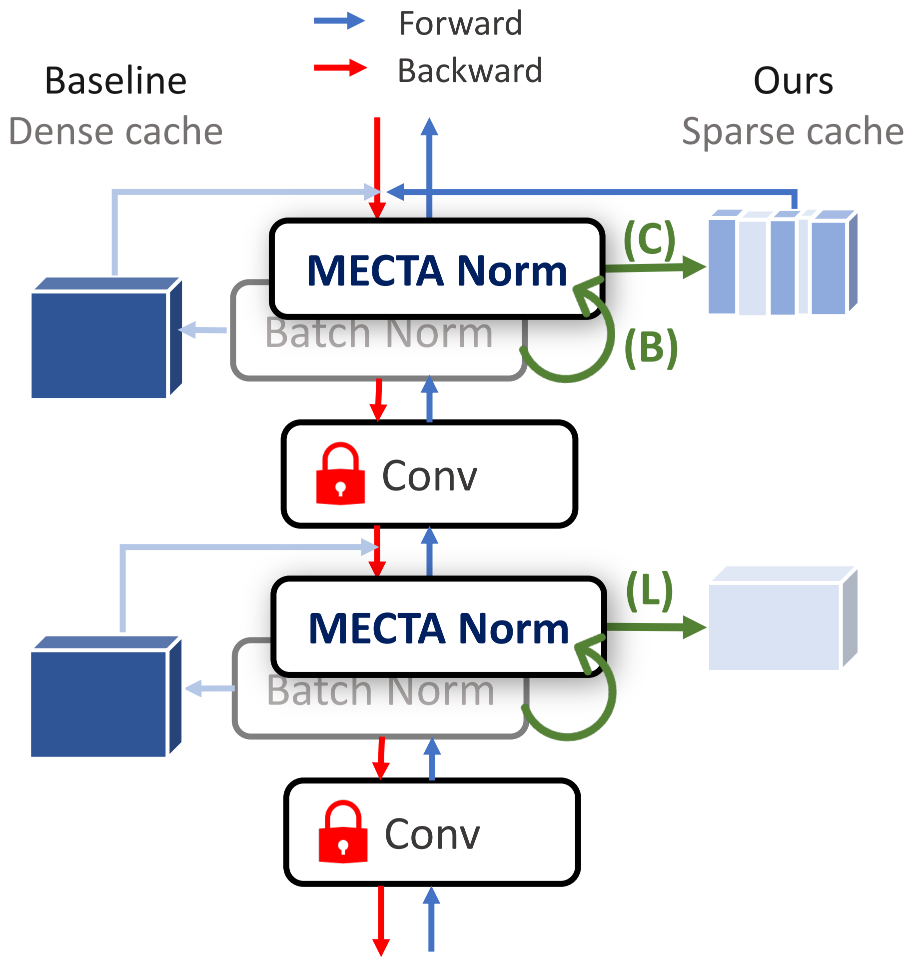
- (B) Reduce batch size and maintain accurate statistic estimation.

$$\mu = \boxed{\frac{1}{B}} \frac{1}{W} \frac{1}{H} \sum_{n,j,k}^{B,W,H} x_{n,i,j,k}$$

Accurate **statistics** $\phi = [\mu, \sigma]$ require more samples in a batch (larger B).

$$z^l_{n,i,j,k} = \frac{x^l_{n,i,j,k} - \boxed{\mu^l_i}}{\sqrt{\boxed{\sigma^2_i} + \epsilon_0}}$$

$$R_{\text{bwd}} = \sum_{l=1}^{L} \boxed{B} \times C^l \times W^l \times H^l \geq R_{\text{fwd}}$$



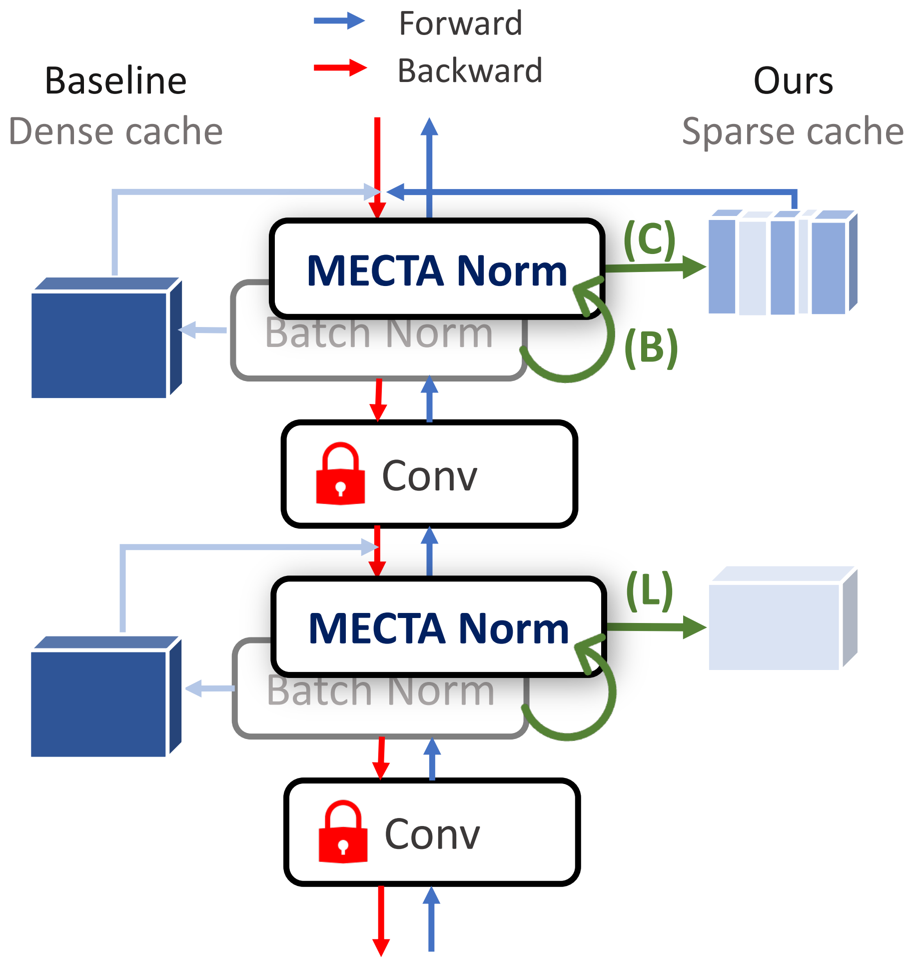
# Memory-Efficient Adaptation by MECTA

- (B) Reduce batch size and maintain accurate statistic estimation.
- (C) Reduce channels to update.
- (L) Reduce layers to update.

$$z^l_{n,i,j,k} = \frac{x^l_{n,i,j,k} - \mu^l_i}{\sqrt{\sigma_i^2 + \epsilon_0}}$$

$$R_{\text{bwd}} = \sum_{l=1}^{\boxed{L}} B \times \boxed{C^l} \times W^l \times H^l \geq R_{\text{fwd}}$$

Drop cache ⇔ Drop gradient
Maintain effective adaptation requires **enough gradient** information.



# Memory-Efficient Adaptation by MECTA

MECTA Norm

Batch Norm

(B)

- (B) Reduce batch size and **maintain** accurate statistic estimation **per BN layer.**

Exponential Moving Average (EMA)

$\phi$ represents $[\mu, \sigma]$

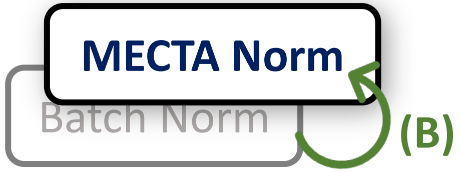
$\boxed{\hat{\phi}_t}$

Statistic of current batch

$$\mu = \boxed{\frac{1}{B}} \frac{1}{W} \frac{1}{H} \sum_{n,j,k}^{B,W,H} x_{n,i,j,k}$$

Inaccurate if $B \downarrow$

# Memory-Efficient Adaptation by MECTA




- (B) Reduce batch size and **maintain** accurate statistic estimation **per BN layer.**
  - Moving average accommodate small batch sizes towards robust and accurate statistic estimations.

Exponential Moving Average (EMA)

$\phi$ represents $[\mu, \sigma]$

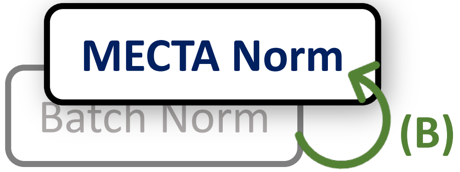
$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

Previous memory

Statistic of current batch

$$\mu = \boxed{\frac{1}{B}}\frac{1}{W}\frac{1}{H}\sum_{n,j,k}^{B,W,H} x_{n,i,j,k}$$

# Memory-Efficient Adaptation by MECTA




- (B) Reduce batch size and **maintain** accurate statistic estimation.
  - Moving average accommodate small batch sizes towards robust and accurate statistic estimations.

Exponential Moving Average (EMA)

$\phi$ represents $[\mu, \sigma]$

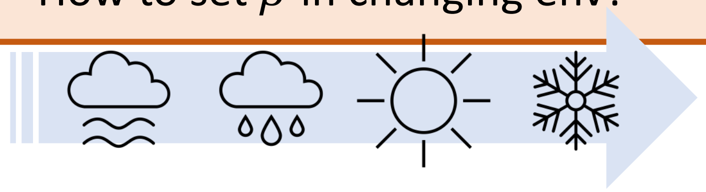
$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

Previous memory

Statistic of current batch

$\beta = 1$ : Only using current **small** batch result in **inaccurate** estimation.

How to set $\beta$ in changing env?



# Memory-Efficient Adaptation by MECTA

- (B) Reduce batch size and **maintain** accurate statistic estimation.
  - Moving average accommodate small batch sizes towards robust and accurate statistic estimations.

Exponential Moving Average (EMA)

$\phi$ represents $[\mu, \sigma]$

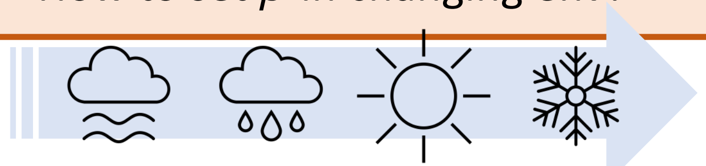
$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

Previous memory

Statistic of current batch

$\beta = 1$ : Only using current **small** batch result in **inaccurate** estimation.

How to set $\beta$ in changing env?



$\beta = 0$ : Only using **training/past** stat (without updates) result in **non-adaptive/non-robust** estimation.

# Memory-Efficient Adaptation by MECTA

- (B) Reduce batch size and maintain accurate statistic estimation.
  - Moving average accommodate small batch sizes towards robust and accurate statistic estimations.

Exponential Moving Average (EMA)

$\phi$ represents $[\mu, \sigma]$

$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

Previous memory

Statistic of current batch

$\beta = 1$ : Only using current **small** batch result in **inaccurate** estimation.

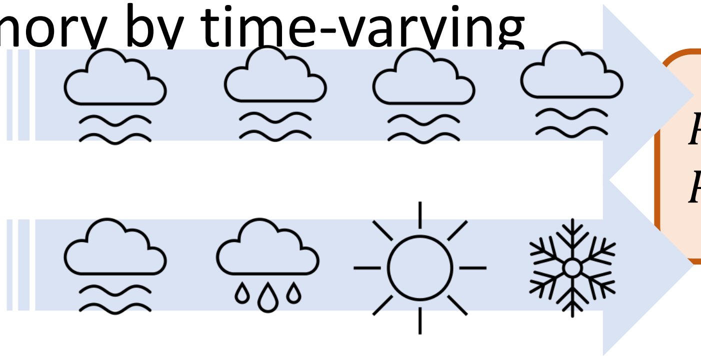
$0 < \beta < 1$

Trade-off between accuracy and adaptivity

$\beta = 0$ : Only using **training** batch (without updates) result in **non-adaptive** estimation.

# Memory-Efficient Adaptation by MECTA

- (B) Reduce batch size and maintain accurate statistic estimation.
  - Moving average accommodate small batch sizes towards robust and accurate statistic estimations.
  - Adaptive memory by time-varying $\beta$.



Exponential Moving Average (EMA)

$\phi$ represents $[\mu, \sigma]$

$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

Previous memory

Statistic of current batch

$\beta = 1$ : Only using current **small** batch result in **inaccurate** estimation.

$P_{t-1} = P_t$ : Stable ➔ $\beta \to 0$ (accurate)

$P_{t-1} \neq P_t$ : On change. ➔ $\beta \to 1$ (fast adapt)

$\beta = 0$ : Only using **training** batch (without updates) result in **non-adaptive** estimation.

# Memory-Efficient Adaptation by MECTA

- (B) Reduce batch size and maintain accurate statistic estimation.
  - Moving average accommodate small batch sizes towards robust and accurate statistic estimations.
  - Adaptive memory by time-varying $\beta$.

$$\beta_t = 1 - e^{-D(\phi_{t-1}, \hat{\phi}_t)},$$

$$D(\phi_{t-1}, \hat{\phi}_t) = \frac{1}{C} \sum_{i=1}^{C} KL(\phi_{t-1,i} \| \hat{\phi}_{t,i}) + KL(\hat{\phi}_{t,i} \| \phi_{t-1,i})$$

Exponential Moving Average (EMA)

$\phi$ represents $[\mu, \sigma]$

$$\phi_t = (1 - \beta)\boxed{\phi_{t-1}} + \beta \boxed{\hat{\phi}_t},$$

Previous memory

Statistic of current batch

$\beta = 1$ : Only using current **small** batch result in **inaccurate** estimation.

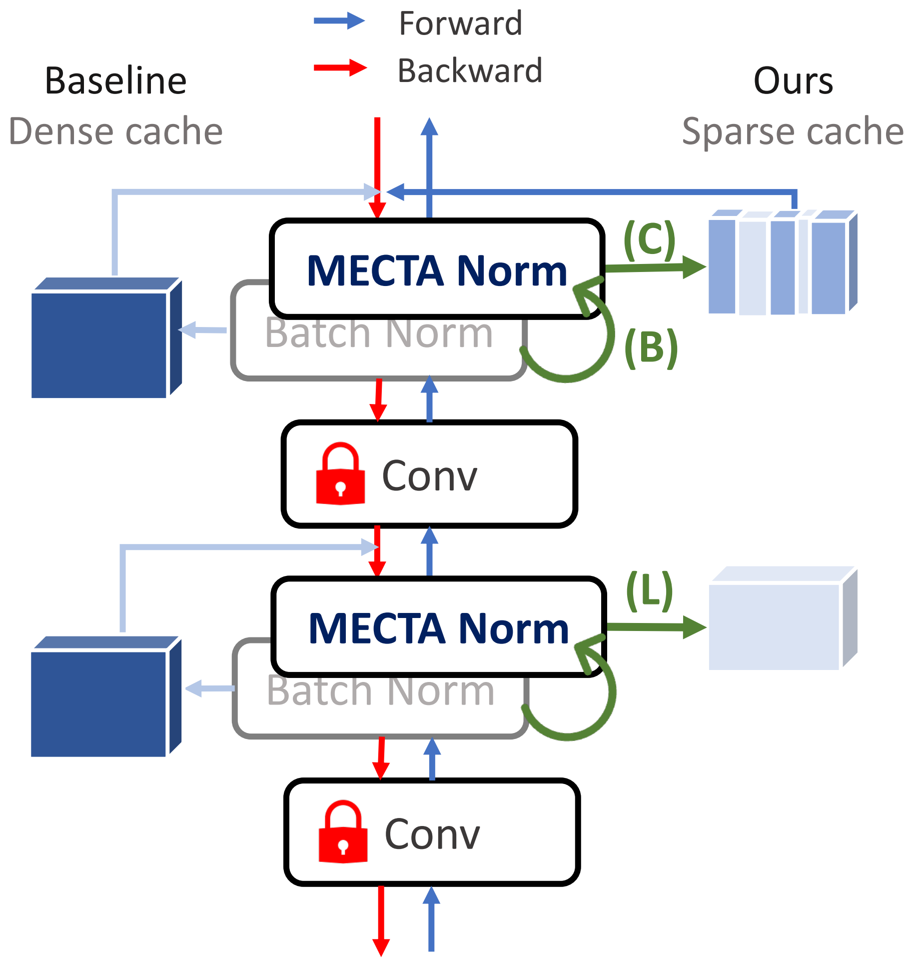
$\phi_{t-1} = \hat{\phi}_t$ : Stable ➔ $\beta \rightarrow 0$ (accurate)

$\phi_{t-1} \neq \hat{\phi}_t$ : On change. ➔ $\beta \rightarrow 1$ (fast adapt)

$\beta = 0$ : Only using **training** batch (without updates) result in **non-adaptive** estimation.

# Memory-Efficient Adaptation by MECTA

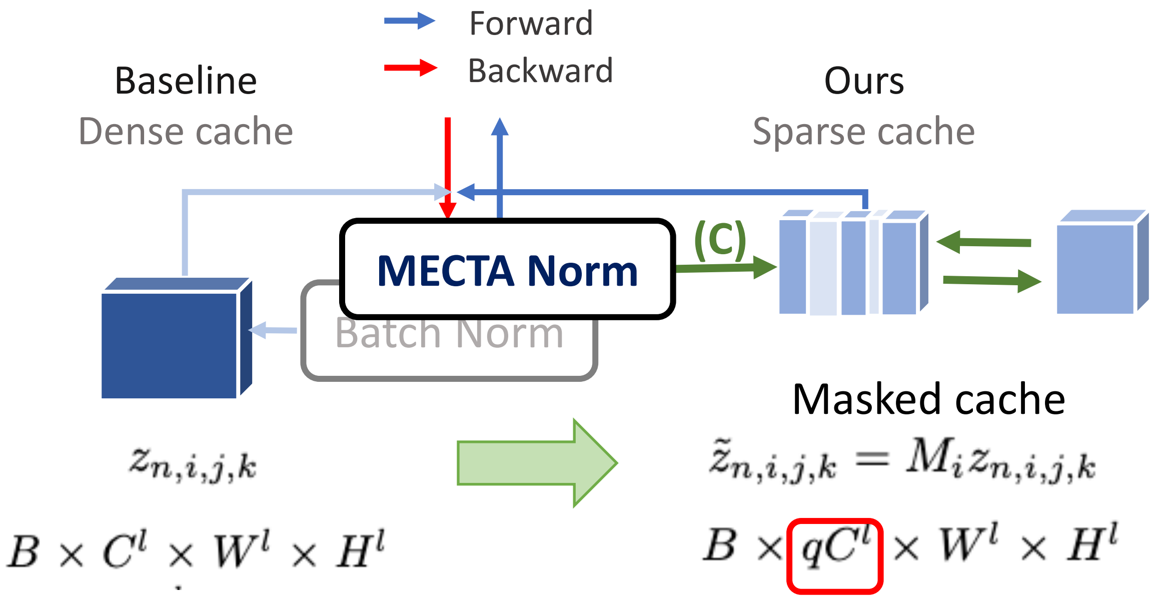
- (B) Reduce batch size and maintain accurate statistic estimation.
- **(C) Reduce channels to update.**
- (L) Reduce layers to update.



# Memory-Efficient Adaptation by MECTA

- (C) Reduce channels to update.
  - Drop $q \times 100\%$ channels in cache.

**Drop cache ⇔ Drop gradient**
Maintain effective adaptation requires **enough gradient** information.



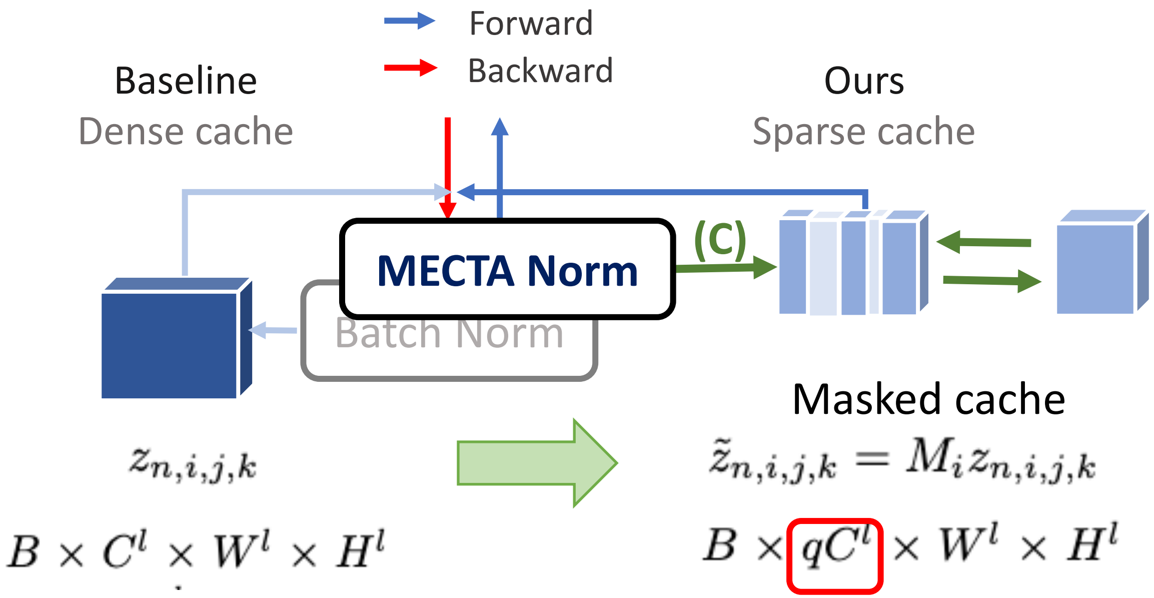
# Memory-Efficient Adaptation by MECTA

- (C) Reduce channels to update.
  - Stochastically drop $q\times 100\%$ channels in cache.

Stoc. drop cache ⇔ **Not** drop gradient.
Maintain effective adaptation requires **enough gradient** information.

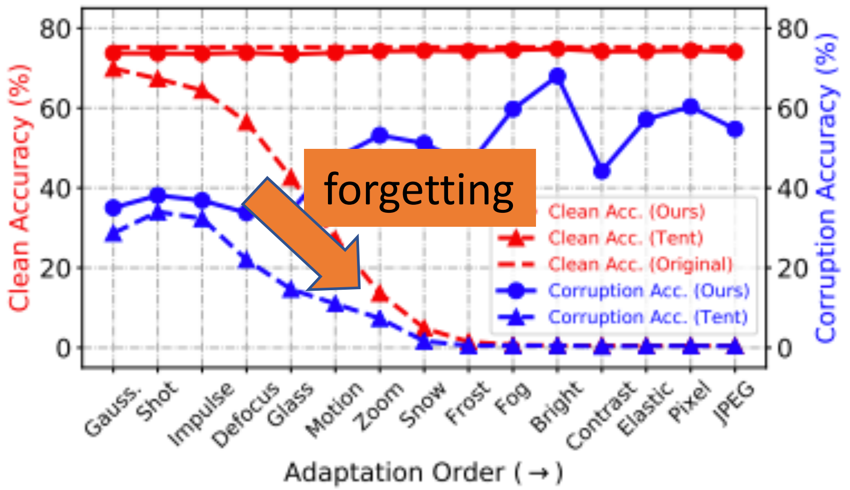
$$\boxed{\mathrm{E}[M_i z_{n,i,k}]} = q\mathrm{E}[\, z_{n,i,j,k}]$$

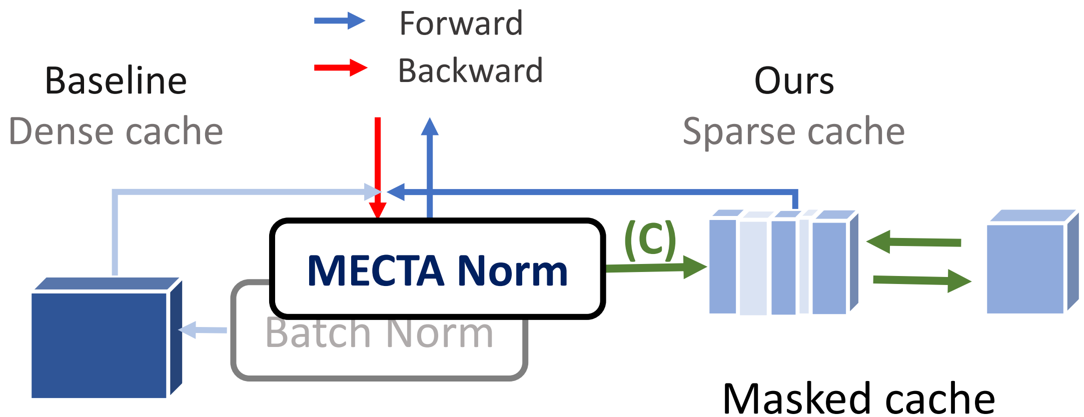
Expectation over continual batches can **restore** the dropped channels



# Memory-Efficient Adaptation by MECTA

- (C) Reduce channels to update.
  - Stochastically drop $q\times100\%$ channels in cache.
  - Implicit gradient regularization which **mitigates forgetting**.




Masked cache

$z_{n,i,j,k}$ $\quad\Rightarrow\quad$ $\tilde{z}_{n,i,j,k} = M_i z_{n,i,j,k}$

$B \times C^l \times W^l \times H^l$ $\quad$ $B \times \boxed{qC^l} \times W^l \times H^l$

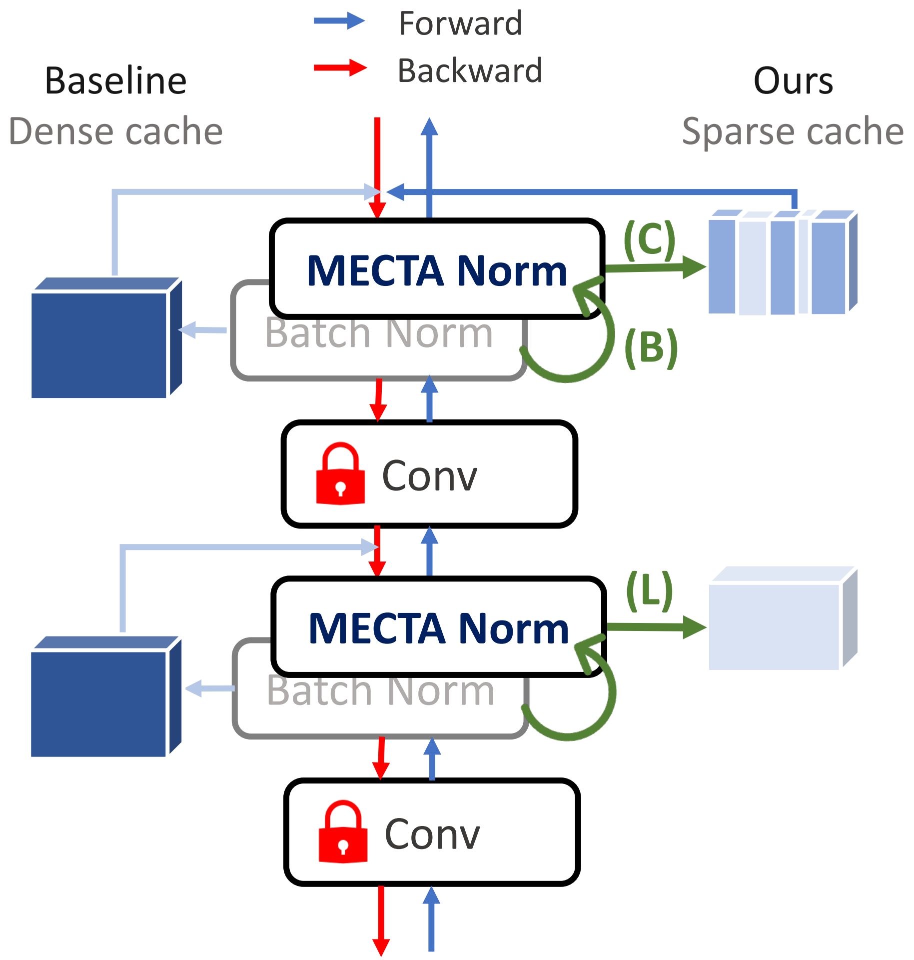
$$\|\theta_t - \theta_0\| = \|\textstyle\sum_t g_t\| \le \mathcal{O}(\textstyle\sum_t \boxed{\|g_t\|})$$

Smaller => less forgetting

Niu, S., Wu, J., Zhang, Y., Chen, Y., Zheng, S., Zhao, P., & Tan, M. (2022). Efficient test-time model adaptation without forgetting. ICML.

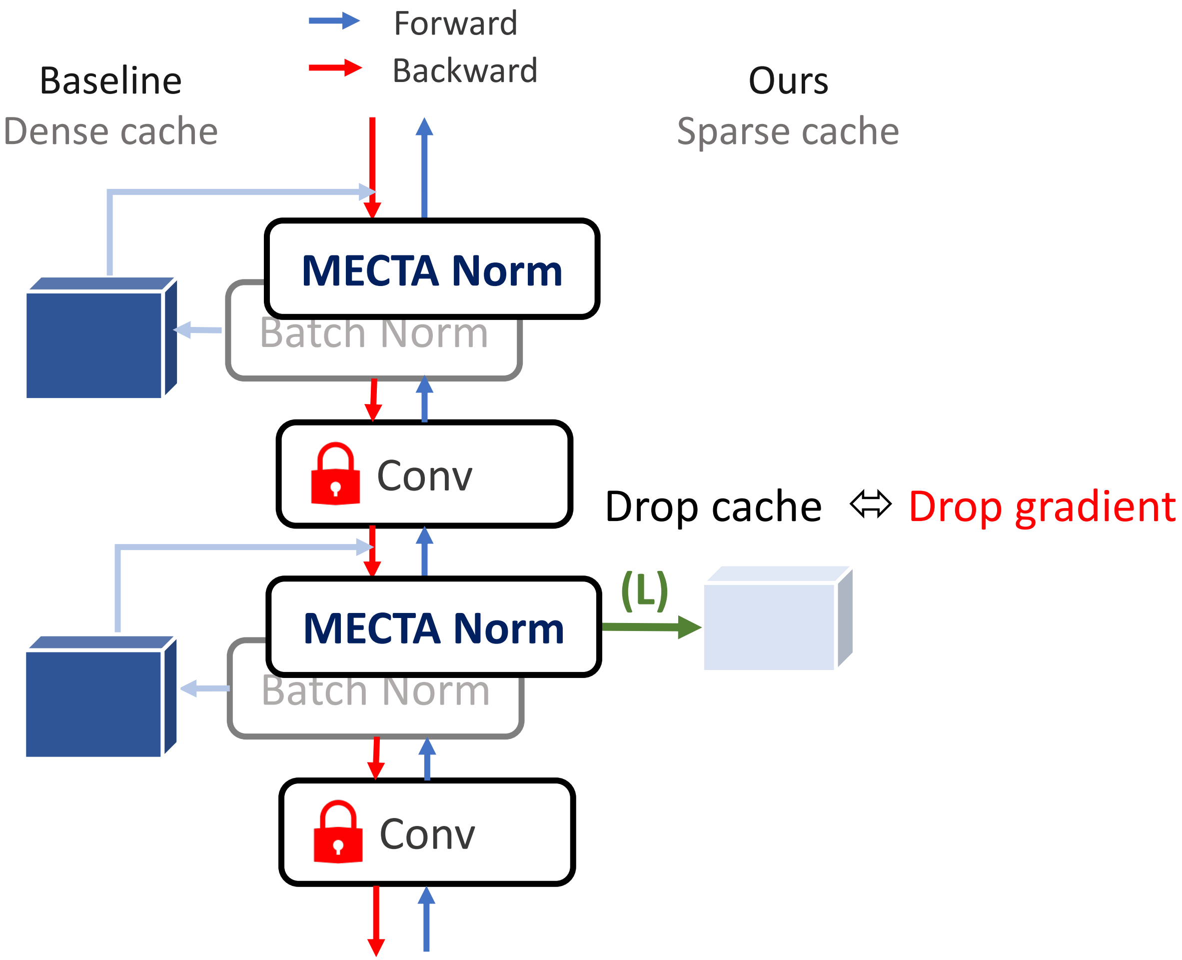
# Memory-Efficient Adaptation by MECTA

- (Reduce B) Adaptive and online statistic estimation on dynamic distributions for accurate statistics on small batch sizes.
- (Reduce C) Channel-sparse gradients via stochastically-pruned caches.
- **(Dynamic L) Cache and train layers on demand.**



# Memory-Efficient Adaptation by MECTA

- (Dynamic L) Cache and train layers on demand.
  - If environment is **stable**, there is no need to continually adapt.
  - Stop gradient to save caches.



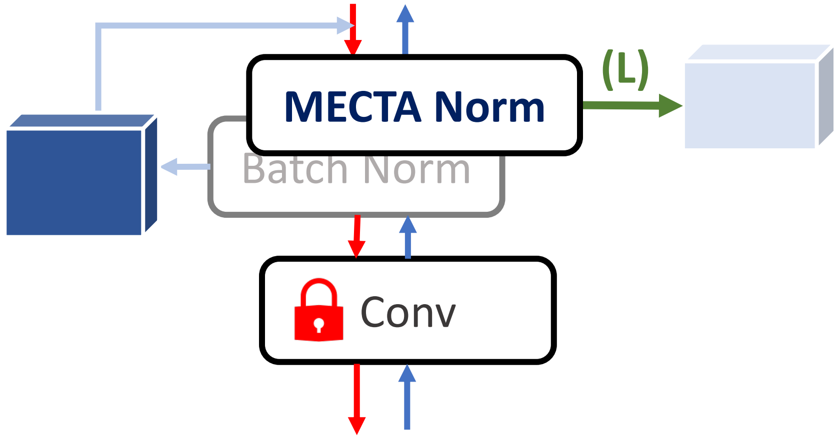
# Memory-Efficient Adaptation by MECTA

- (Dynamic L) Cache and train layers on demand.
  - If environment is **stable**, there is no need to continually adapt.
  - Stop gradient to save caches.

$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

Previous memory     Statistic of current batch

- When to stop? Environment is stable.
- When to restart? Environment changes.



# Memory-Efficient Adaptation by MECTA

- (Dynamic L) Cache and train layers on demand.
  - If environment is **stable**, there is no need to continually adapt.
  - Stop gradient to save caches.

$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

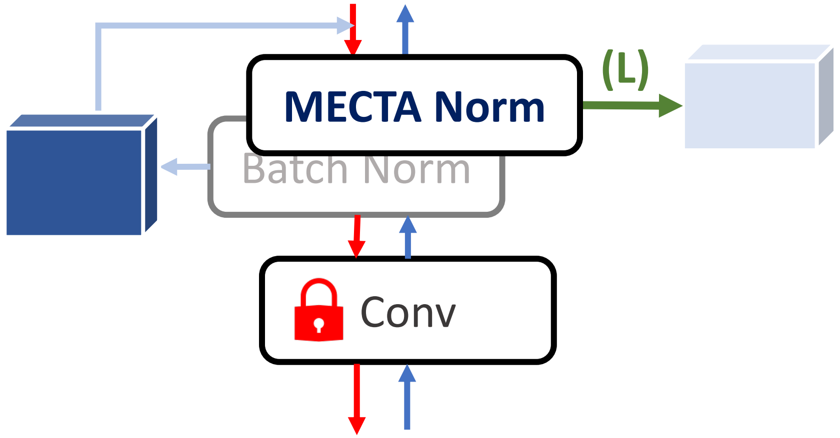
Previous memory

Statistic of current batch

$P_{t-1} = P_t$ : Stable.

$P_{t-1} \neq P_t$ : On change.

- When to stop? Environment is stable.
- When to restart? Environment changes.



# Memory-Efficient Adaptation by MECTA

- (Dynamic L) Cache and train layers on demand.
  - If environment is **stable**, there is no need to continually adapt.
  - Stop gradient to save caches.

$$\phi_t = (1 - \beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

Previous memory

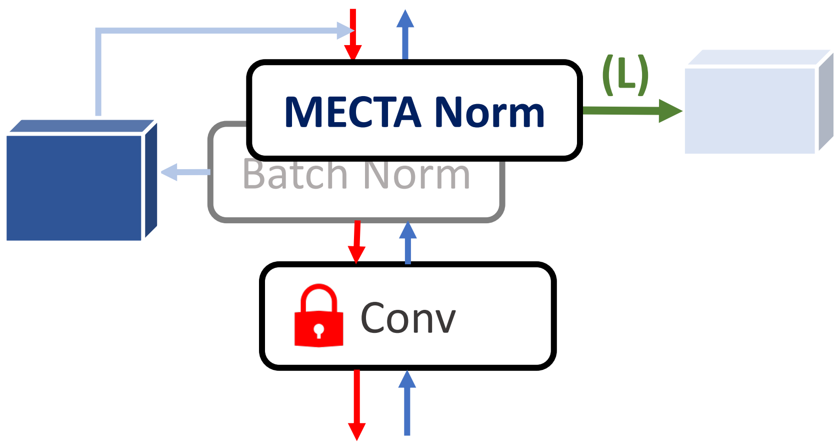
Statistic of current batch

$P_{t-1} = P_t$ : Stable ← $\beta \to 0$

$P_{t-1} \neq P_t$ : On change. ← $\beta \to 1$

$$\beta_t = 1 - e^{-D(\phi_{t-1}, \hat{\phi}_t)},$$

- When to stop? Environment is stable.
- When to restart? Environment changes.



# Memory-Efficient Adaptation by MECTA

- (Dynamic L) Cache and train layers on demand.
  - If environment is **stable**, there is no need to continually adapt.
  - Stop gradient to save caches.

$$\phi_t = (1-\beta)\boxed{\phi_{t-1}} + \beta\boxed{\hat{\phi}_t},$$

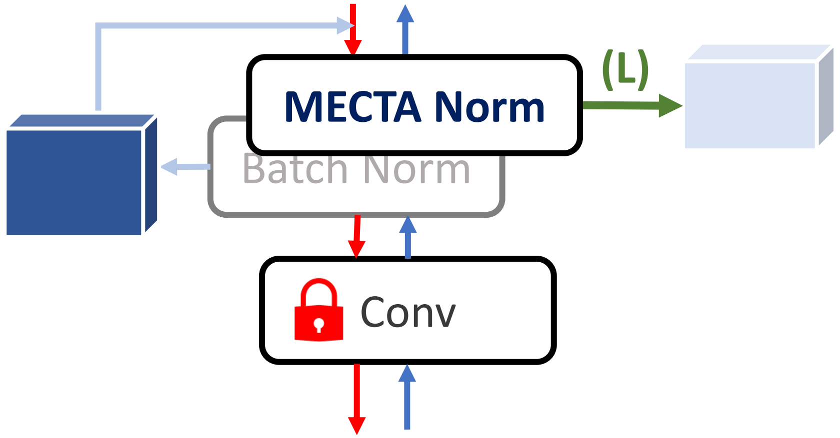
Previous memory — Statistic of current batch

$P_{t-1} = P_t$ : Stable ← $\beta \to 0$

$P_{t-1} \neq P_t$ : On change. ← $\beta \to 1$

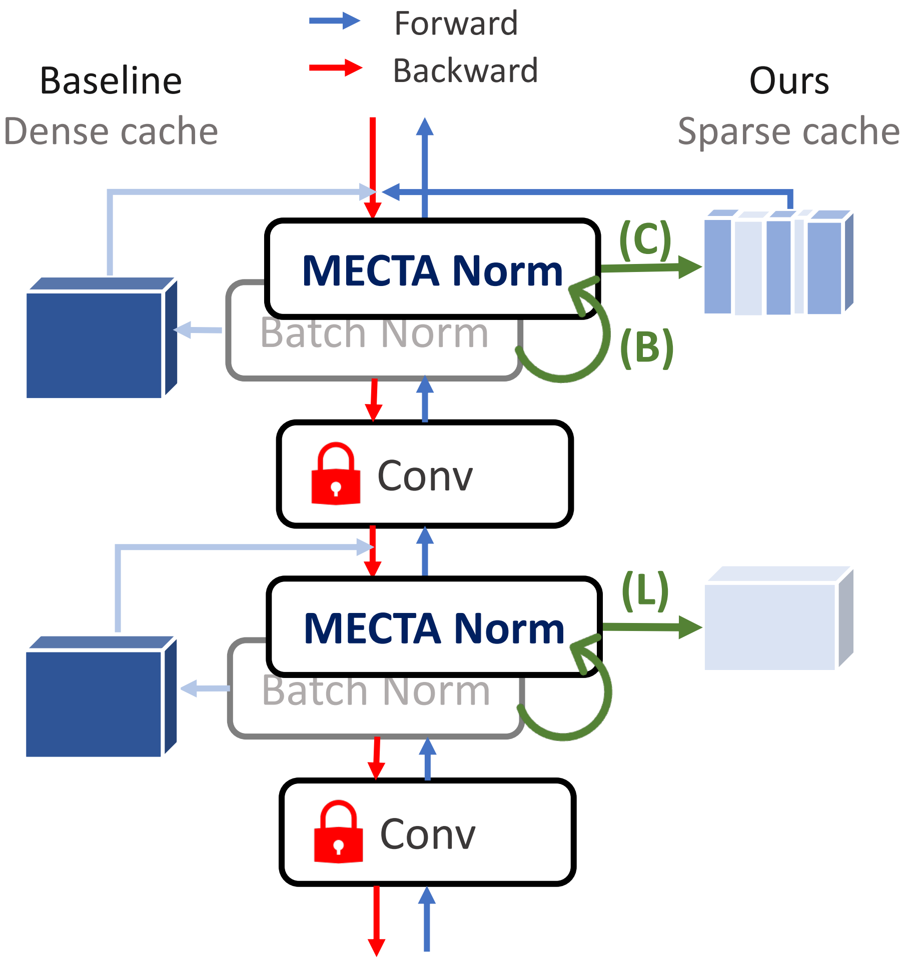
$$\beta_t = 1 - e^{-D(\phi_{t-1}, \hat{\phi}_t)}$$

- When to stop? Environment is stable: $\boxed{\beta_t < \beta_{th}}$
- When to restart? Environment changes: $\boxed{\beta_t > \beta_{th}}$

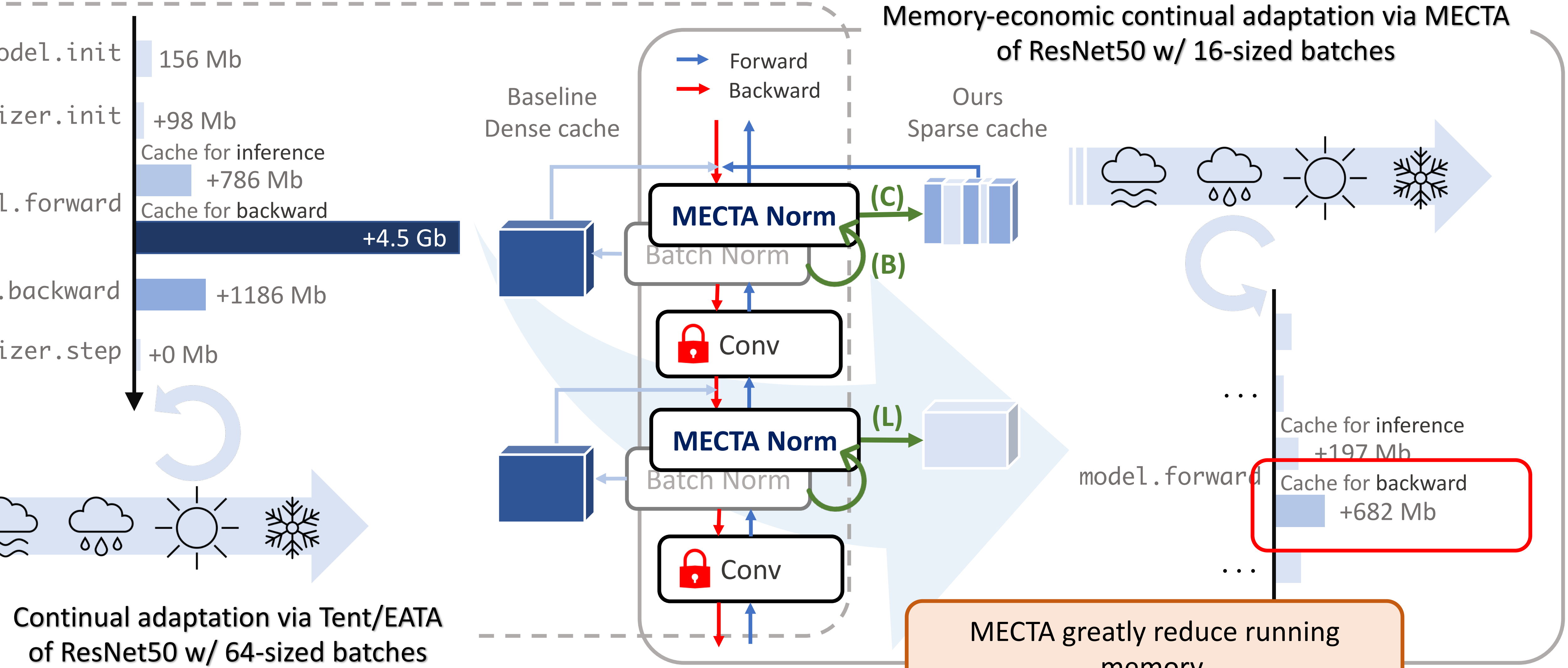


# Memory-Efficient Adaptation by MECTA

- (Reduce B) Adaptive and online statistic estimation on dynamic distributions for accurate statistics on small batch sizes.
- (Reduce C) Channel-sparse gradients via stochastically-pruned caches.
- (Dynamic L) Cache and train layers on demand.



# Memory-Efficient Adaptation by MECTA




MECTA greatly reduce running memory.

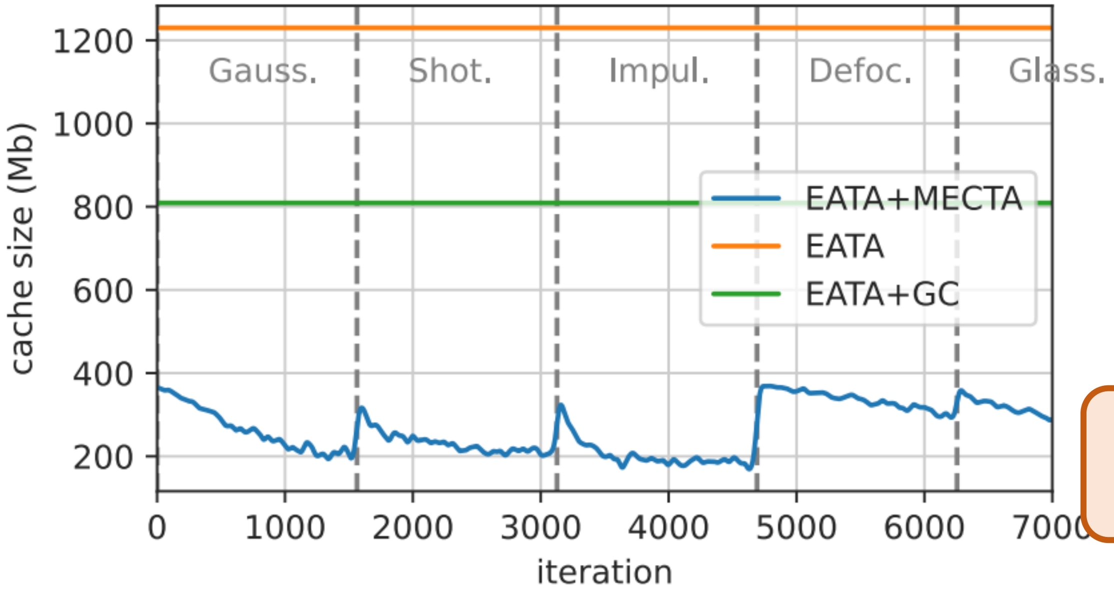
# Benchmark: Accuracy & Memory Efficiency

Changing test-time noise

| Alg. | Noise: Gauss. | Shot. | Impul. | Blur: Defoc. | Glass. | Motion | Zoom. | Weather: Snow | Frost | Fog | Bright. | Digital: Contr. | Elast. | Pixel. | JPEG | Acc. Avg | Cache Max (Mb) | #fwd | #bwd |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | ResNet50; Reset model per perturbation | | | | | | | | | | | | | | | | | | |
| TTT (GN+JT) | 31.0 | 33.6 | 33.4 | 28.1 | 7.8 | 33.2 | 36.8 | 40.9 | 19.0 | 51.0 | 61.8 | 38.9 | 49.4 | 51.7 | 48.0 | 37.6 | 2460×20 | 21 | 20 |
| BN | 15.5 | 16.1 | 16.3 | 20.0 | 20.0 | 28.5 | 40.0 | 34.8 | 35.0 | 48.5 | 65.9 | 24.1 | 45.8 | 50.7 | 41.1 | 33.5 | **206** | 1 | 0 |
| TTA | 4.1 | 4.9 | 4.5 | 12.5 | 8.2 | 12.9 | 25.8 | 14.0 | 19.1 | 21.3 | 53.0 | 12.4 | 14.6 | 24.6 | 33.6 | 17.7 | **206** | 64 | 0 |
| MEMO | 7.5 | 8.7 | 9.0 | 19.7 | 13.0 | 20.7 | 27.6 | 25.3 | 28.8 | 32.1 | 61.0 | 11.0 | 23.8 | 33.0 | 37.5 | 23.9 | 2460×65 | 65 | 65 |
| | ResNet50; Lifelong adaptation | | | | | | | | | | | | | | | | | | |
| CoTTA(+GC) | 16.9 | 20.3 | 22.8 | 20.6 | 22.0 | 31.7 | 42.4 | 34.5 | 34.0 | 47.2 | 58.9 | 24.1 | 44.5 | 48.6 | 42.4 | 34.1 | 2845 (1618) | 33 | 1 |
| Tent(+GC) | 28.4 | 34.1 | 31.7 | 19.3 | 12.2 | 6.9 | 4.0 | 1.4 | 0.8 | 0.7 | 0.9 | 0.4 | 0.6 | 0.7 | 0.6 | 9.5 | 2845 (1618) | 1 | 1 |
| Tent+MECTA | 24.5 | 29.5 | 28.3 | 22.0 | 23.5 | 27.4 | 37.2 | 28.2 | 27.1 | 36.8 | 50.7 | 15.5 | 38.0 | 40.2 | 34.7 | 30.9 | 847 | 1 | 1 |
| EATA(+GC) | **35.0** | 38.1 | 36.8 | **33.8** | **34.2** | **47.3** | **53.2** | **51.1** | **45.6** | **59.7** | **68.0** | **44.2** | **57.2** | **60.4** | **54.7** | **48.0** | 2845 (1618) | 1 | **0.56** |
| EATA+MECTA | 33.7 | **39.1** | **37.8** | 31.7 | 33.1 | 42.2 | 50.3 | 46.3 | 43.0 | 56.9 | 65.4 | 41.2 | 55.2 | 58.2 | 53.7 | 45.9 | 847 | 1 | **0.56** |

Best trade-off between memory and accuracy.

# MECTA dynamically cache data on demand




Cache on demand of environment change

# Benchmark with Constrained Cache

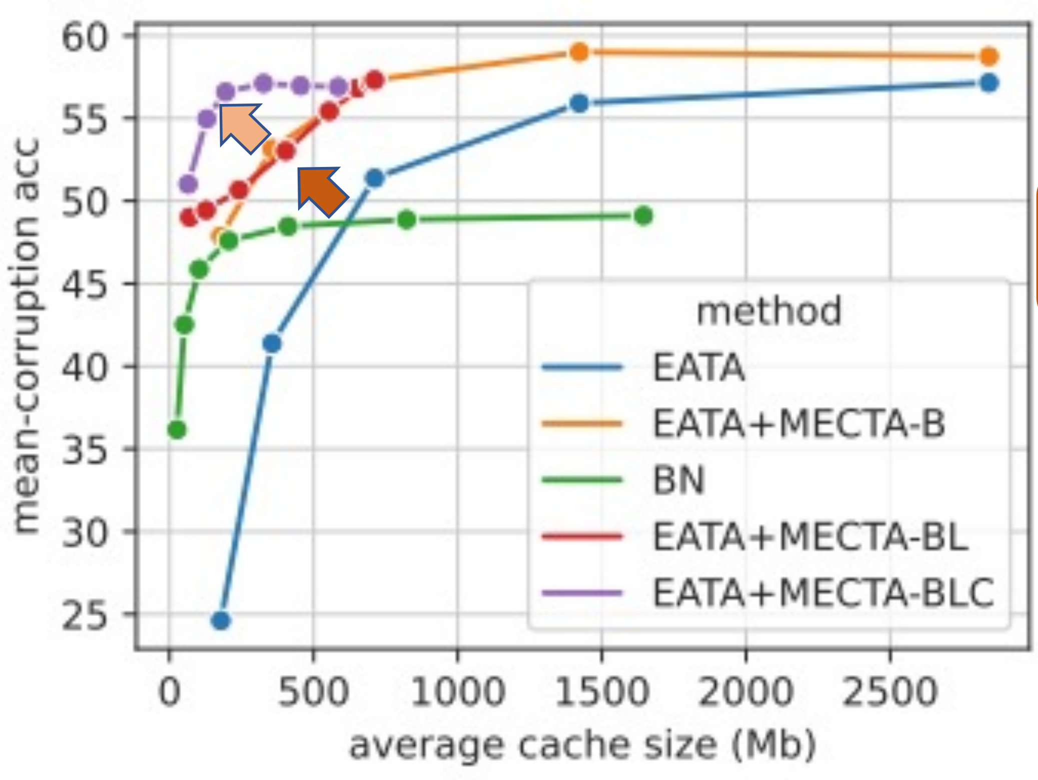
MECTA norm avoid forgetting of TENT.

| | | Noise | | | Blur | | | | Weather | | | | Digital | | | | | Acc. | Cache | | GFLOPs |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alg. | BS | Gauss. | Shot. | Impul. | Defoc. | Glass. | Motion | Zoom. | Snow | Frost | Fog | Bright. | Contr. | Elast. | Pixel. | JPEG | Orig. | Avg | Avg | Max | |
| | | *ImageNet-C* | | | | | | | | | | | | | | | | | | | |
| BN | 128 | 39.2 | 42.6 | 39.6 | 29.9 | 32.9 | 40.8 | 47.4 | 45.0 | 47.7 | 55.8 | 68.5 | 36.0 | 54.8 | 65.4 | 55.7 | 74.2 | 48.5 | 411 | 411 | 4.1 |
| TENT | 8 | 33.8 | 16.5 | 0.8 | 0.4 | 0.3 | 0.4 | 0.4 | 0.4 | 0.4 | 0.3 | 0.4 | 0.2 | 0.3 | 0.4 | 0.3 | 0.4 | 3.5 | 355 | 355 | 8.183 |
| +GC | 16 | 43.3 | 46.1 | 42.8 | 25.8 | 14.8 | 5.0 | 1.3 | 0.7 | 0.7 | 0.7 | 0.8 | 0.6 | 0.7 | 0.7 | 0.7 | 0.7 | 11.6 | 404 | 404 | 10.3 |
| +MECTA | 30 | **48.6** | **50.9** | 48.5 | 35.7 | **38.3** | 39.6 | **44.2** | **37.0** | **37.4** | **42.1** | **51.9** | **31.7** | **42.9** | **47.6** | **42.5** | **53.6** | **43.3** | 338 | 397 | 8.190 |
| EATA | 8 | 34.1 | 37.0 | 35.0 | 27.5 | 28.1 | 35.5 | 38.6 | 39.6 | 39.7 | 47.8 | 56.6 | 35.5 | 44.1 | 53.3 | 46.7 | 63.2 | 41.4 | 355 | 355 | 8.183 |
| +GC | 16 | 44.4 | 47.1 | 45.4 | 39.0 | 39.4 | 47.4 | 49.7 | 49.7 | 48.4 | 57.6 | 64.3 | 47.8 | 54.5 | 61.7 | 56.3 | 69.5 | 51.4 | 404 | 404 | 10.3 |
| +MECTA | 30 | **50.6** | **53.3** | **51.7** | **44.7** | **46.1** | **52.2** | **56.1** | **53.4** | **53.0** | **62.0** | **68.9** | **52.9** | **60.4** | **67.1** | **61.7** | **73.6** | **56.7** | 342 | 397 | 8.190 |

Better accuracy on all noise, reasonable computation load.

# Which component matters more?

Trade-off between robust accuracy and memory (cache size)




B, C are more effective while L is useful in low memory region.

# MECTA: Memory-Economic Continual Test-Time Model Adaptation

Improve **on-device** machine learning memory efficiency on **changing** environments.

- **New Problem**: We initiate the study on the **memory efficiency** of continual test-time adaptation (CTA), revealing the substantial obstacle in practice.
- **New Method**: We propose a novel method with a simple **plug-in MECTA Norm** layer that improves the memory efficiency of different CTA methods.
- **Better Memory-Robustness Trade-off**: Our method maintains comparable performance to full back-propagation methods while significantly reducing the dynamic and maximal cache overheads.

# Thank you!

paper

code