TTT++: When Does Self-Supervised Test-Time Training Fail or Thrive?

NeurIPS 2021

Yuejiang Liu,   Parth Kothari,   Bastien van Delft

Baptiste Bellot-Gurlet,   Taylor Mordan,   Alexandre Alahi

EPFL

Presented by: Daehoon Gwak

Feb 14, 2022

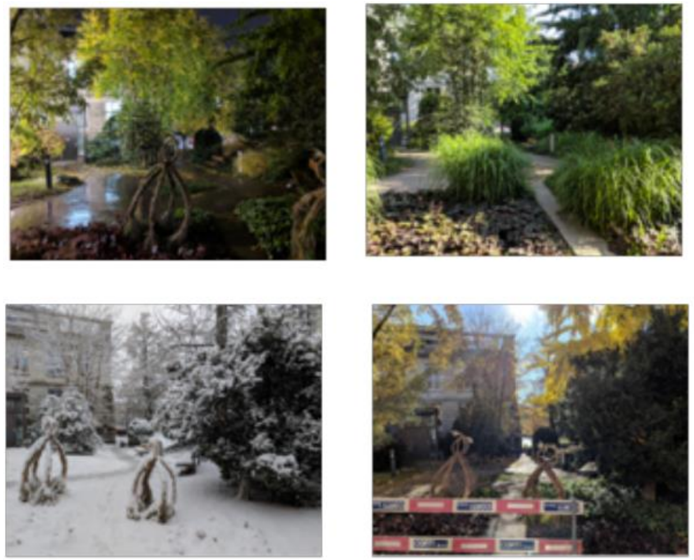
Vision seminar

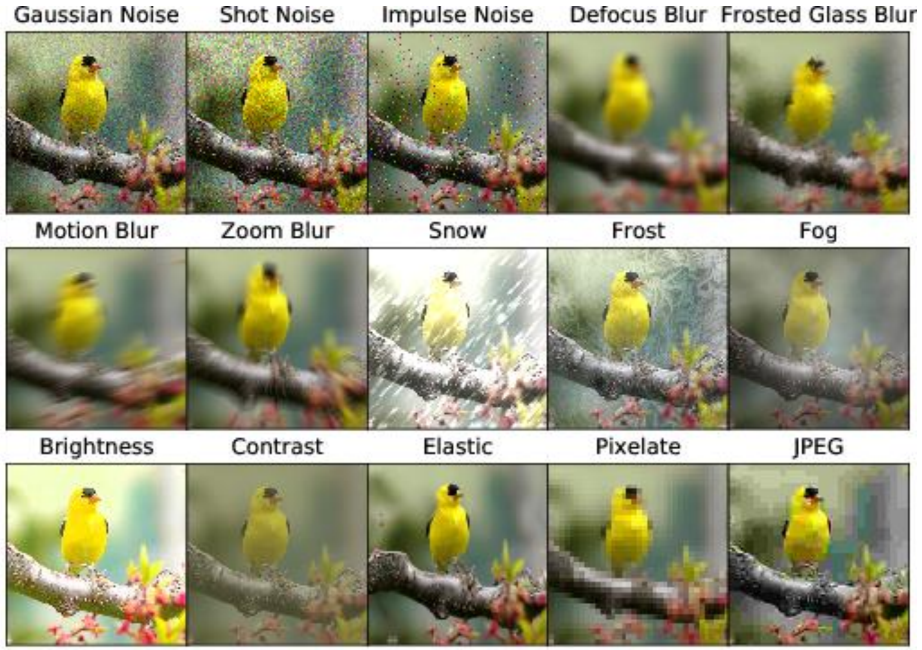
Index

- Introduction
- Background
- Methods
- Experiments
- Conclusion

Introduction

- The standard assumption in ML is that the data distribution at test time will match the training distribution.
- When this assumption does not hold the performance of standard ML methods can deteriorate significantly.
- However, in almost all practical applications, ML systems are regularly tested under unavoidable distribution shift.
- Domain Generalization / Domain Adaptation => Test-Time Adaptation





Introduction

- Domain Adaptation
  - In domain adaptation, the key assumption is that the target data is accessible during training, which allows the model to be adapted to the target domains.

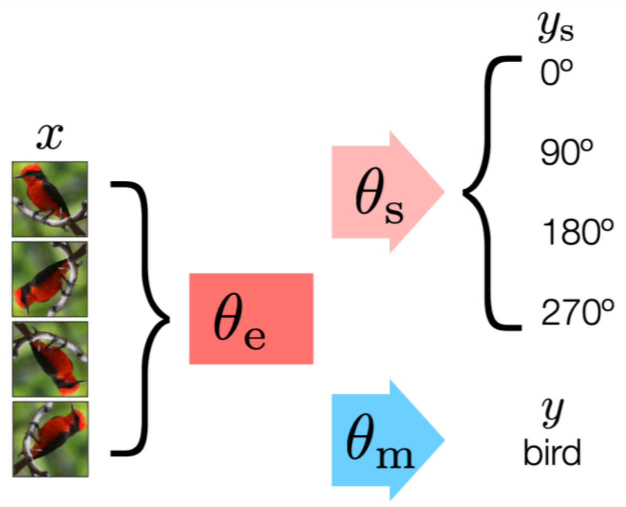
- Domain Generalization
  - Fixed decision boundary ⇒ Since these models are trained on source domains, there will always be an “adaptivity gap” when applying them to target domains without further adaptation

- Test-Time Adaptation
  - Given a trained model, how can we effectively adapt it from one domain to another on the fly, without access to training data and human annotations?

Background

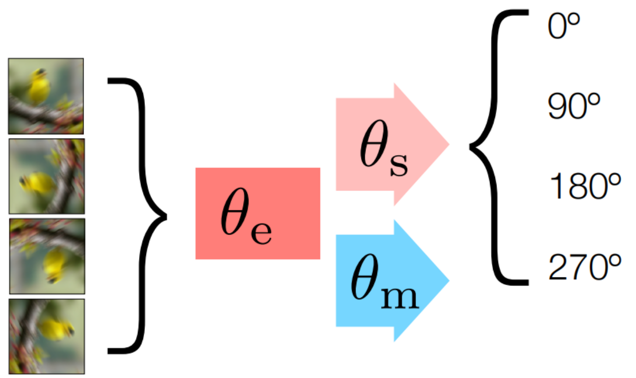
- **Test-Time Training(TTT)**
  - Test-Time Training with Self-Supervision for Generalization under Distribution Shifts (ICML 2020)

Step1: Training



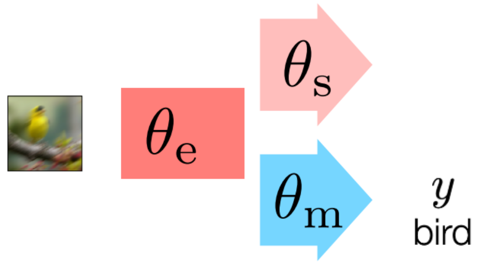
$$\min_{\theta_e,\theta_s,\theta_m} \mathbb{E}_P \left[\begin{array}{l} \ell_m(x, y; \theta_e, \theta_m) \\ + \ell_s(x, y_s; \theta_e, \theta_s) \end{array} \right]$$

Step2: Test-time training



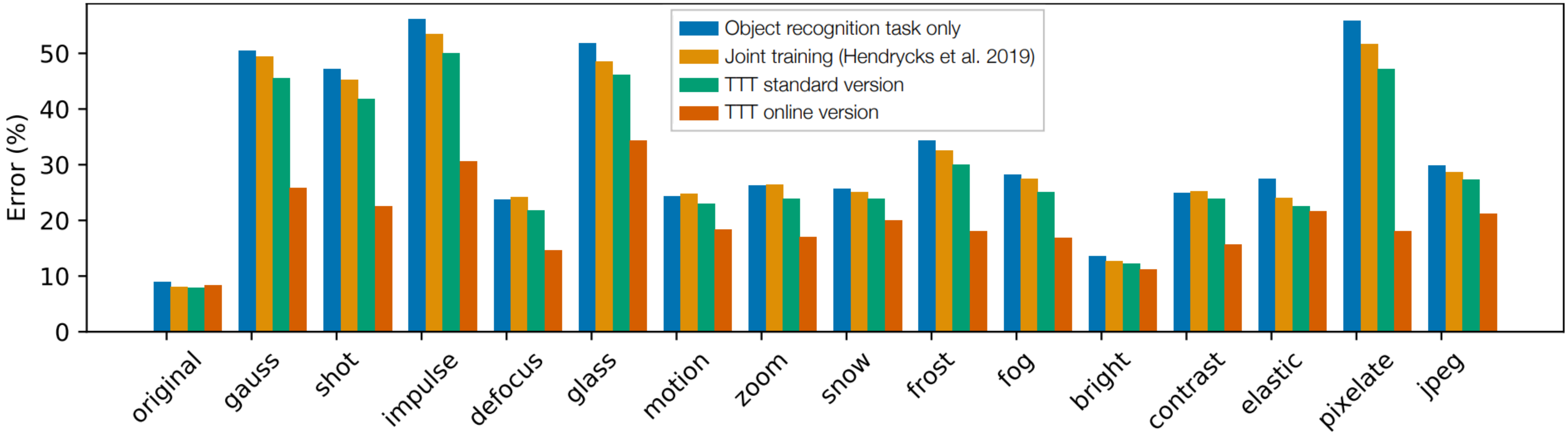
$$\min_{\theta_e,\theta_s} \left[\ell_s(x, y_s; \theta_e, \theta_s) \right]$$

Step3: Test-time prediction



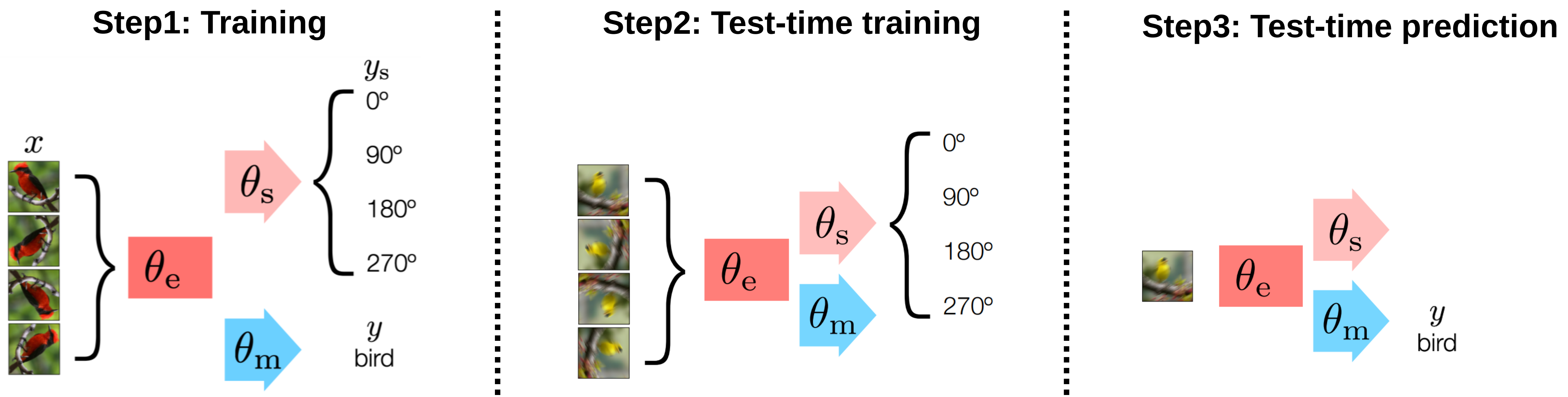
Background

- **Test-Time Training(TTT)**
  - Test-Time Training with Self-Supervision for Generalization under Distribution Shifts (ICML 2020)



Motivation

- **Failure case of TTT**
  - Without any constraints over the feature distribution, TTT may yield an updated encoder that overfits the self-supervised task, which deteriorates the performance on the main task.



$$\min_{\theta_\mathrm{e},\theta_\mathrm{s},\theta_\mathrm{m}} \mathbb{E}_P \left[\begin{array}{l} \ell_\mathrm{m}(x, y; \theta_\mathrm{e}, \theta_\mathrm{m}) \\ + \ell_s(x, y_\mathrm{s}; \theta_e, \theta_s) \end{array} \right]$$

$$\min_{\theta_\mathrm{e},\theta_\mathrm{s}} \left[\ell_s(x, y_\mathrm{s}; \theta_e, \theta_s) \right]$$

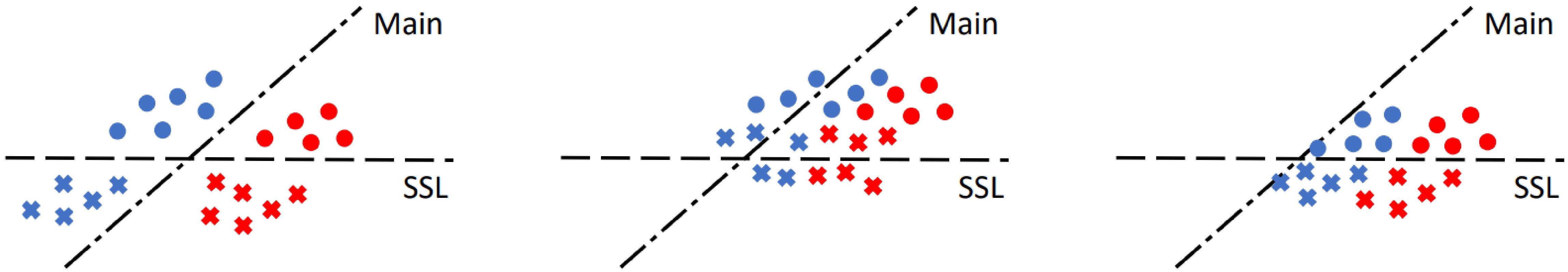
Motivation

- **Failure case of TTT**
  - Illustrative Example of Failures

    (a): The predictive model reaches high accuracy on both the main classification task, $i.e.$, separating red and blue data points, and the auxiliary self-supervised task, $i.e.$, separating circles and crosses.

    (b): Under significant distributional shift, test samples are encoded into a new subspace.

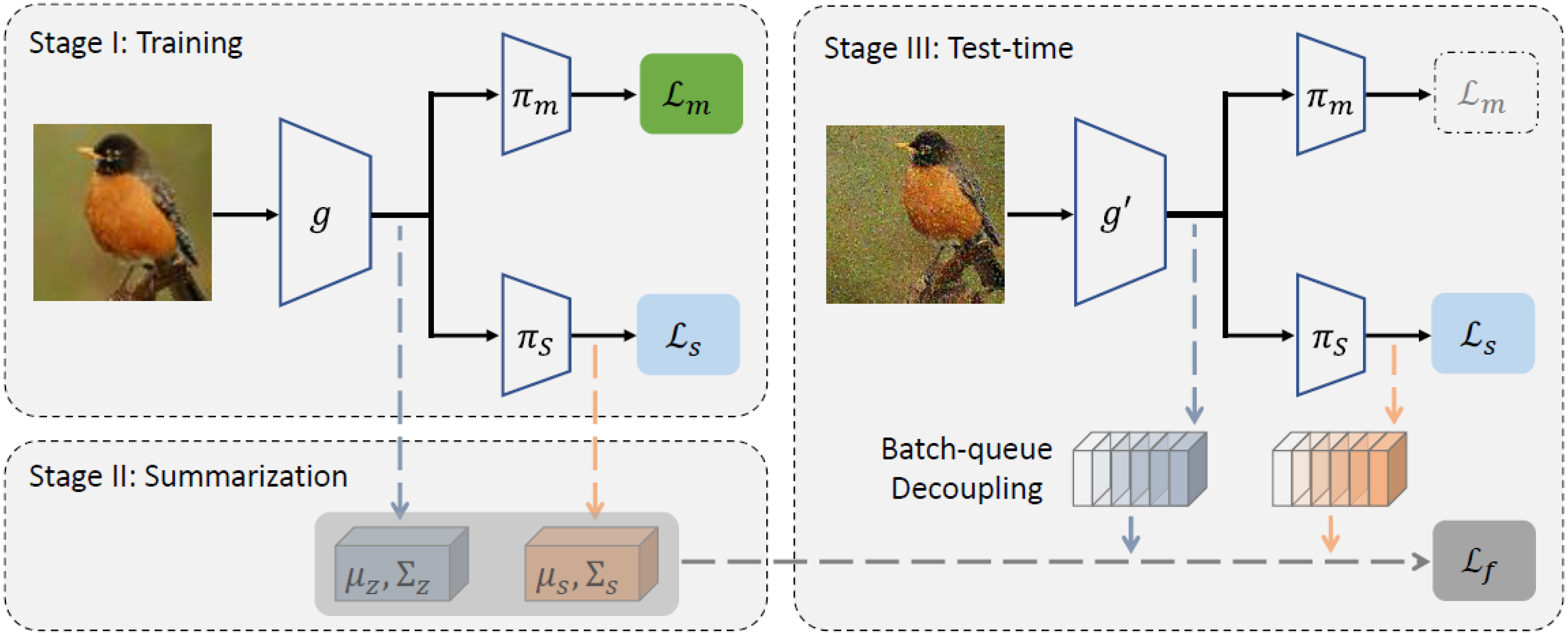
    (c): Failure case of TTT without any constraints over the distribution.



(a) training features (b) test features before TTT (c) test features after TTT

Method

- **Overview**
  - TTT + Regularization (Feature Alignment)



Method

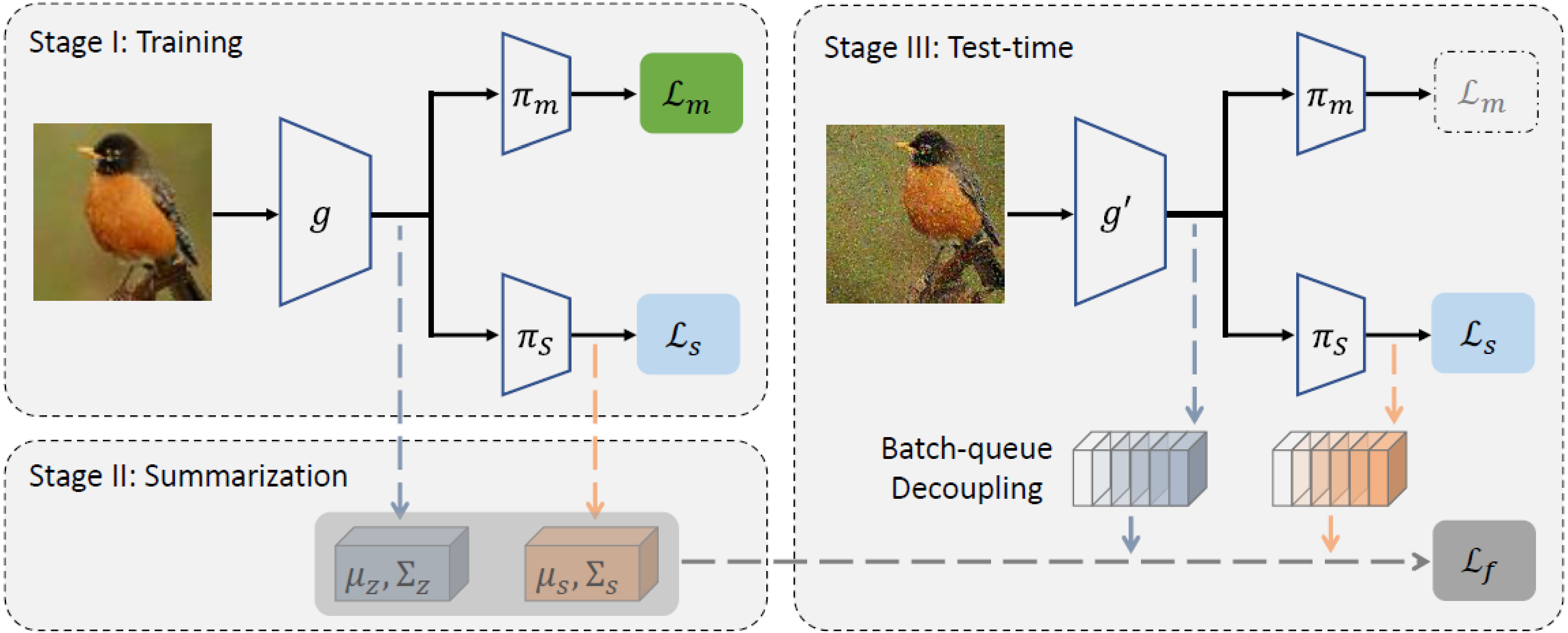
- **Online Feature Alignment**
  - Imposing a constraint over the feature space during TTT
    - Constraint: The feature distribution of test examples remains close to that in the training domain.

    - After training, set of feature vectors encoded from the training data $Z = \{z_1^T, \ldots, z_N^T\}$
    - Mean vector: $\mu_z = \frac{1}{N}\Sigma_{i=1}^{N} z_i$ and Covariance matrix: $\Sigma_Z = \frac{1}{N-1}(Z^T Z - (I^T Z)^T (I^T Z))$
    - This design choice allows us to summarize the distribution of training features in a compact format and store it as part of the model (consider adversarial training, MMD, ...)

  - At test time, we constrain the self-supervised adaptation by minimizing $\mathcal{L}_{f,z}$

$$\mathcal{L}_{f,z} = \|\mu_z - \mu'_z\|_2^2 + \|\Sigma_z - \Sigma'_z\|_F^2$$

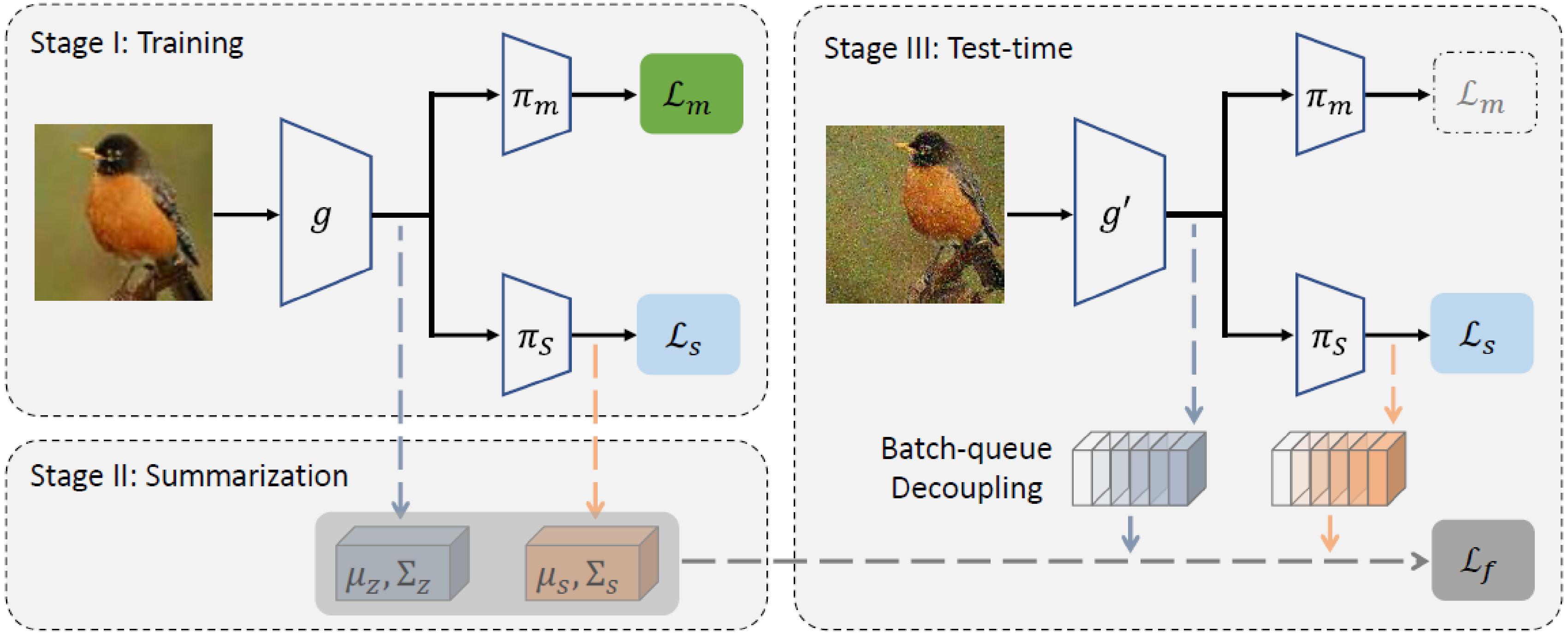
Method



$$\mathcal{L}_{TTT++} = \mathcal{L}_s + \lambda_z \mathcal{L}_{f,z} + \lambda_s \mathcal{L}_{f,s}$$

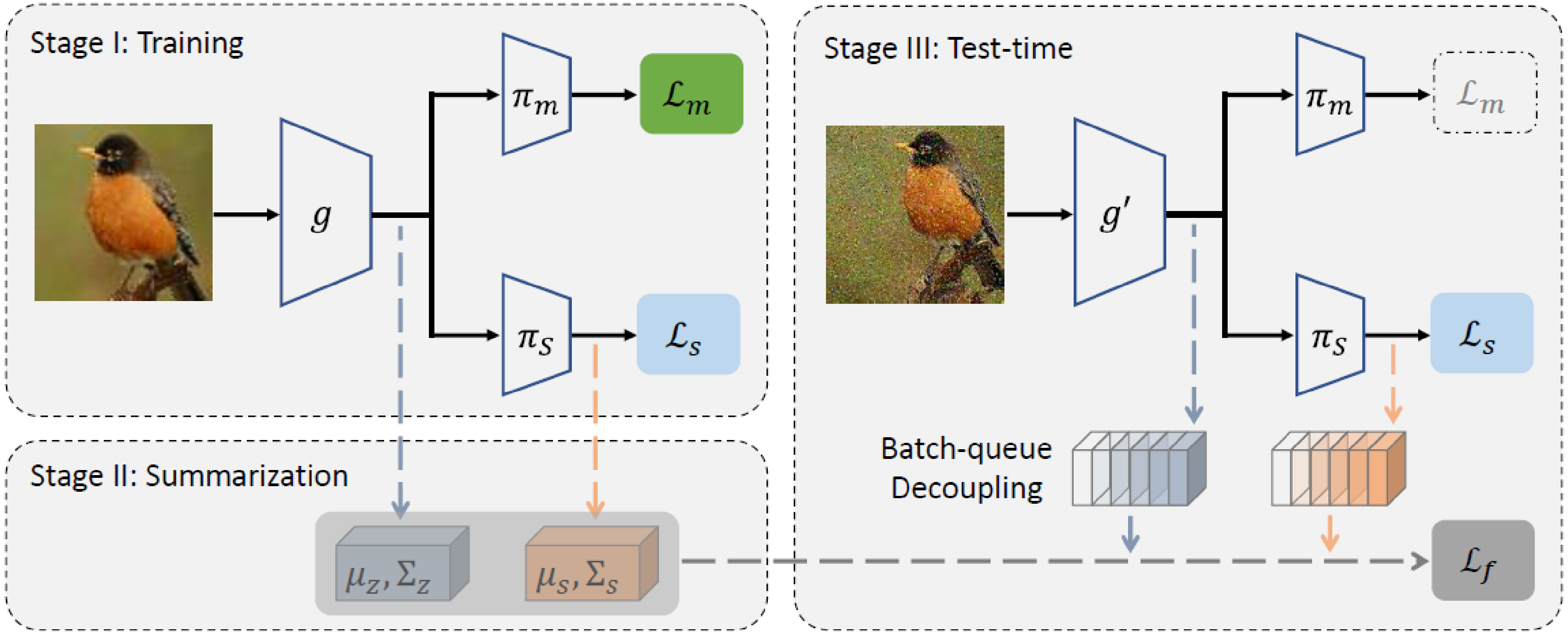
Method

- **Online Dynamic Queue**
  - One practical challenge for online feature alignment: scalability
  - Intuitively, a good estimate of moments of the entire distribution needs at least a handful of samples per class. (~1000 samples in the case of CIFAR-100)
  - However, the computational resources during deployment are often limited to accommodate such a large batch size in the test-time setting.



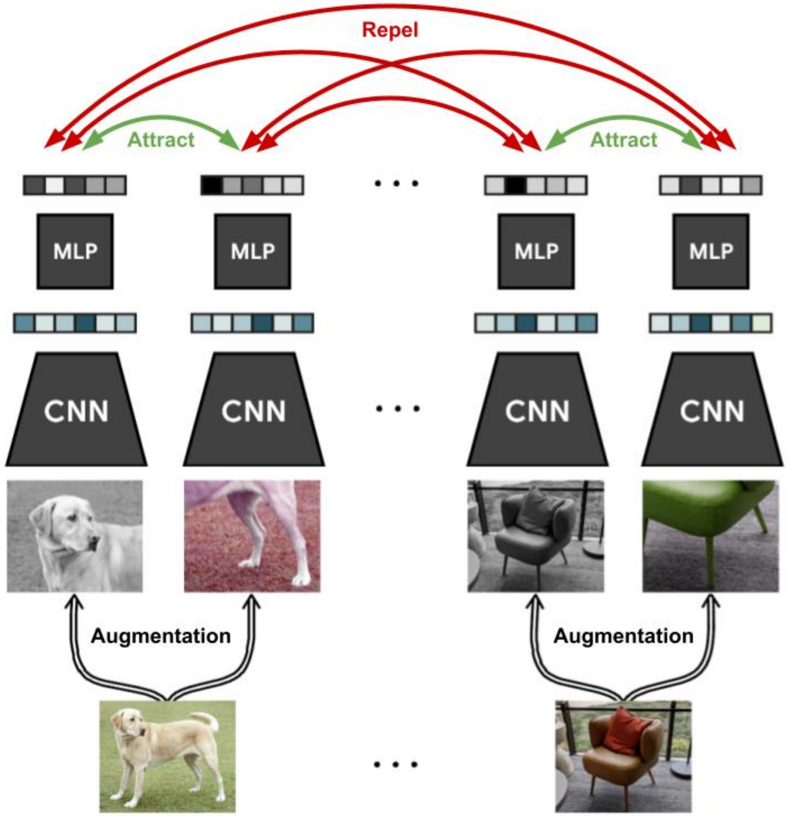
Method

- **Online Dynamic Queue**
  - The dynamic queue contains a few batches of feature vectors encoded at test time.
  - Progressively updating it by appending the latest mini-batch and popping out the oldest one



Method

- **TTT through Contrastive Learning**
  - A lower bound of the test accuracy on the main task is expected to grow rapidly when the SSL task gets closer to the main task.
  - Rotation Prediction ⇒ Contrastive Learning (SimCLR)

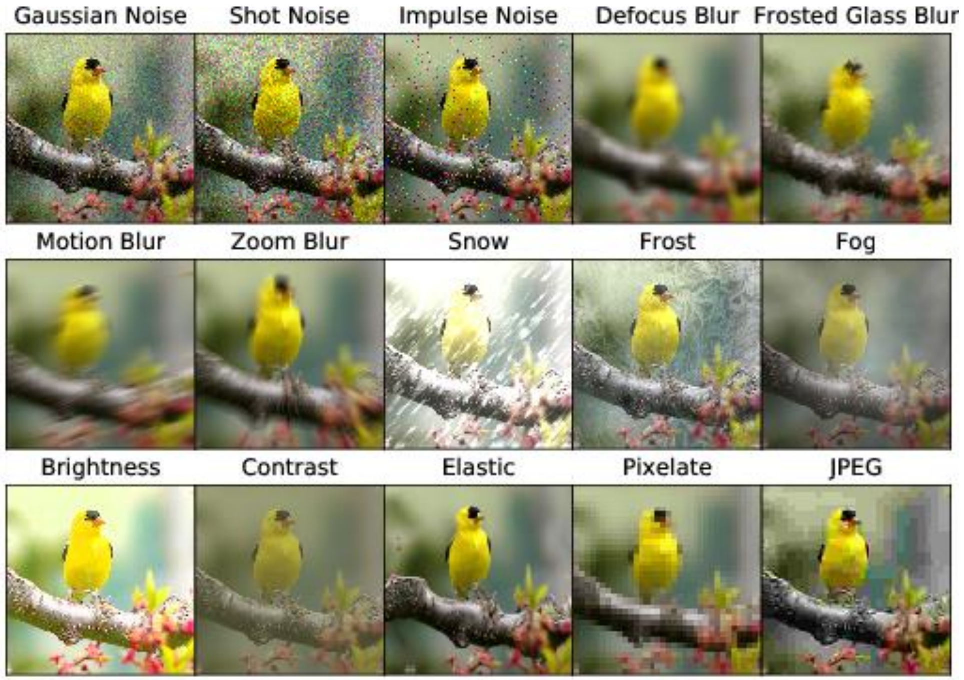


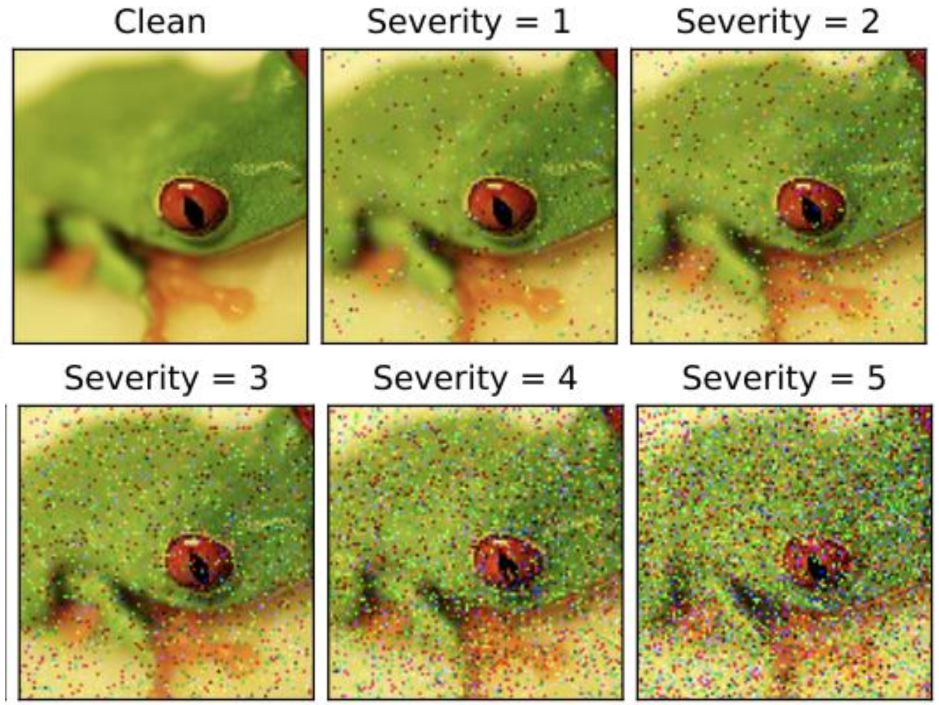
$$\mathcal{L}_s = -\log \frac{\exp(\mathrm{sim}(h_i, h_j)/\tau)}{\sum_{k=1}^{2B} \mathbb{1}_{k \neq i} \exp(\mathrm{sim}(h_i, h_k)/\tau)}$$

$$\mathrm{sim}(u, v) = u^T v / (\|u\| \|v\|)$$

Experiments

- **Common Image Corruption**
  - ResNet -50 on CIFAR 10-C and CIFAR 100-C datasets
    - 15 types of common image corruptions that vary in severity.
    - Batch size of 256 for test-time training
    - Dynamic queue containing 16 batches of feature vectors (CIFAR100-C)





Experiments

- **Common Image Corruption**
  - TTT-C: TTT + SimCLR
  - TFA: TTT + feature alignment

Table 1: Average classification error (%) on CIFAR10-C/100-C [1] and CIFAR10.1 [41]

Method	C10-C	C100-C	C10.1
Test	29.1	61.2	12.1
BN [42]	15.7	43.3	14.1
TTT-R [6]	14.3	40.4	11.0
SHOT [37]	14.7	38.1	11.1
TENT [8]	12.6	36.3	13.4
TFA (Ours)	11.9	35.8	12.1
TTT-C (Ours)	10.7	36.9	9.7
TTT++ (Ours)	**9.8**	**34.1**	**9.5**

Experiments

- **Common Image Corruption**
  - Effect of Batch-Queue Decoupling

	w/o queue			w/ queue			
Sample Size	64	128	256	64×2	64×4	64×8	64×16
Test Error	40.31	38.67	37.01	39.84	37.37	36.18	36.02

Experiments

- **Ablation study**
  - $\mathcal{L}_{TTT++} = \mathcal{L}_s + \lambda_z \mathcal{L}_{f,z} + \lambda_s \mathcal{L}_{f,s}$
  - $\mathcal{L}_{f,z} = \|\mu_z - \mu'_z\|_2^2 + \|\Sigma_z - \Sigma'_z\|_F^2$

TFA	brit	contr	defoc	elast	fog	frost	gauss	glass	impul	jpeg	motn	pixel	shot	snow	zoom
w/o $\mathcal{L}_{f,s}$	7.96	7.57	9.4	17.24	14.38	12.54	14.62	21.05	21.4	12.68	11.92	10.7	13.7	12.74	7.32
w/o $\mathcal{L}_{f,z}$	7.85	7.84	9.18	16.51	14.33	11.99	13.79	20.08	20.17	12.42	12.02	10.5	12.78	13.28	7.44
w/o Σ	7.49	7.56	9.62	18.62	19.22	12.72	16.02	25.07	25.17	13.43	13.63	11.22	15.04	15.11	7.77
w/o μ	**7.43**	**7.37**	8.90	15.92	12.98	11.57	13.46	19.27	**18.95**	11.87	11.11	9.97	12.81	11.76	7.04
Full	7.44	7.40	**8.89**	**15.73**	**12.82**	**11.49**	**12.94**	**18.46**	19.13	**11.66**	**10.77**	**9.93**	**12.67**	**11.73**	**7.03**

Conclusion

README.md

TTT++

This is an official implementation for the paper

TTT++: When Does Self-supervised Test-time Training Fail or Thrive? @ NeurIPS 2021
Yuejiang Liu, Parth Kothari, Bastien van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, Alexandre Alahi

TL;DR: Online Feature Alignment + Strong Self-supervised Learner → Robust Test-time Adaptation

Thank you!