Revisiting Realistic Test-Time Training: Sequential Inference and Adaptation by Anchored Clustering

Yongyi Su[1] Xun Xu[2] Kui Jia[13]

[1]South China University of Technology
[2]Institute for Infocomm Research
[3]Peng Cheng Laboratory

Project Page

Test-Time Training background:

- Test data features a drastic difference from the training data. (Directly testing on target data would give poor results)
- During test-time training, only source domain data is accessible due to storage overhead. (UDA methods don't work)
- Test data arrives in a stream and inference must be taken instantly. (SFDA needs train for multiple epochs in entire target domain)

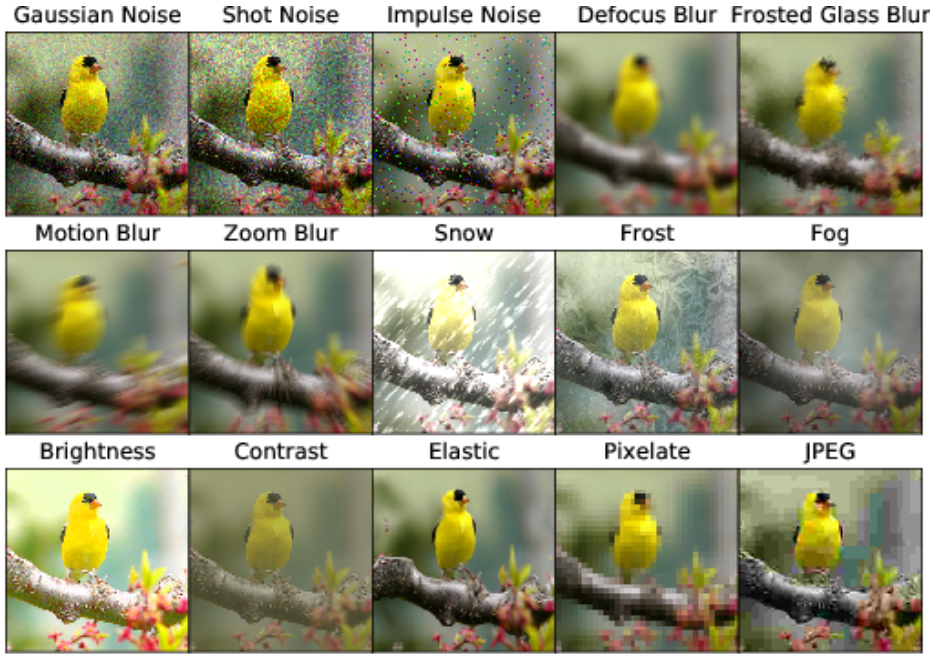

Figure 1. Target data with 15 kinds of corruptions.

Confused definitions of Test-Time Training:

- TENT & BN: make predictions instantly during test time.
- TTT-R: train an additional self-supervised branch for test-time training and make predictions instantly during test time.
- SHOT: train multiple epochs in target domain before making predictions.
- TTT++: train an additional self-supervised branch for test-time training and train multiple epochs in target domain before making predictions.

Different protocols cause unfair comparisons between Test-Time Training methods.

OUR CONTRIBUTION: Categorize TTT by two key factors

- Whether making predictions instantly?
  - **One-pass adaptation** (Denoted as "O" in tables of our paper)
  - **Multi-pass adaptation** (Denoted as "M")
- Whether modifying the training objective?
  - **Modification to source domain training loss** (Denoted as "Y")
  - **No modification to training objective** (Denoted as "N")

A realistic TTT setting, namely **sTTT**, belongs to "N-O" but doesn't restrict access to a light-weight information from the source domain.

Table 1: Comparison under different TTT protocols. Y/N indicates modifying source domain training objective or not. O/M indicate one pass or multiple passes test-time training. C10-C, C100-C and MN40-C refer to CIFAR10-C, CIFAR100-C and ModelNet40-C datasets respectively. All numbers indicate error rate in percentage.

Method	TTT Protocol	Assum. Strength	C10-C	C100-C	MN40-C
TEST	-	-	29.15	60.34	34.62
BN [13]	N-O	Weak	15.49	43.38	26.53
TENT [33]	N-O	Weak	14.27	40.72	26.38
T3A [14]	N-O	Weak	15.44	42.72	24.57
SHOT [21]	N-O	Weak	13.95	39.10	19.71
TTT++ [22]	N-O	Weak	13.69	40.32	-
TTAC (Ours)	N-O	Weak	**10.94**	36.64	22.30
TTAC+SHOT (Ours)	N-O	Weak	10.99	**36.39**	**19.21**
TTT++ [22]	Y-O	Medium	13.00	35.23	-
TTAC (Ours)	Y-O	Medium	**10.69**	**34.82**	-
BN [13]	N-M	Medium	15.70	43.30	26.49
TENT [33]	N-M	Medium	12.60	36.30	21.23
SHOT [21]	N-M	Medium	14.70	38.10	15.99
TTAC (Ours)	N-M	Medium	**9.42**	33.55	16.77
TTAC+SHOT (Ours)	N-M	Medium	9.54	**32.89**	**15.04**
TTT-R [29]	Y-M	Strong	14.30	40.40	-
TTT++ [22]	Y-M	Strong	9.80	34.10	-
TTAC (Ours)	Y-M	Strong	**8.52**	**30.57**	-

For "Y" methods, we can remove the self-supervised branch to adapt them to "N" protocol again, such as TTT++.

Test-Time Anchored Clustering Pipeline:

- Anchored Clustering for Test-Time Training
- Clustering through Pseudo Labeling
- Global Feature Alignment

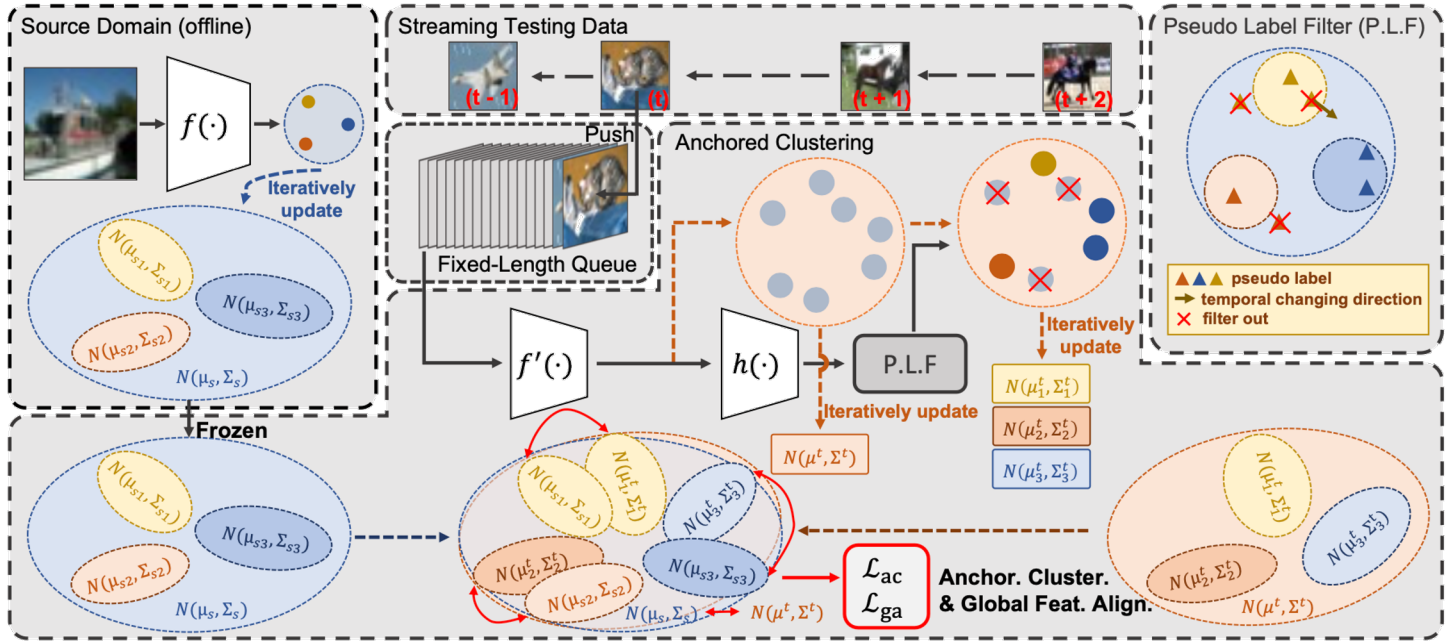

Figure 1: Overview of TTAC pipeline. i) In the source domain, we calculate category-wise and global statistics as anchors. ii) In the testing stage, samples are sequentially streamed and pushed into a fixed-length queue. Clusters in target domain are identified through anchored clustering with pseudo label filtering. Target clusters are then matched to the anchors in source domain to achieve test-time training.

Results:

- We categorize all TTT methods by two key factors, and make the fair comparisons between different TTT methods under the same TTT protocol.
- Our method TTAC consistently achieves the state-of-the-art performance under all categories of TTT protocols.
- TTAC improves about 3% under sTTT, the most realistic and weakest assumption.

Table 1: Comparison under different TTT protocols. Y/N indicates modifying source domain training objective or not. O/M indicate one pass or multiple passes test-time training. C10-C, C100-C and MN40-C refer to CIFAR10-C, CIFAR100-C and ModelNet40-C datasets respectively. All numbers indicate error rate in percentage.

Method	TTT Protocol	Assum. Strength	C10-C	C100-C	MN40-C
TEST	-	-	29.15	60.34	34.62
BN [13]	N-O	Weak	15.49	43.38	26.53
TENT [33]	N-O	Weak	14.27	40.72	26.38
T3A [14]	N-O	Weak	15.44	42.72	24.57
SHOT [21]	N-O	Weak	13.95	39.10	19.71
TTT++ [22]	N-O	Weak	13.69	40.32	-
TTAC (Ours)	N-O	Weak	**10.94**	36.64	22.30
TTAC+SHOT (Ours)	N-O	Weak	10.99	**36.39**	**19.21**
TTT++ [22]	Y-O	Medium	13.00	35.23	-
TTAC (Ours)	Y-O	Medium	**10.69**	**34.82**	-
BN [13]	N-M	Medium	15.70	43.30	26.49
TENT [33]	N-M	Medium	12.60	36.30	21.23
SHOT [21]	N-M	Medium	14.70	38.10	15.99
TTAC (Ours)	N-M	Medium	**9.42**	33.55	16.77
TTAC+SHOT (Ours)	N-M	Medium	9.54	**32.89**	**15.04**
TTT-R [29]	Y-M	Strong	14.30	40.40	-
TTT++ [22]	Y-M	Strong	9.80	34.10	-
TTAC (Ours)	Y-M	Strong	**8.52**	**30.57**	-

Conclusion:

- Confused definitions for TTT. We provide a categorization of TTT protocols by two key factors. Comparison of TTT methods is now fair within each protocol.

- We advocate a realistic TTT setting, namely sTTT. To improve test-time feature learning, we propose TTAC by matching the statistics of the target clusters to the source ones.

- The proposed method is complementary to existing TTT method and is demonstrated on six TTT datasets, achieving the state-of-the-art performance under all categories of TTT protocols.

More information about this work can be obtained from here!

Project Page

Arxiv Link