南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

Unsupervised Domain Adaptation by Backpropagation

Yaroslav Ganin, Victor Lempitsky

ICML 2015

Background

- leaps in performance come only when a large amount of labeled training data is available.But for problems lacking labeled data, but it suffers from the shift in data distribution from the actual data encountered at “test time”.

- Learning a discriminative classifier or other predictor in the presence of a shift between training and test distributions is known as domain adaptation (DA)

- The appeal of the domain adaptation approaches is the ability to learn a mapping between domains in the situation when the target domain data are either fully unlabeled (unsupervised domain annotation) or have few labeled samples

- authors focus on the harder unsupervised case

Methods

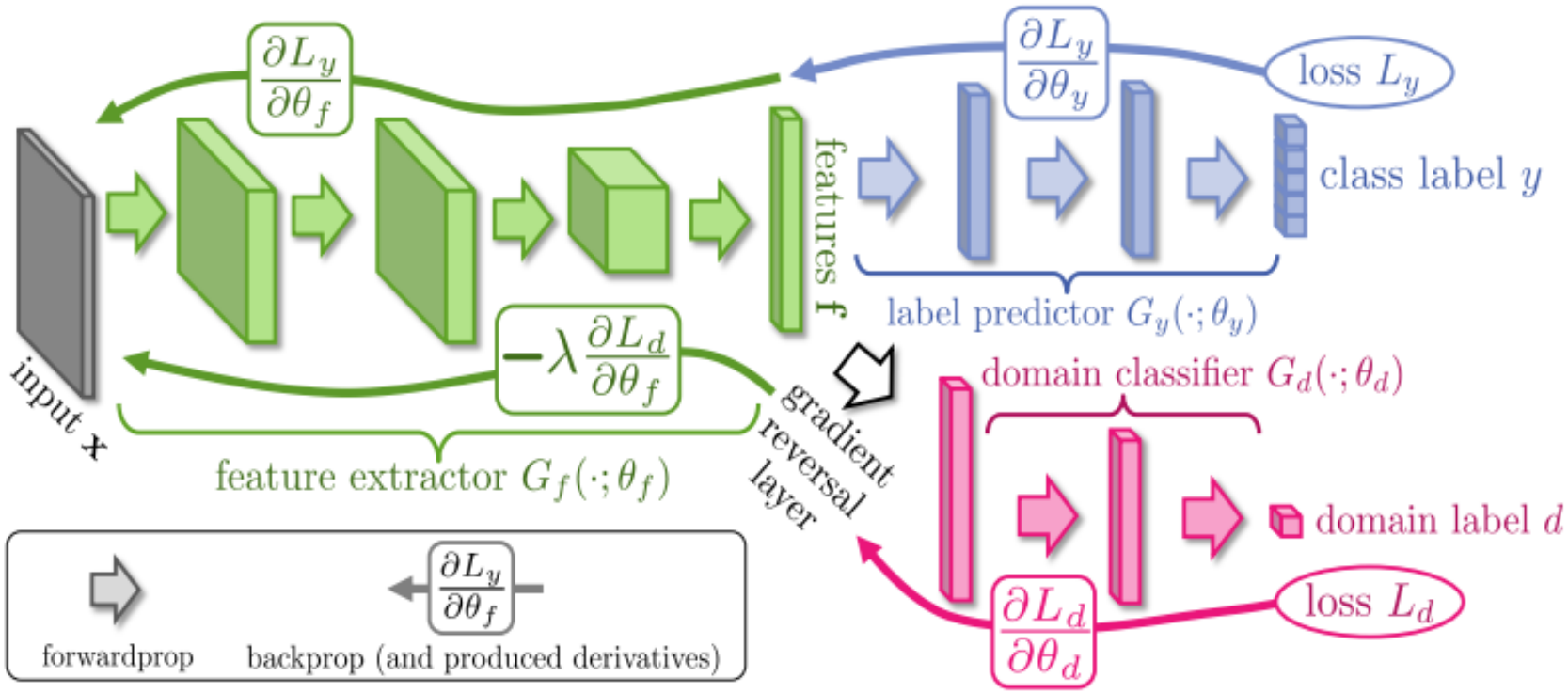

Figure 1. The **proposed architecture** includes a deep *feature extractor* (green) and a deep *label predictor* (blue), which together form a standard feed-forward architecture. Unsupervised domain adaptation is achieved by adding a *domain classifier* (red) connected to the feature extractor via a *gradient reversal layer* that multiplies the gradient by a certain negative constant during the backpropagation-based training. Otherwise, the training proceeds in a standard way and minimizes the label prediction loss (for source examples) and the domain classification loss (for all samples). Gradient reversal ensures that the feature distributions over the two domains are made similar (as indistinguishable as possible for the domain classifier), thus resulting in the domain-invariant features.

Loss计算

$$E(\theta_f, \theta_y, \theta_d) = \sum_{\substack{i=1..N \\ d_i=0}} L_y\left(G_y(G_f(\mathbf{x}_i;\theta_f);\theta_y), y_i\right) - \lambda \sum_{i=1..N} L_d\left(G_d(G_f(\mathbf{x}_i;\theta_f);\theta_d), y_i\right) = \sum_{\substack{i=1..N \\ d_i=0}} L_y^i(\theta_f, \theta_y) - \lambda \sum_{i=1..N} L_d^i(\theta_f, \theta_d) \quad (1)$$

Here, $L_y(\cdot,\cdot)$ is the loss for label prediction (e.g. multinomial), $L_d(\cdot,\cdot)$ is the loss for the domain classification (e.g. logistic), while L_y^i and L_d^i denote the corresponding loss functions evaluated at the i-th training example. Based on our idea, we are seeking the parameters $\hat{\theta}_f, \hat{\theta}_y, \hat{\theta}_d$ that deliver a saddle point of the functional (1):

$$(\hat{\theta}_f, \hat{\theta}_y) = \arg\min_{\theta_f, \theta_y} E(\theta_f, \theta_y, \hat{\theta}_d) \quad (2)$$

$$\hat{\theta}_d = \arg\max_{\theta_d} E(\hat{\theta}_f, \hat{\theta}_y, \theta_d)\,. \quad (3)$$

领域分类器参数θd：这些参数负责最小化领域分类损失。由于领域分类损失在总损失函数中通常带有负号（表示最大化该损失的相反数，即最小化它），这意味着领域分类器试图区分来自不同领域的样本。在领域自适应的上下文中，希望特征表示能够不依赖于数据的领域来源，实际上是在帮助调整特征映射以去除领域特异性。

标签预测器参数θy：这些参数负责最小化标签预测损失，即提高模型对目标任务（如分类、回归等）的预测准确性。这是通过确保特征表示中包含足够的信息来区分不同类别的样本来实现的。

特征映射参数θf：这些参数在优化过程中扮演了关键角色。它们需要同时满足两个目标：最小化标签预测损失和最大化领域分类损失

超参数λ：这个参数用于控制上述两个目标之间的权衡。调整λ的值可以改变模型在追求判别性和领域不变性之间的偏好。

Methods

Optimization with backpropagation

$$\theta_f \longleftarrow \theta_f - \mu\left(\frac{\partial L_y^i}{\partial \theta_f} - \lambda\frac{\partial L_d^i}{\partial \theta_f}\right) \tag{4}$$

$$\theta_y \longleftarrow \theta_y - \mu\frac{\partial L_y^i}{\partial \theta_y} \tag{5}$$

$$\theta_d \longleftarrow \theta_d - \mu\frac{\partial L_d^i}{\partial \theta_d} \tag{6}$$

where μ is the learning rate (which can vary over time). The updates (4)-(6) are very similar to stochastic gradient descent (SGD) updates for a feed-forward deep model that comprises feature extractor fed into the label predictor and into the domain classifier. The difference is the $-\lambda$ factor in (4) (the difference is important, as without such factor,

The GRL as defined above is inserted between the feature extractor and the domain classifier, resulting in the architecture depicted in Figure 1. As the backpropagation process passes through the GRL, the partial derivatives of the loss that is downstream the GRL (i.e. L_d) w.r.t. the layer parameters that are upstream the GRL (i.e. θ_f) get multiplied by $-\lambda$, i.e. $\frac{\partial L_d}{\partial \theta_f}$ is effectively replaced with $-\lambda\frac{\partial L_d}{\partial \theta_f}$. Therefore, running SGD in the resulting model implements the updates (4)-(6) and converges to a saddle point of (1). Mathematically, we can formally treat the gradient reversal layer as a "pseudo-function" $R_\lambda(\mathbf{x})$ defined by two (incompatible) equations describing its forward- and backpropagation behaviour:

$$R_\lambda(\mathbf{x}) = \mathbf{x} \tag{7}$$

$$\frac{dR_\lambda}{d\mathbf{x}} = -\lambda\mathbf{I} \tag{8}$$

梯度反转层（GRL）的作用就是通过在特征提取器和域分类器之间插入这样一个层，使得在训练过程中，特征提取器被“强制”学习到那些能够混淆域分类器的特征，即让域分类器难以区分特征是来自源域还是目标域。这样，学到的特征就更加具有领域不变性，有利于提升模型在目标域上的性能。

Methods

Relation to $H_\Delta H$-distance

$$d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S},\mathcal{T}) = 2 \sup_{h_1,h_2\in\mathcal{H}} |P_{\mathbf{f}\sim\mathcal{S}}[h_1(\mathbf{f}) \neq h_2(\mathbf{f})] - P_{\mathbf{f}\sim\mathcal{T}}[h_1(\mathbf{f}) \neq h_2(\mathbf{f})]| \quad (10)$$

defines a discrepancy distance between two distributions $\mathcal{S}$ and $\mathcal{T}$ w.r.t. a hypothesis set $\mathcal{H}$. Using this notion one can obtain a probabilistic bound (Ben-David et al., 2010) on the performance $\varepsilon_\mathcal{T}(h)$ of some classifier h from $\mathcal{T}$ evaluated on the target domain given its performance $\varepsilon_\mathcal{S}(h)$ on the source domain:

$$\varepsilon_\mathcal{T}(h) \leq \varepsilon_\mathcal{S}(h) + \frac{1}{2} d_{\mathcal{H}\Delta\mathcal{H}}(\mathcal{S},\mathcal{T}) + C\,, \quad (11)$$

where $\mathcal{S}$ and $\mathcal{T}$ are source and target distributions respectively, and C does not depend on particular h.
Consider fixed $\mathcal{S}$ and $\mathcal{T}$ over the representation space produced by the feature extractor G_f and a family of label predictors $\mathcal{H}_p$. We assume that the family of domain classifiers $\mathcal{H}_d$ is rich enough to contain the symmetric difference hypothesis set of $\mathcal{H}_p$:

$$\mathcal{H}_p\Delta\mathcal{H}_p = \{h \,|\, h = h_1 \oplus h_2\,,\ h_1, h_2 \in \mathcal{H}_p\}\,. \quad (12)$$

$$\begin{aligned} &d_{\mathcal{H}_p\Delta\mathcal{H}_p}(\mathcal{S},\mathcal{T}) = \\ &= 2 \sup_{h\in\mathcal{H}_p\Delta\mathcal{H}_p} |P_{\mathbf{f}\sim\mathcal{S}}[h(\mathbf{f}) = 1] - P_{\mathbf{f}\sim\mathcal{T}}[h(\mathbf{f}) = 1]| \leq \\ &\leq 2 \sup_{h\in\mathcal{H}_d} |P_{\mathbf{f}\sim\mathcal{S}}[h(\mathbf{f}) = 1] - P_{\mathbf{f}\sim\mathcal{T}}[h(\mathbf{f}) = 1]| = \\ &= 2 \sup_{h\in\mathcal{H}_d} |1 - \alpha(h)| = 2 \sup_{h\in\mathcal{H}_d} [\alpha(h) - 1] \end{aligned} \quad (13)$$

$d_{H\Delta H}(S,T)$（即源域和目标域之间的差异），就可以有效地减小目标域上的错误率，从而提高领域自适应的性能,提高模型在目标域上的泛化能力.

Experiments

Figure 2. Examples of domain pairs used in the experiments. See Section 4.1 for details.

METHOD	SOURCE	MNIST	SYN NUMBERS	SVHN	SYN SIGNS
	TARGET	MNIST-M	SVHN	MNIST	GTSRB
SOURCE ONLY		.5749	.8665	.5919	.7400
SA (FERNANDO ET AL., 2013)		.6078 (7.9%)	.8672 (1.3%)	.6157 (5.9%)	.7635 (9.1%)
PROPOSED APPROACH		**.8149** (57.9%)	**.9048** (66.1%)	**.7107** (29.3%)	**.8866** (56.7%)
TRAIN ON TARGET		.9891	.9244	.9951	.9987

Table 1. Classification accuracies for digit image classifications for different source and target domains. MNIST-M corresponds to difference-blended digits over non-uniform background. The first row corresponds to the lower performance bound (i.e. if no adaptation is performed). The last row corresponds to training on the target domain data with known class labels (upper bound on the DA performance). For each of the two DA methods (ours and (Fernando et al., 2013)) we show how much of the gap between the lower and the upper bounds was covered (in brackets). For all five cases, our approach outperforms (Fernando et al., 2013) considerably, and covers a big portion of the gap.

南京航空航天大學
NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

A Comprehensive Survey on Test-Time Adaptation under Distribution Shifts

Jian Liang, Member, IEEE, Ran He, Senior Member, IEEE, and Tieniu Tan, Fellow, IEEE

IEEE 2023

Introduction

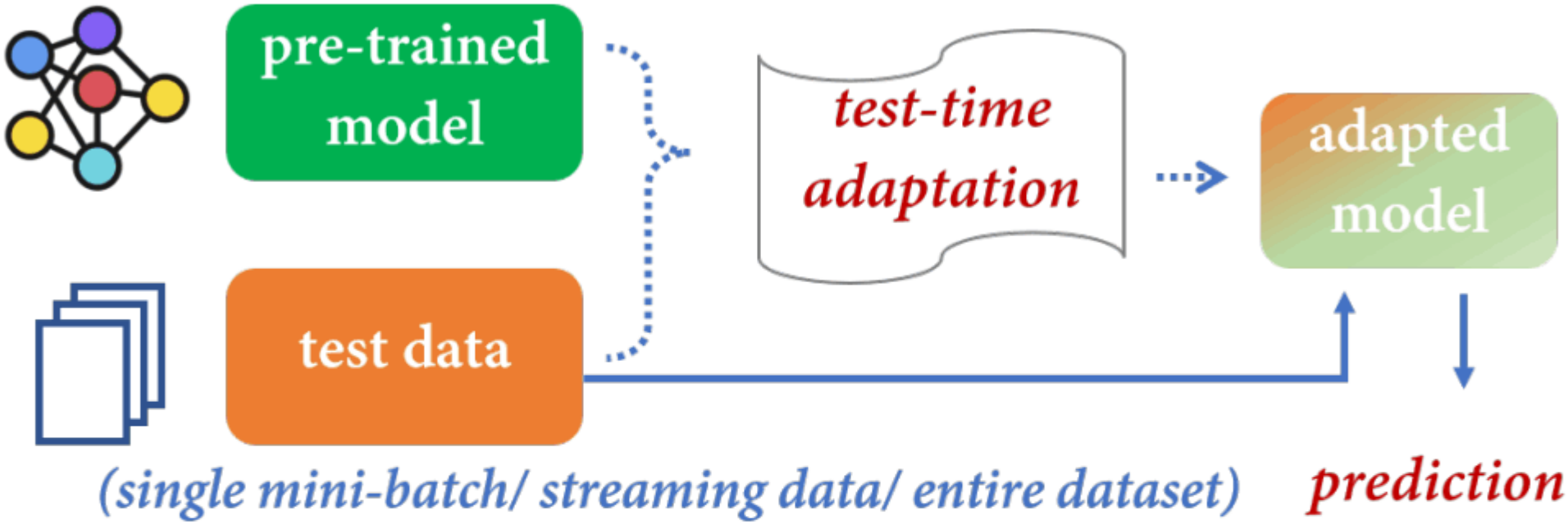

Fig. 1. The test-time adaptation (TTA) paradigm aims to adapt the pre-trained model to various types of unlabeled test data, including single mini-batch, streaming data, or an entire dataset, before making predictions. During the adaptation process, either the model or the input

Introduction

南京航空航天大學 NANJING UNIVERSITY OF AERONAUTICS AND ASTRONAUTICS

Based on the characteristics of the test data

- Firstly, test-time domain adaptation(TTDA), also known as source-free domain adaptation(SFDA) utilizes all m test batches for multi-epoch adaptation before generating final prediction

- Secondly,test-time batch adaptation(TTBA) individually adapts the pre-trained model to one 2 or a few instances. That is to say, the predictions of each mini-batch are independent of the predictions for the other mini-batches.

- Thirdly , online test-time adaptation(OTTA) adapts the pre-trained model to the target data $\{b_1, \cdot\cdot\cdot, b_m\}$ in an online manner, where each mini-batch can only be observed once.

Source-Free Domain Adaptation

Definition 2 (Source-free Domain Adaptation, SFDA). Given a well-trained classifier $f_S : \mathcal{X}_S \to \mathcal{Y}_S$ on the source domain $\mathcal{D}_S$ and an unlabeled target domain $\mathcal{D}_T$, *source-free domain adaptation* aims to leverage the labeled knowledge implied in f_S to infer labels of all the samples in $\mathcal{D}_T$, in a transductive learning [26] manner. Note that, all test data (target data) are required to be seen during adaptation.

So far as we know, the term *source-free domain adaptation* is first proposed by Nelakurthi *et al.* [61], where they try to leverage the noisy predictions of an off-the-shelf classifier and a few labeled examples from the target domain, in order to obtain better predictions for all the unlabeled target samples. The definition here covers [61] as a special case, where the classifier f_S is not accessible but provides the predictions of target data $\{f_S(x)|x \in \mathcal{D}_T\}$.

SFDA 的目标是利用 f_S中隐含的标记知识来推断目标域 D_T 中所有样本的标签。 f_S 是一个在源域 D_S 上训练好的分类器，它能够将源域的特征 X_S 映射到标签 Y_S。然而，在 SFDA 的场景中，不再有直接访问源域数据 D_S 的权限，只能利用 f_S本身（可能是其模型参数、预测结果或其他形式的输出）来进行适应。

1、Pseudo-labeling

To adapt a pre-trained model to an unlabeled target domain, a majority of SFDA methods take inspiration from the semi-supervised learning (SSL) field [146] and employ various prevalent SSL techniques tailored for unlabeled data during adaptation. A simple yet effective technique, pseudo-labeling [101], aims to assign a class label $\hat{y} \in \mathbb{R}^C$ for each unlabeled sample x in $\mathcal{X}_t$ and optimize the following supervised learning objective to guide the learning process,

$$\min_{\theta} \mathbb{E}_{\{x,\hat{y}\}\in\mathcal{D}_t} \; w_{pl}(x) \cdot d_{pl}(\hat{y}, p(y|x;\theta)), \tag{1}$$

where $w_{pl}(x) \in \mathbb{R}$ denotes the weight associated with each pseudo-labeled sample $\{x, \hat{y}\}$, and $d_{pl}(\cdot)$ denotes the divergence between the predicted label probability distribution and the pseudo label probability $\hat{y}$, *e.g.*,

Since the pseudo labels of target data are inevitably inaccurate under domain shift, there exist three different solutions:

(1) improving the quality of pseudo labels via denoising;

(2) filtering out inaccurate pseudo labels with wpl(·);

(3) developing a robust divergence measure dpl(·, ·) for pseudo-labeling.

（1）通过去噪提高伪标签的质量;
（2）用WPL（·）过滤掉不准确的伪标签;
（3）开发一种用于伪标记的鲁棒散度量dpl（·，·）。

2、Consistency Training

As a prevailing strategy in recent semi-supervised learning literature [99], [232], consistency regularization is primarily built on the smoothness assumption or the manifold assumption, which aims to enforce consistent network predictions or features under variations in the input data space or the model parameter space. Besides, another line of consistency training methods tries to match the statistics of different domains even without the source data. In the following, we review different consistency regularizations under data and model variations together with other consistency-based distribution matching methods.

平滑假设（Smoothness Assumption）：该假设认为如果两个数据点在输入空间中彼此接近，那么它们的输出也应该接近。在一致性正则化中，这通常通过向未标记的数据点添加微小的扰动（如数据增强）并强制模型对这些扰动后的数据点的预测与原始数据点的预测保持一致来实现。

流形假设（Manifold Assumption）：该假设认为高维数据通常位于一个低维的流形上。在流形上，相似的数据点（即位于流形上相近位置的数据点）应该具有相似的输出。一致性正则化通过鼓励模型在流形上保持预测的一致性来利用这一假设。

3、Clustering-based Training

Entropy minimization. To encourage confident predictions for unlabeled target data, ASFA [135] borrows robust measures from information theory and minimizes the following α-Tsallis entropy [280],

$$\mathcal{L}_{tsa} = \frac{1}{n_t}\sum_{i=1}^{n_t}\frac{1}{\alpha-1}[1-\sum_{c=1}^{C}p_\theta(y_c|x_i)^\alpha], \qquad (10)$$

where $\alpha > 0$ is called the entropic index. Note that, when α approaches 1, the Tsallis entropy exactly recovers the standard Shannon entropy in $\mathcal{H}(p_\theta(y|x_i)) = \sum_c p_\theta(y_c|x_i)\log p_\theta(y_c|x_i)$. In practice, the conditional Shannon entropy $\mathcal{H}(p_\theta(y|x))$ has been widely used in SFDA

决策边界存在于低密度的区域。基于聚类的SFDA方法致力于减少网络预测的不确定性或者鼓励目标域特征聚类。

4、Source Distribution Estimation

Another favored family of SFDA approaches compensates for the absence of source data by inferring data from the pre-trained model, which turns the challenging SFDA problem into a well-studied DA problem. Existing distribution estimation methods could be divided into three categories: data generation from random noises [12], [305], [306], data translation from target samples [157], [288], [307], and data selection from the target domain [98], [187], [274].

SFDA转换为一个域适应

5、Self-supervised Learning

Self-supervised learning is another learning paradigm tailored to learn feature representation from unlabeled data based on auxiliary prediction tasks (pretext tasks) [87], [90], [91], [92], [93]. As mentioned above, the centroid-based pseudo labels are similar to the learning manner of DeepCluster [90]. Inspired by rotation prediction [87], SHOT++ [98] further comes up with a relative rotation prediction task and introduces a 4-way classification head in addition to the C-dimensional semantic classification head during adaptation, which has been adopted by later methods [166], [171], [217]. Besides, OnTA [264] and CluP [247] exploit the self-supervised learning frameworks [91], [92] for learning discriminative features as initialization, respectively. TTT++ [272] learns an extra self-supervised branch using contrastive learning [93] in the source model, which facilitates the adaptation in the target domain with the same objective. Recently, StickerDA [145] designs three self-supervision objectives (*i.e.*, sticker location, sticker rotation, and sticker classification) and optimizes the sticker intervention-based pretext task with the auxiliary classification head in both the source training and target adaptation phases.

通过构建任务来从无标签的数据中学习知识

Test-Time Batch Adaptation

Definition 4 (Test-Time Batch Adaptation, TTBA). Given a classifier $f_{\mathcal{S}}$ learned on the source domain $\mathcal{D}_{\mathcal{S}}$, and a mini-batch of unlabeled target instances $\{x_t^1, x_t^2, \cdots, x_t^B\}$ from $\mathcal{D}_{\mathcal{T}}$ under distribution shift ($B \geq 1$), *test-time batch adaptation* aims to leverage the labeled knowledge implied in $f_{\mathcal{S}}$ to infer the label of each instance at the same time.

It is important to acknowledge that the inference of each instance is not independent, but rather influenced by the other instances in the mini-batch. Test-Time Batch Adaptation (TTBA) can be considered a form of SFDA [7] when the batch size B is sufficiently large. Conversely, when the batch size B is equal to 1, TTBA degrades to TTIA [8]. Typically, these schemes assume no access to the source data or the ground-truth labels of data on the target distribution. In the following, we provide a taxonomy of TTBA (including TTIA) algorithms, as well as the learning scenarios.

1、Batch Normalization Calibration

Normalization layers (*e.g.*, batch normalization [378] and layer normalization [379]) are considered essential components of modern neural networks. For example, a batch normalization (BN) layer calculates the mean and variance for each activation over the training data $\mathcal{X}_{\mathcal{S}}$, and normalizes each incoming sample x_s as follows,

$$\hat{x}_s = \gamma \cdot \frac{x_s - \mathbb{E}[X_{\mathcal{S}}]}{\sqrt{\mathbb{V}[X_{\mathcal{S}}] + \epsilon}} + \beta, \tag{19}$$

where γ and β denote the scale and shift parameters (*a.k.a.*the learnable affine transformation parameters), and ϵ is a small constant introduced for numerical stability. The BN statistics (*i.e.*, the mean $\mathbb{E}[\mathcal{X}_{\mathcal{S}}]$ and variance $\mathbb{V}[\mathcal{X}_{\mathcal{S}}]$) are typically approximated using exponential moving averages over batch-level estimates $\{\mu_k, \sigma_k^2\}$,

$$\hat{\mu}_{k+1} = (1-\rho)\cdot\hat{\mu}_k + \rho\cdot\mu_k,\ \hat{\sigma}_{k+1}^2 = (1-\rho)\cdot\hat{\sigma}_k^2 + \rho\cdot\sigma_k^2, \tag{20}$$

where ρ is the momentum term, k denotes the training step, and the BN statistics over the k-th mini-batch $\{x_i\}_{i=1}^{B_s}$ are

$$\mu_k = \frac{1}{B_s}\sum\nolimits_i x_i,\ \sigma_k^2 = \frac{1}{B_s}\sum\nolimits_i (x_i - \mu_k)^2, \tag{21}$$

Test-Time Batch Adaptation

2、Model Optimization

(1) training with auxiliary tasks

(2) fine-tuning with unsupervised objectives

and the test instance. Specifically, they adopt a common multi-task architecture, comprising the primary classification head $h_c(\cdot;\theta_c)$, the SSL head $h_s(\cdot;\theta_s)$, and the shared feature encoder $f_e(\cdot;\theta_e)$. The following joint objective of TTT or OSHOT is optimized at the training stage,

$$\theta_e^*, \theta_c^*, \theta_s^* = \arg\min_{\theta_e,\theta_c,\theta_s} \sum_{i=1}^{n_s} \mathcal{L}_{pri}(x_i, y_i; \theta_c, \theta_e) + \mathcal{L}_{ssl}(x_i; \theta_s, \theta_e), \tag{24}$$

where $\mathcal{L}_{pri}$ denotes the primary objective (*e.g.*, cross-entropy for classification tasks), and $\mathcal{L}_{ssl}$ denotes the auxiliary SSL objective (*e.g.*, rotation prediction [87] and solving jigsaw puzzles [51]). For each test instance x_t, TTT [8] first adjusts the feature encoder $f_e(\cdot;\theta_e)$ by optimizing the SSL objective,

$$\theta_e(x_t) = \arg\min_{\theta_e} \mathcal{L}_{ssl}(x_t; \theta_s^*, \theta_e), \tag{25}$$

then obtains the prediction with the adjusted model as $\hat{y} = h_c(f_e(x;\theta_e(x_t));\theta_s^*)$. By contrast, OSHOT [390] mod-

3、Meta-Learning

MAML , a notable example of meta-learning, learns a meta-model that can be quickly adapted to perform well on a new task using a small number of samples and gradient steps.

two distinct categories: backward propagation、forward propagation

Online Test-Time Adaptation

Definition 5 (Online Test-Time Adaptation, OTTA). Given a well-trained classifier f_S on the source domain $\mathcal{D}_S$ and a sequence of unlabeled mini-batches $\{\mathcal{B}_1, \mathcal{B}_2, \cdots\}$, *online test-time adaptation* aims to leverage the labeled knowledge implied in f_S to infer labels of samples in $\mathcal{B}_i$ under distribution shift, in an online manner. In other words, the knowledge learned in previously seen mini-batches could be accumulated for adaptation to the current mini-batch.

方法：

1、BatchNorm Calibration（批量归一标准化）

和TTBA的差不多，只不过是可以继续多个batch的batch norm统计。

2、Entropy Minimization（熵最小化）

文中没有概括这个方法的流程，之说了是处理无标签数据常用的方法。

3、Pseudo-labeling（伪标签）

与SFDA的伪标签类似，一边利用伪标签学习，一边标记下一个batch的伪标签。

4、Consistency Regularization（一致性训练）感觉与SFDA的一致性训练类似。

Online Test-Time Adaptation

灾难性遗忘问题

Previous studies [483], [484] find that the model optimized by TTA methods suffers from severe performance degradation (named forgetting) on original training samples. To mitigate the forgetting issue, a natural solution is to keep a small subset of training data that is further learned at test time as regularization [516], [526], [538]. PAD [541] comes up with an alternative approach that keeps the relative relationship of irrelevant auxiliary data unchanged after test-time optimization. AUTO [514] maintains a memory bank to store easily recognized samples for replay and prevents overfitting towards unknown samples at test time.

Another anti-forgetting solution lies in using merely a few parameters for test-time model optimization. For exam-

（1）保留一小部分训练数据，该子集在测试时作为正则化进一步学习

（2）减少改变的参数，仅使用几个参数进行测试，从而提高模型的优化

Thanks