# Domain Adaptation

Week-6

# Announcements

- No lecture on this Friday (Mar. 29th, Good Friday, public holiday and no make-up session).
- Submission deadline extension for CLAB-2. see the announcement by Dylan.

Dear Students,

Welcome to Week 6! Two important announcements this week:

1.**Friday** is a **public holiday**: there are no labs and no lecture. While the Friday labs have been rescheduled (see MyTimetable), we understand that not everyone will be able to attend their rescheduled lab. In recognition of this, Assignment 2 will be extended by one week.
2.Assignment 2 extension: due 23:59 **Friday 26 April.**
3.Tutorial 4 continues this week during lab time.

**Best wishes,**
Dylan

# Addressing students' concerns

- Didn't fully understand the requirement on the deep learning lectures, theory or high-level concepts?
  - Both. First, students need to complete network training listed in CLAB-2. Second, students are required to understand the basic calculations involved, BP, parameters of the network calculation, receptive field calculation.
- Concerns on lab sessions and tutorials.
  - We will work with tutors on improving the quality for the coming lab sessions.
- Slides are not updated (sometimes with new slides.)
  - I will double check the slides and update it with the ones shown in the recording.
- What types of questions are included in the final exam for the theoretical calculation part?
  - See the questions we released for each tutorial per week.
  - Release exam papers during the break.

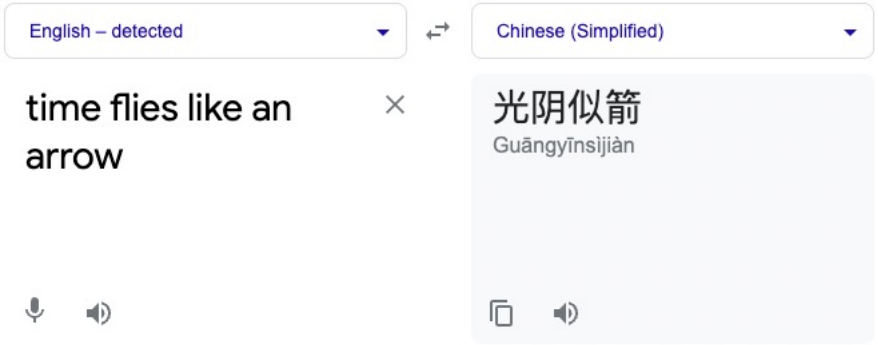
# Addressing students' concerns

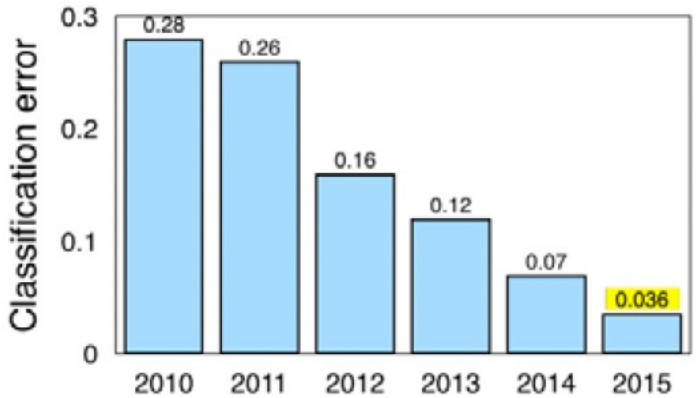
- Having more systematic tutorials about the tools we need for assignments.
- The assignments are way too big for how much of the total grade they are worth.
- Coding examples.
- Cannot capture the whole idea of the lecture(s)
- Feel issues for understanding the basic concepts of traditional Machine Learning.
- Tutors are not available for the first 30 minutes of the lab session.

# Success of deep learning

## Deep learning for self-driving cars

## Google translate

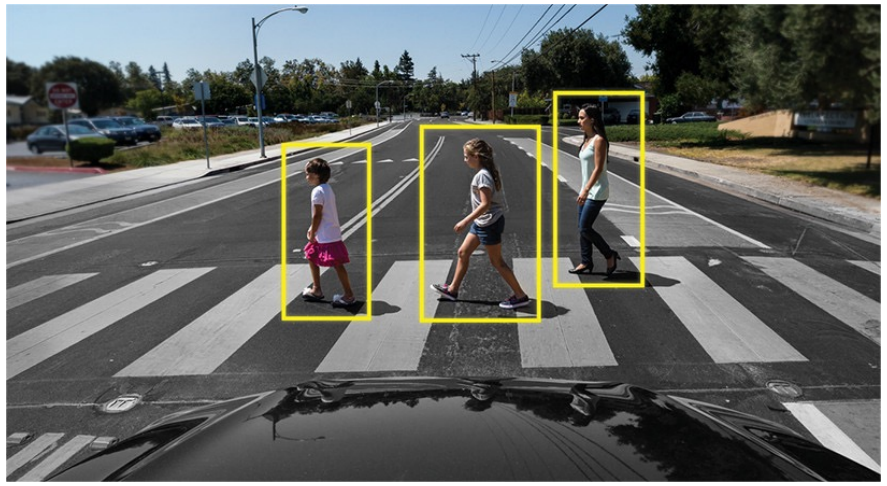



IMAGENET

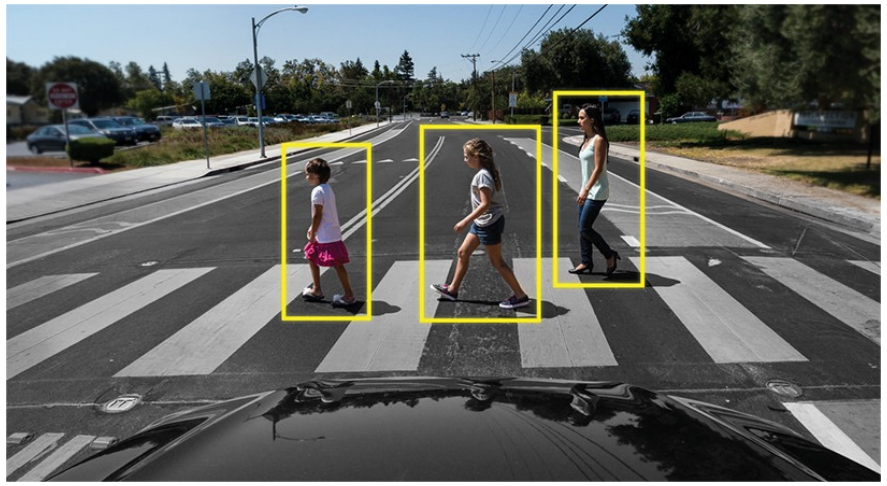
# A major limitation of deep learning

- Not data efficient: learning requires millions of labeled examples to achieve good performance
- Models do not generalize well to new domains: not like humans!



New domain

# A major limitation of deep learning



What your neural network is trained on

New domain


What it is asked to test on

“dataset bias”
“domain shift”
“domain adaptation”
“domain transfer”

Assume that training a classifier is not a big deal for you. ☺

Training Data

Testing Data

99.5%

57.5%

The results are from: http://proceedings.mlr.press/v37/ganin15.pdf

Domain shift: Training and testing data have different distributions.

→ **Domain adaptation**

# Definition of Domain Adaptation

- Domain adaptation is a field of computer vision, where our goal is to train a neural network on a **source dataset** and secure/achieve a good accuracy on the **target dataset** which is significantly different from the source dataset.

**Transfer learning:** https://en.wikipedia.org/wiki/Transfer_learning

# Domain shift example: image classification

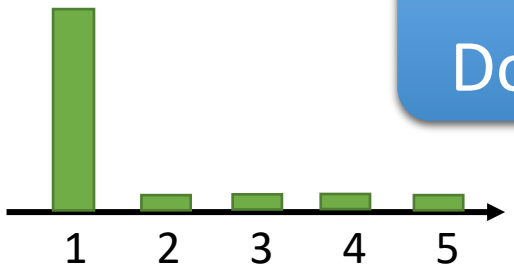
Training Data

Test Data

Source Domain

Target Domain

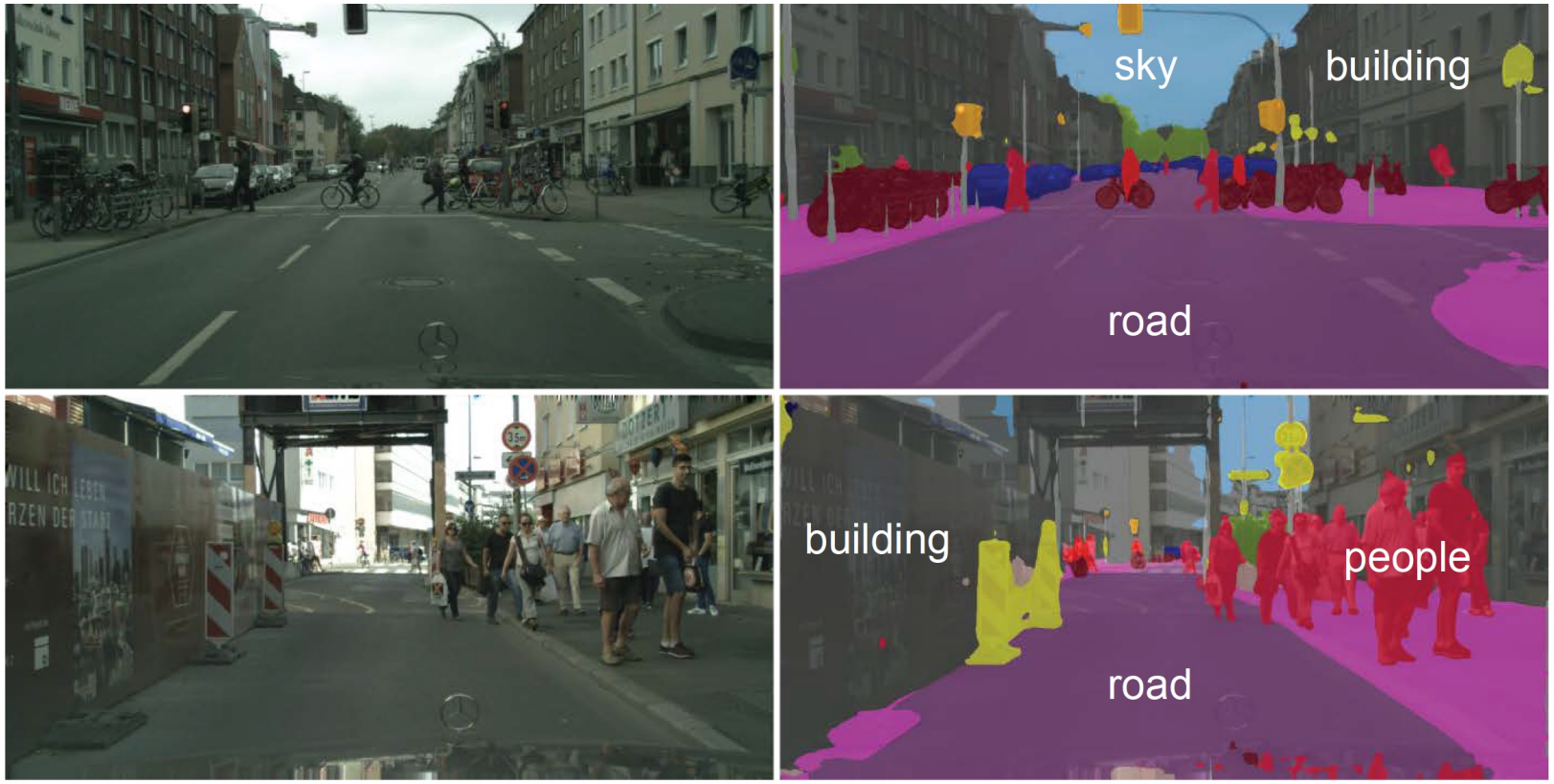
different distribution


This is “0”.

This is “1”.

# Domain shift example: scene segmentation*

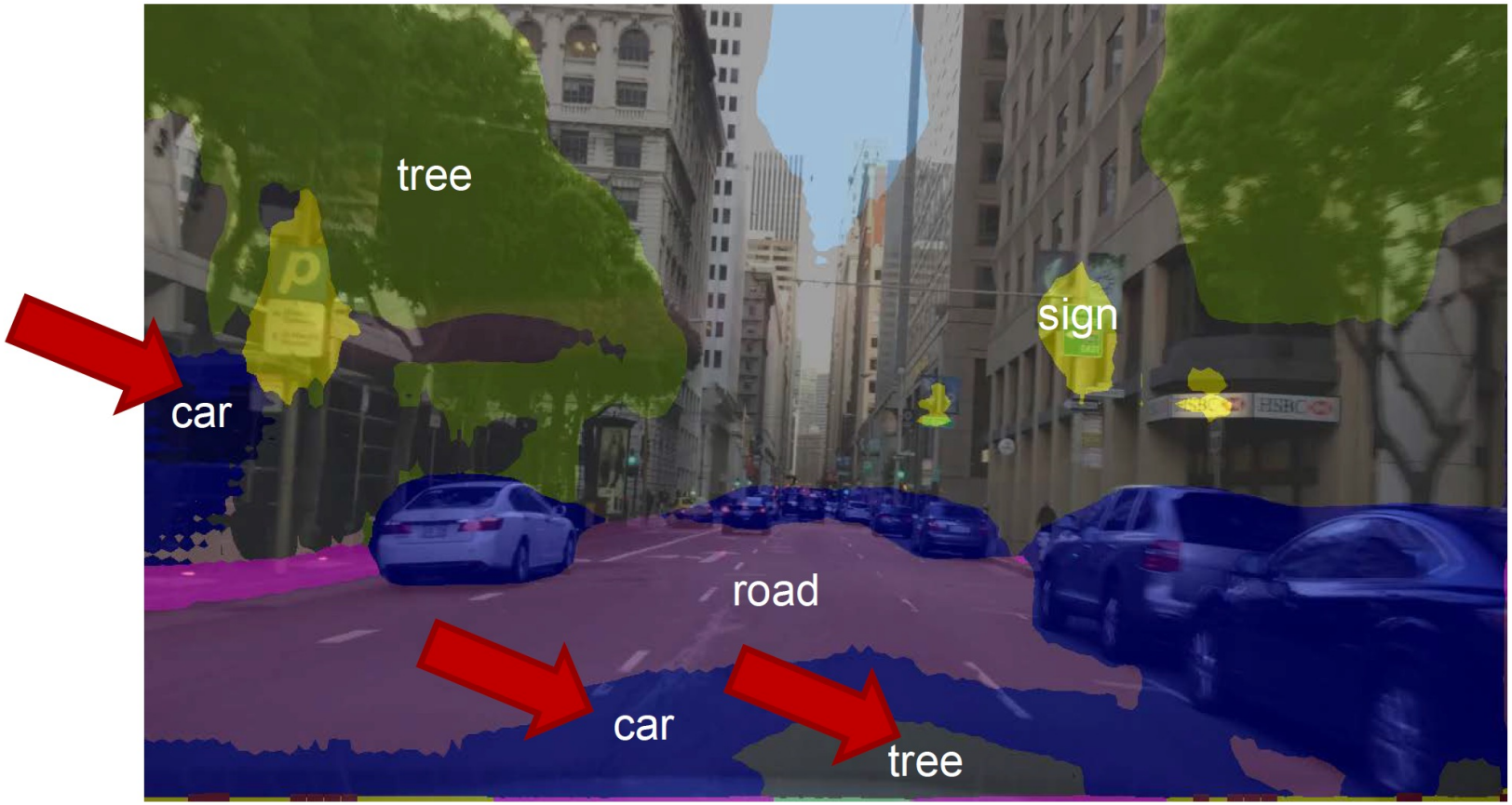



Source domain: Cityscapes

In this example: a segmentation neural network is trained on Cityscape and tested on Cityscapes

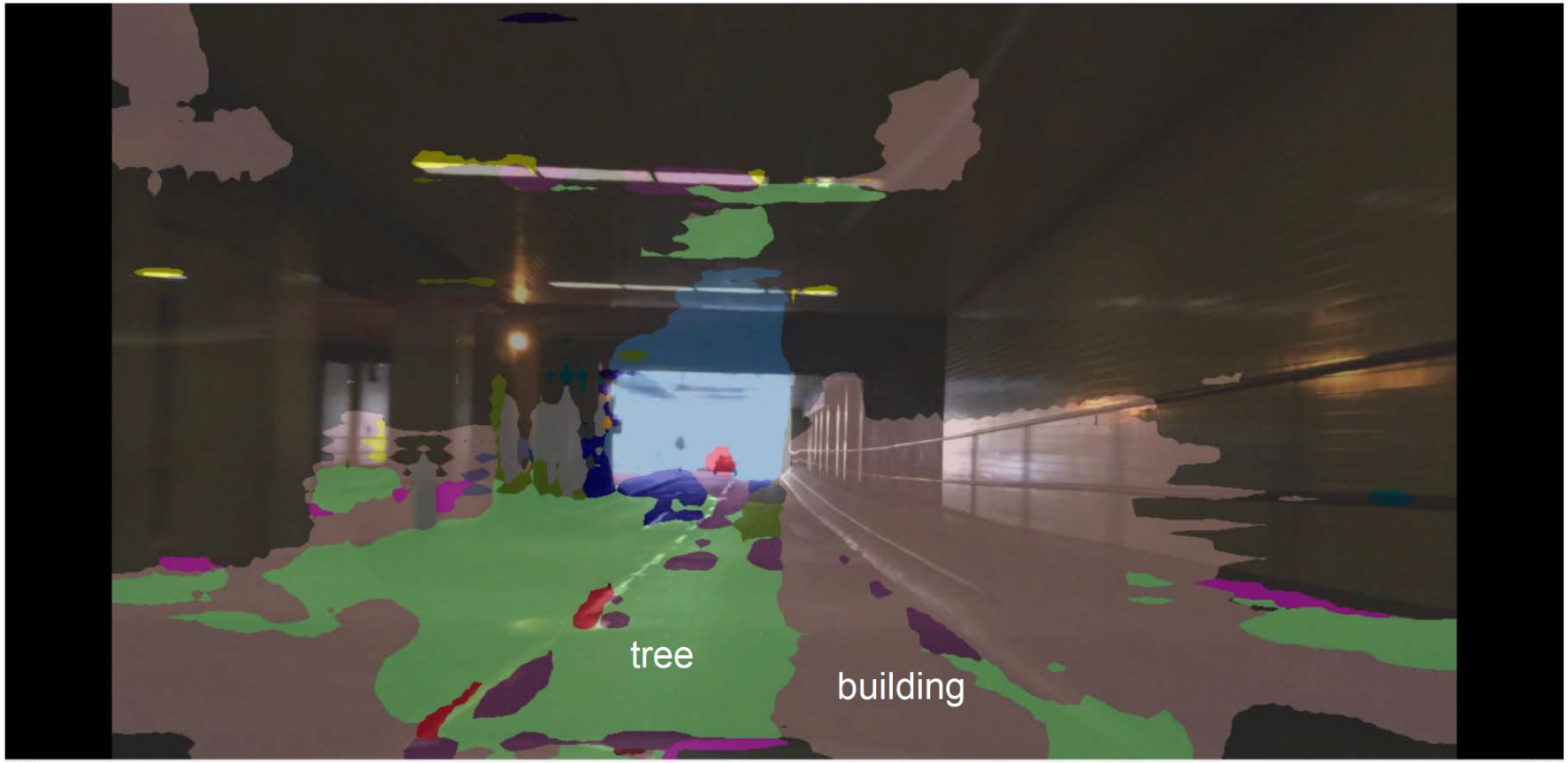
* High level vision contents will be covered in week 11.

# Domain shift example: scene segmentation




When test the previous segmentation model on the target domain, i.e., San Francisco street view in this example, segmentation results are much worse.

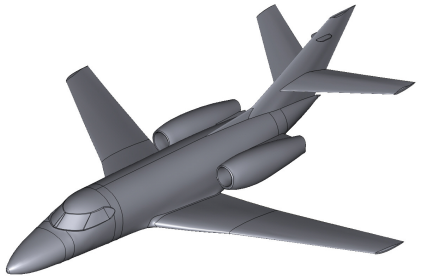
# No tunnels in Cityscapes?



# Different types of domain shift

From dataset to dataset

From CAD models to real images



From low density to high density

# Domain Adaptation

Source Domain
(with labeled data)

“4” “0” “1”

Knowledge of target domain

- Idea: training a model by source data, then fine-tune the model by target data
- Challenge: only limited target data, so be careful about overfitting

“8”

Little but labeled

# Domain Adaptation

- Fine-tuning (Implementation)
  - In deep learning, **fine-tuning** is an approach to transfer learning in which the weights of a pre-trained model are trained on new data. Fine-tuning can be done on the entire neural network, or on only a subset of its layers, in which case the layers that are not being fine-tuned are "frozen" (not updated during the backpropagation step).
  - In deep learning coding, please check the model loading function,

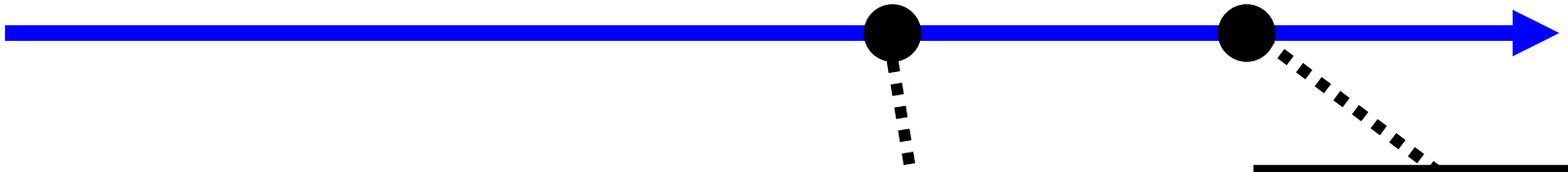
```
• model.load_state_dict(torch.load('pre-trained-Model.pth'))

• for name, param in model.named_parameters():
      if 'layer' in name and int(name.split('.')[1]) < 3: # Freeze the first 3 layers
          param.requires_grad = False
```

# Domain Adaptation

Source Domain
(with labeled data)

“4” “0” “1”

Knowledge of target domain



Large amount of
unlabeled data

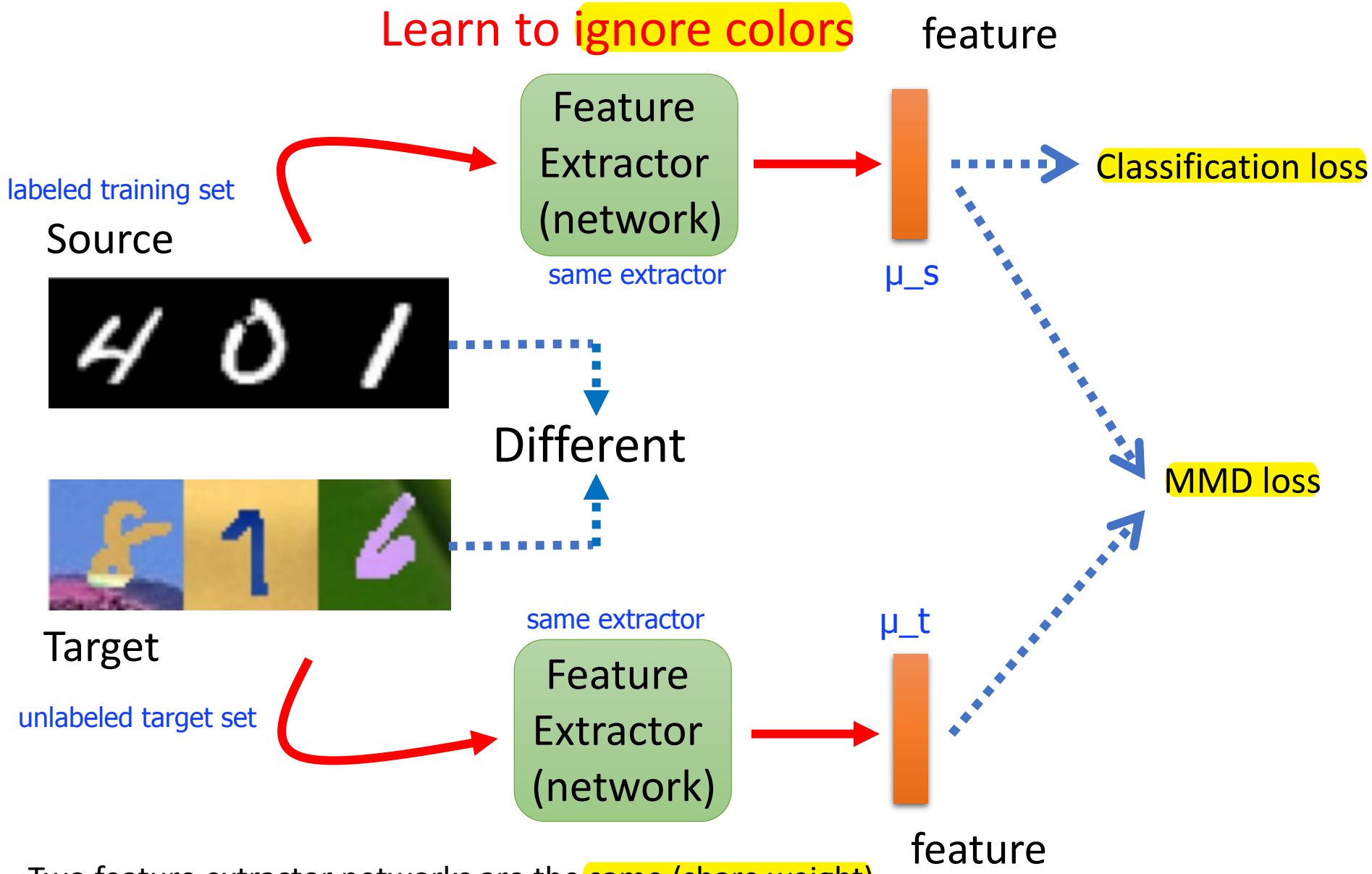
“8”

Little but
labeled

# Today

- We show that deep models can be adapted to the target domain **without labels**
  - It's named unsupervised domain adaptation
- Mainly introduce three domain adaptation methods
  - Maximum mean discrepancy (MMD)
  - Correlation alignment
  - Adversarial alignment

# Maximum mean discrepancy (MMD)




Two feature extractor networks are the same (share weight)

# Maximum mean discrepancy (MMD)

- In each training mini-batch, we compute the mean of source features:

$$\mu_S = \frac{1}{n_S}\sum_i f_S^i$$

$f_S^i$ is the feature of the $i$th source image
$n_S$ is the number of source features

- We compute the mean of target features

$$\mu_T = \frac{1}{n_T}\sum_i f_T^i .$$

$f_T^i$ is the feature of the $i$th target image
$n_T$ is the number of target features

- The MMD loss is written as:

$$L_{MMD} = \|\mu_S - \mu_T\|^2.$$

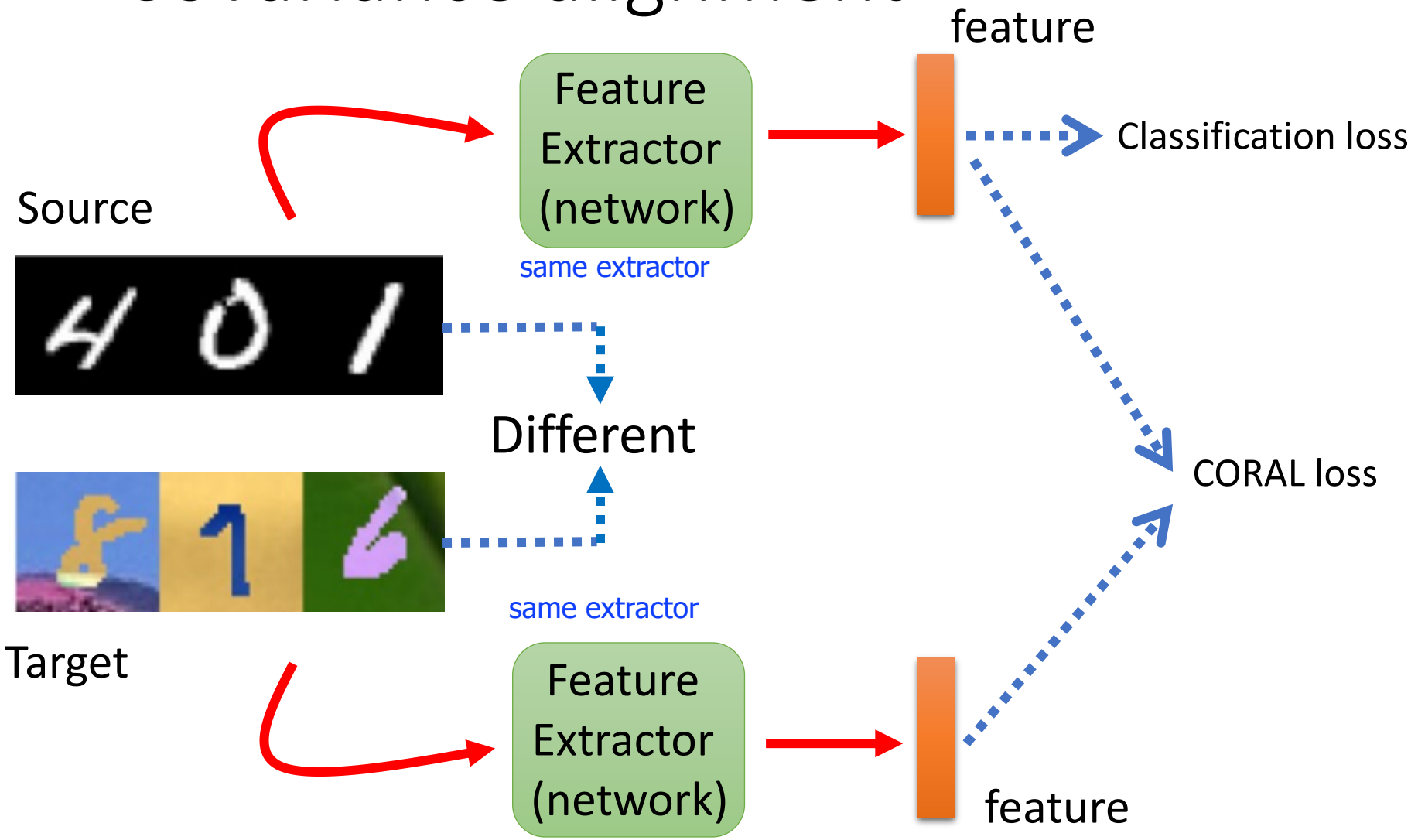
diff. between Src and Tgt mean features

- Overall loss function:

$$L = L_{class} + \lambda L_{MMD}$$

where $\lambda$ controls the balance between the two losses.
We use $L_2$ norm in MMD loss

# Covariance alignment




Two feature extractor networks are the same (share weight)

Sun et al., Deep CORAL: Correlation Alignment for Deep Domain Adaptation. ECCV 2016

# CORAL loss

The CORAL loss is defined as the distance between the second-order statistics (covariances) of the source and target features

$$L_{coral} = \frac{1}{4d^2} \|C_S - C_T\|_F^2,$$

where d is the feature dimension, and $C_S$ and $C_T$ represent covariance matrix of the source and target domains, respectively.

$L = L_{class} + \sum_{i=1}^{t} L_{coral}$, where $t$ is the number of layers that apply this loss

Sun et al., Deep CORAL: Correlation Alignment for Deep Domain Adaptation. ECCV 2016

# CORAL Loss

- Categorical Cross-entropy loss

Given the predicted probability distribution, it is defined as the average negative log-likelihood of the true class. The formula for categorical cross-entropy loss is expressed as

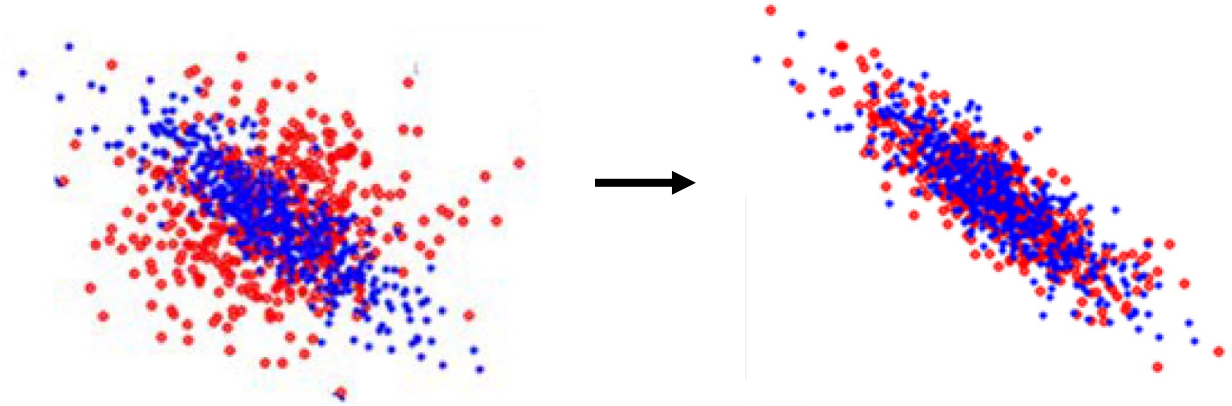
$$L = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{i,j} log(p_{i,j}), \tag{19}$$

softmax

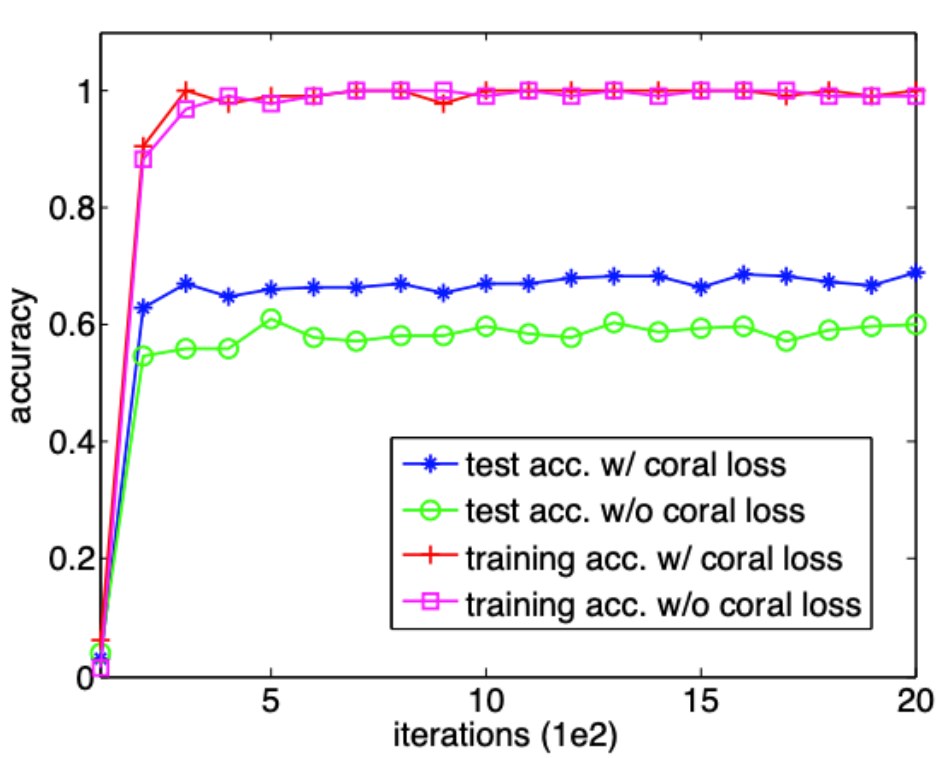
where $N$ is the number of samples, $C$ is the number of classes, $y$ is the true label, and $p$ is the predicted probability of the true class. The loss is calculated for each sample and averaged over the entire dataset.

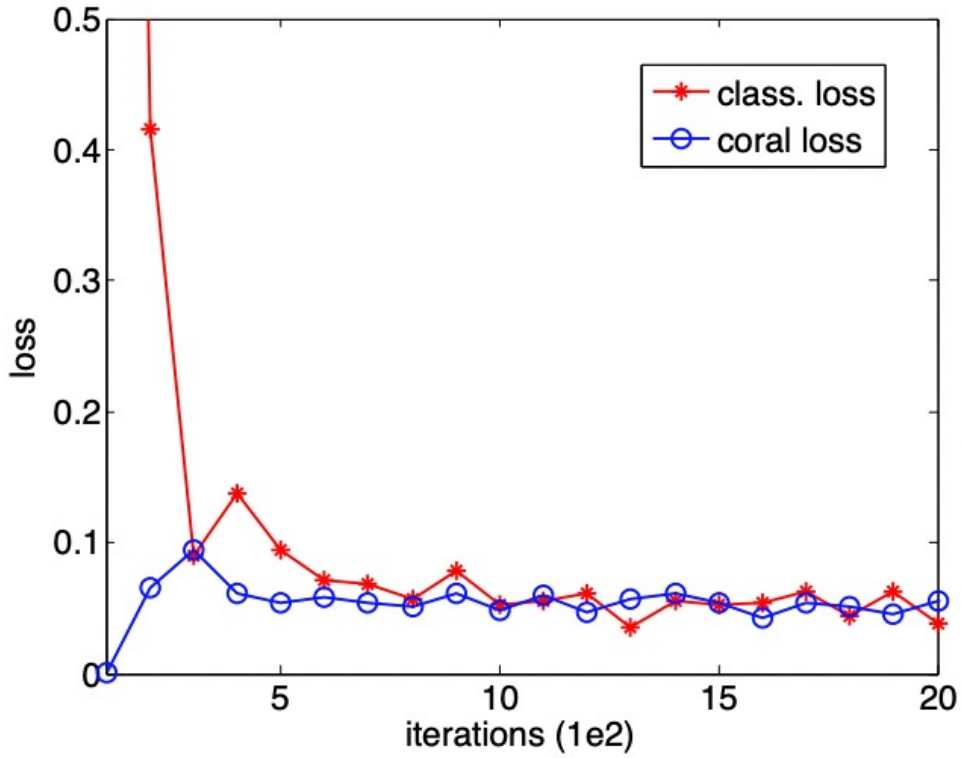
# Some results

Cov matrix before using CORAL



Cov matrix after using CORAL

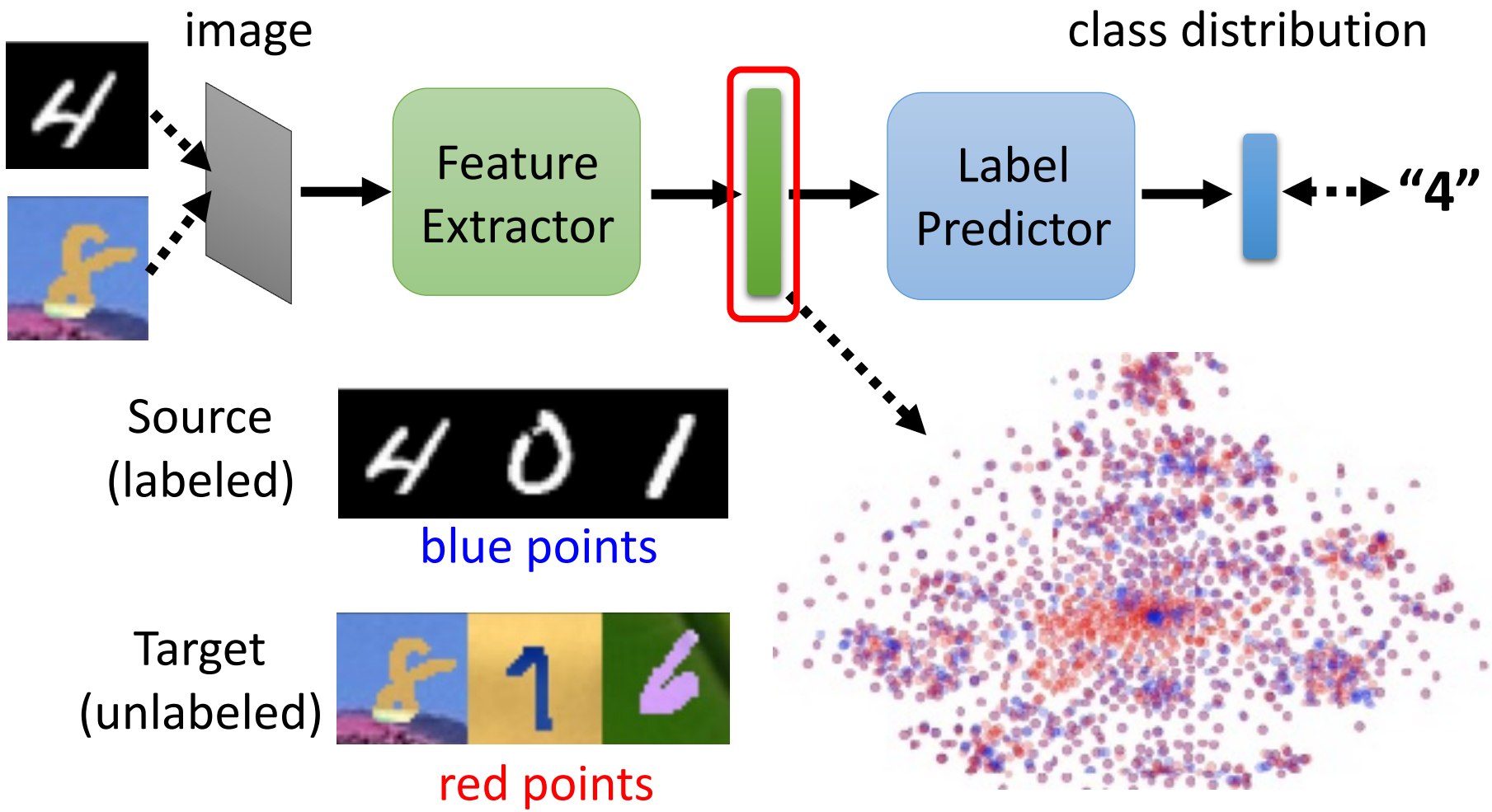


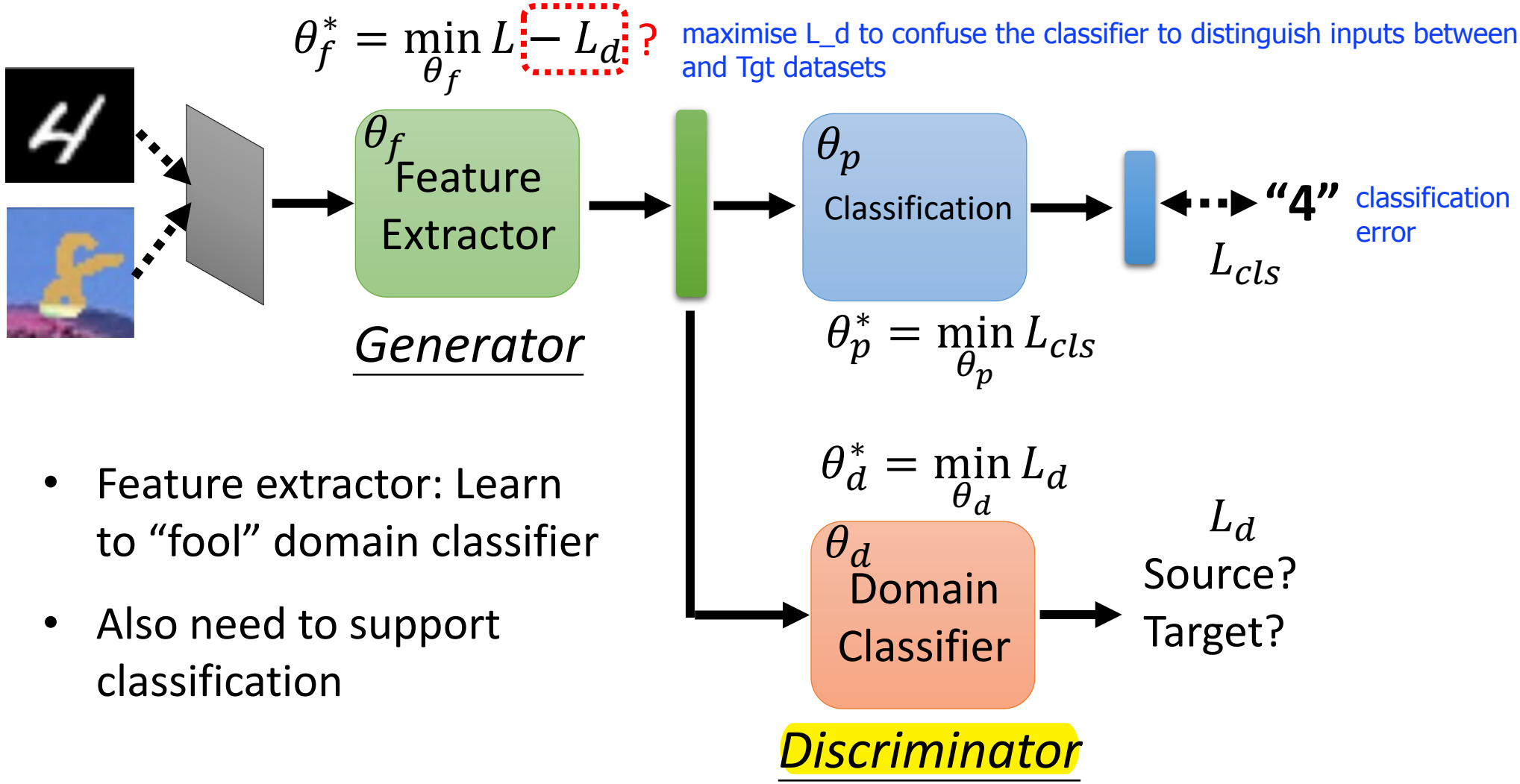
# Domain Adversarial Training

Adversarial networks

# Domain Adversarial Training



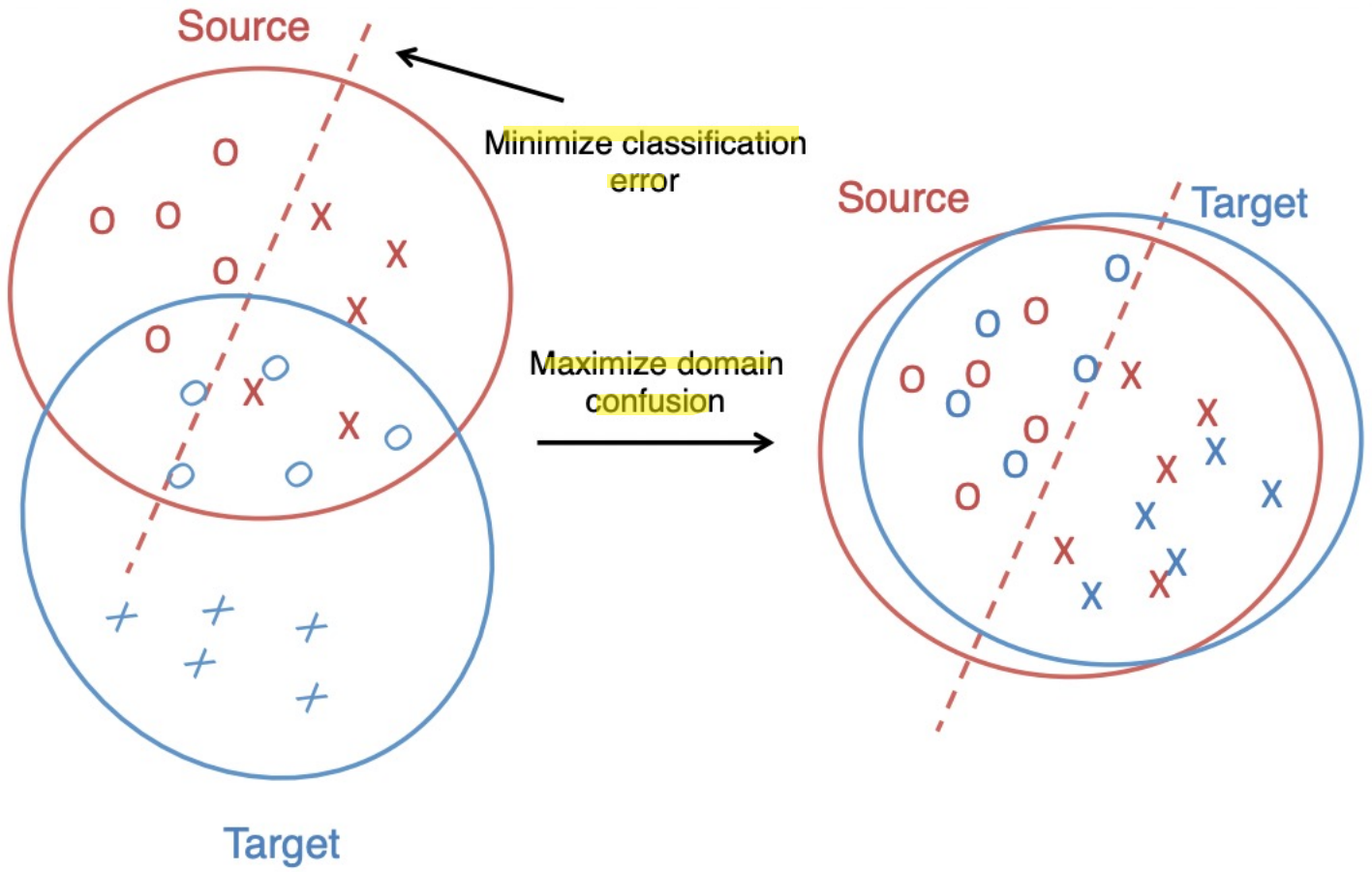
# Domain Adversarial Training




- Feature extractor: Learn to "fool" domain classifier
- Also need to support classification

Simultaneous Deep Transfer Across Domains and Tasks. ICCV 2015

# An illustration adversarial training



# Adversarial loss functions

- To achieve domain invariance, we have two loss functions
- Domain classification loss

$$L_{domain}(x_S, x_T, \theta_{feature}; \theta_{domain}) = -\sum_d 1[y_D = d] \log p_d$$

softmax

- Domain confusion loss

$$L_{conf}(x_S, x_T, \theta_{domain}; \theta_{feature}) = -\sum_d \frac{1}{D} [y_D = d] \log p_d$$

The domain confusion loss makes output predicted domain labels similar to a uniform distribution over domain labels

where $x_S, x_T$ indicate source, target domain samples; $\theta_{feature}$ is the feature extraction network; $\theta_{domain}$ is the domain classifier. $y_d$ means the predicted domain of an input sample, and $D$ is the ground truth domain of this sample. $p$ is the softmax of the domain classifier activation: $p = softmax\left(\theta_{domain}^T f\left(x; \theta_{feature}\right)\right)$.

# Problem

- Domain classification loss

if y_d is from the Src domain

minimise $$L_{domain}(x_S, x_T, \theta_{feature}; \theta_{domain}) = -\sum_d 1[y_d = D] \log p_d$$

- Domain confusion loss

maximise $$L_{conf}(x_S, x_T, \theta_{domain}; \theta_{feature}) = -\sum_d \frac{1}{D}[y_d = D] \log p_d$$

- Ideally, we want to minimize both
  - A domain classifier is trained
  - Domain confusion is achieved.
- However, the two losses stand in direct opposition to one another:
  - learning a fully domain invariant representation means the domain classifier must do poorly; and
  - learning an effective domain classifier means that the representation is not domain invariant

# Solution – iterative updates

$$\min_{\theta_{domain}} L_{domain}(x_S, x_T, \theta_{feature}; \theta_{domain})$$

$$\min_{\theta_{feature}} L_{conf}(x_S, x_T, \theta_{domain}; \theta_{feature})$$

We perform iterative updates for the following two objectives given the fixed parameters from the previous iteration

# Results

Before domain confusion

After domain confusion

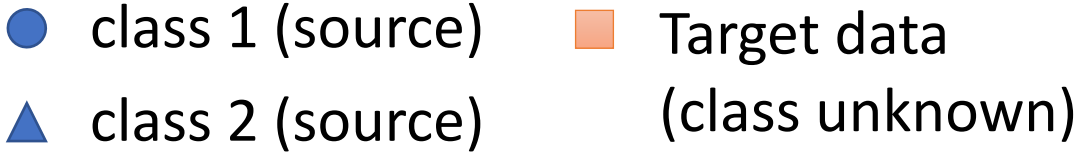
# Results

Yaroslav Ganin, Victor Lempitsky, Unsupervised Domain Adaptation by Backpropagation, ICML, 2015

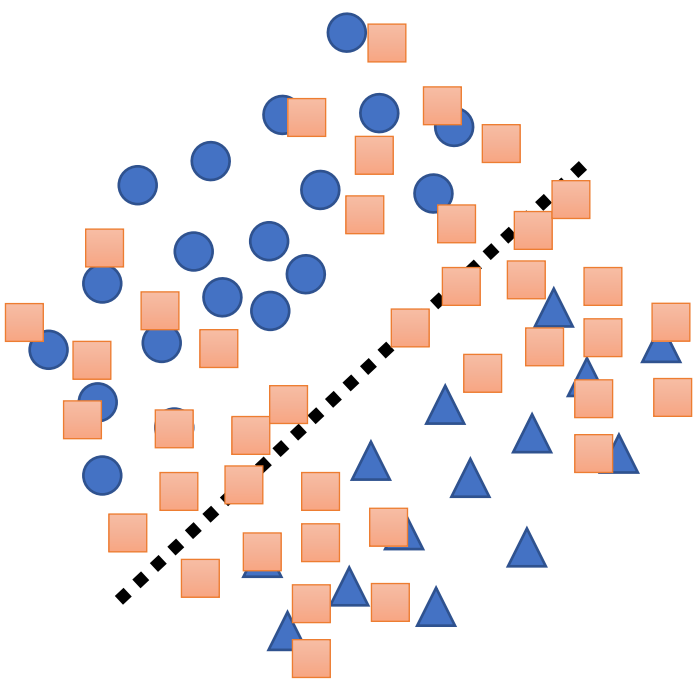
Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, Domain-Adversarial Training of Neural Networks, JMLR, 2016


| Method | Source / Target | MNIST / MNIST-M | Syn Numbers / SVHN | SVHN / MNIST | Syn Signs / GTSRB |
|---|---|---|---|---|---|
| Source only | | .5225 | .8674 | .5490 | .7900 |
| SA (Fernando et al., 2013) | | .5690 (4.1%) | .8644 (−5.5%) | .5932 (9.9%) | .8165 (12.7%) |
| DANN | | **.7666** (52.9%) | **.9109** (79.7%) | **.7385** (42.6%) | **.8865** (46.4%) |
| Train on target | | .9596 | .9220 | .9942 | .9980 |

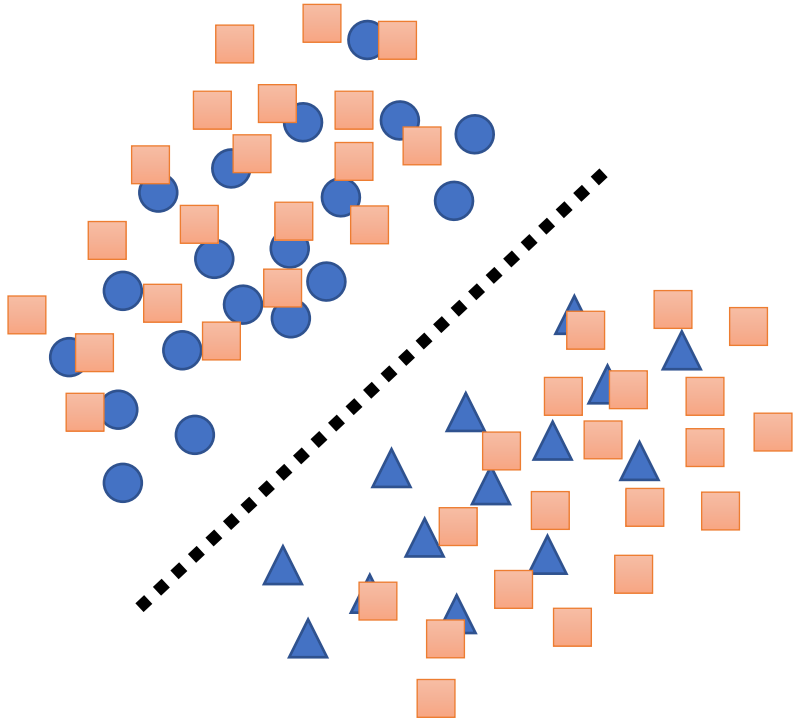
# Limitation




Decision boundaries learned from source domain

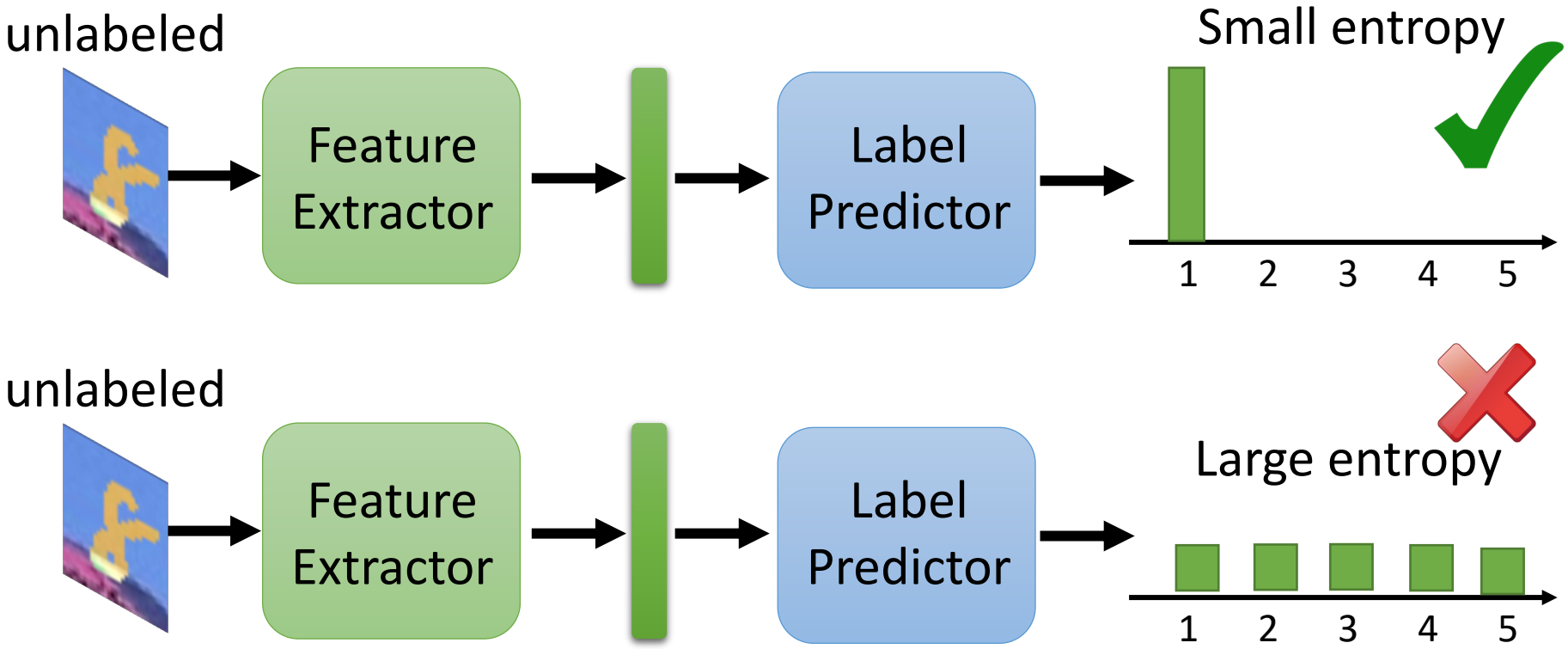


Source and target data are aligned, but ……



Target data (unlabeled far from boundary)

# Considering decision boundary




Used in Decision-boundary Iterative Refinement Training with a Teacher (DIRT-T) https://arxiv.org/abs/1802.08735

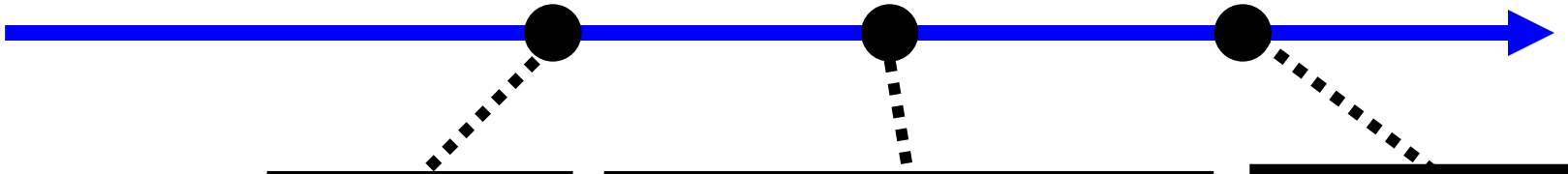
Maximum Classifier Discrepancy https://arxiv.org/abs/1712.02560

# Domain Adaptation

Source Domain
(with labeled data)

“4” “0” “1”

Knowledge of target domain



Testing Time Training (TTT)

https://arxiv.org/abs/1909.13231

little & unlabeled

Large amount of unlabeled data

“8”

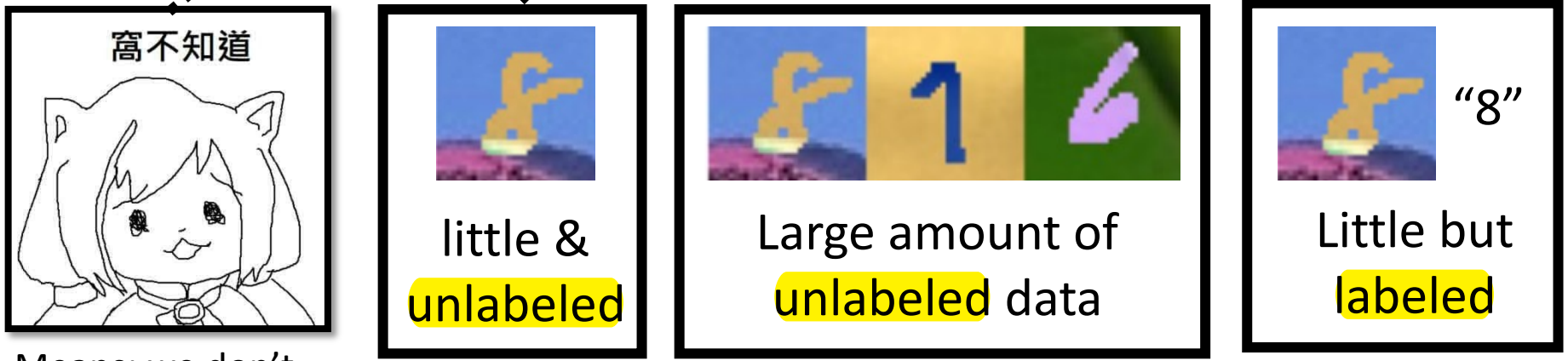
Little but labeled

# Domain Adaptation

Source Domain
(with labeled data)

“4” “0” “1”

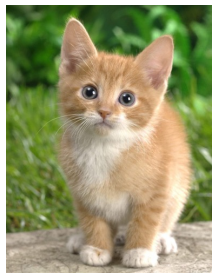
Knowledge of target domain




Means: we don’t know anything about it

# ***Domain Generalization***

https://ieeexplore.ieee.org/document/8578664


cat


dog


cat


dog


cat

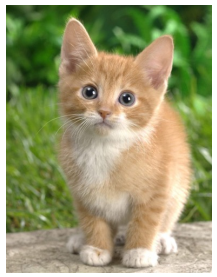

dog

Training


cat


dog

Testing


cat


dog

Training


cat


dog


cat


dog


cat


dog

Testing

https://arxiv.org/abs/2003.13216

# Concluding Remarks

Source Domain
(with labeled data)

“4” “0” “1”

Knowledge of target domain


Means: we don’t know anything about it