Domain Adaptation

Let us take a quick test

- Wie is de president van India (Dutch)
- Wer ist der Präsident von Indien (German)
- Who is the president of India
- ભારતના રાષ્ટ્રપતિ કોણ છે?
- भारताचे राष्ट्रपती कोण आहेत?
- भारत के राष्ट्रपति कौन हैं?

Have you observed the following?

- A CNN pre-trained on Imagenet yields very high accuracy on CIFAR-10 database, but when tested on LFW database (without fine-tuning filters on LFW train set) yields low accuracies, why?

Expectation

- A model trained on large training data can generalize and yield high accuracies on unseen test data

Reality-1

- Big assumption: Train and test data follow the same distribution
- Unseen testing or cross database testing are hard problems (and low accuracies are achieved)
  - Train on Imagenet and test on LFW database (without fine-tuning on LFW) -> low accuracies
  - Fine-tune Imagenet trained model with LFW train data and test on LFW test data -> improved accuracy

Reality-2: Example in Face Recognition

For the same task, algorithm trained with one domain (good quality) images fail with images from domain 2 (low resolution)

Reality-3

- Let us take an example of Aadhaar (UIDAI) project in India
  - in 2010 when it started, every day 1 million users were getting enrolled (started from zero)
  - data distribution varies from one part of the country to another (e.g. Punjab vs Kerala)
  - Same algorithm may not work directly (without fine-tuning)
  - Not all (training) data is available upfront to train the classifiers

Era of Deep Models

- Deep models require large databases for training

- Training a face recognition model for two different countries requires large training samples individually from the two countries

- Training a model for driving in USA does not work for Indian roads

Dataset Shift or Concept Drift

Train

Model

Test

Model will not yield good performance

Dataset Shift or Concept Drift

- Dataset shift - Pattern Recognition term
- Concept drift - Machine Learning term

Data Shift in Machine Learning, MIT Press 2009

Ideal ways to Handle

- Distinguishes between shift and noise
- Detect a shift and update the model

Domain Adaptation

Domain Adaptation

- One of the key human capabilities: ability to generalize from one domain to another

Source and Target Domains

- Source Domain where large training data is available and an algorithm learns the "task"
- Target Domain where very few samples are available and algorithm has to adapt for the "new (similar) task"

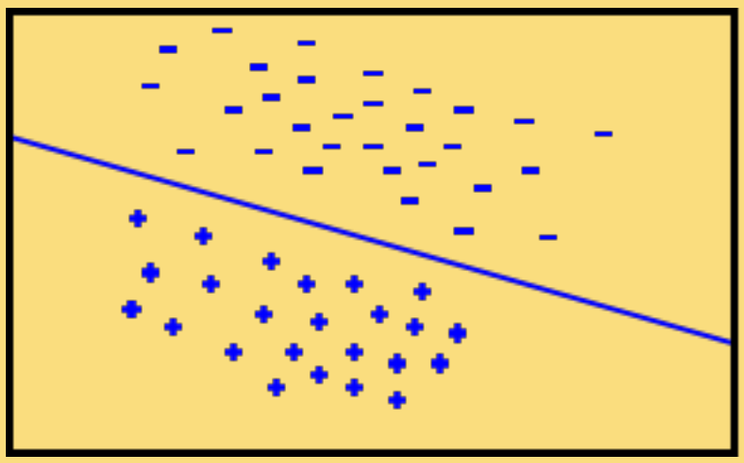
Source Target

Domain Adaptation (Formally)

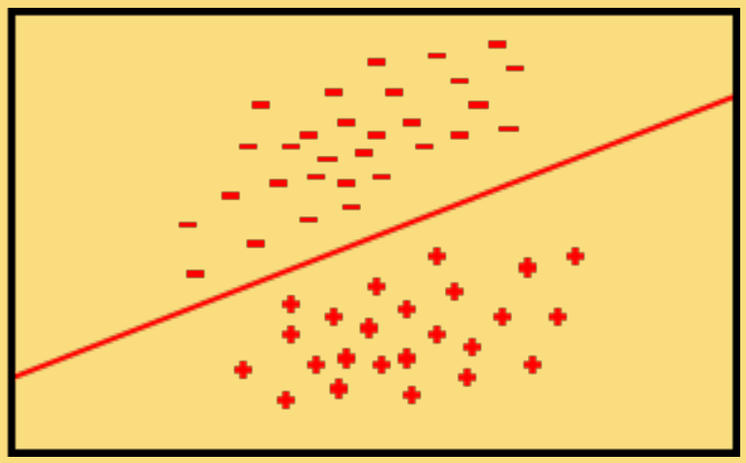
- Given
  - Source domain and task: D_s, T_s
  - Target domain and task: D_t, T_t
- Domain adaptation aim to improve the learning of target predictive function $f_T(.)$ in D_t using knowledge learnt in D_s and T_s

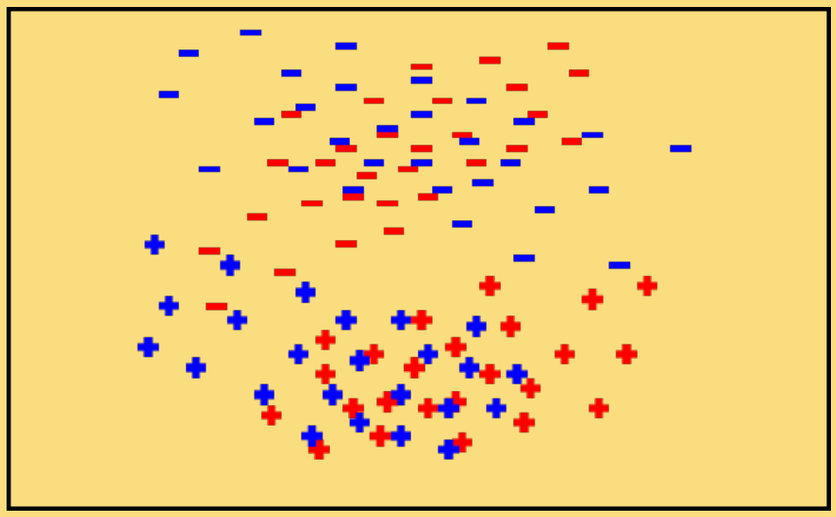
Another View

Source



Target





How to adapt?

What are the key takeouts so far?

- Variations in data distribution from one domain to another
- Training data may come in sequence - not all data is available for training upfront
- Similar tasks can be "pre-trained" in another domain or another dataset
- Limited training data in target domain

Some questions?

- A model is trained, now new samples pertaining to all classes are available, how to update the model?

Some questions?

- A model is trained, now new samples pertaining to some classes are available, how to update the model?

Some questions?

- A model is trained, now new samples pertaining to some new classes are available, how to update the model?

What we will discuss?

- Transfer Learning - most popular concept of domain adaptation
- Incremental / Online Learning
- Co-training
- Co-transfer Learning
- Mapping Approach

Concept

Transfer Learning

A Survey on Transfer Learning, TKDE, 2010

Transfer Learning

- People often transfer knowledge to novel situations
  - Chess → Checkers
  - C++ → Java
  - Maths → Computer Science

The ability of a system to recognize and apply knowledge and skills learned in previous tasks to novel tasks (or new domains)

Transfer Learning

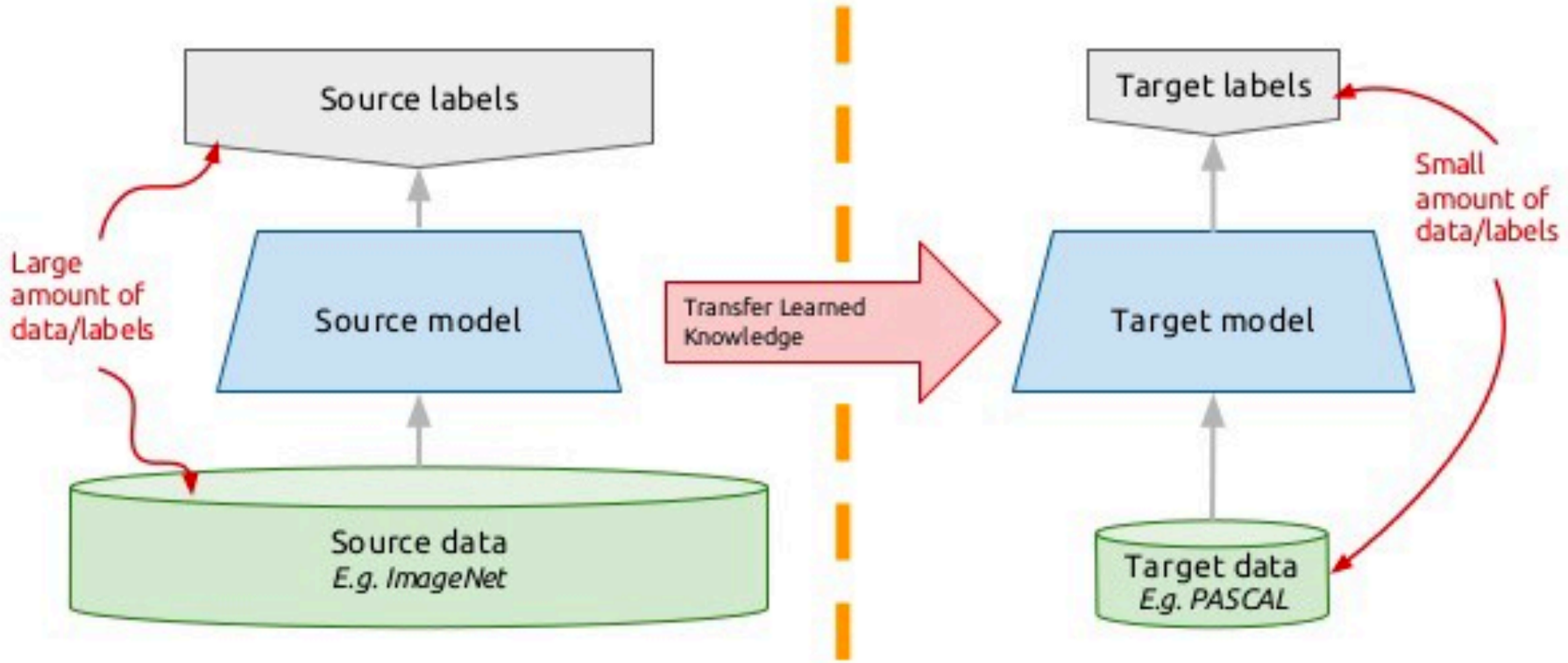
	Source Domain	Target Domain
Training Data	Labeled/ Unlabeled	Labeled/ Unlabeled
Test Data		Unlabeled

Formal Definition

Given a source domain D_s and learning task T_s, a target domain D_t and a learning task T_t, transfer learning aims to help improve the learning of the target prediction function $f_t(.)$ in D_t using the knowledge in D_s and D_t, where

$$D_s \neq D_t \text{ or } T_s \neq T_t$$

Transfer learning: idea

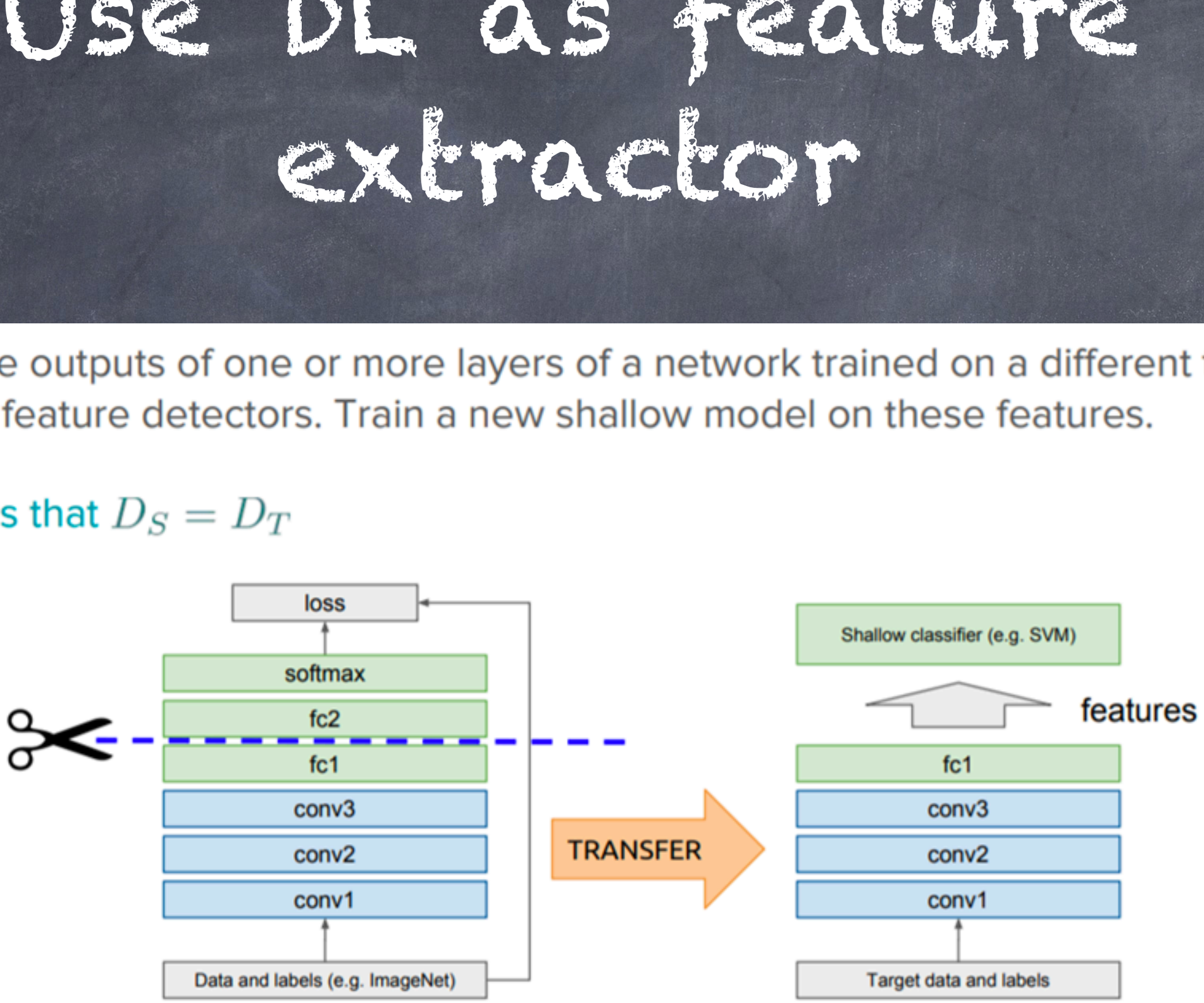


- What, how, and when to transfer?

Use DL as feature extractor

Idea: use outputs of one or more layers of a network trained on a different task as generic feature detectors. Train a new shallow model on these features.

Assumes that $D_S = D_T$



Simplest Transfer Learning

- Fine-tuning

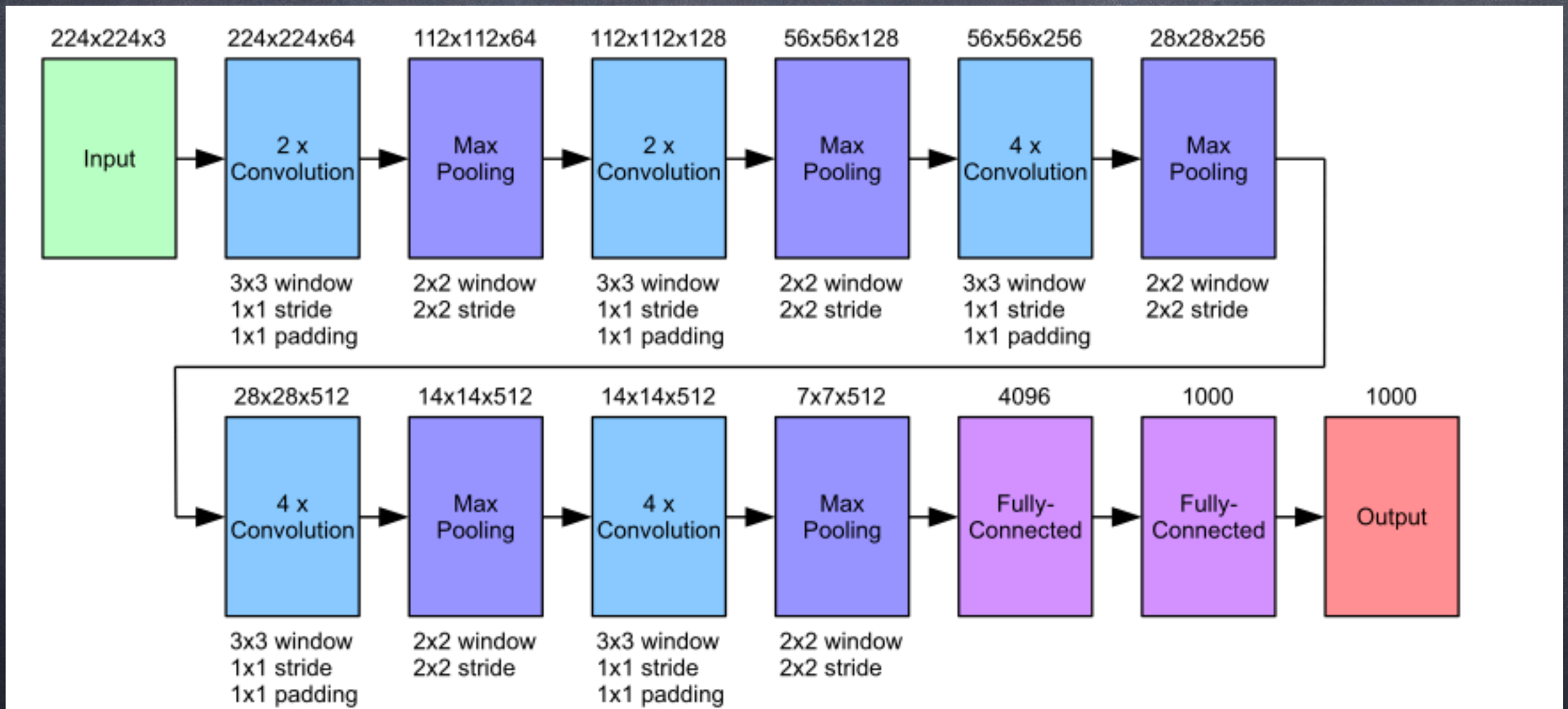
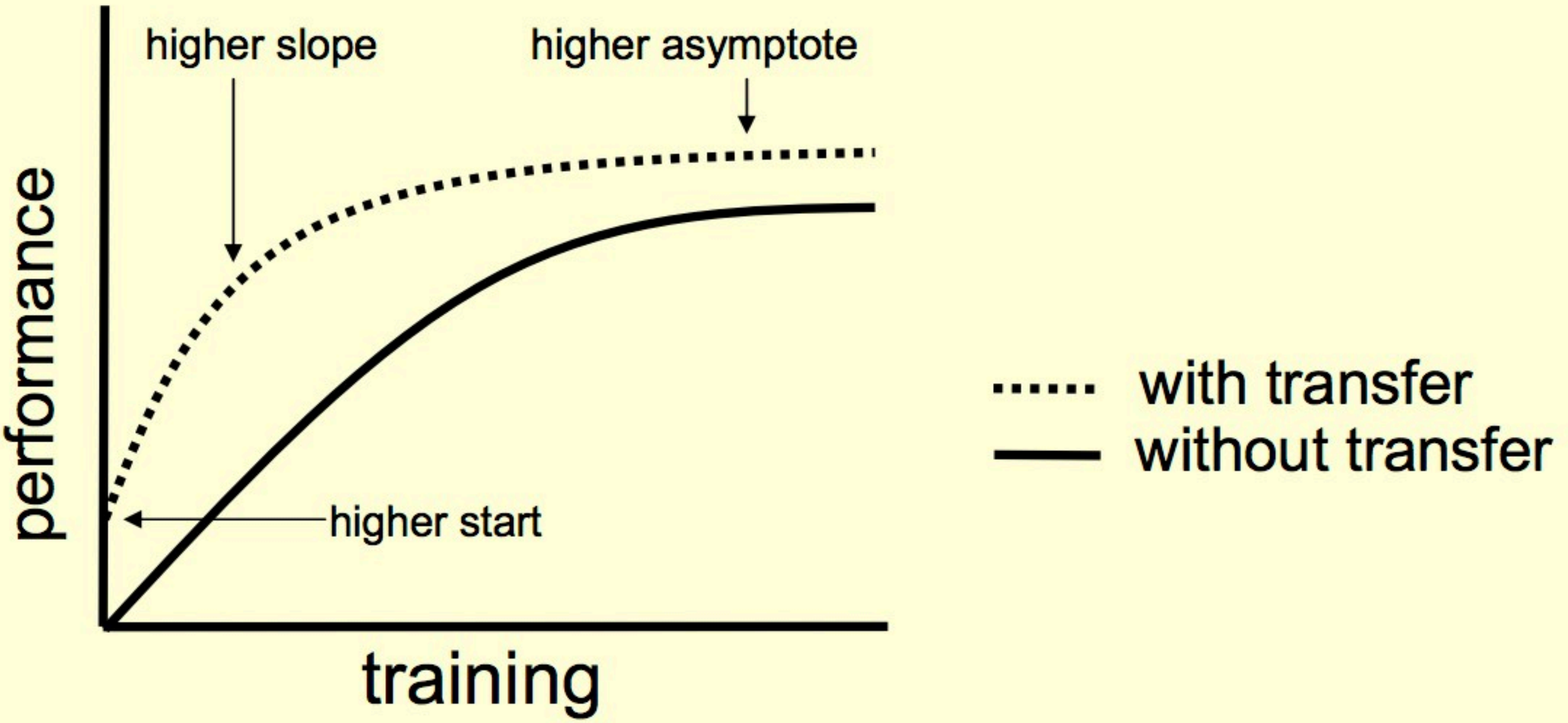


Fig. 13. The VGGNet, containing a total of 24 hidden layers, consisting of 16 3×3 convolutional layers, five 2×2 max pooling layers, as well as three fully-connected layers


higher slope
higher asymptote
performance
higher start
training
with transfer
without transfer

Transfer Learning

- Three types based on how to transfer:
  - Inductive transfer
  - Transductive transfer
  - Unsupervised transfer

Inductive Transfer Learning

- $T_S \neq T_T$: source and target tasks are different
- D_S and D_T may be same or different

Transductive Transfer Learning

- $D_S \neq D_T$
- $T_S = T_T$
- Some labeled target-domain data must be available at training time.

Unsupervised Transfer Learning

- $T_S \neq T_T$
- labels are not available

Relationship between Traditional Machine Learning and Various Transfer Learning Settings

Learning Settings		Source and Target Domains	Source and Target Tasks
Traditional Machine Learning		The same	The same
Transfer Learning	Inductive Transfer Learning	The same	Different but related
	Unsupervised Transfer Learning	Different but related	Different but related
	Transductive Transfer Learning	Different but related	The same

Different Settings of Transfer Learning

Transfer Learning Settings	Source Domain Labels	Target Domain Labels	Tasks
Inductive Transfer Learning	Available	Available	Regression, Classification
	Unavailable	Available	Regression, Classification
Transductive Transfer Learning	Available	Unavailable	Regression, Classification
Unsupervised Transfer Learning	Unavailable	Unavailable	Clustering, Dimensionality Reduction

Three major issues

What to transfer?

- asks which part of knowledge can be transferred across domains or tasks.

- Some knowledge is specific for individual domains or tasks, and some knowledge may be common between different domains such that they may help improve performance for the target domain or task.

How to transfer?

- After discovering which knowledge can be transferred, learning algorithms need to be developed to transfer the knowledge, which corresponds to the "how to transfer" issue.

When to transfer?

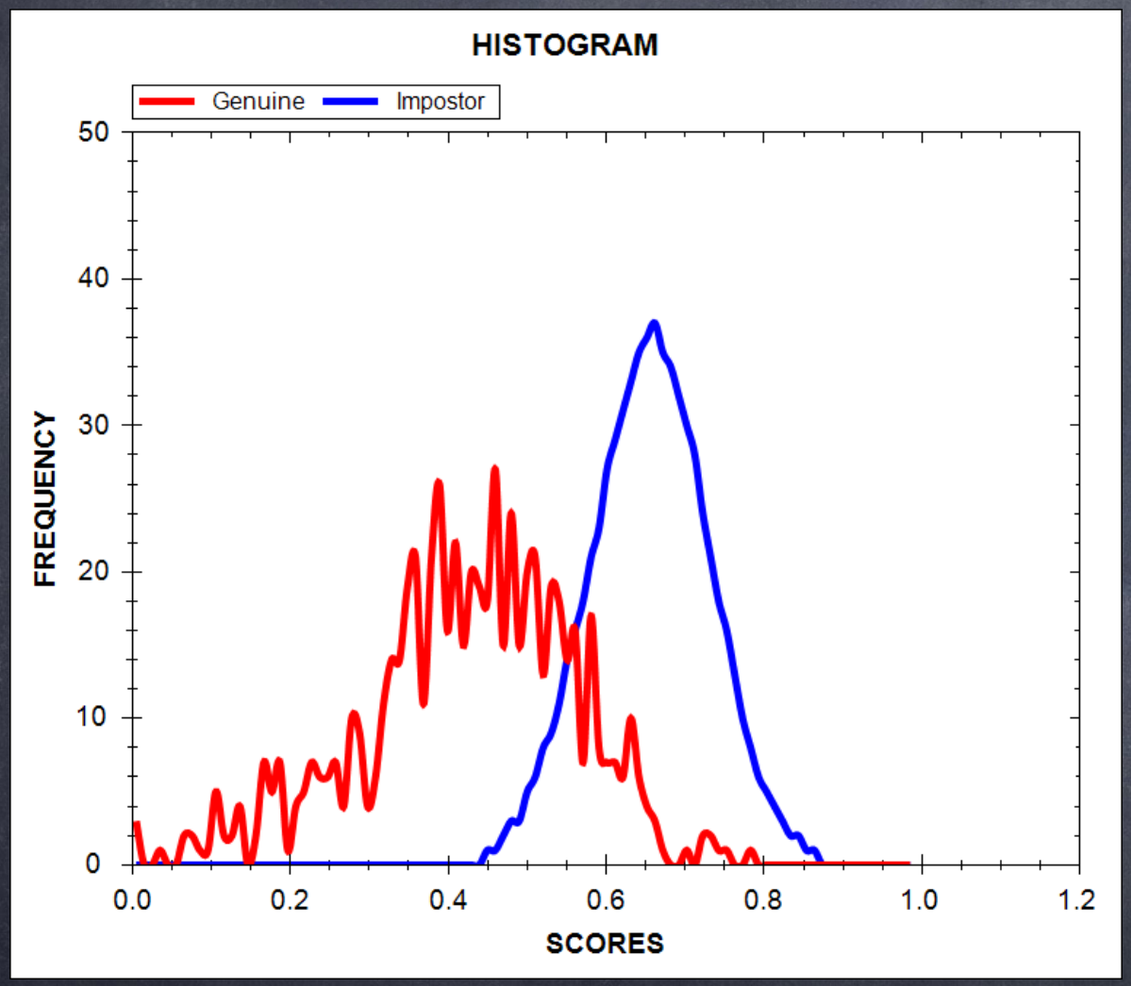
- asks in which situations, transferring skills should be done.
- in which situations, knowledge should not be transferred.

- In some situations, when the source domain and target domain are not related to each other, brute-force transfer may un-succeed.
- In the worst case, it may even hurt the performance of learning in the target domain, a situation which is often referred to as **negative transfer.**

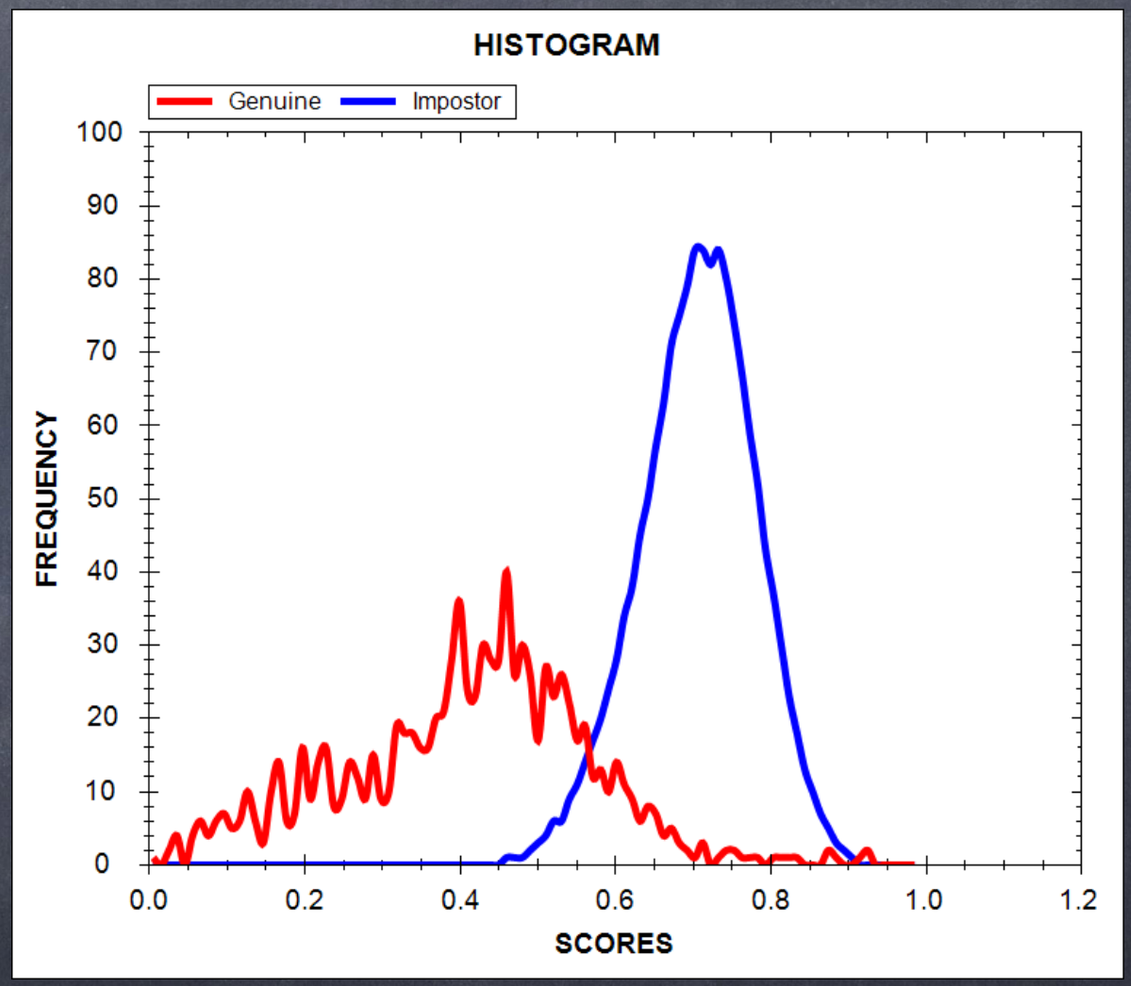
Incremental (Online) Learning

- Online learning is a concept inspired by human cognition
- Starting from childhood, a natural learning process constantly trains the human mind to develop new abilities and sustain old (useful) behavior
- It also learns to remove behaviors or patterns that become extraneous and redundant over time

Example - Match Score Distribution



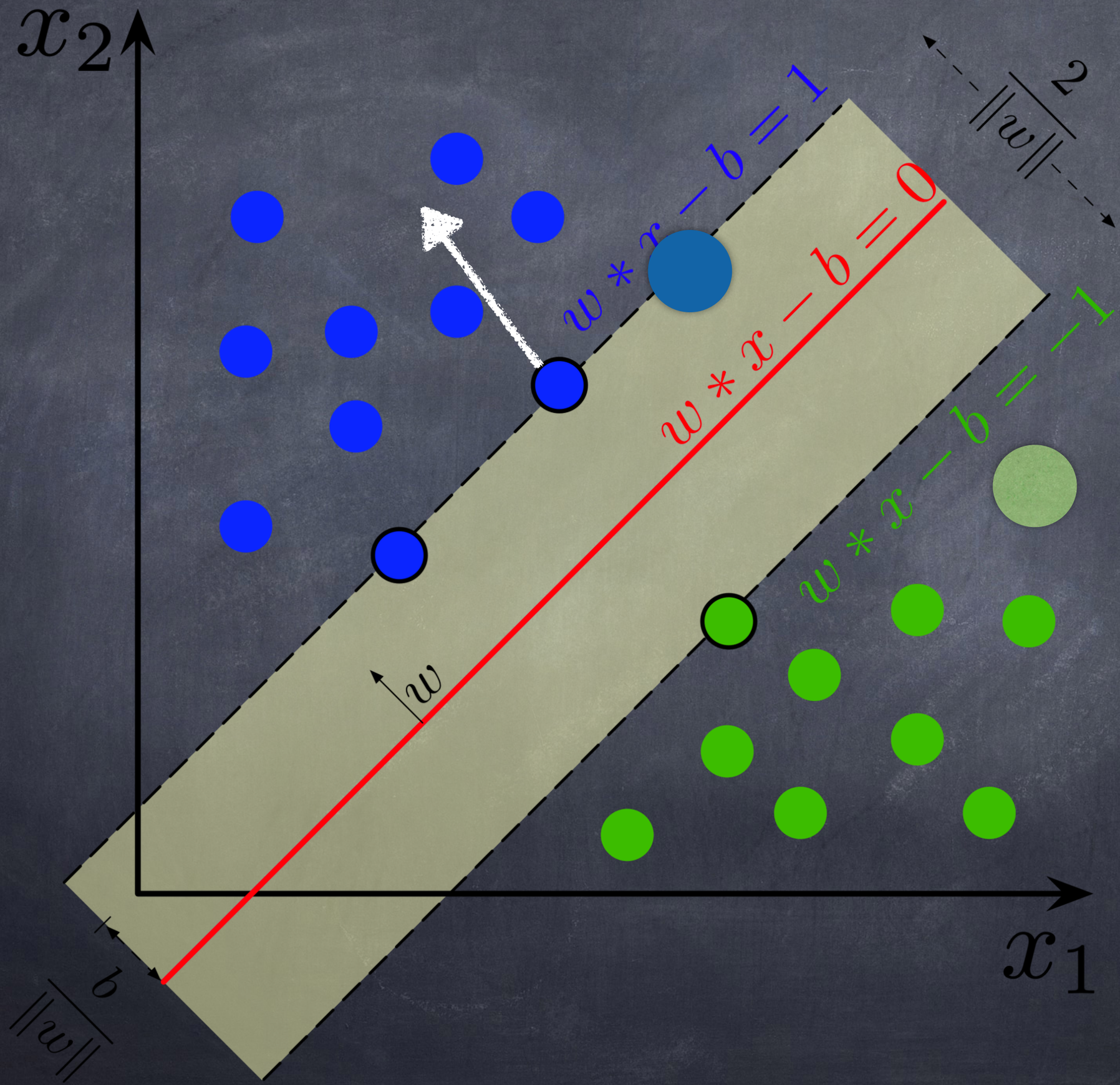
Match Score Histogram
with 100 Individuals



Match Score Histogram
with 300 Individuals

Incremental (Online) Learning in SVM

- Large margin classifier trained on entire data
- What if we have data coming in incremental fashion?
- Basic Idea: update Support Vectors to accommodate newer data points which cannot be classified using "old" model while retaining the KKT conditions on all previously seen data


x_2
$w * x - b = 1$
$w * x - b = 0$
$w * x - b = -1$
$\frac{2}{\|w\|}$
w
$\frac{b}{\|w\|}$
x_1

Online SVM

- SVM is trained using an initial training database and a decision hyperplane is obtained with m numbers of support vectors
- Check the prediction output for each new training data x_i,
  - if the prediction in incorrect, re-train using this new data

Other Online Classifiers

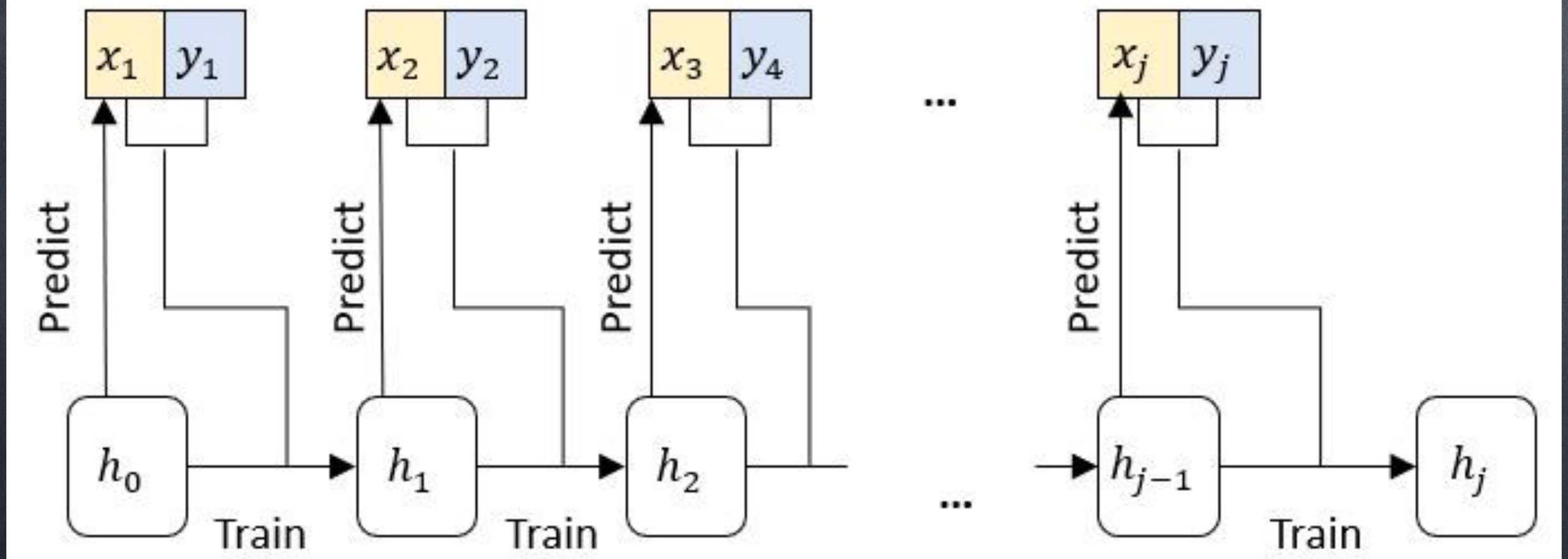
- Online Bayes: Recompute the probability distributions of individual classes and recompute the decision boundaries

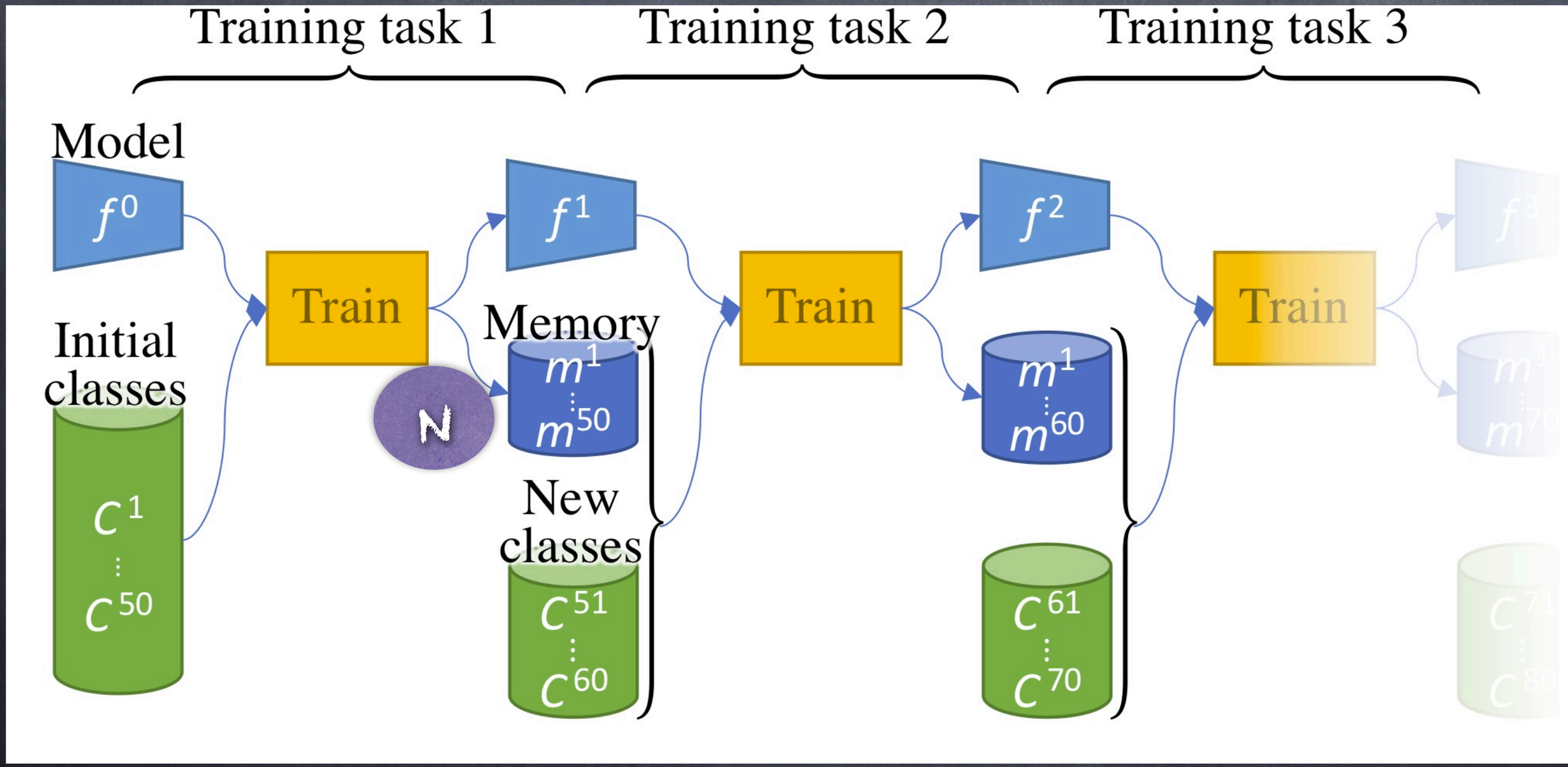
- Neural Network: Batch wise training can be considered as incremental learning

Types of Incremental Learning

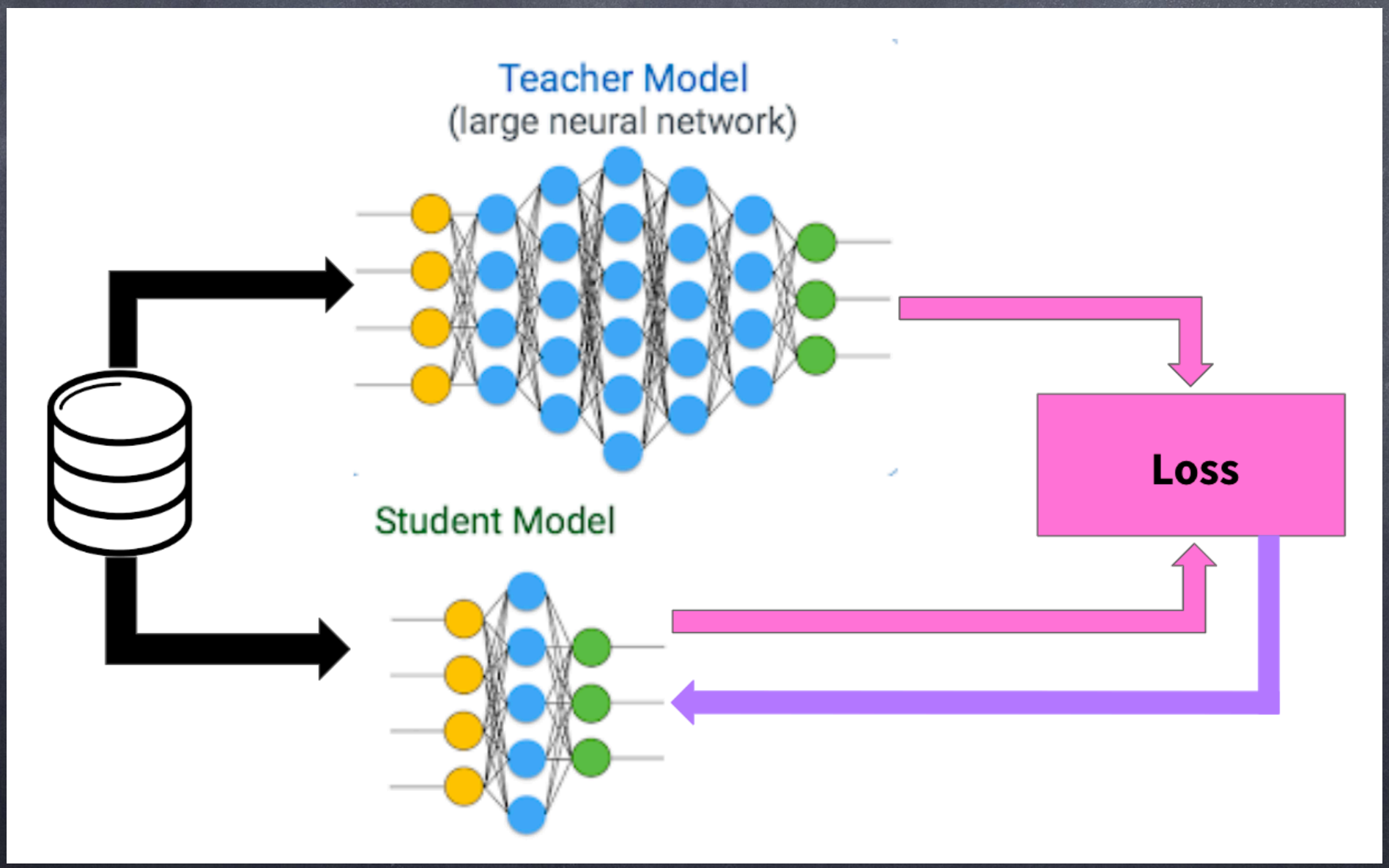
- Sample incremental learning
- Class incremental learning
- Task incremental learning
- Domain incremental learning

Incremental Neural Network

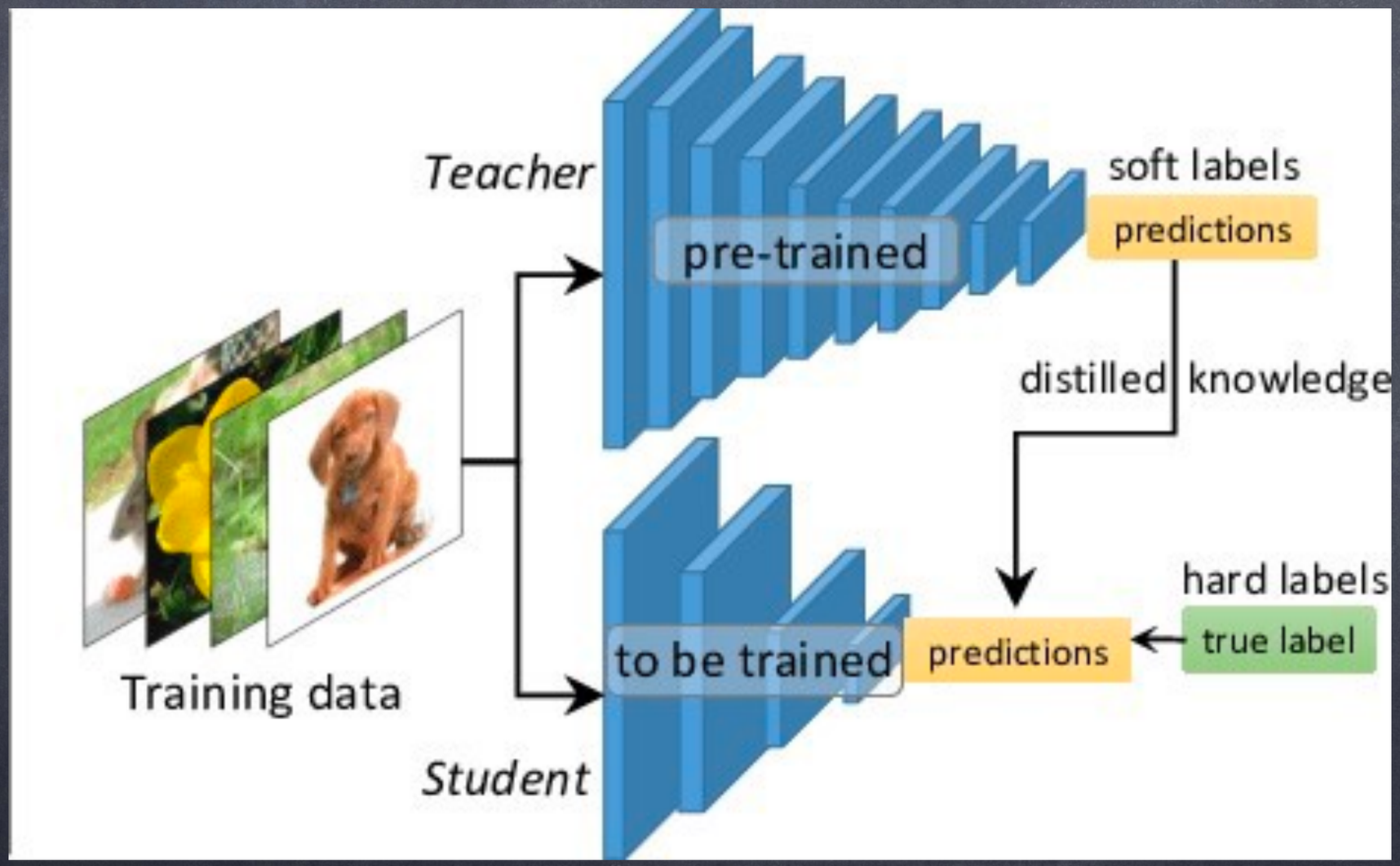



Training task 1
Training task 2
Training task 3
Model
f^0
f^1
f^2
Train
Train
Train
Initial classes
C^1
C^{50}
N
Memory
m^1
m^{50}
New classes
C^{51}
C^{60}
m^1
m^{60}
C^{61}
C^{70}

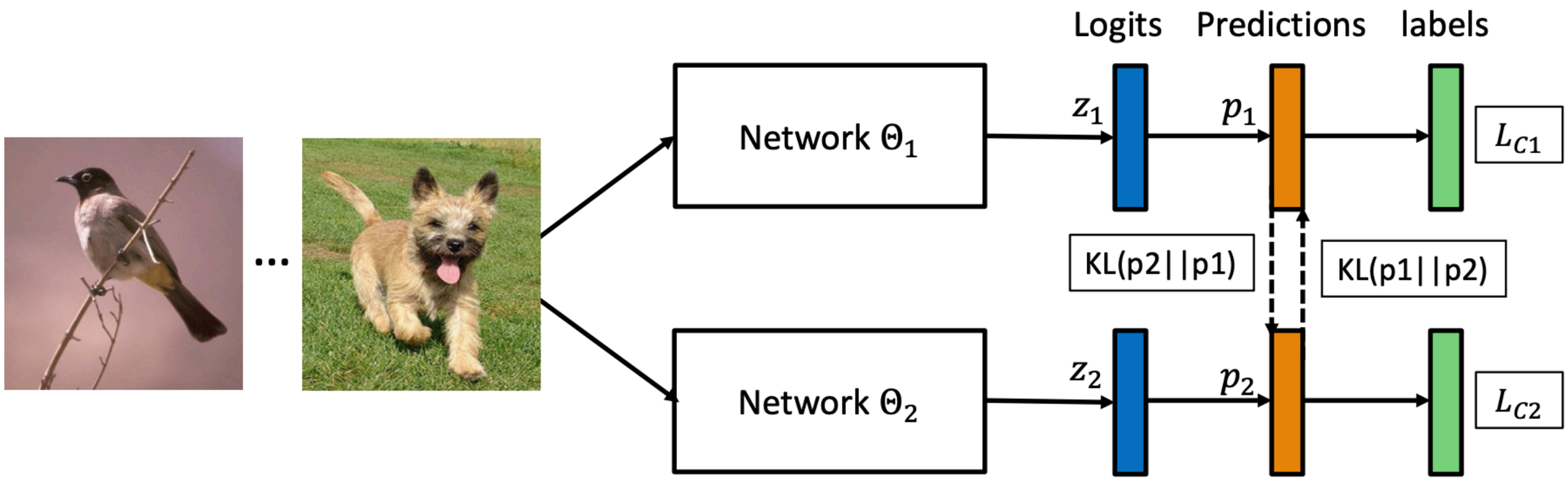
Student Teacher Network



Student Teacher Network



Deep Mutual Network



Co-training

- Two classifiers "co-train" each other
- Why two classifiers?

Co-training

- Semi-supervised learning technique with two "views", i.e. x1 and x2
- Each of the two feature sets is sufficient to classify the data
- Views should be sufficiently independent

How training works in Co-training

Step1: Train the classifiers using initial training data (e.g. two SVMs trained on individual views)

View 1	View 2	Label
x1(1)	x2(1)	+
x1(2)	x2(2)	+
x1(3)	x2(3)	-
x1(4)	x2(4)	-

How training works in Co-training

Step2: The classifiers classify the unlabeled training data

Use probability in SVM

C1(x1)

C2(x2)

Label the most confident instances only

partial labels

View 1	Lab	View 2	Label
x'1(1)		x'2(1)	+
x'1(2)	-	x'2(2)	
x'1(3)		x'2(3)	-
x'1(4)	+	x'2(4)	

partial labels

How training works in Co-training

Step3: Label all instances from one view to other

C1(x1)

C2(x2)

(e.g. Label from SVM1 is used to label instances in x2 and vice versa)

View 1	Label
x'1(1)	+
x'1(2)	-
x'1(3)	-
x'1(4)	+

View 2	Label
x'2(1)	+
x'2(2)	-
x'2(3)	-
x'2(4)	+

How training works in Co-training

Step4: Re-train the classifiers using these newly labeled instances

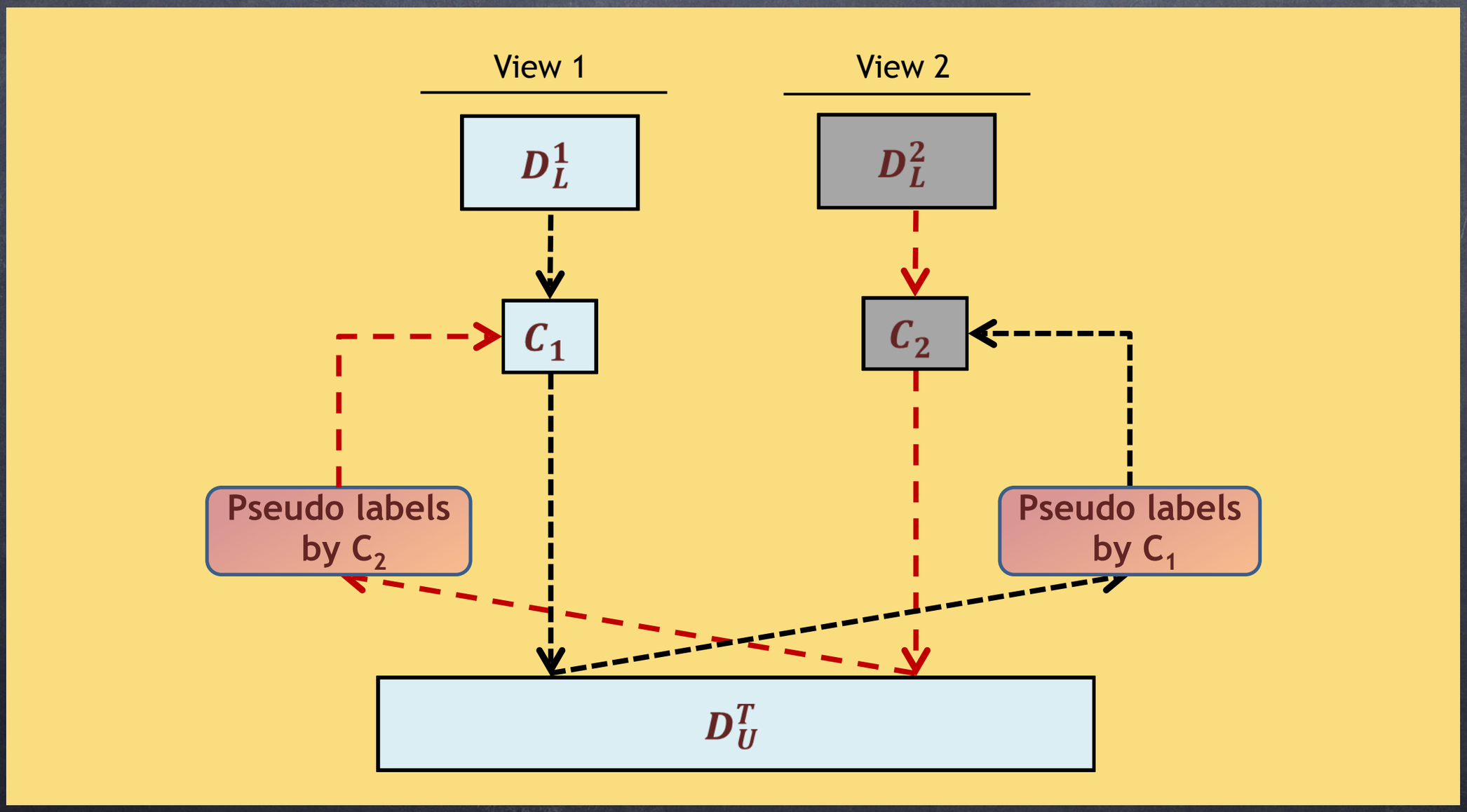
C1(x'1)

View 1	Label
x'1(1)	+
x'1(2)	-
x'1(3)	-
x'1(4)	+

C2(x'2)

View 2	Label
x'2(1)	+
x'2(2)	-
x'2(3)	-
x'2(4)	+

Co-training - in Summary



Co-training: What happens at the test time?

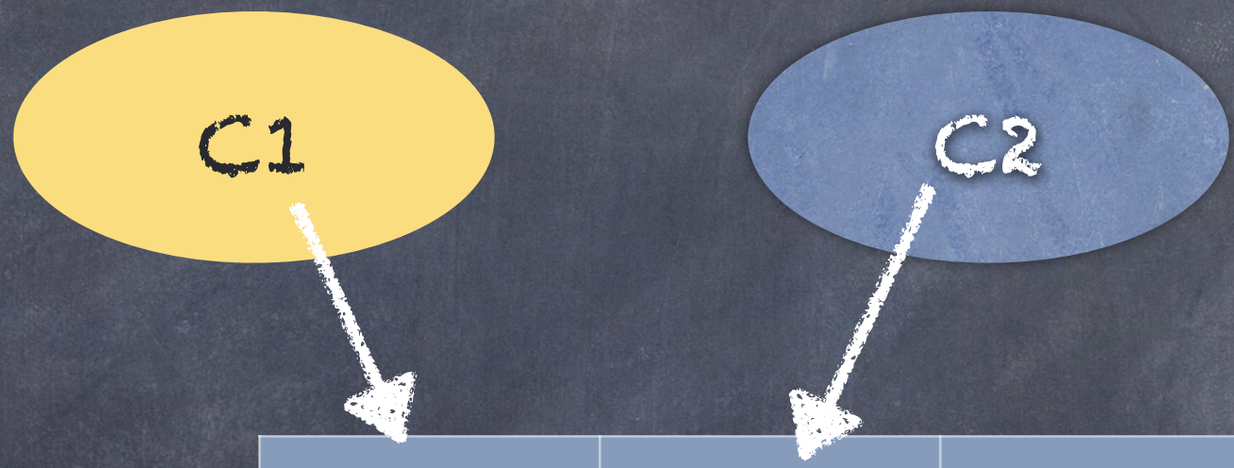
C1

C2

View 1	View 2	Label	
xt1(1)	xt2(1)	+	+
xt1(2)	xt2(2)	-	
xt1(3)	xt2(3)	-	+
xt1(4)	xt2(4)	+	-

More confident

Co-training: What happens at the test time?

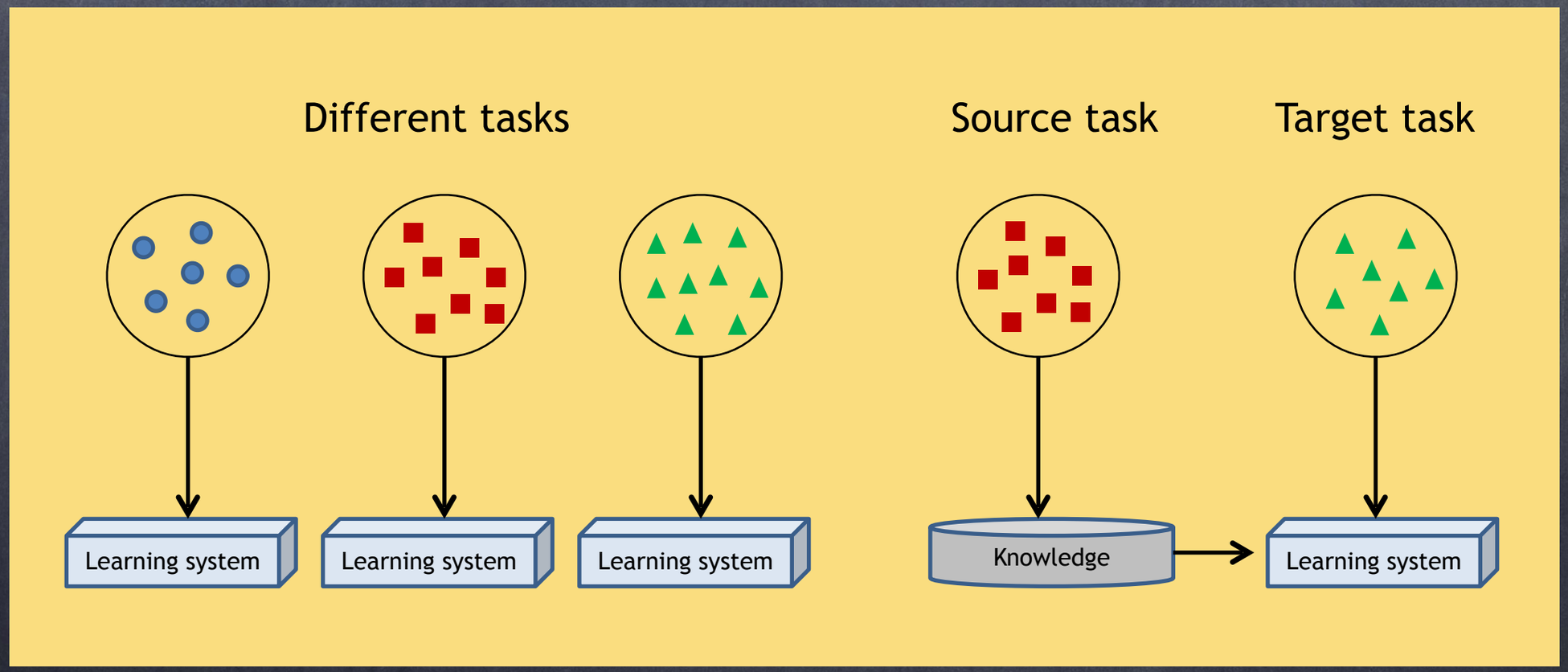


View 1	View 2	Label
xt1(1)	xt2(1)	+
xt1(2)	xt2(2)	-
xt1(3)	xt2(3)	-
xt1(4)	xt2(4)	-

Concept

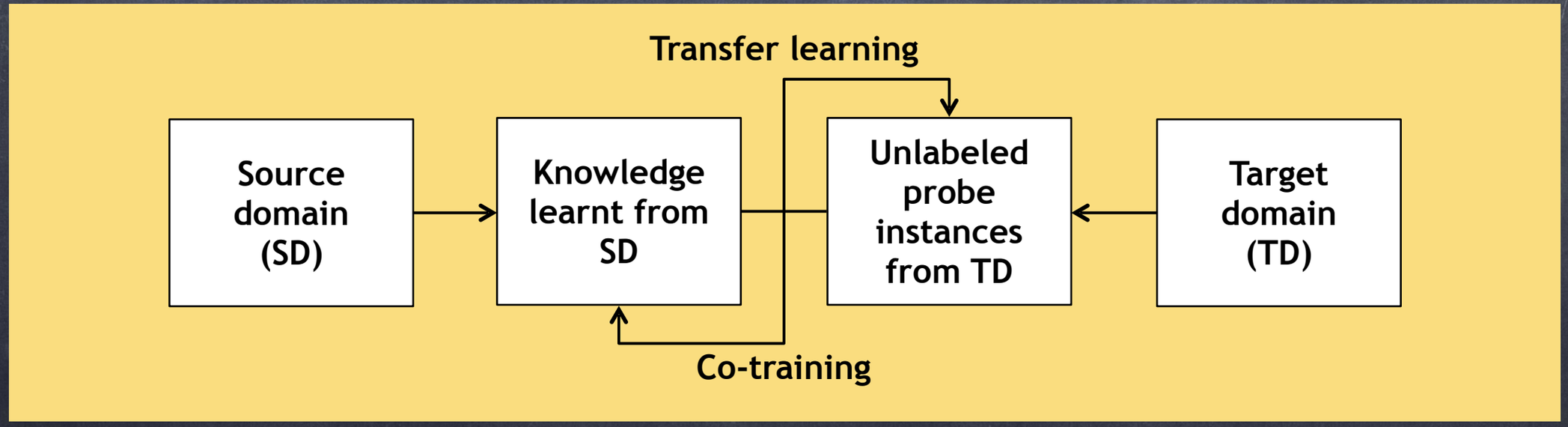
Co-Transfer Learning

Transfer Learning



Co-transfer Learning

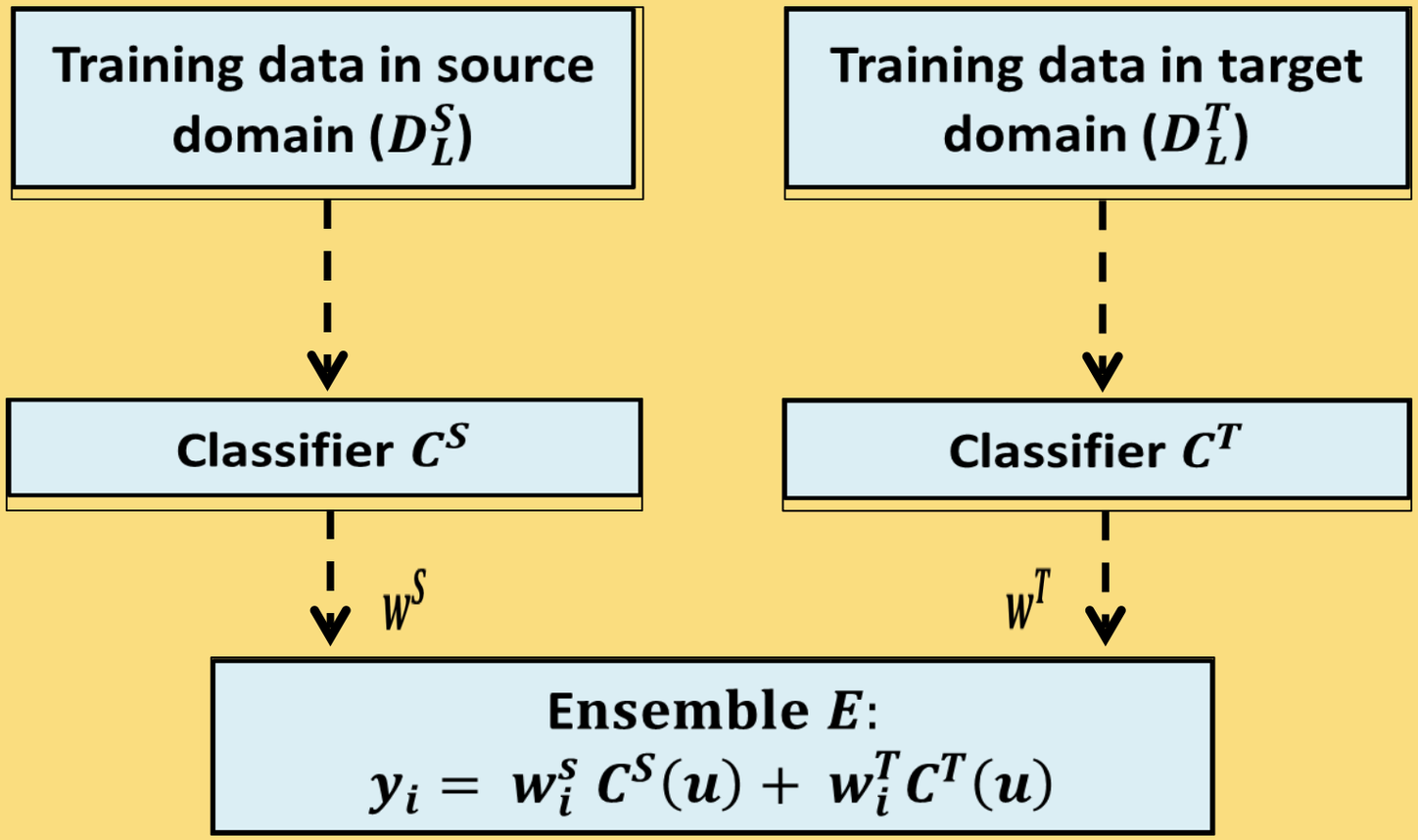
- Cross-pollination of transfer learning and co-training



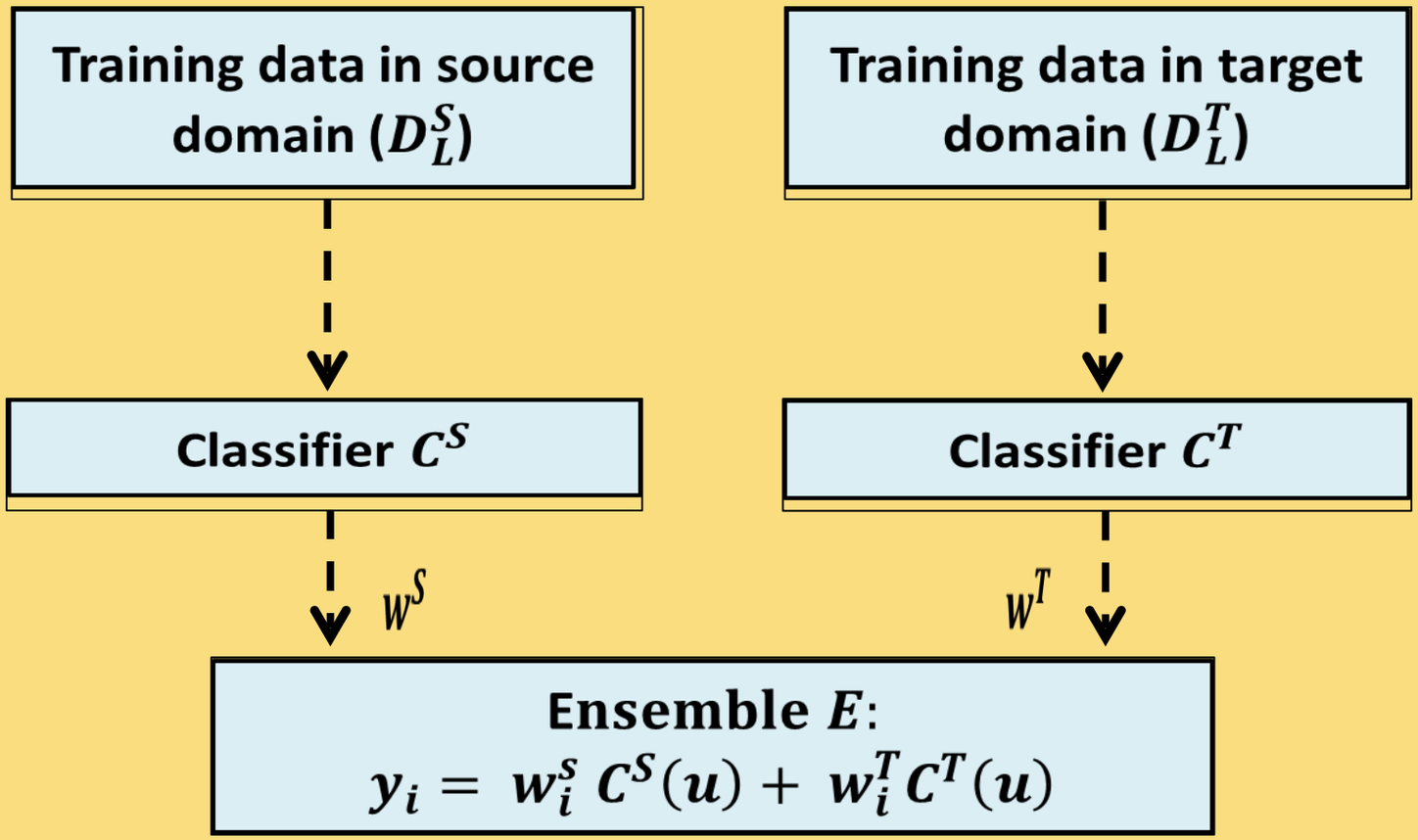
Co-transfer Learning

- Three major steps:
  - construct ensembles from different views using source and target domains
  - transfer learning in the source and target domains
  - co-train ensembles

Constructing Ensemble

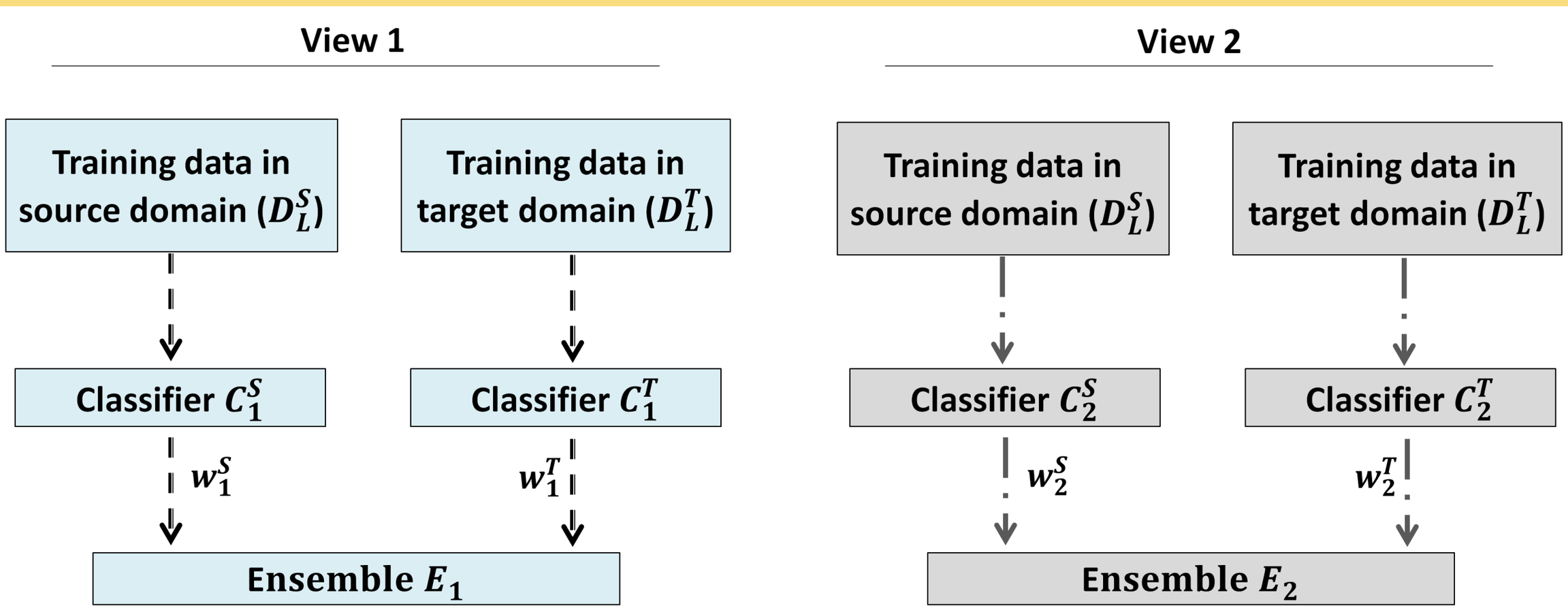


Transfer Learning

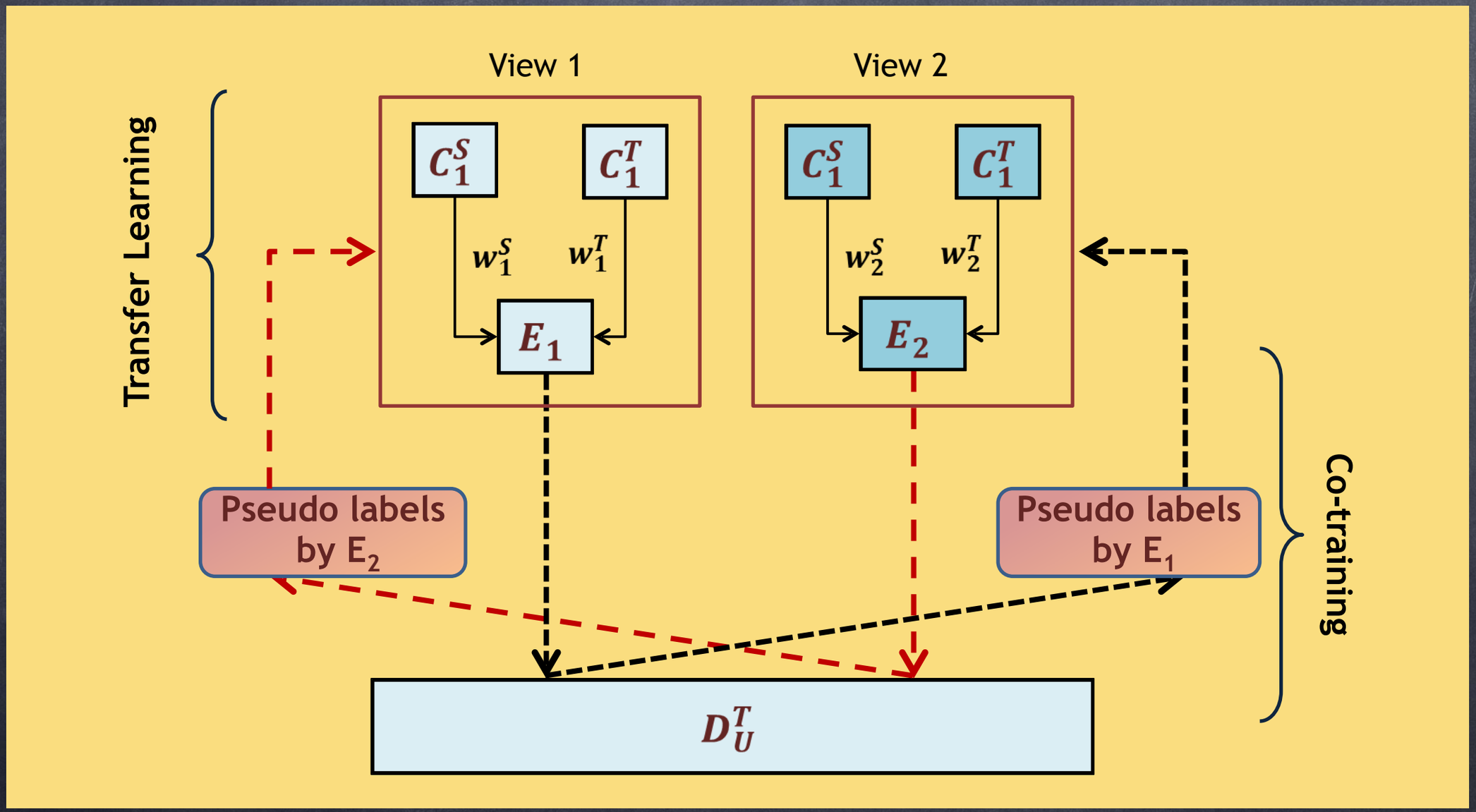


transfer learning with weight updates from Source to target

Co-training from Two Views



Co-transfer Learning



End-to-end incremental learning - ECCV2018