Web-Mining Agents: Transfer Learning TrAdaBoost

R. Möller
Institute of Information Systems
University of Lübeck

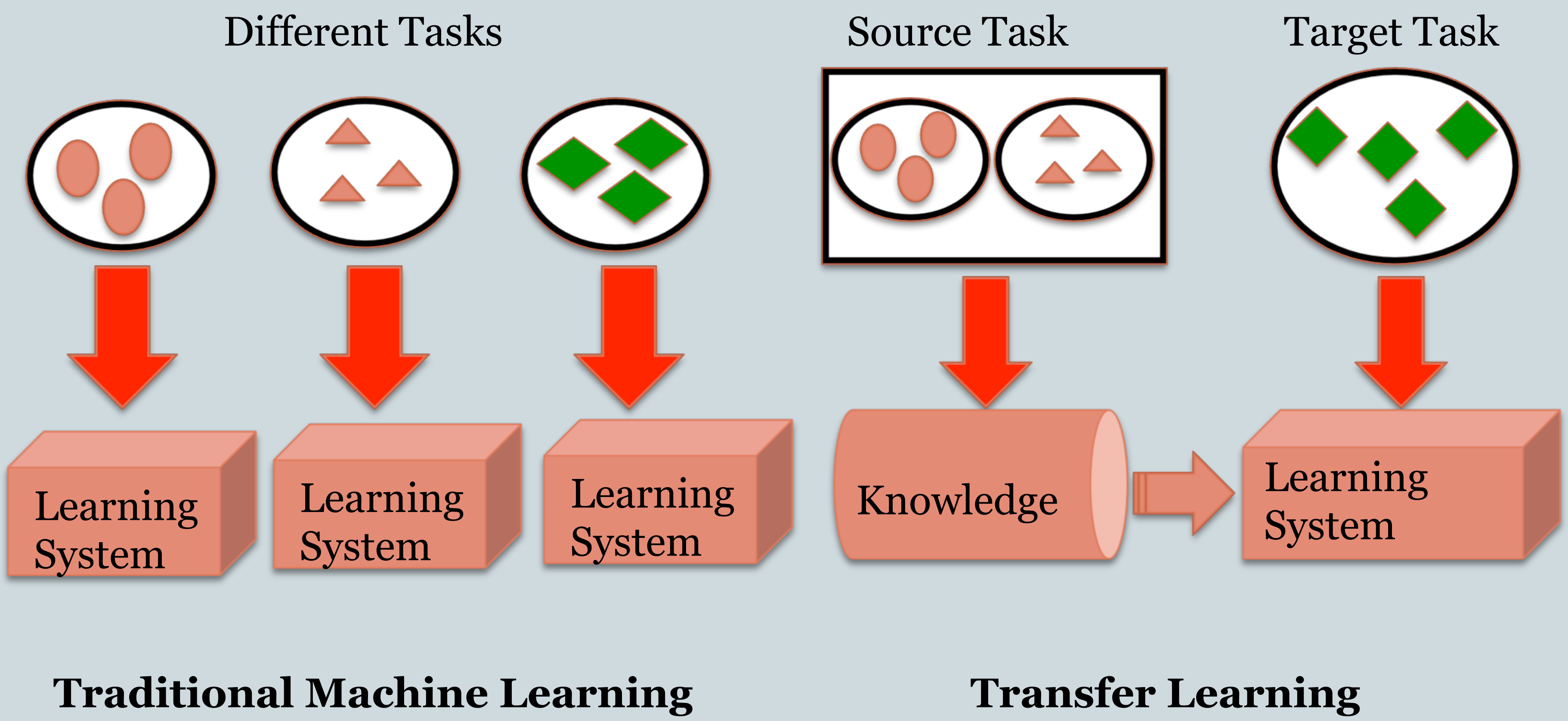
Based on an excerpt of:

Transfer for Supervised Learning Tasks

by: HAITHAM BOU AMMAR

MAASTRICHT UNIVERSITY

Traditional Machine Learning vs. Transfer



Transfer Learning Definition

- Notation:
  - Domain $\mathcal{D}$:
    - Feature Space: $\mathcal{X}$
    - Marginal Probability Distribution: $P(X)$
      - with $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$
  - Given a domain then a task is :

$$\mathcal{T} = \{\mathcal{Y}, f(.)\}$$

Label Space

P(Y|X)

Transfer Learning Definition

Given a source domain and source learning task, a target domain and a target learning task, transfer learning aims to help improve the learning of the target predictive function using the source knowledge, where

$$\mathcal{D}_s \neq \mathcal{D}_T \quad \textit{or} \quad \mathcal{T}_s \neq \mathcal{T}_T$$

Transfer Definition

- Therefore, if either :

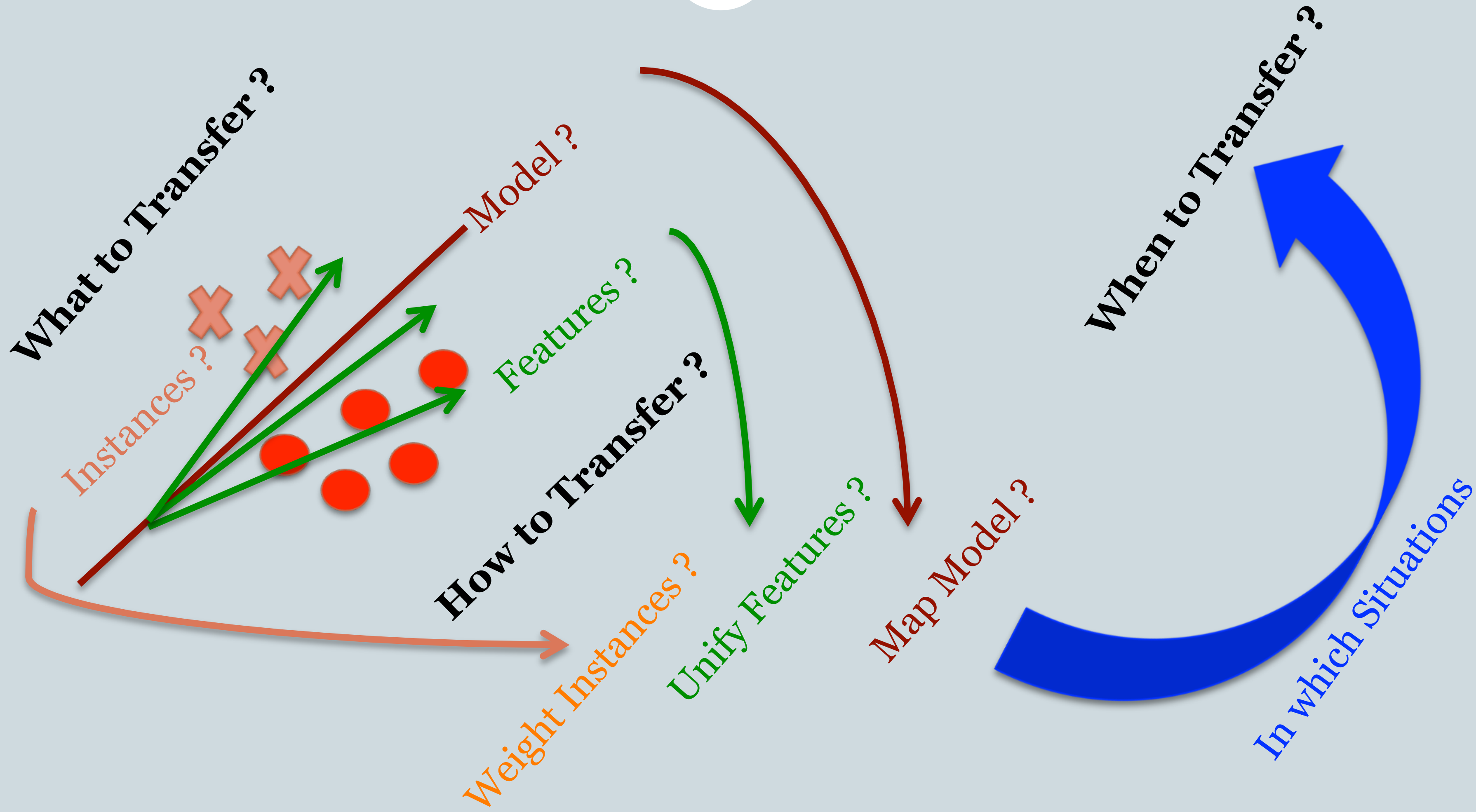
Domain Differences

$$\mathcal{X}_S \neq \mathcal{X}_T \qquad \mathcal{P}_S(X) \neq \mathcal{P}_T(X)$$

Task Differences

$$\mathcal{Y}_S \neq \mathcal{Y}_T \qquad P(Y_S|X_S) \neq P(Y_T|X_T)$$

Questions to answer when transferring



Algorithms: TrAdaBoost

- Assumptions:
  - Source and Target task have same feature space:

$$\mathcal{X}_S = \mathcal{X}_T$$

  - Marginal distributions are different:

$$P_S(X) \neq P_T(X)$$

Not all source data might be helpful !

Algorithm: TrAdaBoost

- Idea:
  - Iteratively reweight source samples such that:
    - reduce effect of “bad” source instances
    - encourage effect of “good” source instances
- Requires:
  - Source task labeled data set
  - Very small Target task labeled data set
  - Unlabeled Target data set
  - Base Learner

Algorithm: TrAdaBoost

Algorithm 1 TrAdaBoost Framework

1: **Require:** two labeled data sets $\mathcal{T}_D$ and $\mathcal{T}_s$, the unlabeled data set $\mathcal{S}$, a base learning algorithm Ξ, and the maximum number of iterations N.

2: **Initialize:** the weight vector $\mathbf{w}^{(1)} = [w_1^1, \ldots, w_{n+m}^{(1)}]^T$

3: **for** $t = 1$ to N **do**

4: Set $\mathbf{p}^{(t)} = \mathbf{w}^{(t)} / \left(\sum_{i=1}^{n+m} w_i^{(t)}\right)$

5: Learn a hypothesis $h^{(t)} : \mathcal{X} \rightarrow \mathcal{Y}$ by calling Ξ and passing the distribution $\mathbf{p}^{(t)}$ over the combined data set $\mathcal{T}$.

6: Compute the prediction error of $h^{(t)}$ on $\mathcal{T}_s$ using:

$$\epsilon^{(t)} = \sum_{i=n+1}^{n+m} \frac{w_i^{(t)} |h^{(t)}(x^{(i)}) - c(x^{(i)})|}{\sum_{i=n+1}^{n+m} w_i^{(t)}}$$

7: **end for**

8: Set $\beta^{(t)} = \frac{\epsilon^{(t)}}{1-\epsilon^{(t)}}$ and $\beta = \frac{1}{1+\sqrt{2\ln(n/N)}}$

9: Update the weight vector according to:

$$w_i^{(t+1)} = \begin{cases} w_i^{(t)} \beta^{|h^{(t)}(x^{(i)}) - c(x^{(i)})|} & \text{for } i = 1, \ldots, n \\ w_i^{(t)} \beta^{-|h^{(t)}(x^{(i)}) - c(x^{(i)})|} & \text{for } i = n+1, \ldots, n+m \end{cases} \quad (5)$$

10: **Output:**

$$h^{(f)}(x) = \begin{cases} 1 & \text{if } \prod_{t=\frac{N}{2}}^{N} \beta_t^{-h^{(t)}(x)} \geq \prod_{t=\frac{N}{2}}^{N} \beta_t^{-\frac{1}{2}} \\ 0 & \text{otherwise} \end{cases} \quad (6)$$

Weights Initialization

Hypothesis Learning and error calculation

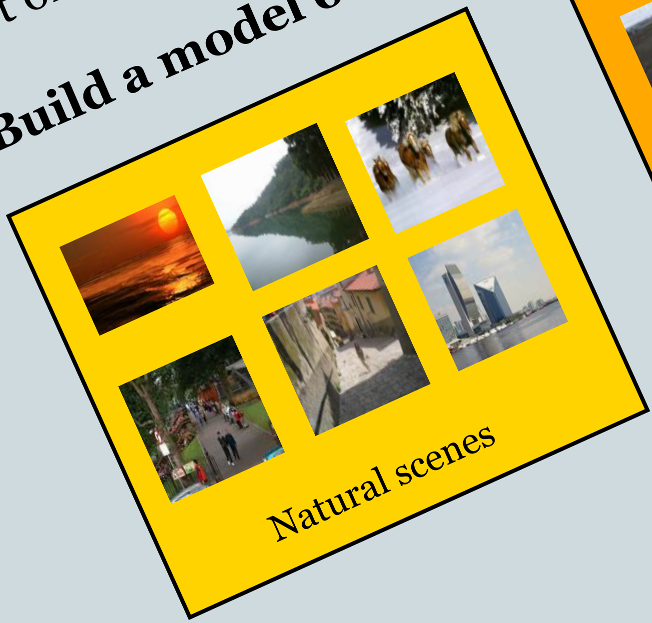
Weights Update

Algorithms: Self-Taught Learning

Problem Targeted is :

1. Little labeled data
2. A lot of unlabeled data

Build a model on the labeled data



Algorithms: Self-Taught Learning

- Assumptions:
  - Source and Target task have different feature space:

$$\mathcal{X}_S \neq \mathcal{X}_T$$

  - Marginal distributions are different:

$$P_S(X) \neq P_T(X)$$

  - Label Space is different:

$$\mathcal{Y}_S \neq \mathcal{Y}_T$$

Algorithms: Self-Taught Learning

- Framework:
  - Source Unlabeled data set:

$$D_S = \{(x_s^{(i)})\}_{i=1}^{m}$$

  - Target Labeled data set:

$$D_T = \{(x_T^{(j)}, y_T^{(j)})\}_{j=1}^{n} \text{ with } n <<< m$$

Build classifier for
cars and Motorbikes

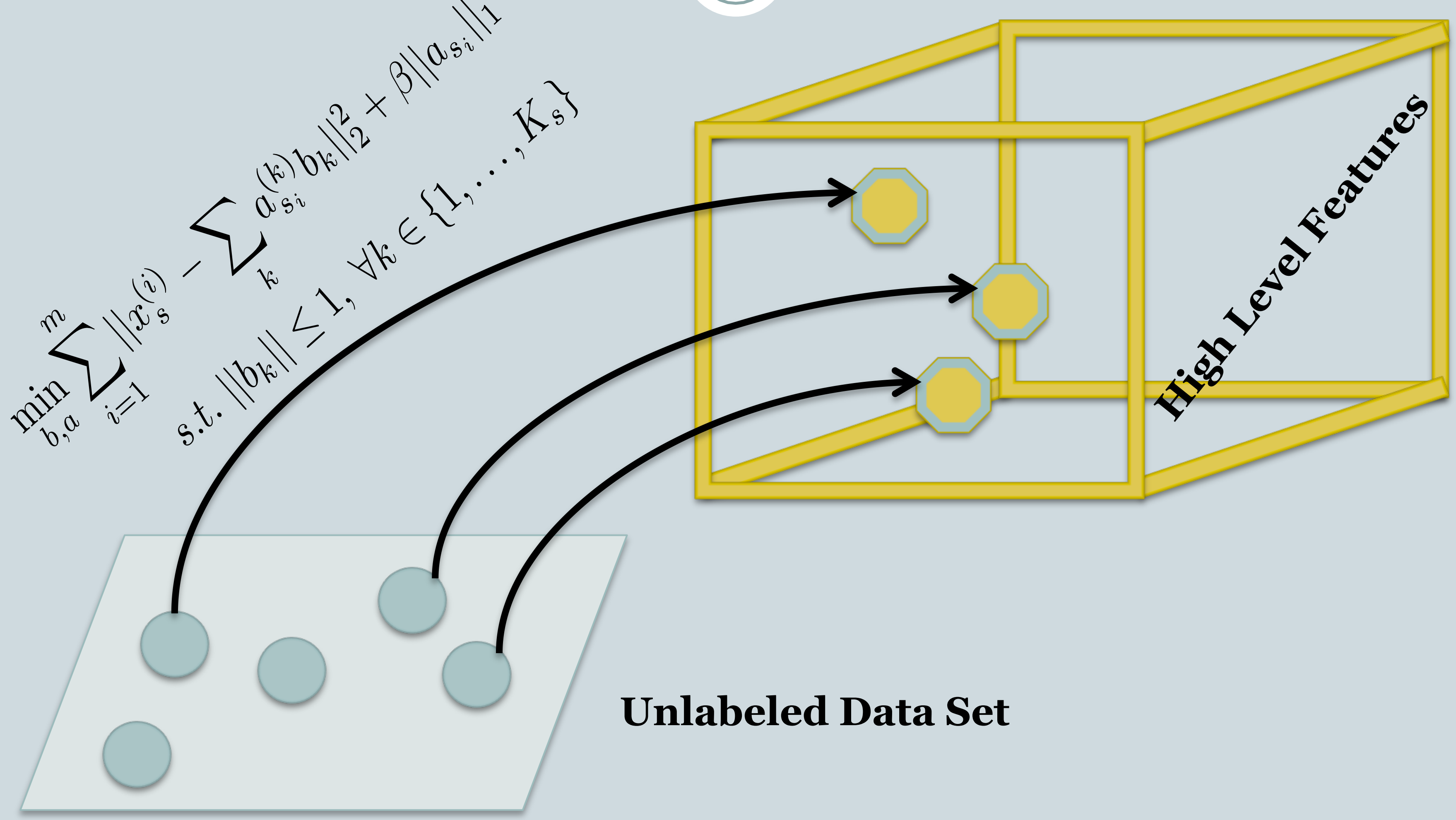
Algorithms: Self-Taught Learning

- Step One: *Discover high level features from Source data by*

$$\min_{b,a} \sum_{i=1}^{m} \underbrace{||x_s^{(i)} - \sum_k a_{s_i}^{(k)} b_k||_2^2}_{\text{Re-construction Error}} + \underbrace{\beta ||a_{s_i}||_1}_{\text{Regularization Term}}$$

$$\underbrace{s.t. \ ||b_k|| \leq 1, \ \forall k \in \{1, \ldots, K_s\}}_{\text{Constraint on the Bases}}$$

Algorithm: Self-Taught Learning



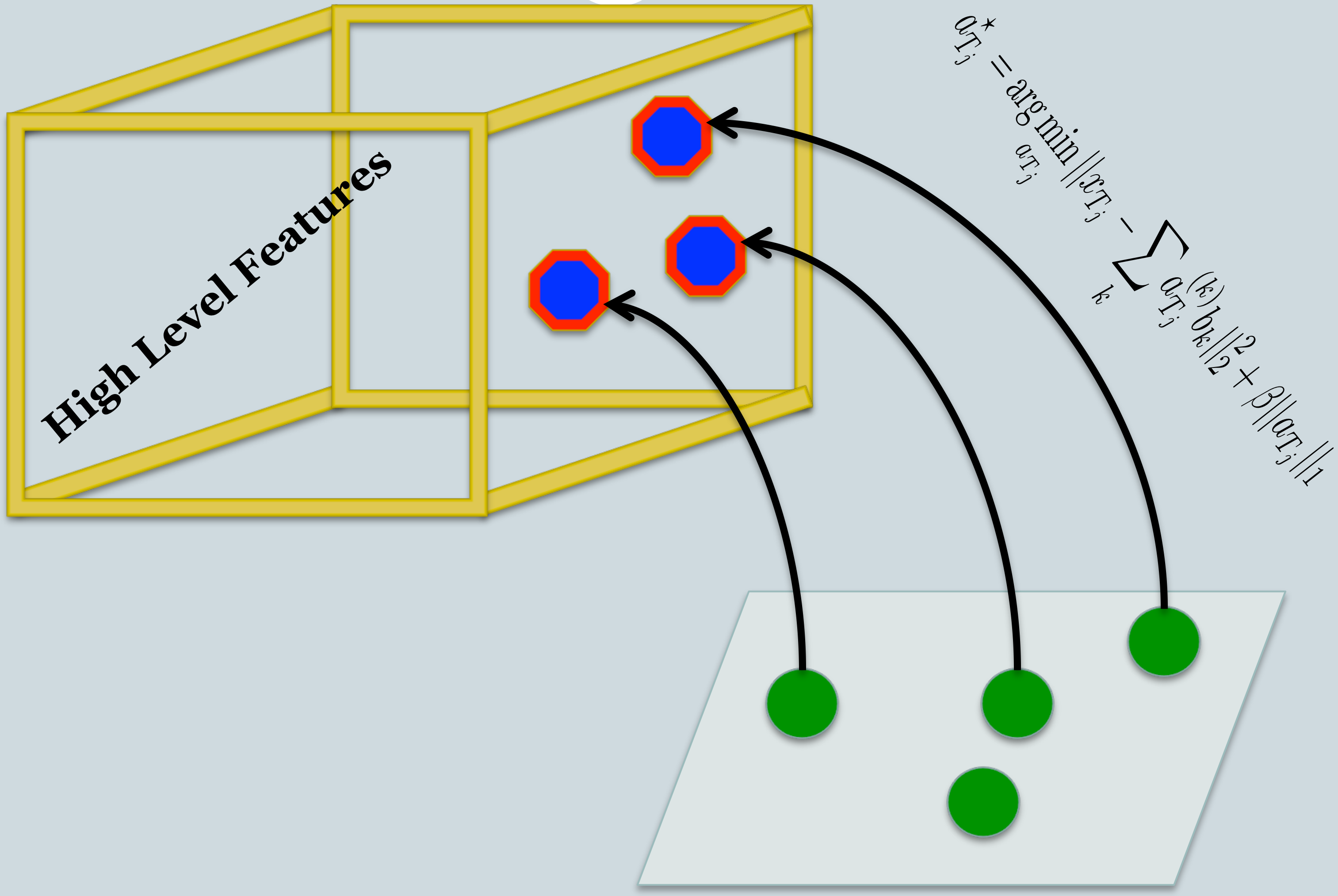
Algorithm: Self-Taught Learning

- Step Two: *Project target data onto the attained features by*

$$a^{\star}_{T_j} = \arg\min_{a_{T_j}} ||x_{T_j} - \sum_k a^{(k)}_{T_j} b_k||_2^2 + \beta ||a_{T_j}||_1$$

Informally, find the activations in the attained bases such that:
1. Re-construction is minimized
2. Attained vector is sparse

Algorithms: Self-Taught Learning



Algorithms: Self-Taught Learning

- Step Three: *Learn a Classifier with the new features*

input Labeled training set

$T = \{(x_l^{(1)}, y^{(1)}), (x_l^{(2)}, y^{(2)}), \ldots, (x_l^{(m)}, y^{(m)})\}.$ → Target Task

Unlabeled data $\{x_u^{(1)}, x_u^{(2)}, \ldots, x_u^{(k)}\}.$ → Source Task

output Learned classifier for the classification task.

algorithm Using unlabeled data $\{x_u^{(i)}\}$, solve the optimization problem (1) to obtain bases b. → Learn new features (Step 1)

Compute features for the classification task to obtain a new labeled training set $\hat{T} = \{(\hat{a}(x_l^{(i)}), y^{(i)})\}_{i=1}^{m}$, where

$\hat{a}(x_l^{(i)}) = \arg\min_{a^{(i)}} \|x_l^{(i)} - \sum_j a_j^{(i)} b_j\|_2^2 + \beta \|a^{(i)}\|_1.$ → Project target data (Step 2)

Learn a classifier $\mathcal{C}$ by applying a supervised learning algorithm (e.g., SVM) to the labeled training set $\hat{T}$. → Learn Model (Step 3)

return the learned classifier $\mathcal{C}$.

Conclusions

- Transfer learning is to re-use source knowledge to help a target learner
- Transfer learning is not generalization
- TrAdaBoost transfers instances
- Self-Taught Learning transfers unlabeled features

Next in Web-Mining Agents:

Unlabeled Features Revisited
Unsupervised Learning: Clustering