Proceedings of Machine Learning Research 129:145–160, 2020 ACML 2020

Boosting-Based Reliable Model Reuse

Yao-Xiang Ding DINGYX@LAMDA.NJU.EDU.CN
Zhi-Hua Zhou ZHOUZH@LAMDA.NJU.EDU.CN
National Key Laboratory for Novel Software Technology, Nanjing University, China.

Editors: Sinno Jialin Pan and Masashi Sugiyama

Abstract

We study the following model reuse problem: a learner needs to select a subset of models from a model pool to classify an unlabeled dataset without accessing the raw training data of the models. Under this situation, it is challenging to properly estimate the *reusability* of the models in the pool. In this work, we consider the model reuse protocol under which the learner receives *specifications* of the models, including reusability indicators to verify the models' prediction accuracy on any unlabeled instances. We propose MoreBoost, a simple yet powerful boosting algorithm to achieve effective model reuse under the idealized assumption that the reusability indicators are noise-free. When the reusability indicators are noisy, we strengthen MoreBoost with an active rectification mechanism, allowing the learner to query ground-truth indicator values from the model providers actively. The resulted MoreBoost.AR algorithm is guaranteed to significantly reduce the prediction error caused by the indicator noise. We also conduct experiments on both synthetic and benchmark datasets to verify the performance of the proposed approaches.

1. Introduction

As the scale of machine learning tasks gets larger, model training usually involves high time and sample complexity, thus it becomes increasingly expensive to learn from scratch. This promotes the growing need to utilize existing models for solving a new task. Modeling essential aspects under this scenario, the model reuse problem studies the following situation: given a pool of pre-trained models without their training data, a learner tries to reuse some of these models to make the current task to have a better performance. There are several properties to consider. First, the models are not necessarily to be trained from the same/similar task. In other words, not all the existing models are helpful for the current task. Moreover, the training data of these models are unavailable to the current user. As a result, there are two big challenges to deal with: (1) *to identify which models may be helpful to the current task;* (2) *to exploit these helpful models.* The good news is, though the training data are not observed, as described in the learnware paradigm (Zhou, 2016), each model is associated with a specification, which can be carefully designed to help the future learner to understand which model can be utilized and how to exploit it. The specification is the key to characterize the reusability of a model. It can have various of formulations, such as (1) descriptions of the original task from which the model is learned, e.g. information of the original feature/label space, learning objective and data distribution statistics; (2) a reusability indicator taking future instances as inputs, and outputting whether the model

can make accurate predictions; (3) high-level knowledge such as logical rules allowing to understand how the model can be utilized.

There have been several model reuse scenarios studied before. When we assume that all models are useful, the simplest way is to directly build the majority voting classifier with all existing models (Zhou, 2012). Alternatively, (Li et al., 2012) considers to reuse the existing models to adapt to new objective functions, and (Yang et al., 2017) considers to utilize the existing models as feature generators. These approaches do not consider the design of specifications, thus the model reusability is not explored. In (Wu et al., 2020), the reducible kernel mean embedding (RKME) specification is proposed, which is the first work to consider the design of the specification. The RKME specification provides data density information of the original task under which the model is trained. On the other hand, RKME is only designed for the scenario where the specific mixture density assumption holds. In this work, we consider a more general model reuse scenario without relying on such problem-specific assumption: for any model, there is a reusability indicator included in the specification, which takes any unlabeled instance as input, and outputs a score to measure the model's ability to make accurate prediction on it. Furthermore, we aim at dealing with the following challenge: the reusability indicator can only be prepared by the local model provider with the limited local information, thus it is not guaranteed to correctly estimate reusability on any future instances. To deal with this issue, we consider the setting where the target task learner can query ground-truth indicator outputs from model providers to rectify the indicators. This provides a preliminary exploration to realize the *evolvability* demanded by learnware (Zhou, 2016), that is, the learnware is able to evolve to adapt to environmental changes and improve its own performance/usability.

Our contributions are listed as follows. In Section 2, we introduce the reusability-aware model reuse protocol with reusability indicator specifications. In Section 3, we propose MoreBoost (Model reuse Boosting), a simple yet powerful boosting-based model reuse algorithm. We show that under the noise-free case, MoreBoost is guaranteed to minimize the prediction error. In Section 4, we show that MoreBoost suffers from significant performance degeneration when the reusability indicators are inaccurate. To solve this issue, we propose an active rectification mechanism to rectify the reusability indicators. We show that the resulted MoreBoost.AR algorithm (MoreBoost with Active Rectification) again enjoys strong theoretical guarantees. In Section 6, we provide experimental results on synthetic and benchmark datasets to show the desired performance of our approaches.

We utilize bolded characters to denote a set of enumerated elements by their subscripts. The length of enumeration is clear from the content. For example, $\mathbf{w}$ denotes the set $\{w_s|s \in [T]\}$ in which $[T] \equiv \{1, 2, \ldots, T\}$. We also utilize $\mathbb{I}$ to denote the indicator function.

2. Reusability-Aware Model Reuse Protocol

Assume that the global input and output spaces are $\mathcal{X}, \mathcal{Y}$, where $\mathcal{Y}$ is a finite set of classes of size L, i.e. $[L]$. We assume that there is a global labeling function f^* mapping from $\mathcal{X}$ to $\mathcal{Y}$, which generate labels for data sampled from any input distributions $\mathcal{D}$ over $\mathcal{X}$.

There are K model providers involved in our problem. Each provider k observes her local dataset $S_k = (X_k, Y_k) = \{(x, y) \in \mathcal{X} \times \mathcal{Y}_k\}$. For each S_k, the instances are generated by sampling from an unknown input distribution $\mathcal{D}_k$ over $\mathcal{X}$, and then labeling by f^*. Note

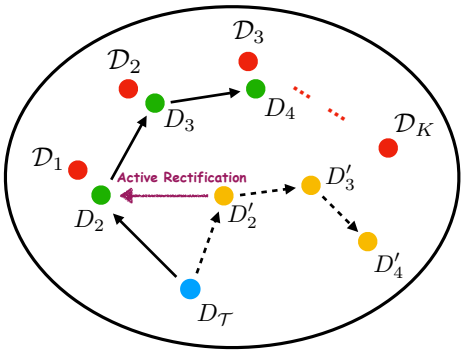


Figure 1: Illustration of MoreBoost and MoreBoost.AR. The initial new task data distribution $D_{\mathcal{T}}$ on $\mathcal{U}$ may not be close to any of the existing task data distributions $\mathcal{D}_k, k \in [K]$. Under the noise-free case, the changed data distributions D_t over MoreBoost could get much closer to $\mathcal{D}_k$. While under the noisy case, the mismatch between D_t and $\mathcal{D}_k$ may accumulate over iterations. Thus it is necessary to introduce the active rectification mechanism in MoreBoost.AR.

Algorithm 1 Reusability-Aware Model Reuse Protocol

Model Upload: Models and reusability indicators $\mathbf{f}, \mathbf{h}$.
Input: New task data $\mathcal{U}$, error upper bounds $\bar{\boldsymbol{\sigma}}$.
repeat {*Model Selection*}
  repeat {*Function Indicator Rectification*}
    Choose $k \in [K], x \in \mathcal{U}$ to query $h_k^*(x)$.
    Update $h_k(x) = h_k^*(x)$.
  until Stop condition reached.
  Do model reuse algorithm iteration.
until Total number of iterations reaches.
Output: $F(\{f_{k_m}\}, \{h_{k_m}\}), m \in [M]$.

that the input spaces for all parties are the global one $\mathcal{X}$, while the label spaces are subsets of the global label space, i.e. $\mathcal{Y}_k \subseteq \mathcal{Y}$. The cardinality of $\mathcal{Y}_k$ is L_k. We assume that $\mathcal{Y}_k$ can have intersections. Furthermore, $\cup \mathcal{Y}_k = \mathcal{Y}$. Each provider learns a *local model* f_k using her local dataset S_k, which is a multi-class classifier mapping from $\mathcal{X}$ to the product space $[0,1]^{L_k}$. For the convenience of model reuse, we assume that the provider knows the global label space $\mathcal{Y}$, and f_k is augmented into a classifier $f_{k,\mathcal{Y}} : \mathcal{X} \times [0,1]^L$ in the global label space by setting the model outputs in $\mathcal{Y}_k$ as f_k, and $\mathcal{Y} \setminus \mathcal{Y}_k$ as zeros. The label assignment rule is given by $\bar{f}_k(x) = \arg\max_{l \in [L]} f_{k,\mathcal{Y}}(x, l)$, in which $f_{k,\mathcal{Y}}(x, l)$ is the l-th output of $f_{k,\mathcal{Y}}(x)$.

For each provider, besides the classification model, there is also a certificate of model reusability, i.e. reusability indicator h_k trained with local dataset S_k. h_k is a function from $x \in \mathcal{X}$ to $\{0, 1\}$. We assume that for $\forall k \in [K]$, the noise-free reusability indicator h_k^* satisfies the following relationship:

Assumption 1

$$h_k^*(x) = 1 \implies p(f_k(x) \neq f^*(x)) \leq \epsilon_k, \forall x \in \mathcal{X},$$

in which ϵ_k is a small non-negative error rate.

Thus if $h_k^*(x) = 1$, then f_k is reusable on x, otherwise this is not guaranteed. The reusability indicators can be implemented in many ways. For example, they can be any predictors measuring the distance between a future instance to the original task data distribution, such as outlier detectors (Schölkopf et al., 2001; Liu et al., 2008). Furthermore, if there is side information available for any instance x (such as additional text descriptions for image instances), h_k can also be predictors based on it. However, since h_k can only be trained with local dataset S_k, it is unlikely to make correct predictions on all future instances. Thus we model h_k as an approximation of h_k^* under Bernoulli noise, such that for $\forall x \in \mathcal{X}$, $h_k(x)$ is generated by flipping $h_k^*(x)$ with a certain probability $\sigma_k(x)$. We assume that the noise upper bounds

$$\bar{\sigma}_k \geq \sigma_k(x) = p(h_k(x) \neq h_k^*(x)), \forall x \in \mathcal{X}$$

can be observed by any future learner. In practice, $\bar{\sigma}_k$ can be obtained through various ways, such as testing h_k under a small labeled hold-out validation set collected by future learners.

The objective of our model reuse problem is as follows. **A new user needs to reuse the existing models to solve her own task**: There are N unlabeled data $\mathcal{U} = \{x_i, i \in [N]\}$ sampled from an unknown input distribution $\mathcal{D}_\mathcal{T}$ over $\mathcal{X}$. We assume that the underlying true label space is global, i.e. $\mathcal{Y}$. The learner faces with a transductive learning problem: She chooses a subset $f_{k_1}, f_{k_2}, \ldots, f_{k_M}$ out of the K existing models, and outputs $F(\{f_{k_m}\}, \{h_{k_m}\}), m \in [M]$, a transductive classifier to predict on all $x \in \mathcal{U}$. As described above, we assume that the learner receives all reusability indicators from the K model providers as well as their noise rate upper bounds to identify models' reusability.

Since the reusability indicators could be inaccurate, the learner is assumed to have the privilege of rectifying the reusability indicators. In other words, she can actively choose any $x \in \mathcal{U}$, and query for $h_k^*(x)$ from any model provider k to refine h_k during the running of the model reuse algorithm. Note that querying ground-truth indicator value is much easier than querying ground-truth labels, since the model provider only needs to verify whether the instance is indeed similar to the model training data. The overall model reuse protocol is illustrated in Algorithm 1.

3. Model Reuse by Boosting

In this section, we introduce the idealized assumption that reusability indicators are free from error, i.e. $\mathbf{h} = \mathbf{h}^*$. Under this situation, there is no need for the learner to query the ground-truth indicator values. Motivated by the deep connection between boosting and online learning with expert advice (Freund and Schapire, 1997), we proposed MoreBoost, a simple yet powerful boosting approach to solve the model reuse problem under the noise-free case. We will see that MoreBoost has a fast convergence rate to output a transductive classifier to maximize the labeling accuracy on $\mathcal{U}$. Meanwhile, it could achieve near-optimal performance in selecting the smallest subset of models to reuse. The process of MoreBoost is illustrated in Algorithm 2. As in classical boosting, the key idea lies in maintaining a weight distribution D over all current task instances. In Line 4, we calculate the reusability score of each model by the weighted sum of reusability indicator predictions. It is easy to see

Algorithm 2 MoreBoost

1: **Input**: Current task data $\mathcal{U}$, existing models and reusability indicators $\mathbf{f}, \mathbf{h}$, weight update parameter η, number of total iterations T.
2: Initialize $D_1(x_i) = 1/N$ for $i \in [N]$.
3: **for** $t = 1$ **to** T **do**
4: Choose the optimal model index as $k_t = \arg\max_{k\in[K]} \sum_{i=1}^{N} D_t(x_i)h_k(x_i)$.
5: Set loss $c_t(x_i) = h_{k_t}(x_i), i \in [N]$.
6: Update weights: $D_{t+1}(x_i) = D_t(x_i)\exp(-\eta c_t(x_i))/\sum_{i=1}^{N} D_t(x_i)\exp(-\eta c_t(x_i))$.
7: **end for**
8: **Output**: Predict $x \in \mathcal{U}$: $F_T(x_i) = \arg\max_{l\in[L]} \sum_{t=1}^{T} h_{k_t}(x_i)\mathbb{I}[\bar{f}_{k_t}(x_i) = l]$.

that the higher the score, the better the model will perform on high-weight instances. The optimal model is selected greedily according to the reusability score. The next key step is to update the sample distribution (Line 6) with the loss defined in Line 5. The key intuition is that if the instance is *covered* by the current reusability indicator, then its weight should be decreased, otherwise we should pay more attention to it by increasing its weight. We introduce a weight update parameter η to control the aggressiveness of weight shrinkage once an instance is covered. In practice, the more times that we want an instance to be covered, the smaller η can be used. Finally, different from common boosting, a transductive classifier F_T is outputted by MoreBoost, which can be utilized to label the instances in $\mathcal{U}$. To proceed on analysing the theoretical properties of MoreBoost, we propose its weak learning conditions below.

Assumption 2 (Weak learning condition for MoreBoost.) *For any data distribution D over $\mathcal{U}$, there exists $k \in [K]$, such that $\sum_{i=1}^{N} D(x_i)h_k^*(x_i) > \gamma > 0$.*

The weak learning condition ensures that in any iteration, the fraction of instances which are covered is at least γ. The following result provides the convergence rate of MoreBoost.

Theorem 1 *Let the prediction error of F_T be $\epsilon_{F_T} = \frac{1}{N}\sum_{i=1}^{N} p(F_T(x_i) \neq f^*(x_i))$, and denote by $\bar{\epsilon} = \max_{k\in[K]} \epsilon_k$. When $\eta = \sqrt{(\ln N)/T}$, after running MoreBoost for $T \geq \mathcal{O}(\ln N/\gamma^2)$ iterations, we have $\epsilon_{F_T} \leq \bar{\epsilon}$.*

From Theorem 1, we can see that MoreBoost requires very few iterations to run if γ is not too small. On the other hand, even if γ is small, the algorithm still guarantees to converge. To further verify whether MoreBoost tends to choose a small set of useful models, we take an alternative view of the model reuse problem. We can treat $\mathbf{f}$ and $\mathcal{U}$ as two sets of graph vertices. Then the indicators $\mathbf{h}$ define the edge between these two sets, such that x_i connects to f_k only when $h_k(x_i) = 1$. Thus $\mathbf{f}, \mathcal{U}, \mathbf{h}$ jointly define a bipartite graph. It is easy to see that MoreBoost tries to find the minimum subset of $\mathbf{f}$ that do the best on covering more instances. This exactly coincides with the target of choosing a small set of useful models. In particular, if we only require each instance to be covered by at least one model, then this is exactly the set cover problem, which is among Karp's 21 NP-complete problems (Karp, 1972). It is also proved that the greedy algorithm can achieve the near-optimal approximation ratio for set cover under the P≠NP conjecture (Dinur and Steurer, 2014).

Table 1: Example to show why error rates grow for MoreBoost. The checkmark in row k, column i indicates that $h_k^*(x_i) = 1$, otherwise $h_k^*(x_i) = 0$. Numbers indicate the indicator error rates.

	x_1	x_2	x_3	x_4	x_5	x_6
f_1	✓, 0.1	✓, 0.1	✓, 0	✓, 0	0	0
f_2	0	0	0	0	✓, 0.1	✓, 0.1
f_3	1	1	0	0	1	0
f_4	0	0	1	1	0	1

We can show the close relationship between MoreBoost and the greedy algorithm for set cover.

Theorem 2 *When $\eta \to \infty$, MoreBoost is equivalent to the greedy set cover algorithm.*

This shows that MoreBoost can be treated as the relaxed version of the greedy set cover algorithm, which utilizes a soft weight update rule. This verifies that MoreBoost achieves significant performance in keeping the set of chosen models as small as possible.

4. Rectifying Function Indicators

In the previous section, we show that MoreBoost is effective when reusability indicators $\mathbf{h}$ are accurate. However, this is usually not the case in practice, since it is unreasonable to assume that $\mathbf{h}$ can be trained well only using data within a single task. In this section, we first show that even small indicator errors could lead to significant performance degeneration of MoreBoost. We further propose an active rectification mechanism, which enables MoreBoost to achieve the desired accuracy even when indicator noise is large.

4.1. Risk of Indicator Error

What performance will MoreBoost achieve when the reusability indicators are not accurate? The following result shows that even when the error rates are not large, since the prediction error grows w.r.t. T, MoreBoost performs much worse under this situation.

Theorem 3 *Let the prediction error of F_T be $\epsilon_{F_T} = \frac{1}{N}\sum_{i=1}^{N} p(F_T(x_i) \neq f^*(x_i))$. Assume that $\forall t \in [T], \max\{\sum_{i=1}^{N} D_t(x_i)\mathbb{I}[h_{k_t}(x_i) \neq h_{k_t}^*(x_i)], \frac{1}{N}\sum_{i=1}^{N}\mathbb{I}[h_{k_t}(x_i) \neq h_{k_t}^*(x_i)]\} \leq \sigma < \gamma$, in which $h_{k_t}^*$ are noise-free versions of h_{k_t}. When $\eta = \sqrt{(\ln N)/T}$, after running MoreBoost for $T \geq \mathcal{O}(\ln N/(\gamma - \sigma)^2)$ iterations, we have that $\epsilon_{F_T} \leq \bar{\epsilon} + \sigma T$.*

We utilize a simple example in Table 1 to show why this could happen. Assume that η is not too small. If no error is made, then MoreBoost will select f_1 and f_2 in the first two iterations to correctly label all the instances. However, in any iteration, once errors are made on both x_1 and x_2, then MoreBoost will turn to choose f_3 and f_4, leading to error on all instances. This example shows the key reason for the error accumulation in Theorem 3: on any iteration of MoreBoost, once a wrong model is chosen, the prediction error of the final obtained model could keep growing by choosing more incorrect models

Algorithm 3 MoreBoost.AR

1: **Input**: $\mathcal{U}, \mathbf{f}, \mathbf{h}, \eta$, error parameters σ_0, δ, specification error upper bounds $\bar{\boldsymbol{\sigma}}$, number of total iterations T.
2: Initialize $h_{0,k} = h_k, D_1(x_i) = \frac{1}{N}, I_{1,k}(x_i) = 1$ for $i \in [N], k \in [K]$.
3: **for** $t = 1$ **to** T **do**
4: Set $\tilde{\sigma}_t = \frac{\sigma_0}{T}, \tilde{\delta}_t = \frac{\delta}{T}$.
5: $I_{t+1}, \mathbf{h}_t, k_t \leftarrow \text{RecSpec}(\mathcal{U}, I_t, D_t, \mathbf{h}_{t-1}, \bar{\boldsymbol{\sigma}}, \tilde{\sigma}_t, \tilde{\delta}_t)$.
6: Set loss $c_t(x_i) = D_t(x_i) h_{k_t}(x_i), i \in [N]$.
7: Update weights: $D_{t+1}(x_i) = D_t(x_i) \exp(-\eta c_t(x_i)) / \sum_{i=1}^{N} D_t(x_i) \exp(-\eta c_t(x_i))$.
8: **end for**
9: **Output**: Predict $x \in \mathcal{U} : F_T(x_i) = \arg\max_{l \in [L]} \sum_{t=1}^{T} D_t(x_i) h_{T,k_t}(x_i) \mathbb{I}[\bar{f}_{k_t}(x_i) = l]$.

in the following iterations. Thus it is crucial to rectify the reusability indicators in each iteration to boost their accuracy, meanwhile to introduce effective mechanism to prevent the errors from growing large.

To address the above issues, we propose MoreBoost with Active Rectification algorithm named MoreBoost.AR, which is illustrated in Algorithm 3. The major changes w.r.t. MoreBoost are two-fold: (1) to deal with indicator error, the loss function and the model combination rule are modified; (2) instead of directly calculating the reusability score, an active rectification mechanism named RecSpec is introduced, to shrink the indicator error rates to pre-defined levels $\tilde{\sigma}_t, \tilde{\delta}_t$. We detailedly introduce RecSpec below.

4.2. The Active Rectification Mechanism

The RecSpec mechanism is illustrated in Algorithm 4. The objective for it is to actively query $\mathbf{h}^*$ to update $\mathbf{h}$, in order to identify the model which is near-optimal under the current iteration. It utilizes $I_k(x_i)$ to record whether an instance x_i has been queried for model k (0 for queried, 1 otherwise). RecSpec includes a greedy error pre-shrinking stage defined in Line 2-13: it shrinks the indicator error rate for each model k under D down to a sufficiently small level. Note that the instances are chosen greedily: The unqueried instance with the largest sample weight is chosen first (Line 4). Then in the main loop, motivated by (Kalyanakrishnan et al., 2012), the confidence bound based best arm identification strategy is utilized to identify the near-optimal model quickly. Detailedly, it utilizes the query records $\mathbf{I}$ to calculate the query counts (Line 15) as well as the confidence bound for each model (Line 16). It then chooses the arm u with the maximum lower confidence bound, as well as v with maximum upper confidence bound besides u (Line 17) to perform greedy queries (Line 18-19, similar to Line 4-5). The intuition is that these are two most promising models to be the best, thus they are worth querying. From the stopping criterion in Line 20, we see that if u has high confidence to be better than v, then we output u as the best arm. The theoretical guarantee on the query complexity for a single call of RecSpec is given as follows. First we define a complexity term. Let $d_{k,n_k} = \sum_{i=1}^{N} \big\{ D(x_i) \big[\big(1 - I_k(x_i)\big) h_k^*(x_i) + I_k(x_i) \big[\sigma_k(x_i) + \big(1 - 2\sigma_k(x_i)\big) h_k^*(x_i) \big] \big] \big\}$ such that $\sum_{i=1}^{N} I_k(x_i) = n_k$. When n_k instances have not been queried for h_k, $d_{k,n}$ is the expected proportion of instances such that $h_k(x_i) = 1$ under D. Furthermore, we set $d^* = \max_{k \in [K]} \big[\min_{n \in [N]} d_{k,n} \big]$, and the corresponding k as

Algorithm 4 RecSpec

Input: Current task data $\mathcal{U}$, query indicator I, sample weights D, reusability indicators $\mathbf{h}$, error upper bounds $\bar{\boldsymbol{\sigma}}$, error parameters $\tilde{\sigma}, \tilde{\delta}$.
for $k = 1$ **to** K **do**
  repeat
    Choose an instance to query: $q = \arg\max_{i\in[N]} D(x_i)I_k(x_i)$.
    Set $h_k(x_q) = h_k^*(x_q),\ I_k(x_q) = 0,\ n_k = \sum_{i=1}^N I_k(x_i)$.
    Set $U(k, n_k) = \sqrt{\frac{1}{2n_k}\ln\Big(4K^2N^2/\tilde{\delta}\Big)}$.
    if $\bar{\sigma}_k \geq 1/4 \vee \tilde{\sigma} \geq \max\{\sqrt[3]{(18\bar{\sigma}_k/N)\ln\Big(4K^2N^2/\tilde{\delta}\Big)}, 72\bar{\sigma}_k/N\}$ **then**
      stopping condition $\leftarrow \sum_{i=1}^N I_k(x_i)D(x_i)\bar{\sigma}_k \leq \tilde{\sigma}/8$.
    else
      stopping condition $\leftarrow \sum_{i=1}^N I_k(x_i)D(x_i) \leq \tilde{\sigma}/2$.
    end if
  until stopping condition.
end for
repeat
  Set $\hat{d}_k = \sum_{i=1}^N D_k(x_i)h_k(x_i), n_k = \sum_{i=1}^N I_k(x_i), k \in [K]$.
  Set $U(k, n_k) = \sqrt{\frac{n_k}{2N^2}\ln\Big(4K^2N^2/\tilde{\delta}\Big)}, k \in [K]$.
  Choose $u = \arg\max_{k\in[K]}(\hat{d}_k - U(k, n_k)), v = \arg\max_{k\in[K]/\{u\}}(\hat{d}_k + U(k, n_k))$.
  Choose an instance to query for each of $b \in \{u, v\}$: $i_b = \arg\max_{i\in[N]} D(x_i)I_b(x_i)$.
  Update $h_u(x_{i_u}) = h_u^*(x_{i_u}),\ h_v(x_{i_v}) = h_v^*(x_{i_v}),\ I_u(x_{i_u}) = 0,\ I_v(x_{i_v}) = 0$.
until $\tilde{\sigma}/8 + \hat{d}_u - U(u, n_u) > \hat{d}_v + U(v, n_v)$.
Return: $I, \mathbf{h}, u$.

k^*. We then define

$$\Delta_k = \begin{cases} \min_{n\in[N]} \max\{d^* - d_{k,n}, 0\}, & k \neq k^*, \\ \min_{n\in[N],k\in[K]/\{k^*\}} \max\{d^* - d_{k,n}, 0\}, & k = k^*. \end{cases}$$

Δ_k measures how easy h_k can be distinguished from or recognized as the optimal one. Furthermore, we define the query complexity measure for the pre-shrinking stage:

$$m_k = \begin{cases} \max\big(0, N(1 - \tilde{\sigma}/(8\bar{\sigma}_k)\big),\ \bar{\sigma}_k \geq 1/4 \vee \tilde{\sigma} \geq \max\{\sqrt[3]{(18\bar{\sigma}_k/N)\ln\Big(4K^2N^2/\tilde{\delta}\Big)}, 72\bar{\sigma}_k/N\}, \\ \max\big(0, N(1 - \tilde{\sigma}/2)\big),\ \text{otherwise.} \end{cases}$$

Theorem 4 *Let the indicator error of h_u be $\hat{\sigma} = \sum_{i=1}^N D(x_i)\mathbb{I}[h_u(x_i) \neq h^*(x_i)]$, where $h^* = \arg\max_{h_k^*,k\in[K]} \sum_{i=1}^N D(x_i)h_k^*(x_i)$. The following events will happen simultaneously with probability at least $1-\tilde{\delta}$: (1) The query complexity of a single call of RecSpec is smaller than $\min\big\{KN, \sum_{k=1}^K \max\{m_k, N[1 - 2N(\Delta_k/4 + \tilde{\sigma}/32)^2\ln\Big(4K^2N^2/\tilde{\delta}\Big)]\}\big\}$; (2) $\hat{\sigma} \leq \tilde{\sigma}$.*

Theorem 4 shows that the query complexity affects by the cover gaps Δ_k. The query complexity will be significantly reduced if the cover gaps are large. By the above guarantee, we can show that MoreBoost.AR enjoys desired performance which is close to the noise-free case with high probability, under the following weak learning assumptions.

Assumption 3 (Weak learning conditions for MoreBoost.AR) *(1) For any data distribution D over $\mathcal{U}$, there exists $k \in [K]$, such that $\sum_{i=1}^{N} D(x_i)h_k^*(x_i) > \gamma > 0$. (2) Setting σ_0 properly: $\sigma_0 < \gamma T$.*

Theorem 5 *Let the error rate of F_T be $\epsilon_{F_T} = \frac{1}{N}\sum_{i=1}^{N} p(F_T(x_i) \neq f^*(x_i))$, and denote by $\bar{\epsilon} = \max_{k\in[K]} \epsilon_k$. When $\eta = \sqrt{(\ln N)/T}$, after running MoreBoost.AR for $T \geq \mathcal{O}(N^2 \ln N/\gamma^4)$ iterations, we have $\epsilon_{F_T} \leq \bar{\epsilon} + 2\sigma_0$ with probability at least $1 - \delta$.*

5. Related Work

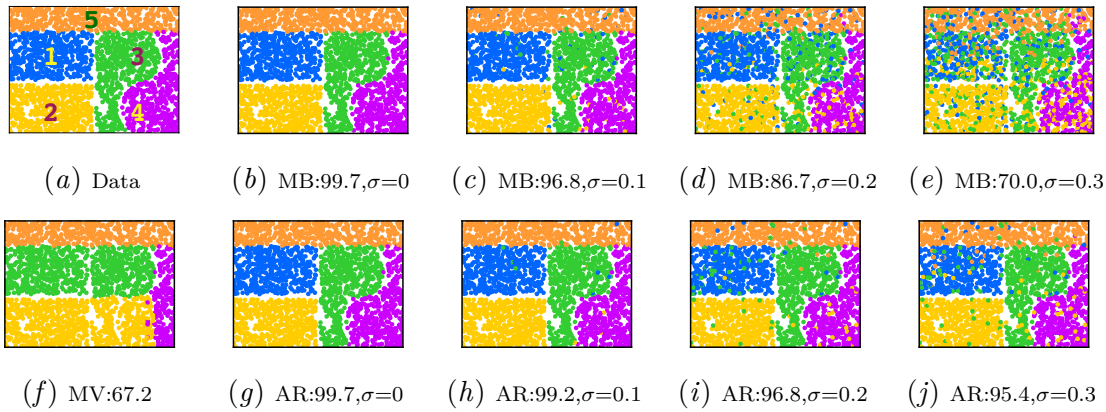
The model reuse problem has a significant difference from other scenarios exploiting existing models in solving a new task. First, it does not assume that all existing models are related to the current task, which is usually assumed in transfer learning (Pan et al., 2010) scenarios such as hypothesis transfer (Dredze et al., 2010; Kuzborskij and Orabona, 2017). Moreover, it is also different from multi-party and federated learning paradigms (Li et al., 2010; McMahan et al., 2017; Yang et al., 2019; Wu et al., 2019). The central problem under these scenarios is how multiple parties can cooperate to solve the same/similar learning tasks. To achieve this target, rich information such as data distribution statistics can be exchanged among different parties in privacy-preserving ways. While for model reuse, the models may be trained under very different tasks to the current one, meanwhile all information on those tasks can only be observed through the models and their specifications. In recent years, some interesting studies have discussed how transferability of learned models can be measured (Achille et al., 2019; Nguyen et al., 2020). While they mainly study how the transferability can be measured on target data without using model specifications. When target data only provide limited information, e.g. the number of them are limited or they are unlabeled, providing the specifications are strongly helpful for measuring the transferability. There have been a number of boosting-based approaches in related areas such as transfer learning and domain adaptation. While they usually focus on how source and target domain data can be both utilized to train a new model (Dai et al., 2007; Becker et al., 2013; Habrard et al., 2013; Wang and Pineau, 2015), or utilizing boosting to solve a sequence of related learning tasks (Rettinger et al., 2006). Thus they consider very different learning problems to our work.

The analysis of error accumulation under the noisy situation is inspired by the distributional shift issue in imitation learning (Ross et al., 2011). We design different techniques to solve our problem since we consider a very different learning scenario.

6. Experiments

6.1. Experiements on Synthetic Data

We create a 2D toy example with five classes to verify our approaches. There are five classes 1-5 represented by color blocks. We divide them into four tasks (1,2), (3,4,5), (2,3),



(a) Data (b) MB:99.7,σ=0 (c) MB:96.8,σ=0.1 (d) MB:86.7,σ=0.2 (e) MB:70.0,σ=0.3

(f) MV:67.2 (g) AR:99.7,σ=0 (h) AR:99.2,σ=0.1 (i) AR:96.8,σ=0.2 (j) AR:95.4,σ=0.3

Figure 2: Results on synthetic data. Figure 2a shows the target data and their true labels. Figure 2b-2j show prediction results and accuracy (%) of different approaches. σ are indicator noise rates.

(1,4), where we train gradient boosting decision trees as existing models with training data sampled from the corresponding blocks. We then generate instances uniformly for each class as the current task data, which are illustrated in Figure 2a. The reusability indicators are built according to the ground-truth class with fixed noise σ added, which is known to the learner. We consider the following settings and methods: (1) majority voting (MV, Figure 2f); (2) MoreBoost(MB, Figure 2b-2e); (3) MoreBoost.AR (AR, Figure 2g-2j). We run both MoreBoost and MoreBoost.AR for 50 iterations. We observe that MoreBoost performs well under the small noise case, meanwhile RecSpec performs better under the large noise case.

6.2. Experiments on Benchmark Data

First, we consider simpler experiments on three ten-class benchmark datasets: MNIST (LeCun et al., 1998), fashion-MNIST (Xiao et al., 2017) and CIFAR-10 (Krizhevsky, 2009). One each dataset, to obtain existing models, we form five existing tasks according to classes. Detailedly, we randomly partition the ten classes into three tasks with proportion (4, 3, 3) first. Furthermore, we randomly choose three of the four classes in the first task to form two more tasks. For each task, we randomly choose 3000 training data out of the original training set for each class to formulate the model training data. Three-layer CNNs on MNIST and Fashion-MNIST, 16-layer wide ResNets (Zagoruyko and Komodakis, 2016) on CIFAR-10 are utilized to obtain the existing models. To form the target task, we randomly choose 50 instances for each class out of the original testing data for predicting their true labels. We compare the following approaches: (1) the best single existing model (MAX); (2) majority voting (MV); (3) the state-of-the-art model reuse approach RKME (Wu et al., 2020); (4) MoreBoost (MB); (5) MoreBoost.AR with random query strategy (RND), which randomly selects equal number of instances to query in each iteration, and (6) MoreBoost.AR (AR). We set the total query number the same for RND and AR. To obtain reusability indicators for MoreBoost-based approaches, we utilize pre-trained CNNs or wide ResNets to extract features of the model training data from the outputs of the penultimate layers. Afterwards,

Table 2: Accuracy on benchmark data over five random splits. The last two rows show the proportion of the query budget used and the indicator error rates. All results are Value(%)±Std. Dev.(%).

	MNIST	Fashion-MNIST	CIFAR-10	CIFAR-100	CUB-200-2011	Caltech-256
MAX	39.7±0.2	38.4±1.2	37.1±3.3	20.4±6.3	10.2±2.0	12.0±2.4
MV	48.0±3.2	39.5±1.6	37.1±0.7	29.0±7.5	27.6±5.9	48.0±3.0
RKME	98.6±0.3	90.3±1.0	84.7±9.5	65.0±6.6	27.6±7.9	20.4±4.8
MB	92.1±1.4	81.6±0.7	83.7±1.3	55.6±8.1	41.2±6.5	10.4±2.2
RND	96.9±0.7	91.1±0.9	91.6±0.8	62.0±10.6	80.6±6.0	57.6±12.3
AR	**99.4±0.1**	**95.3±1.8**	**95.4±0.4**	**80.4±5.1**	**90.4±2.4**	**87.8±5.5**
Query Prop.	55.8±2.1	58.2±5.1	55.4±4.4	81.0±3.9	89.9±1.4	93.0±2.6
Ind. Error	3.2±1.4	12.2±1.8	7.9±1.2	9.4±3.3	5.2±3.3	49.7±13.8

we train isolation forests (Liu et al., 2008) on them. We also learn RKME specifications for the RKME approach on the same pre-trained features for fair comparison. To estimate indicator noise, we sample five instances from the original test data for each class to form the validation data. From Table 2, we can see that MoreBoost.AR achieves the optimal performance as in the synthetic data experiment.

We further consider larger-scale experiments on three more benchmark datasets: CIFAR-100 (Krizhevsky, 2009), CUB-200-2011 (Wah et al., 2011) and Caltech-256 (Griffin et al., 2007). We keep the comparison methods and feature extraction process similar to above. For CIFAR-100, the original training data are from 100 classes, which are grouped into 20 super-classes. We form 20 tasks accordingly to obtain the existing models, which are 16-layer wide ResNets. We further sample instances from original testing data to form the target task. We choose ten classes randomly, and sample ten instances for each class. We further sample five instances for each class for validating indicator noise rates. For CUB-200-2011 and Caltech-256, we randomly partition the classes into 40 and 25 existing tasks, and fine-tune an ImageNet pre-trained ResNet-101 model on them to obtain the existing models. The formulation of the target task is the similar to CIFAR-100. We observe the similar performance gain for MoreBoost.AR.

7. Conclusion

In this work, we studied the reusability-aware model reuse problem, in which a learner chooses multiple existing models to reuse with reusability indicator specifications. We proposed MoreBoost, a simple but powerful model reuse algorithm. An active rectification mechanism was also proposed under the noisy situation. Applying our algorithms on different kinds of model reuse settings could be interesting future work to study.

Acknowledgement

This research was supported by NSFC (61921006) and Collaborative Innovation Center of Novel Software Technology and Industrialization. The authors would like to thank the

anonymous reviewers for constructive suggestions, as well as Peng Zhao and Xi-Zhu Wu for helpful discussions.

Appendix A. Proofs

A.1. Proof of Theorem 1

Proof Assume that $D_1, D_2, \ldots, D_T$ are data distributions generated by running MoreBoost for T iterations. We consider an online learning view of the learning processs. Define the cost function of distribution D at iteration t as $c_D(t) = \sum_{i=1}^{N} D(x_i)c_t(x_i)$. The regret of MoreBoost can be defined as $R_T = \sum_{t=1}^{T} \left(c_{D_t}(t) - c_{D^*}(t)\right)$, in which $D^* = \arg\min_D \sum_{t=1}^{T} c_D(t)$. Since the cost on any single instance is equivalent to one-hot data distribution, we have that for $\forall i \in [N]$, $\frac{1}{T}\sum_{t=1}^{T} c_{D_t}(t) - \frac{1}{T}\sum_{t=1}^{T} c_t(x_i) \leq \frac{R_T}{T}$. According to the weak learning condition, on any data distribution, the proportion of covered instances is at least γ. Thus $\frac{1}{T}\sum_{t=1}^{T} c_{D_t}(t) > \gamma$. Then we have $\gamma - \frac{R_T}{T} \leq \frac{1}{T}\sum_{t=1}^{T} c_t(x_i)$. If $\frac{1}{T}\sum_{t=1}^{T} c_t(x_i) > 0$, then we know that $\exists h_t^*, h_t^*(x_i) = 1$. Furthermore, by the assumption on reusability indicators, we know the error rate on x_i is below $\bar{\epsilon}$. To guarantee that $\frac{1}{T}\sum_{t=1}^{T} c_t(x_i) > 0$, we can set $\gamma - \frac{R_T}{T} > 0$. If we guarantee that the regret $R_T \leq \mathcal{O}(\sqrt{T \ln N})$, it is easy to see that we can set $T \geq \mathcal{O}(\ln N/\gamma^2)$. Notice that the online learning procedure for MoreBoost is designed exactly to fit for the regret minimization process of Hedge (Freund and Schapire, 1997). As a result, the regret bound holds as expected. ■

A.2. Proof of Theorem 2

Proof When $\eta \to \infty$, then once $c_t(x_i) = \mathbb{I}[h_t(x_i) = 1] = 1$, $D_{t+1}(x_i) = 0$. Then we have the following two observations: (1) When $\eta \to \infty$, h_k are chosen according to the number of instances they can cover; (2) Each instance is covered by exactly one h_k. These two observations exactly fit for what the greedy set cover algorithm does. ■

A.3. Proof of Theorem 3

Proof Follow the similar proof of Theorem 1, we have that for $\forall i \in [N]$, $\frac{1}{T}\sum_{t=1}^{T} c_{D_t}(t) - \frac{1}{T}\sum_{t=1}^{T} c_t(x_i) \leq \frac{R_T}{T}$. Since for $\forall h, h', \mathbb{I}[h = 1 \wedge h' = 0] \leq \mathbb{I}[h \neq h']$, by the assumption in the theorem, $\forall t, c_{D_t}(t) = \sum_{i=1}^{N} D_t(x_i)\mathbb{I}[h_{k_t}(x_i) = 1] \geq \sum_{i=1}^{N} D_t(x_i)(\mathbb{I}[h_{k_t}^*(x_i) = 1] - \mathbb{I}[h_{k_t}(x_i) \neq h_{k_t}^*(x_i)]) \geq \gamma - \sigma$. Then after running MoreBoost for $T \geq \mathcal{O}(\frac{\ln N}{(\gamma-\sigma)^2})$ iterations, we have that $\forall x_i \in \mathcal{U}, \sum_{t=1}^{T} h_t(x_i) > 0$. Furthermore, $\forall x_i \in \mathcal{U}$, $p(F_T(x_i) \neq f^*(x_i))$ can be upper bounded by $p(F_T(x_i) \neq f^*(x_i), (\nexists h_{k_t}(x_i) \neq h_{k_t}^*(x_i))) + p(\exists h_{k_t}(x_i) \neq h_{k_t}^*(x_i)) \leq p(F_T(x_i) \neq f^*(x_i), (\nexists h_{k_t}(x_i) \neq h_{k_t}^*(x_i))) + \sum_{t=1}^{T} \mathbb{I}[h_{k_t}(x_i) \neq h_{k_t}^*(x_i)]$. Then $\epsilon_{F_T} \leq \frac{1}{N}\sum_{i=1}^{N} \Big(p(F_T(x_i) \neq f^*(x_i), (\nexists h_{k_t}(x_i) \neq h_{k_t}^*(x_i))) + \sum_{t=1}^{T} \mathbb{I}[h_{k_t}(x_i) \neq h_{k_t}^*(x_i)]\Big) \leq \bar{\epsilon} + \sigma T$. ■

A.4. Proof of Theorem 4

For simplicity, we analyse the first call of RecSpec, whose guarantee obviously holds for the rest calls. The following lemma shows that the pre-training stage controls the empirical errors of indicators close to their expectations.

Lemma 6 *After the pre-training stage in Line 2-13, then* $\forall k \in [K], \frac{1}{N}\sum_{i=1}^{N} \mathbb{I}[h_k(x_i) \neq h_k^*(x_i)] \geq \tilde{\sigma}/2$ *holds with probability at most* $\tilde{\delta}/4$.

Proof We study arbitrary $k \in [K]$. First, $\sum_{i=1}^{N} D(x_i)\mathbb{I}[h_k(x_i) \neq h_k^*(x_i)] \leq \sum_{i=1}^{N} I_k(x_i)D(x_i)$. Next, we study different cases. (1) If $\bar{\sigma}_k \geq 1/4$, then $\sum_{i=1}^{N} I_k(x_i)D(x_i) \leq \tilde{\sigma}/(8\bar{\sigma}_k) \leq \tilde{\sigma}/2$. (2) If $\bar{\sigma}_k < 1/4 \wedge \tilde{\sigma} < \max\{72\bar{\sigma}_k/N, \sqrt[3]{(18\bar{\sigma}_k/N)\ln\left(4K^2N^2/\tilde{\delta}\right)}\}$, then $\sum_{i=1}^{N} I_k(x_i)D(x_i) \leq \tilde{\sigma}/2$ holds by the stopping condition. (3) If $\tilde{\sigma} \geq \max\{72\bar{\sigma}_k/N, \sqrt[3]{(18\bar{\sigma}_k/N)\ln\left(4K^2N^2/\tilde{\delta}\right)}\}$, from the greedy query rule in Line 4 of Algorithm 4, $\sum_{i=1}^{N} D(x_i)I(x_i)\bar{\sigma}_k \leq n_k\bar{\sigma}_k/N$. Furthermore, since $\tilde{\sigma} \geq 72\bar{\sigma}_k/N$, we have $\tilde{\sigma}N/(8\bar{\sigma}_k) \geq \tilde{\sigma}N/(9\bar{\sigma}_k) + 1$. As a result, we have that when the stopping condition reaches, $n_k \geq (\tilde{\sigma}N)/(9\bar{\sigma}_k)$. By Hoeffding's inequality, we have $p(\sum_{i=1}^{N} D(x_i)\mathbb{I}[h_k \neq h_k^*] \geq \tilde{\sigma}/2) \leq p(\sum_{i=1}^{N} I_k(x_i)D(x_i)\mathbb{I}[h_k(x_i) \neq h_k^*(x_i)] - \sum_{i=1}^{N} I_k(x_i)D(x_i)\bar{\sigma}_k \geq \tilde{\sigma}/2) \leq \exp\left(-2n_k(\tilde{\sigma}/2)^2\right) \leq \exp\left(-N\tilde{\sigma}^3/(18\bar{\sigma}_k)\right) \leq \tilde{\delta}/(4K^2N^2)$, where the last inequality holds by the assumption that $\tilde{\sigma} \geq \sqrt[3]{(18\bar{\sigma}_k/N)\ln\left(4K^2N^2/\tilde{\delta}\right)}$. By union bound over maximum number of total loops KN, models K and instances N, we arrive at the final result. ■

Next we show the correctness of the stopping criterion in Line 20 of Algorithm 4.

Lemma 7 *If the stopping criterion in Line 20 of Algorithm 4 is reached, then we have that* $\hat{\sigma} \geq \tilde{\sigma}$ *with probability at most* $\tilde{\delta}/2$.

Proof Assmue that when the stopping condition is reached, we have $d_{u,n_u} > \hat{d}_u - U(u, n_u)$ and for $\forall k \in [K]$, we have $d_{k,n_k} < \hat{d}_k + U(k, n_k)$. From the definition of v, for $\forall k \in [K]$,

$$\tilde{\sigma}/8 + \hat{d}_u - U(u, n_u) > \hat{d}_k + U(k, n_k) \implies \tilde{\sigma}/8 + d_{u,n_u} > d_{k,n_k} \implies$$
$$\tilde{\sigma}/8 + \sum_{i=1}^{N} D(x_i)h_u^*(x_i) + \sum_{i=1}^{N} I_u(x_i)D(x_i)\bar{\sigma}_u(x_i) > \sum_{i=1}^{N} D(x_i)h_k^*(x_i) - 2\sum_{i=1}^{N} I_k(x_i)D(x_i)\bar{\sigma}_k(x_i).$$

Due to the stopping condition in Line 12, we have $\tilde{\sigma} + \sum_{i=1}^{N} D(x_i)h_u^*(x_i) > \sum_{i=1}^{N} D(x_i)h_k^*(x_i)$, which indicates that u is close to the true best model. The next step is to ensure that the confidence bounds are valid for all iterations before the stopping criterion is reached. By Hoeffding's inequality, for $\forall k \in [K], n \in [N]$, we have that $p(|d_{k,n} - \hat{d}_k| > U(k, n)) \leq 2e^{-2\frac{N^2}{n}U^2(k,n)}$. Notice that the stopping criterion is reached within $KN/2$ iterations since the maximum query time is KN and two instances are chosen to query in each iteration. Then by the union bound, we need to guarantee that $KN\sum_{k=1}^{K}\sum_{n=1}^{N} e^{-2\frac{N^2}{n}U(k,n)} \leq \tilde{\delta}/4$, which is indeed the case by the design of the confidence bound $U(k, n)$. Combining with Lemma 6, we arrive at the final result. ■

The following lemma shows how small each $n_k, k \in [K]$ should be to make the stopping criterion in Line 20 reached.

Lemma 8 *Let sufficient query condition be* $n_k \leq 2N^2(\Delta_k/4+\tilde{\sigma}/32)^2/(\ln\big(4K^2N^2/\delta\big)), \forall k \in [K]$. *If the sufficient query condition is reached, then the stopping criterion is not reached with probability at most* $\tilde{\delta}/(2KN)$.

Proof Assume that on one iteration, all $n_k, k \in [N]$ satisfy the condition. Then we have $U(k^*, n_{k^*}) \leq \Delta_{k^*}/4 + \tilde{\sigma}/32, U(v, n_v) \leq \Delta_v/4 + \tilde{\sigma}/32$. On the other hand, if the stopping criterion has not been reached, we have $\tilde{\sigma}/8 + \hat{d}_{k^*} - U(k^*, n_{k^*}) \leq \hat{d}_v + U(v, n_v)$. Then we know that $\hat{d}_{k^*} \leq \hat{d}_v + \Delta_{k^*}/4 + \Delta_v/4 - \tilde{\sigma}/16 \leq \hat{d}_v + \Delta_v/2 - \tilde{\sigma}/16$. By construction of Δ_{k^*}, Δ_v, we know that $\exists k \in [K], \hat{d}_k - d_k \geq U(k, n_k) \vee d_k - \hat{d}_k \geq U(k, n_k)$. By Hoeffding's inequality and union bound, we know the probability of this event is within $\sum_{k=1}^{K}\sum_{n_k=1}^{N} 2\exp\bigg(((-2N^2)/n_k)\Big(\sqrt{(n_k/2N^2)\ln\Big(4K^2N^2/\tilde{\delta}\Big)}\Big)^2\bigg) = \tilde{\delta}/(2KN)$. ■

Then we prove Theorem 4.

Proof Assume that the stopping condition is reached when querying $\lfloor \frac{T}{2} \rfloor \leq t \leq T$ instances, and the stopping condition is reached iff the sufficient query condition is satisfied. Then $t = \lfloor T/2 \rfloor + \sum_{t=\lfloor T/2 \rfloor}^{T} \mathbb{I}\{\neg\text{sufficient query condition}\} \leq \sum_{k=1}^{K}\sum_{t=\lfloor T/2 \rfloor}^{T} \mathbb{I}\{n_k \leq 2N^2(\Delta_k/4 + \tilde{\sigma}/32)^2/(\ln\Big(4K^2N^2/\tilde{\delta}\Big))\} \leq \sum_{k=1}^{K} \max\big\{0, N\big(1-2N(\Delta_k/4+\tilde{\sigma}/32)^2/(\ln\Big(4K^2N^2/\tilde{\delta}\Big)\big)\big\}$. We can see that if $T \geq \sum_{k=1}^{K} \max\big\{0, N\big(1-2N(\Delta_k/4+\tilde{\sigma}/32)^2/(\ln\Big(4K^2N^2/\tilde{\delta}\Big)\big)\big)\big\}$, then the stopping condition is reached. By Lemma 8 and union bound, we know that this event would happen with probability at least $1-(KN)(\tilde{\delta}/(2KN)) = 1-\tilde{\delta}/2$. Furthermore, the query complexity of the training stage, i.e. m_k can be calculated easily from the greedy query criterion in Line 4. Combining Lemma 7, we arrive at the final result. ■

A.5. Proof of Theorem 5

Proof We have that for $\forall i \in [N]$, $\frac{1}{T}\sum_{t=1}^{T} c_{D_t}(t) - \frac{1}{T}\sum_{t=1}^{T} c_t(x_i) \leq \frac{R_T}{T}$, which follows from the similar proof of Theorem 1. Assume that the high probability events in Theorem 4 hold for all T iterations. Since for $\forall h, h', \mathbb{I}[h = 1 \wedge h' = 0] \leq \mathbb{I}[h \neq h']$, by Theorem 4 and the weak learning assumption, $\sum_{i=1}^{N} D_t(x_i)h_{k_t}(x_i) \geq \sum_{i=1}^{N} D_t(x_i)(h_t^*(x_i) - \mathbb{I}[h_{k_t}(x_i) \neq h_t^*(x_i)]) \geq \gamma - \sigma_0/T$, in which $h_t^* = \arg\max_{h_k^*, k\in[K]} \sum_{i=1}^{N} D_t(x_i)h_k^*(x_i)$. Furthermore, $\forall t, c_{D_t}(t) = \sum_{i=1}^{N} D_t^2(x_i)h_{k_t}(x_i) = \sum_{i=1}^{N} D_t^2(x_i)h_{k_t}^2(x_i) \geq (\sum_{i=1}^{N} D_t(x_i)h_t(x_i))^2/N \geq (\gamma - \sigma_0/T)^2/N$. As a result, after running MoreBoost.AR for $T \geq \mathcal{O}(N^2 \ln N/\gamma^4)$ iterations, we have that $\forall x_i \in \mathcal{U}, \sum_{t=1}^{T} D_t(x_i)h_t(x_i) > 1$. Furthermore, we have that $\forall x_i \in \mathcal{U}$, $p(F_T(x_i) \neq f^*(x_i))$ can be upper bounded by $p\Big(F_T(x_i) \neq f^*(x_i) \wedge \sum_{t=1}^{T} D_t(x_i)\mathbb{I}[h_{k_t}(x_i) = 1, h_{k_t}^*(x_i) = 0] < 1/2\Big) + p\Big(F_T(x_i) \neq f^*(x_i) \wedge \sum_{t=1}^{T} D_t(x_i)\mathbb{I}[h_{k_t}(x_i) = 1, h_{k_t}^*(x_i) = 0] \geq 1/2\Big) \leq \bar{\epsilon} + 2\sum_{t=1}^{T} D_t(x_i)\mathbb{I}[h_{k_t}(x_i) \neq h_{k_t}^*(x_i)]$. We then have that $\epsilon_{F_T} \leq \frac{1}{N}\sum_{i=1}^{N} p(F_T(x_i) \neq f^*(x_i)) \leq \bar{\epsilon} + 2\sum_{t=1}^{T}\Big(\frac{1}{N}\sum_{i=1}^{N} D_t(x_i)\mathbb{I}[h_{k_t}(x_i) \neq h_{k_t^*}(x_i)]\Big) \leq \bar{\epsilon} + 2\sigma_0$. The desired result follows from the union bound w.r.t. T. ■

References

Alessandro Achille, Michael Lam, Rahul Tewari, Avinash Ravichandran, Subhransu Maji, Charless C Fowlkes, Stefano Soatto, and Pietro Perona. Task2vec: Task embedding for meta-learning. In *CVPR*, 2019.

Carlos J Becker, Christos M Christoudias, and Pascal Fua. Non-linear domain adaptation with boosting. In *NeurIPS*, 2013.

Wenyuan Dai, Yang Qiang, Gui-Rong Xue, and Yong Yu. Boosting for transfer learning. In *ICML*, 2007.

Irit Dinur and David Steurer. Analytical approach to parallel repetition. In *STOC*, 2014.

Mark Dredze, Alex Kulesza, and Koby Crammer. Multi-domain learning by confidence-weighted parameter combination. *Machine Learning*, 79(1-2):123–149, 2010.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

Amaury Habrard, Jean-Philippe Peyrache, and Marc Sebban. Boosting for unsupervised domain adaptation. In *ECML-PKDD*, 2013.

Shivaram Kalyanakrishnan, Ambuj Tewari, Peter Auer, and Peter Stone. Pac subset selection in stochastic multi-armed bandits. In *ICML*, 2012.

Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pages 85–103. Springer, 1972.

Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, 2009.

Ilja Kuzborskij and Francesco Orabona. Fast rates by transferring from auxiliary hypotheses. *Machine Learning*, 106(2):171–195, 2017.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Ming Li, Wei Wang, and Zhi-Hua Zhou. Exploiting remote learners in internet environment with agents. *Science in China Series F: Information Sciences*, 53(1):64–76, 2010.

Nan Li, Ivor W Tsang, and Zhi-Hua Zhou. Efficient optimization of performance measures by classifier adaptation. *IEEE transactions on pattern analysis and machine intelligence*, 35(6):1370–1382, 2012.

Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *ICDM*, 2008.

Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *AISTATS*, 2017.

Cuong V Nguyen, Tal Hassner, Cedric Archambeau, and Matthias Seeger. Leep: A new measure to evaluate transferability of learned representations. In *ICML*, 2020.

Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010.

Achim Rettinger, Martin Zinkevich, and Michael Bowling. Boosting expert ensembles for rapid concept recall. In *AAAI*, 2006.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, 2011.

Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

Boyu Wang and Joelle Pineau. Online boosting algorithms for anytime transfer and multitask learning. In *AAAI*, 2015.

Xi-Zhu Wu, Song Liu, and Zhi-Hua Zhou. Heterogeneous model reuse via optimizing multiparty multiclass margin. In *ICML*, 2019.

Xi-Zhu Wu, Wenkai Xu, Song Liu, and Zhi-Hua Zhou. Model reuse with reduced kernel mean embedding specification. *arXiv preprint arXiv:2001.07135*, 2020.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Qiang Yang, Yang Liu, Tianjian Chen, and Yongxin Tong. Federated machine learning: Concept and applications. *ACM Transactions on Intelligent Systems and Technology*, 10 (2):12, 2019.

Yang Yang, De-Chuan Zhan, Ying Fan, Yuan Jiang, and Zhi-Hua Zhou. Deep learning for fixed model reuse. In *AAAI*, 2017.

Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

Zhi-Hua Zhou. *Ensemble methods: foundations and algorithms*. Chapman and Hall/CRC, 2012.

Zhi-Hua Zhou. Learnware: on the future of machine learning. *Frontiers of Computer Science*, 10(4):589–590, 2016.