Adaptive Ensemble Learning with Confidence Bounds

Cem Tekin, Jinsung Yoon and Mihaela van der Schaar

Extracting actionable intelligence from distributed, heterogeneous, correlated and high-dimensional data sources requires run-time processing and learning both locally and globally. In the last decade, a large number of meta-learning techniques have been proposed in which local learners make online predictions based on their locally-collected data instances, and feed these predictions to an ensemble learner, which fuses them and issues a global prediction. However, most of these works do not provide performance guarantees or, when they do, these guarantees are asymptotic. None of these existing works provide confidence estimates about the issued predictions or rate of learning guarantees for the ensemble learner. In this paper, we provide a systematic ensemble learning method called Hedged Bandits, which comes with both long run (asymptotic) and short run (rate of learning) performance guarantees. Moreover, we show that our proposed method outperforms all existing ensemble learning techniques, even in the presence of concept drift. This work has numerous applications, including network monitoring and security, healthcare informatics, online recommendation systems, social networks, smart cities, etc.

Introduction

Huge amounts of data streams are now being produced by more and more sources and in increasingly diverse formats: sensor readings, physiological measurements, GPS events, network traffic information, documents, emails, transactions, tweets, audio files, videos etc. These streams are then mined in real-time to provide actionable intelligence for a variety of applications: patient monitoring (Simons 2008), recommendation systems (Cao and Li 2007), social networks (Beach et al. 2008), targeted advertisement (Swix, Stefanik, and Batten 2004), network security (Eskicioglu and Delp 2001), medical diagnosis (Arsanjani et al. 2013) etc. Hence, online data mining algorithms have emerged that analyze the correlated, high-dimensional and dynamic data instances captured by one or multiple heterogeneous data sources, extract actionable intelligence from these instances and make decisions in real-time. To mine these data streams, the following questions need to be answered continuously, for each data instance: Which processing/prediction/decision rule should a *local learner* (LL) select? How should the LLs adapt and learn their rules to maximize their performance? How should the processing/predictions/decisions of the LLs be combined/fused by a meta-learner to maximize the overall performance?

Existing works on meta-learning (Littlestone and Warmuth 1989), (Freund and Schapire 1995) have aimed to provide solutions to these questions by designing *ensemble learners* (ELs) that fuse the predictions[1] made by the LLs into global predictions. A majority of the literature treats the LLs as black box algorithms, and proposes various fusion algorithms for the EL with the goal of issuing predictions that are at least as good as the best LL in terms of prediction accuracy. In some of these works, the obtained result holds for any arbitrary sequence of data instance-label pairs, including the ones generated by an adaptive adversary. However, the performance bounds proved for the EL in these papers depend on the performance of the LLs. In this work, we go one step further and study the joint design of learning algorithms for both the LLs and the EL.

In this paper, we present a novel learning method which continuously learns and adapts the parameters of both the LLs and the EL, after each data instance, in order to achieve strong performance guarantees - both confidence bounds and regret bounds. We call the proposed method *Hedged Bandits* (HB). The proposed system consists of a contextual bandit algorithm for the LLs and Hedge algorithm (Freund and Schapire 1995) for the EL. The proposed method is able to exploit the adversarial regret guarantees of Hedge and the data-dependent regret guarantees of the contextual bandit algorithm to derive a data-dependent regret bound for the EL. It was proven for the Hedge algorithm that it has a $O(\sqrt{T \log M})$ bound on the regret of the EL with respect to the best fixed LL, where T is time and M is the number of LLs. We utilize this property of Hedge to derive a regret bound with respect to the best data-dependent prediction rule of the best LL.

The contributions of this paper are:

- We prove regret and confidence bounds for each LL with

[1] Throughout this paper the term prediction is used to denote a variety of tasks from making predictions to taking actions.

respect to the best data-dependent prediction rule of that LL.

- Using the regret bounds proven for each LL, we prove a regret bound for the EL with respect to the best data-dependent prediction rule of the best LL.
- We numerically compare HB with state-of-the-art machine learning methods in the context of medical informatics and show the superiority of HB.

Problem Description

Consider the system model given in Figure 1. There are M LLs indexed by the set $\mathcal{M} := \{1, 2, \ldots, M\}$. Each LL receives streams of data instances, sequentially, over discrete time steps ($t = 1, 2, \ldots$). The instance received by LL i at time t is denoted by $x_i(t)$. Without loss of generality, we assume that $x_i(t)$ is a d_i dimensional vector in $\mathcal{X}_i := [0, 1]^{d_i}$.[2]

The collection of data instances at time t is denoted by $\boldsymbol{x}(t) = \{x_i(t)\}_{i\in\mathcal{M}}$. As an example, in a medical diagnosis application $\boldsymbol{x}(t)$ can include real-valued features such as lab test results; discrete features such as age and number of previous conditions; and categorical features such as gender, smoker/non-smoker, etc. In this example each LL corresponds to a different medical expert.

The set of prediction rules of LL i is denoted by $\mathcal{F}_i$. After observing $x_i(t)$, LL i selects a prediction rule $a_i(t) \in \mathcal{F}_i$. The selected prediction rule produces a prediction $\hat{y}_i(t)$. Then, all LLs send their predictions $\hat{\boldsymbol{y}}(t) := \{\hat{y}_i(t)\}_{i\in\mathcal{M}}$ to the EL, which combines them to produce a final prediction $\hat{y}(t)$. The set of possible predictions is denoted by $\mathcal{Y}$, which is equivalent to the set of true labels (ground truth). We assume that the true label $y(t) \in \mathcal{Y}$ is revealed after the final prediction, by which the LLs and the EL can update their prediction rule selection strategy, which is a mapping from the history of past observations, decisions, and the current instance to the set of prediction rules.[3]

In our setup each LL is only required to observe its own data instance and know its own prediction rules. However, the accuracy of the prediction rules are unknown and data dependent. The EL knows nothing about the instances and prediction rules of the LLs. The accuracy of prediction rule $f \in \mathcal{F}_i$ for instance $x_i(t)$ is denoted by $\pi_f(x_i(t)) := \Pr(\hat{y}_i(t) = y(t)|x_i(t), a_i(t) = f)$. We assume that the accuracy of a prediction rule obeys the following Hï¿œlder rule, which represents a similarity measure between different data instances.

Assumption 1. *There exists $L > 0$, $\alpha > 0$ such that for all $i \in \mathcal{M}$, $f \in \mathcal{F}_i$, and $x, x' \in \mathcal{X}_i$, we have*

$$|\pi_f(x) - \pi_f(x')| \leq L||x - x'||^\alpha.$$

We assume that L and α are known by the LLs. Going back to our medical informatics example, we can interpret Assumption 1 as follows. Consider two patients. If all the lab tests and demographic information of both patients are similar, it is expected that they have the same underlying condition. Hence, the prediction made by f should be similar for these patients.

[2]The unit hypercube is just used for notational simplicity. Our methods can easily be generalized to arbitrary bounded, finite dimensional data spaces, including spaces of categorical variables.

[3]Missing and erroneous labels will be discussed later in the extensions section.

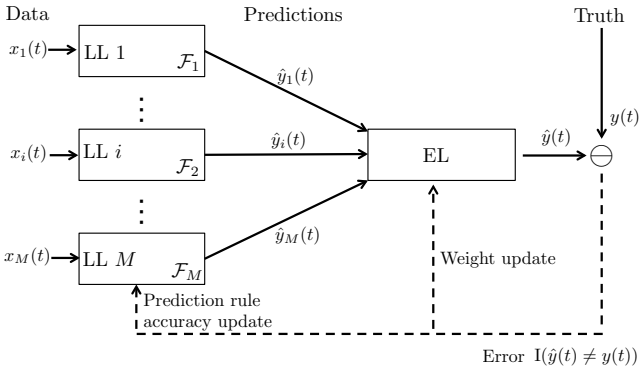

Figure 1: Block diagram of the HB. After observing the instance, each LL selects one of its prediction rules to produce a prediction, and sends its prediction to the EL which makes the final prediction. Then, both the LLs and the EL update their prediction policies based on the received feedback $y(t)$.

Definition of the Regret

In this section we define the regret for each LL and the EL, which defines the expected total number of excess prediction errors due to not knowing the optimal prediction rules for each instance. The optimal prediction rule of LL i for an instance $x \in \mathcal{X}_i$ is defined as

$$f_i^*(x) := \arg\max_{f\in\mathcal{F}_i} \pi_f(x).$$

The (data-dependent) regret of LL i is defined as its total expected loss due to not knowing $f_i^*(x)$, $x \in \mathcal{X}_i$ perfectly, and is given by

$$R_i(T) := \sum_{t=1}^{T} \pi_{f_i^*(x_i(t))}(x_i(t)) - \mathrm{E}\left[\sum_{t=1}^{T} \mathrm{I}(\hat{y}_i(t) = y(t))\right] \quad (1)$$

for an arbitrary sequence of instances $x_i(1), \ldots, x_i(T)$, where $\mathrm{I}(\cdot)$ is the indicator function.

Consider an arbitrary, length T sequence of instance collections: $\boldsymbol{x}^T := (\boldsymbol{x}(1), \boldsymbol{x}(2), \ldots, \boldsymbol{x}(T))$. The best LL for $\boldsymbol{x}^T$ is defined as

$$i^*(\boldsymbol{x}^T) := \arg\max_{i\in\mathcal{M}} \sum_{t=1}^{T} \pi_{f_i^*(x_i(t))}(x_i(t)) \quad (2)$$

which is the LL whose total predictive accuracy is greatest among all LLs.

The regret of the EL for $\boldsymbol{x}^T$ is defined as

$$R_{\text{ens}}(T) := \sum_{t=1}^{T} \left(\pi_{f_{i^*}^*(x_{i^*}(t))}(x_{i^*}(t)) - \mathrm{E}\left[\mathrm{I}(\hat{y}(t) = y(t))\right]\right) \quad (3)$$

where $i^* = i^*(\boldsymbol{x}^T)$.

The goal of this work is to develop algorithms for the LLs and the EL that minimize the growth rate of $R_{\text{ens}}(T)$. We will prove in the next section that the *average regret* $R_{\text{ens}}(T)/T$ of the proposed algorithms will converge asymptotically to 0, meaning that algorithms are optimal in terms of their average predictive accuracy, and prove a sublinear regret bound on $R_{\text{ens}}(T)$ with respect to the optimal data-dependent prediction rule of the best expert, meaning that the proposed algorithms have a provably fast rate of learning.

A Instance-based Uniform Partitioning Algorithm for the LLs

Each LL uses the *Instance-based Uniform Partitioning* (IUP) algorithm given in Figure 2. IUP is designed to exploit the similarity measure given in Assumption 1 when learning accuracies of the prediction rules. Basically, IUP partitions $\mathcal{X}_i$ into a finite number of equal sized, identically shaped, non-overlapping sets, whose granularity determine the balance between approximation accuracy and estimation accuracy. Evidently, increasing the size of a set in the partition results in more past instances falling within that set, and this positively effects the estimation accuracy. However, increasing the size of the set also allows more dissimilar instances to lie in the same set, which negatively effects the approximation accuracy. IUP strikes this balance by adjusting the granularity of the data space partition based on the information contained within the similarity measure (Assumption 1) and the time horizon T.[4]

Let m_i be the partitioning parameter IUP used for LL i, which is used for partitioning $[0,1]^{d_i}$ into $(m_i)^{d_i}$ identical hypercubes. This partition is denoted by $\mathcal{P}_i$.[5] IUP estimates the accuracy of each prediction rule for each hypercube $p \in \mathcal{P}_i$, $i \in \mathcal{M}$ separately, by only using the past history from instance arrivals that fall into hypercube p. For each LL i, IUP keeps and updates the following parameters during its operation for LL i:

- $D_i(t)$: A non-decreasing function used to control the deviation probability of the sample mean accuracy of the prediction rule $f \in \mathcal{F}_i$ from its true accuracy.
- $N^i_{f,p}(t)$: Number of times an instance arrived to hypercube $p \in \mathcal{P}_i$ and prediction rule f of expert i is used to make the prediction.
- $\hat{\pi}^i_{f,p}(t)$: Sample mean accuracy of prediction rule f by time t.

IUP alternates between two phases for each LL: exploration and exploitation. The exploration phase enables learning more about the accuracy of a prediction rule for which IUP has high uncertainty. In other words, IUP explores a prediction rule if it has a low confidence on the estimate of the accuracy of that prediction rule. The exploitation phase uses the prediction rule that is assumed to be the best so far in order to minimize the number of incorrect predictions. The decision to explore or exploit is given by checking whether the accuracy estimates of the prediction rules for the hypercube that contains the current instance are close enough to their true values with respect to the number of instances that have arrived so far. For this purpose, the following set is calculated for LL i at time t.

$$\mathcal{F}^{\text{ue}}_{i,p_i(t)}(t) := \{f \in \mathcal{F}_i : N^i_{f,p_i(t)}(t) \leq D_i(t)\} \qquad (4)$$

where $p_i(t)$ is the hypercube in $\mathcal{P}_i$ that contains $x_i(t)$. If $\mathcal{F}^{\text{ue}}_{i,p_i(t)}(t) = \emptyset$, then LL i enters the exploitation phase. In the exploitation phase, the prediction rule with the highest estimated accuracy is chosen to make a prediction, i.e.,

$$a_i(t) := \arg\max_{f \in \mathcal{F}_i} \hat{\pi}^i_{f,p_i(t)}(t)$$

where

$$\hat{\pi}^i_{f,p_i(t)}(t) = \frac{\sum_{t'=1}^{t-1} \text{I}(a_i(t') = f, \hat{y}_i(t') = y(t'))}{N^i_{f,p_i(t)}(t)}.$$

Otherwise, if $\mathcal{F}^{\text{ue}}_{i,p_i(t)}(t) \neq \emptyset$, then LL i enters the exploration phase. In the exploration phase, an under-explored prediction rule in $\mathcal{F}^{\text{ue}}_{i,p_i(t)}(t)$ is randomly chosen to learn about its accuracy.

IUP for LL i:
  Input: $D_i(t)$, T, m_i
  Initialize sets: Create partition $\mathcal{P}_i$ of $[0,1]^{d_i}$ into $(m_i)^{d_i}$ identical hypercubes
  Initialize counters: $N^i_{f,p} = 0, \forall f \in \mathcal{F}_i, p \in \mathcal{P}_i$
  Initialize estimates: $\hat{\pi}^i_{f,p} = 0, \forall f \in \mathcal{F}_i, p \in \mathcal{P}_i$
  while $t \geq 1$ **do**
    Find the set in $\mathcal{P}_i$ that $x_i(t)$ belongs to, i.e., $p_i(t)$
    Let $p^* = p_i(t)$
    Compute the set of under-explored arms $\mathcal{F}^{\text{ue}}_{i,p^*}(t)$ given in (4)
    if $\mathcal{F}^{\text{ue}}_{i,p^*}(t) \neq \emptyset$ **then**
      Select a_i randomly from $\mathcal{F}^{\text{ue}}_{i,p^*}(t)$
    else
      Select a_i randomly from $\arg\max_{f \in \mathcal{F}_i} \hat{\pi}^i_{f,p^*}$
    end if
    Produce a prediction $\hat{y}_i$ based on a_i.
    Observe the true label $y(t)$ (delay possible).
    $r = \text{I}(\hat{y}_i(t) = y(t))$
    $\hat{\pi}^i_{a_i,p^*} = \frac{\hat{\pi}^i_{a_i,p^*} N^i_{a_i,p^*} + r}{N^i_{a_i,p^*}+1}$
    $N^i_{a_i,p^*}++$
    $t = t+1$
  end while

Figure 2: Pseudocode of IUP for LL i.

Hedge Algorithm for the EL

Let $\hat{\boldsymbol{y}}(t) = (\hat{y}_1(t), \ldots, \hat{y}_M(t))$ be the vector of predictions made by the LLs at time t, by their prediction rules $\boldsymbol{a}(t) = (a_1(t), \ldots, a_M(t))$. We assume that the EL uses the Hedge

[4] It is also possible to strike this balance without knowing the time horizon by using a standard method called the doubling trick.

[5] Instances lying at the edges of the hypercubes can be assigned to one of the hypercubes in a random fashion without affecting the derived performance bounds.

algorithm (Freund and Schapire 1995), to produce the final prediction $\hat{y}(t)$.[6]

Hedge keeps a weight vector $\boldsymbol{w}(t) = (w_1(t), \ldots, w_M(t))$, where $w_i(t)$ denotes the weight of LL i, which represents the algorithm's trust on the predictions of LL i. After observing $\hat{\boldsymbol{y}}(t)$, Hedge samples its final prediction from this set according to the following probability distribution:

$$\Pr(\hat{y}(t) = \hat{y}_i(t)) := w_i(t) / \left(\sum_{j=1}^{M} w_j(t) \right).$$

This implies that Hedge will choose the LLs with higher weights with higher probability. Whenever the prediction of an LL is wrong, its weight is scaled down by $\kappa \in [0, 1]$, where κ is an input parameter of Hedge. As shown in (Freund and Schapire 1995), the regret of Hedge with respect to the best fixed LL is bounded by $O(\sqrt{T \log M})$ for an arbitrary sequence of predictions made by the LLs. We will use this property of Hedge together with the regret bound for LLs to obtain a data-dependent regret bound for the EL.

Analysis of the Regret

In this section we prove bounds on the regrets given in Equations 1 and 3, when the LLs use IUP and the EL uses Hedge as their algorithms. Due to limited space the proofs are given in the supplemental material. The following theorem bounds the regret of each LL.

Theorem 1. ***Regret bound for an LL.*** *When LL i uses IUP with control function $D_i(t) = t^{2\alpha/(3\alpha+d_i)} \log t$ and partitioning parameter $m_i = \lceil T^{1/(3\alpha+d_i)} \rceil$, for any sequence $\boldsymbol{x}^T$ we have*

$$R_i(T) \leq T^{\frac{2\alpha+d_i}{3\alpha+d_i}} \left(2^{d_i} |\mathcal{F}_i| \log T + C_{i,1} \right) + T^{\frac{d_i}{3\alpha+d_i}} C_{i,2}$$

where

$$C_{i,1} := 3Ld_i^{\alpha/2} + \frac{2Ld_i^{\alpha/2} + 2}{(2\alpha + d_i)/(3\alpha + d_i)}$$
$$C_{i,2} := 2^{d_i} |\mathcal{F}_i| + \beta 2^{d_i+1}$$
$$\beta := \sum_{t=1}^{\infty} 1/t^2.$$

Theorem 1 states that the difference between the expected number of correct predictions made by IUP and the highest expected number of correct predictions LL i can achieve, increases as a sublinear function of the sample size T. This implies that the average excess prediction error of IUP compared to the optimal set of predictions that LL i can make converges to zero as the sample size approaches infinity. Since the regret bound is uniform over time, this lets us exactly calculate how far IUP is from the optimal strategy for any finite T, in terms of the average number of correct predictions. Basically, we have $R_i(T)/T = O\left(T^{-\frac{\alpha}{3\alpha+d_i}}\right)$.

As a corollary of the above theorem, we have the following *confidence bound* on the accuracy of the predictions of LL i made by using IUP.

Corollary 1. ***Confidence bound for IUP.*** *Assume that IUP is run with the set of parameters given in Theorem 1. Let $ACC_{i,\epsilon}(t)$ be the event that the prediction rule chosen by IUP for LL i at time t has accuracy greater than or equal to $\pi_{f_i^*(x_i(t))}(x(t)) - \epsilon_t$. If IUP exploits at time t, we have*

$$\Pr(ACC_{i,\epsilon_t}(t)) \geq 1 - 2|\mathcal{F}_i|/t^2$$

where $\epsilon_t = (5Ld_i^{\alpha/2} + 2)t^{-\alpha/(3\alpha+d_i)}$.

Corollary 1 gives a confidence bound on the predictions made by IUP for each LL. This guarantees that the prediction made by IUP is very close to the prediction of the best prediction rule that can be selected given the instance. For instance, in a medical application, the result of this corollary can be used to calculate the patient sample size required to achieve a desired level of confidence in the predictions of the LLs. For instance, for every (ϵ, δ) pair, we can calculate the minimum number of patients N^* such that, for every new patient $n > N^*$, IUP will not choose any prediction rule with suboptimality greater than $\epsilon > 0$ with probability at least $1 - \delta$, when it exploits.

The next theorem shows that the regret of the ensemble learner grows sublinearly over time. Moreover, the term with the highest regret order scales by $|\mathcal{F}_{i^*}|$ but not the sum of the number of prediction rules of all the LLs.

Theorem 2. ***Regret bound for the EL*** *When the EL runs Hedge(κ) with $\kappa = 1/\left(1 + \sqrt{\frac{2 \log M}{T}}\right)$, the regret of the EL with respect to the optimal data-dependent prediction rule of the best expert is bounded by*

$$R_{ens}(T) \leq \sqrt{2T \log M} + \log M$$
$$+ T^{\frac{2\alpha+d_{i^*}}{3\alpha+d_{i^*}}} \left(2^{d_{i^*}} |\mathcal{F}_{i^*}| \log T + C_{i^*,1} \right) + T^{\frac{d_{i^*}}{3\alpha+d_{i^*}}} C_{i^*,2}$$

Hence for $\alpha < d_{i^*}$, $R_{ens}(T) = O\left(|\mathcal{F}_{i^*}| T^{\frac{2\alpha+d_{i^*}}{3\alpha+d_{i^*}}}\right)$.

Theorem 2 implies that the highest time order of the regret does not depend on M. Hence, the algorithm's learning speed scales well with the number of experts. Since regret is measured with respect to the optimal data-dependent strategy of the best expert, it is expected that the ELs performance will improve when better experts are added to the system.

Extensions

Active EL: The prediction accuracy of an LL can be low when it explores. Since the EL combines the predictions of the LLs, taking into account the prediction of an LL which explores can reduce the prediction accuracy of the EL. In

[6] We decided to use Hedge as the ensemble learning algorithm due to its simplicity and regret guarantees. In practice, Hedge can be replaced with other ensemble learning algorithms. For instance, we evaluate the performance when LLs use IUP and the EL uses Weighted Majority (WM) algorithm (Littlestone and Warmuth 1989) in the numerical results section.

order to overcome this, we propose the following modification: Let $\mathcal{A}(t) \subset \mathcal{M}$ be the set of LLs that exploit at time t. If $\mathcal{A}(t) = \emptyset$, the EL will randomly choose one of the LLs' prediction as its final prediction. Otherwise, the EL will apply an ensemble learning algorithm (such as Hedge or WM) using only the LLs in $\mathcal{A}(t)$. This means that only the predictions of the LLs in $\mathcal{A}(t)$ will be used by the EL and only the weights of the LLs in $\mathcal{A}(t)$ will be updated by the EL. Our numerical results illustrate that such a modification can result in an accuracy that is much higher than the accuracy of the best LL.

Missing labels: The proposed HB method can also tackle missing and erroneous labels. In this case, the parameters of IUP and weights of Hedge will only be updated for instances whose labels are observed later. Assuming that the label of an instance is observed with probability q, independently from other instances, our regret bounds will scale with $1/q$.

If there is a positive probability p_ϵ that the label is incorrect, then our algorithms will still work, and the same regret bounds will hold. However, the benchmark we compare against will change from the best prediction rule to a p_ϵ-near optimal prediction rule. This means that our algorithms will be guaranteed to have sublinear regret with respect to a prediction rule whose accuracy is at least p_ϵ smaller than the accuracy of the best prediction rule given each instance.

Active learning: Our method has the ability to perform active learning, especially in settings where acquiring labels are costly. The same regret bound for IUP will hold, even when it is only allowed to observe the label at time steps it explores. Since the rate of exploration is sublinear in time, the average number of time steps in which there exists one LL that explores tends to zero. This means that the average active learning cost of our method (when observations of labels are costly) tends to zero.

Correlation-based ensemble prediction: In this paper we assumed that the EL makes its predictions solely based on the predictions of the LLs without taking into account the instances seen by the LLs. A modified version of the EL can consider the instances observed by the LLs when making its prediction. This modification can improve the predictive accuracy of the EL especially at times when the LLs' instances are highly correlated.

Assume that the EL uses a distance $D_{i,j}(x_i, x_j)$ to measure the similarity between the instances seen by LL i and LL j. Before taking the weighted average of the predictions of all the LLs, the EL filters out LLs with inconsistent predictions. To do this, the EL clusters LLs based on their similarities, such that for each cluster $\mathcal{C}$, $D_{i,j}(x_i, x_j) < \delta$ for all $i, j \in \mathcal{C}$. Here, $\delta > 0$ is a threshold parameter which is set (and can be adapted) by the EL. Then, within each cluster, the EL performs majority voting to output the cluster's prediction. Only the LLs whose predictions are consistent with the prediction of their cluster are included in the current ensemble.

Illustrative Results

In this section, we evaluate the performance of HB (with active EL defined in Extensions Section, using both IUP for LLs, Hedge for EL and IUP for LLs, WM for EL) with various machine learning techniques on a breast cancer diagnosis dataset from the UCI archive (Mangasarian, Street, and Wolberg 1995). More specifically, we compare HB with three state-of-the-art machine learning techniques (Logistic regression (LR), support vector machine with radial based kernerl function (SVMs), and Adaptive boosting algorithm (AdaBoost)) and the best and the worst LL.

Description of the dataset: The original dataset consists of 569 instances and 30 attributes that are clinically relevant to the breast cancer diagnosis (e.g. tumor radius, texture, perimeter, etc.). The diagnosis outcome (label) is binary, either malignant or benign.

Simulation setup: Each algorithm runs 50 times and the average performances of 50 runs are reported as final result. For HB algorithm, we create 3 LLs, and randomly assign 10 attributes to each LL as its feature types. The LLs do not have any common attributes for fair comparison with non-ensemble algorithms. Hence, $d_i = 10$ for all $i \in \{1, 2, 3\}$. For each run, we simulate 10000 instances by drawing an instance from the original 569 instances uniformly at random at each time step. For the offline benchmarks (LR, SVMs and AdaBoost) are trained using 285 (50%) randomly drawn instances from the original 569 instances. Like HB, they are also tested on 10,000 instances drawn uniformly at random from the original 569 instances (excluding the 285 (50%) training instances). In other word, no training instances were used in testing set, but 50 different training sets were used to compute the average performace. LR and SVMs used all 30 attributes, while the 3 weak learners of AdaBoost used 10 randomly assigned attributes as their feature types.

Results on prediction accuracy: All algorithms make one of the two predictions (benign or malignant) for every instance. We consider three performance metrics: prediction error rate (PER), false positive rate (FPR), and false negative rate (FNR). PER is defined as the fraction of times the prediction is different from the label. FPR and FNR are defined as the rate of prediction error among benign cases and the rate of prediction error among malignant cases, respectively.

Table 1: Comparison of HB with other benchmarks

Units(%)	**Average**			**Std**		
Performance Metric	**PER**	**FPR**	**FNR**	**PER**	**FPR**	**FNR**
HB(IUP+WM)	2.01	1.27	2.98	0.48	0.62	0.71
HB(IUP + Hedge)	2.57	2.22	2.94	0.77	0.82	0.88
Logistic Regression	6.04	8.48	2.94	2.18	4.07	1.3
AdaBoost	6.91	9.55	2.99	2.58	4.82	1.83
SVMs	9.73	14.21	2.98	2.5	4.19	1.98
Best LL of IUP	2.41	2.07	2.97	0.75	0.86	0.88
Average LL of IUP	3.82	3.59	2.91	0.53	0.7	0.8
Worst LL of IUP	5.21	4.97	2.95	0.75	1.08	1.26

As the Table 1 shows, HB (IUP + WM) has 2.01% PER, 1.27% FAR, and 2.98% FNR. Hence, its PER is only 66.7% of PER of the second best of the benchmark algorithms (LR). We also note that PER of the best LL is 60% less than PER of LR. This implies that IUP used by the LLs yields high classification accuracy, because it learns the best data dependent prediction rule of the LLs.

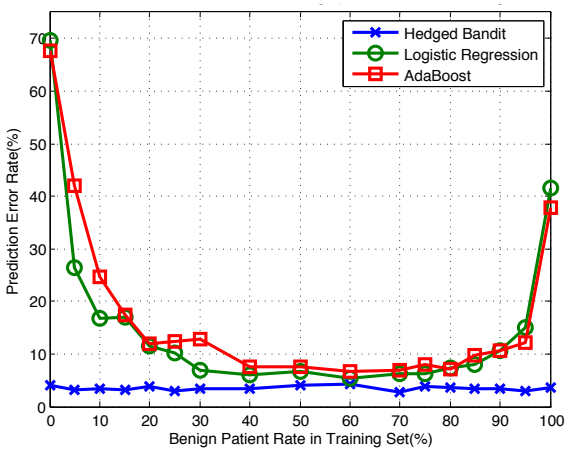

Figure 3: PER of HB, LR and AdaBoost as a function of the composition of the training set.

HB with IUP and WM outperforms the best LL, because it takes a weighted majority of the predictions of LLs as its final prediction, and as we can see from Table 1, all LL are reasonably close in terms of their PER to the best LL, since the PER of the worst LL is 5.82%. In contrast, Hedge puts a probability distribution over the LLs based on their weights, and follows the prediction of the chosen LL. Due to this, it is expected that the deterministic approach (WM) performs better than the probabilistic approach (Hedge), when all LLs have low PER.

Another advantage of HB is that it results in low standard deviation for PER, FPR and FNR, which is expected since IUP provides tight confidence bounds on the accuracy of the prediction rule chosen for any instance for which it exploits.

In Figure 3, the performances of HB (with WM), LR and AdaBoost are compared as a function of the training set composition. Since both LLs and the EL learns online in HB, its performance is invariant to the training set composition. On the other hand, the performance of LR and AdaBoost, the offline algorithms, highly depends on the composition of the training set. This shows the adaptivity of HB to various data distributions.

Although LR, SVMs and Adaboost algorithms are able to change into an online version by retraining each model after arriving every instance, the computational complexity of this online implementations is much higher than that of HB. For instance, the computational complexity of online logistic regression is $O(k \cdot N)$ higher than our HB algorithm where k is the iteration number for optimizing the parameters in logistic regression. Therefore, implementing the online version of benchmarks are not feasible for big data, and spontaneous decision making setting.

Related Works

In this section, we compare our proposed method with other online learning and ensemble learning methods in terms of the underlying assumptions and performance bounds.

Heterogeneous data observations: Most of the existing ensemble learning methods assume that the LLs make predictions by observing the same set of instances (Littlestone and Warmuth 1989), (Fan, Stolfo, and Zhang 1999), (Masud et al. 2009), (Street and Kim 2001), (Minku and Yao 2012). Our methods allow the LLs to act based on heterogeneous data streams that are related to the same event. Moreover, we impose no statistical assumptions on the correlation between these data streams. This is achieved by isolating the decision making process of the EL from the data. Essentially, the EL acts solely based on the predictions it receives from the LLs.

Data-dependent oracle benchmark vs. empirical risk minimization: Our method can be viewed as online supervised learning with bandit feedback, because only the estimated accuracies of the prediction rules chosen by the LLs can be updated after the label is observed. Most of the prior works in this field use empirical risk minimization techniques. In this context, it is assumed that the set of instances and labels $(\{x_i(t)\}_{i\in\mathcal{M}}, y(t))$ are drawn from an unknown i.i.d. distribution (Vapnik 1992). The goal of empirical risk minimization is to come up with a hypothesis which maps the instances to a prediction such that the sample mean of the loss is minimized. This problem translates into an optimization problem which can be solved by standard methods such as linear or convex programming. In contrast, our methods approximate the unknown hypothesis incrementally, by partitioning the data space and estimating the prediction rule accuracies for each set in the partition separately. Due to this property, we are able to obtain strong theoretical performance guarantees both for the short-run (in terms of confidence bounds) and the long-run (in terms of regret bounds).

Moreover, our regret guarantees that hold for any arbitrary data arrival process, hence require no statistical assumptions on how the data is generated. The only required assumption is that the conditional distribution of the label given the data is stationary. Under this condition, we can prove regret bounds with respect to the best data-dependent policy of the best LL, while prior works showed regret bounds only with respect to the best LL (Blum 1997), (Littlestone and Warmuth 1989), (Freund and Schapire 1995). The reason for this is that prior works treat each LL as a black-box, and do not consider how the LLs should learn to optimize the performance of the EL.

Reduced computational complexity: Most ensemble learning methods process the data in chunks. For instance, (Fan, Stolfo, and Zhang 1999) considers an online version of AdaBoost, in which weights are updated in a batch fashion after the processing of each data chunk is completed. Unlike this work, our method processes each instance only once upon its arrival, and do not need to store any past instances. Moreover, the LLs only learn from their own instances, and no data exchange between LLs are necessary. The above properties make our method computationally efficient and suitable for distributed implementation.

Dealing with dynamic data distribution: Since our method does not require any statistical assumptions on the data generation process, it works even when the data distribution is dynamically changing, i.e., when there is concept drift (Žliobaitė 2010). To deal with concept drift (Minku and Yao 2012) considers two ensembles, where one ensemble

is used for making predictions, while the other ensemble is used for tracking the changes in data distribution. Dynamic WM algorithms, which adapt the LLs based on the ELs performance are proposed in (Kolter and Maloof 2005), (Kolter and Maloof 2007). However, the bounds on the performance of the proposed algorithms are derived with respect to an online learner trained on each concept individually, and no confidence bound is derived for each individual instance. A perceptron WM rule is proposed in (Canzian, Zhang, and van der Schaar 2013), where each LL receives predictions of other LLs before making the final prediction. An asymptotic bound on the prediction error probability is provided, under the strong assumption that the prediction error probability of the best prediction rule tends to zero. All of these previous works tackle the concept drift using ad-hoc methods, and do not have strong regret and confidence bound guarantees, as our method does.

References

Arsanjani, R.; Xu, Y.; Dey, D.; Vahistha, V.; Shalev, A.; Nakanishi, R.; Hayes, S.; Fish, M.; Berman, D.; Germano, G.; et al. 2013. Improved accuracy of myocardial perfusion spect for detection of coronary artery disease by machine learning in a large population. *Journal of Nuclear Cardiology* 20(4):553–562.

Beach, A.; Gartrell, M.; Akkala, S.; Elston, J.; Kelley, J.; Nishimoto, K.; Ray, B.; Razgulin, S.; Sundaresan, K.; Surendar, B.; et al. 2008. Whozthat? Evolving an ecosystem for context-aware mobile social networks. *Network, IEEE* 22(4):50–55.

Blum, A. 1997. Empirical support for winnow and weighted-majority algorithms: Results on a calendar scheduling domain. *Machine Learning* 26(1):5–23.

Canzian, L.; Zhang, Y.; and van der Schaar, M. 2013. Ensemble of distributed learners for online classification of dynamic data streams. *arXiv preprint arXiv:1308.5281*.

Cao, Y., and Li, Y. 2007. An intelligent fuzzy-based recommendation system for consumer electronic products. *Expert Systems with Applications* 33(1):230–240.

Eskicioglu, A. M., and Delp, E. J. 2001. An overview of multimedia content protection in consumer electronics devices. *Signal Processing: Image Communication* 16(7):681–699.

Fan, W.; Stolfo, S. J.; and Zhang, J. 1999. The application of adaboost for distributed, scalable and on-line learning. In *Proc. Fifth ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 362–366. ACM.

Freund, Y., and Schapire, R. E. 1995. A decision-theoretic generalization of on-line learning and an application to boosting. In *Computational Learning Theory*, 23–37. Springer.

Kolter, J. Z., and Maloof, M. A. 2005. Using additive expert ensembles to cope with concept drift. In *Proc. 22nd Int. Conf Machine Learning*, 449–456. ACM.

Kolter, J. Z., and Maloof, M. A. 2007. Dynamic weighted majority: An ensemble method for drifting concepts. *The Journal of Machine Learning Research* 8:2755–2790.

Littlestone, N., and Warmuth, M. K. 1989. The weighted majority algorithm. In *30th Annual Symposium on Foundations of Computer Science*, 256–261. IEEE.

Mangasarian, O. L.; Street, W. N.; and Wolberg, W. H. 1995. Breast cancer diagnosis and prognosis via linear programming. *Operations Research* 43(4):570–577.

Masud, M. M.; Gao, J.; Khan, L.; Han, J.; and Thuraisingham, B. 2009. Integrating novel class detection with classification for concept-drifting data streams. In *Machine Learning and Knowledge Discovery in Databases*. Springer. 79–94.

Minku, L. L., and Yao, X. 2012. DDD: A new ensemble approach for dealing with concept drift. *Knowledge and Data Engineering, IEEE Transactions on* 24(4):619–633.

Simons, D. 2008. Consumer electronics opportunities in remote and home healthcare. *Philips Research*.

Street, W. N., and Kim, Y. 2001. A streaming ensemble algorithm (SEA) for large-scale classification. In *Proc. Seventh ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 377–382. ACM.

Swix, S. R.; Stefanik, J. R.; and Batten, J. C. 2004. Method and system for providing targeted advertisements. US Patent 6,718,551.

Vapnik, V. 1992. Principles of risk minimization for learning theory. In *Advances in Neural Information Processing Systems*, 831–838.

Žliobaitė, I. 2010. Learning under concept drift: an overview. *arXiv preprint arXiv:1010.4784*.