Department of Mathematics and Informatics
Faculty of Sciences
University of Novi Sad

Multi-class boosting with adversarial multi-arm bandits on incomplete views

Student research report

Andrea Mihajlović

Supervisors:
Miloš Radovanović, PhD
Sanja Brdar, PhD

It will be published

Abstract

Motivation: General problem when dealing with microbiome data is which approach (sequencing or preprocessing) to use in order to get the input feature set (view) to learn predictive models through machine learning approaches. Moreover, microbiome data comes from various sources and often view incompleteness is inevitable.
Results: The proposed algorithm is the extension of an existing multi-view boosting algorithm based on multi-arm bandits, now able to work in multi-class setting and with incomplete views (views with missing sample representation). At each time step, it proclaims one view as the winning using adversarial multi-arm bandits and uses prediction information from it to update the final model weights and prediction in a boosting process. Three data sets were created from several microbiome studies and then used to examine the performance of created algorithm. One of the experiments showed 7% of increase in F1 score compared to the single-view classifier, while the other one showed 54%. The application domain is not restricted only on microbiome data. Further steps involve examination in other domains.

Acknowledgment

This study was conducted under the support from COST Action 18131 “Statistical and machine learning techniques in human microbiome studies” and its Grant Short-term Scientific Mission (STSM). I would like to thank for the opportunity to collaborate with colleagues from University Bari Aldo Moro (Uniba) in Italy. It was a pleasure to work with Gianvito Pio, PhD, Paolo Mignone, PhD, Michelangelo Ceci, PhD and PhD students from Department of Informatics, Uniba, Italy. Special thanks also go to my colleague from BioSense Institute, Marko Panić, PhD, for great discussion on multi-view and Adaboost approaches. Last but not least, thanks to my supervisors Miloš Radovanović, PhD and Sanja Brdar, PhD, as much as, to Tatjana Lončar-Turukalo, PhD for the support during past years.

Contents

Introduction

In recent years, microbiome dysbiosis is associated with many diseases not only as a main diagnostic tool but as an accompanying symptom that occurs in much higher prevalence than usual. Through the so called microbiota-gut-brain axis, the alterations of microbiome can be related to neurodevelopmental conditions, like Autistic Spectrum Disorder (ASD) [1, 2] or on the other hand to the some type of cancers [3, 4, 5]. Sequencing of microbiome samples is convenient, especially in specific cases when certain tests are impossible or hard to conduct. However, despite reducing the sequencing costs in past couple of years, they can still represent a big milling stone, peculiarly for large microbiome studies or whole genome studies.

16s rRNA sequencing became very popular, since not only that the costs were less, but the data itself, compared to the shotgun sequencing. On the other hand, now, a lot of studies are turned towards using both approaches, with more or less samples underwent to 16s rRNA sequencing [2]. Combining data from such different sources is challenging.

Many prediction problems involve using of different feature subsets on the same data as input for machine learning (ML) models. However, standard ML algorithms do not provide a way for analyzing them jointly. Promising results come from *multiview learning*. Views are defined as different feature subsets on the same data (e.g., pictures of some object from different perspectives or feature tables obtained after different sequence preprocessings). Multi-view learning algorithms combine information from different views in order to make the final decision. By combining information, we can get an extra knowledge which can help multi-view algorithm to outperform single-view one. In the core is the way of combining information.

One solution to this problem is feature concatenation. However, as stated in [6], feature level fusion can lead to the curse of dimensionality, even if views are low dimensional in terms of number of features. The other solution, followed here, is decision level fusion. It comprises average voting system between the view classifiers to obtain the final decision, similar to the ensemble technique.

There are several solutions for decision level fusion, that can be divided into several areas depending on the methodology used. Although very popular, co-training technique is limited on the number of views. It was originally proposed by Blum *et al.* [7] for semi-supervised problem which is solved based on the agreement between two views. Different variants were proposed after the original paper, like co-regularization technique [8, 9]. Regularization was used as a method for reducing overfitting. Sindhwani *et al.* [9] extended it to be able to work with multiple views. The well known ML boosting algorithm, Adaboost [10], is extended in multi-class [11] and multi-view [12] case. There are also works on accelerating Adaboost using adversarial multi-arm bandits [13]. The problem with optimizing the base classifier

over the whole space is solved in iterations by choosing one of the subsets based on its usefulness.

Other approaches for multi-view learning are kernel algorithms, like MVLing [14], which uses regularization, and MKL [15], in which co-labeling has been introduced. Both MVLing and MKL were compared with rBoost.SH from Peng *et al.* [6] along with Adaboost [10] on concatenated views, Mumbo [16], eBoost.SH (learning the best expert strategy from a pool of strategies) and iBoost (each view has its own distribution weight) also proposed by Peng *et al.* in the same article [6]. rBoost.SH outperformed them all. rBoost.SH is based on partial information game and it is briefly explained in further sections.

The biggest problem with aforementioned algorithms is that they can only work when samples are fully observed on views. More realistic situation is when some of the representations of a sample are missing in the corresponding views (partially observed). Xu *et al.* [17] considered not only incomplete views in terms of samples but also in terms of features. They proposed multi-view model to complete the missing features with multiple views (including the case when all features are missing) using SOR-like optimization algorithm. Two very recent paper have drawn the attention onto incomplete views in multi-view learning. Model SURE [18] includes data incompleteness in multi-view clustering task. Model LHGN [19] uses graph learning for classification with incomplete views. Further, as incomplete view we will consider only the view in which the total feature representation of a sample is missing.

Our aim is to extend an existing multiview learning approach proposed by Peng *et al.* [6] to work in multi-class case with incomplete views and evaluate combinations of different types of microbiome views (e.g., 16s rRNA / shotgun sequences, pipeline and parameter settings) in a classification task. However, there are no strict limits for application domain. Other similar solutions uses different methodology or solves different machine learning task. For example, Mumbo is also a multi-view algorithm for multi-class classification, but it maintains one distribution of examples on each view, unlike our method which is multi-class classification algorithm with shared weight distribution over the views. Compared to AdaBoost.MH.Exp3.P [13], we use different reward definition.

Since the proposed algorithm is mixture of adversarial multi-arm bandits, boosting process, multi-class classification and incomplete views, in Section 2 each of these concepts will be explained along with the proposed algorithm in detail. Section 3 presents the benchmark datasets and Section 4 experiment settings and results. In the end, in Section 5 closing remarks and future works are given.

Methodology

To be able to understand the proposed extension, first, core algorithms will be reviewed. The proposed model is a combination of different algorithms. At the core of proposed model is a boosting with a shared weight (Section 2.2), which will give us a convergence guarantees [20]. Furthermore, it uses adversarial multi-arm bandits (Section 2.1) for choosing the best view. In Section 2.3 more details are given on the proposed algorithm.

2.1 Adversarial multi-arm bandits

Multi-arm bandits problem is originally proposed by Robbins [21]. In this problem, in sequence of trials (T), a player chooses one out of K actions (in gambling: pulling of slot arms) such that the total reward it receives is maximized. The player has to make a trade-off between trying out different machines (*exploration*) and playing with what is believed to be the best one (*exploitation*) [20]. Under the assumption that each arm has a different distribution of rewards, the player needs to find the arm with the best expected return as early as possible and keep gambling with it [22]. This problem is not exclusive for gambling, but is applicable to other domains like finances, marketing, medicine, traffic engineering, etc.

In several papers, Auer *et al.* [20, 22, 23] proposed *adversarial multi-arm bandit algorithms*, as a partial information game, in which a player and an *adversary* compete and the reward is the only information propagated to the player. Unlike other bandit problem studies, in this, no statistical assumptions are made about the process generating the rewards. Stochastic bandits, for instance, use the assumption that reward is generated from a fixed distribution.

Further, we review the formalization of this game, described in [22]. There are K possible actions i, $1 \leqslant i \leqslant K$, over a sequence of trials $t = 1, 2, ..., T$. Adversarial multi-arm bandit problem is defined as a partial information game which relies on *full information game* [22], another variant of bandit algorithm. In both variants, for each trial t: *i)* adversary chooses a vector $\mathbf{r}(t) \in [0, 1]^K$ of current rewards by mapping the past history of play $i_1, ..., i_{t-1}$; *ii)* while player chooses an action by picking a number $i_t \in 1, 2, ..., K$ and receives the corresponding reward $r_{i_t}(t)$ associated with taking action i_t, which is assumed to be in a interval $[0, 1]$. However, it is pointed out that generalization on any interval $[a, b]$, for arbitrary $a < b$, is applicable, too [22].

In the partial information game (Algorithm 2 [22]), player observes only the reward $r_{i_t}(t)$ for the chosen action i_t, while on the contrary, in the full information game (Algorithm 1 [20]), it observes full vector $\mathbf{r}(t)$. Total reward of player choosing actions $(i_1, i_2, ..., i_T)$ is defined with

Algorithm 1 Hedge

Input: A real number $\eta > 0$

Initialize $R_i(0) = 0$ for $i = 1, ..., K$

For each $t = 1, 2, ...$ until game ends **do**

  Choose action i_t according to the distribution $\mathbf{p}(t)$ where $p_i(t) = \frac{e^{\eta G_i(t-1)}}{\sum_{j=1}^{K} e^{\eta G_j(t-1)}}$

  Receive the reward vector $\mathbf{r}(t)$ and score gain $r_{i_t}(t)$

  $G_i(t) = G_i(t-1) + r_i(t)$ for $i = 1, ..., K$

Algorithm 2 Exp3

Input: Real $\gamma \in (0, 1]$

Initialize $p_i(1) = 1$ for $i = 1, ..., K$

For each $t = 1, 2, ...$ until game ends **do**

  $q_i(t) = (1-\gamma)\frac{p_i(t)}{\sum_{j=1}^{K} p_j(t)} + \frac{\gamma}{K} i = 1, ..., K$

  Draw i_t randomly accordingly to the probabilities $\mathbf{q}(t) = (q_1(t), ..., q_K(t))$

  Receive reward $r_i(t) \in [0, 1]$

  For $j = 1, ..., K$ set

$$\hat{r}_j(t) = \begin{cases} r_{i_t}(t)/q_i(t) & \text{if } j = i_t \\ 0 & \text{otherwise} \end{cases}$$

  $p_j(t+1) = p_j(t) e^{(\gamma \hat{r}_j / K)}$

$$G_A(T) = \sum_{t=1}^{T} r_{i_t}(t)$$

The performance of a player (algorithm) is measured with *worst-case regret* or difference between $G_A(T)$ and total reward when strategy B for choosing the actions $j_1, ..., j_T$ is followed: $G_B(T) - G_A(T)$, where

$$G_B(T) = \sum_{t=1}^{T} r_{j_t}(t)$$

Depending on the sign of difference, we can say that player gained or lost by following strategy A instead of choosing strategy B [20]. Special case of this regret is *weak regret* of strategy A and the reward for globally best action: $G_{max}(T) - R_A(T)$, where

$$G_{max}(T) = max_j \sum_{t=1}^{T} r_j(t)$$

However, we are usually concerned with the expected regret of the algorithm: $R_A = EG_max - \mathbf{E}[G_A]$, where expected reward of algorithm A is given with

$$\mathbf{E}[G_A] = \mathbf{E}_{i_1,...,i_T}[\sum_{t=1}^{T} r_{i_t}]$$

In 2002, Auer *et al.* [20] presented several extensions of the partial information game algorithm Exp3 [23, 22], but we will now focus on **Exp3.P** (Algorithm 3).

Algorithm 3 Exp3.P

Input: Reals $\sigma > 0$ and $\gamma \in (0, 1]$

1: Initialize $p_i(1) = e^{\frac{\sigma\gamma}{3}}\sqrt{\frac{T}{K}}$ for $i = 1, ..., K$

2: **For each** $t = 1, 2, ..., T$ **do**

3: $q_i(t) = (1-\gamma)\frac{p_i(t)}{\sum_{j=1}^{K} p_j(t)} + \frac{\gamma}{K}, i = 1, ..., K$

4: Choose i_t randomly according to distribution $\mathbf{q}_t = (q_1(t), ..., q_K(t))$

5: Receive reward $r_i(t) \in [0, 1]$

6: **For each** $j = 1, ..., K$

i): $\hat{r}_j(t) = \begin{cases} r_{i_t}(t)/q_i(t) & \text{if} j = i_t \\ 0 & \text{otherwise} \end{cases}$ ii): $p_j(t+1) = p_j(t)e^{(\frac{\gamma}{3K}(\hat{r}_j(t) + \frac{\sigma}{q_j(t)\sqrt{KT}}))}$

In order to ensure exploration of Exp3.P, distribution of probabilities for taking an action is represented by weighted combination of the probability of taking the action uniformly and the one that is exponentially proportional to the cumulative rewards resulting from taking the action in the past (line 3) [6]. The problem might be that some views have always very high probability and that others might not be selected at all. To encourage actions that are unlikely to be taken, due to low probabilities, instead of reward for taken action j_t, estimated reward $\hat{r}_{j_t} = r_j(t)/q_j(t), j$ is used (line 6), as in Exp3 (Algorithm 2). Note that Exp3.P is a no-regret algorithm [6].

2.2 Boosting with shared weight distribution

Here we describe the simplest multi-view algorithm for boosting with shared sample weight distribution i.e. **Boost.SH** and its extension to adversarial multi-arm bandits. They were introduced by Peng *et al.* [24, 6], but only for binary classification. Unlike AdaBoost, where each view has its own re-sampling weight [13], this algorithm uses a single re-sampling distribution for all views at each boosting round [24]. The algorithm is given in the following.

Boost.SH (Algorithm 4) starts with training data $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$, where $\mathbf{x}_i = \{x_i^1, x_i^2, ...x_i^K\}$ is training instance with $x_i^j \in \Re^{q_j}$ as the jth view of instance x_i and $y_i \in \mathcal{Y} = \{-1, +1\}$ as class label of instance x_i. There is an assumption that instances $(\mathbf{x}_i, y_i)$ are independent and identically distributed according to a probability distribution D (unknown) over $\mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \Re^q$ and $q = \sum_{j=1}^{K} q_j$ [6].

At each time step, the algorithm trains *weak classifiers* on each view independently (line 4) and based on the output predictions it computes an *edge* per view (line 5). Afterwards, it chooses the view with the largest edge as the winning view (line 6). The winning view edge and classifier are used to update the shared weight, which was initially uniformly distributed among instances. In testing phase, Boost.SH builds the final classifier as a weighted sum of base classifiers for the winning view at time t.

Algorithm 4 Boost.SH

Input: $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$

1: Initialize $w_i^1 = \frac{1}{n}$ for $i = 1, ..., n$

2: **For each** $t = 1, 2, ..., T$ **do**

3: **For each** $j = 1, 2, ..., K$ **do**

4: Compute base classifier $h_t^j(\mathbf{x}^j)$ with $\mathbf{w}^t$

5: Compute edge $\theta_t^j = \sum_{i=1}^{n} w_i^t y_i h_t^j(x_i^j)$

6: $\alpha_t = \frac{1}{2}\ln(\frac{1+\theta_t^{j^*}}{1-\theta_t^{j^*}}), j^* = \text{argmax}_j\, \theta_t^j$

7: $w_i^{(t+1)} = \frac{w_i^t}{Z_t^{j^*)}} \cdot e^{-\alpha_t y_i h_t^{j^*}(x_i^{j^*})}$

where $Z_t^{j^*}$ is chosen such that $\sum_{i=1}^{n} w_i^{(t+1)} = 1$

Output: $H(\hat{\mathbf{x}}) = sign(\sum_{t=1}^{T} \alpha_t h_t^{j_t}(\hat{\mathbf{x}}^{j_t})$

Despite establishing *consistency* by using shared weight distribution, *diversity* problem remains with using Boost.SH [6]. In general, the algorithm will be limited to learn only from the views with the maximal edge (line 6), and may never choose the one with lower edges at all. The extreme would be the case when a view is constantly the winning one, which is not a different case than learning from a single view.

This was a motive for introducing randomized version of Boost.SH or **rBoost.SH** (Algorithm 5) [6]. It incorporates adversarial multi-arm bandits (Algorithm 3) into a boosting process (Algorithm 4) by learning probabilities $\mathbf{p}_{1\times M}$ for choosing actions (a.k.a. views). The adversarial multiarmed bandit problem is addressed using exponentially weighted average forecaster algorithm [6, 25]. The forecaster algorithm modifies a probability distribution over actions, such that the probability of choosing an action increases or decreases exponentially with the average rewards associated with it [6]. In contrast to Boost.SH, rBoost.SH computes classifier only along the chosen view j (line 5). The reward for chosen view is divided by probability for choosing that view (line 7 *i)*), which encourages the views with low probability to be chosen. Output is expressed as sign of weighted sum of test sample predictions with the winning view weak classifiers at each time step.

Algorithm 5 rBoost.SH

Input: $S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}$; $\sigma > 0$; $\gamma \in (0, 1]$

Initialize $w_i^1 = \frac{1}{n}$ for $i = 1, ..., n$; $p_j^1 = e^{\frac{\sigma\gamma}{3}\sqrt{\frac{T}{K}}}$ for $j = 1, ..., K$

For each $t = 1, 2, ..., T$ **do**

$q_j^t = (1-\gamma)\frac{p_j^t}{\sum_{k=1}^{K} p_k^t} + \frac{\gamma}{K}, j = 1, ..., K$

Let j^* be the view chosen according to $\mathbf{q}_t$

Compute base classifier $h_t^{j^*}(\mathbf{x}^{j^*})$ with $\mathbf{w}^t$

Compute edge $\theta_t^{j^*} = \sum_{i=1}^{n} w_i^t y_i h_t^{j^*}(x_i^{j^*})$ and $r_t^{j^*} = 1 - \sqrt{1 - (\theta_t^{j^*})^2}$

For $k = 1, ..., K$ set

i): $\hat{r}_t^k = \begin{cases} r_t^k / q_k^t & \text{if } k = j^* \\ 0 & \text{otherwise} \end{cases}$ ii): $p_{t+1}^k = p_t^k e^{(\frac{\gamma}{3K}(\hat{r}_t^k + \frac{\sigma}{q_t^k\sqrt{KT}}))}$

$\alpha_t = \frac{1}{2}\ln(\frac{1+\theta_t^{j^*}}{1-\theta_t^{j^*}})$

$w_i^{(t+1)} = \frac{w_i^t}{Z_t^{j^*)}} \cdot e^{-\alpha_t y_i h_t^{j^*}(x_i^{j^*})}$

where $Z_t^{j^*}$ is chosen such that $\sum_{i=1}^{n} w_i^{(t+1)} = 1$

Output: $H(\hat{\mathbf{x}}) = sign(\sum_{t=1}^{T} \alpha_t h_t^{j_t}(\hat{\mathbf{x}}^{j_t})$

A large number of views requires more trials to explore them so that the best views can be exploited [6]. Since the probability of selecting the views increases exponentially, rBoost.SH can quickly discover the views with a large edge. This is highly desirable, especially when there exist noisy views, but also since it increases computational efficiency, resulting in a reduced number of weak classifiers [6].

2.3 irBoost.SH: rBoost.SH extension

Despite good results, rBoost.SH has a disadvantage that it only comprises binary classification tasks. It was explicitly defined that ground truth values $y \in \{-1, +1\}$ [6]. Additionally, if some of the samples misses representation in some view(s), then rBoost.SH fails. We propose an extension for using rBoost.SH in case of multi-class classification and incomplete views. We denoted it with **irBoost.SH**. It is presented in the Algorithm 6.

Our problem set S consists of views of unequal shape. For each view j, $j = 1, ..., K$, $\mathcal{N}_j$ is a sample set with samples that have representation in that view. A set of all samples is given with

$$\mathcal{N} = \bigcup_{j=1}^{K} \mathcal{N}_j$$

Total number of samples for prediction or number of samples in $\mathcal{N}$ is denoted with $|\mathcal{N}|$. Every sample (not sample representation) is labeled with class label $y_i \in \mathcal{Y} \subset \mathbb{N}$. Number of

features per view can also be unequal, but the views must have disjoint feature sets.

We start from initializing weights w_i for each sample $i \in \mathcal{N}$ and view probabilities p_j for each view $j = 1, ..., K$ (line 1). Steps 3 and 4 in irBoost.SH are same as steps 3 and 4 in rBoost.SH. Besides outputting predicted label (Equation 2.1), weak classifier in irBoost.SH outputs a *class probability vector* $\mathbf{h}_i$ for each sample (line 5) - a confidence vector with components in $[0, 1]$ for each class $l \in \mathcal{Y}$. Predicted label is used in edge calculation (line 7), thus indirectly in reward calculation (line 8), and in shared weight update (line 10). On the contrary, the proposed algorithm uses class probabilities in the output prediction.

However, because of our setting with incomplete views, unknown number of samples does not have representation in all views. To resolve this, we define modified predicted label (Equation 2.1) and modified class probability vector (Equation 2.2) for $i \in \mathcal{N}$. They will help us to formally introduce missing sample representation. Zero in Equation 2.1 does not represent 0-class, but it is an artificial label (notice $\mathcal{Y} \subset \mathbb{N}$) for unknown class label prediction of sample representation not contained in that view. In that case, artificial prediction, and hence unknown sample representation, will not contribute to the edge nor weight calculations. Also, in Equation 2.2, $\vec{0}$ is unknown probability vector for the same reason, but used in the prediction phase.

$$\hat{h}^j(x_i^j) = \begin{cases} h^j(x_i^j) & \text{, if } i \in \mathcal{N}_j \\ 0 & \text{otherwise} \end{cases} \tag{2.1}$$

$$\hat{\mathbf{h}}^j(x_i^j) = \begin{cases} \mathbf{h}^j(x_i^j) & \text{, if } i \in \mathcal{N}_j \\ \vec{0} & \text{otherwise} \end{cases} \tag{2.2}$$

Since our task is multi-class classification, we adjust edge formula from rBoost.SH for any natural number as class label. Notice that multiplication of real and predicted label $y_i h_t^j$ when they are -1 or +1 as in rBoost.SH, would be $+1$ for equal real and predicted label i.e. 1×1, $(-1) \times (-1)$, and -1 for unequal i.e. $1 \times (-1)$, $(-1) \times 1$. Define function $\mathbb{I}$ as in Equation 2.3.

$$\mathbb{I}[a = b] = \begin{cases} 1 & \text{, if } a = b \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

It is an indicator function which accounts agreement of two passed values. We modify sum of multiplications $y_i h(x_i)$, for $i = 1, ..., n$, $n = |\mathcal{N}|$, $y_i \in \{-1, +1\}$ and $\mathbb{I}$ in Equation 2.3, as follows

$$\begin{aligned}
\sum_{i \in \mathcal{N}} y_i h(x_i) &= \sum_{\substack{y_i = h(x_i) \\ i = 1, ..., n}} 1 - \sum_{\substack{y_i \neq h(x_i) \\ i \in \mathcal{N}}} 1 = \sum_{\substack{y_i = h(x_i) \\ i \in \mathcal{N}}} 1 - (n - \sum_{\substack{y_i = h(x_i) \\ i \in \mathcal{N}}} 1) \\
&= \sum_{\substack{y_i = h(x_i) \\ i \in \mathcal{N}}} 1 - (\sum_{i \in \mathcal{N}} 1 - \sum_{\substack{y_i = h(x_i) \\ i \in \mathcal{N}}} 1) = 2 \sum_{\substack{y_i = h(x_i) \\ i \in \mathcal{N}}} 1 - \sum_{i \in \mathcal{N}} 1 \\
&= 2 \sum_{i \in \mathcal{N}} \mathbb{I}[y_i = h(x_i)] - \sum_{i \in \mathcal{N}} 1 = \sum_{i \in \mathcal{N}} 2\mathbb{I}[y_i = h(x_i)] - \sum_{i \in \mathcal{N}} 1 \\
&= \sum_{i \in \mathcal{N}} (2\mathbb{I}[y_i = h(x_i)] - 1) = \sum_{i \in \mathcal{N}} 2(\mathbb{I}[y_i = h(x_i)] - \frac{1}{2})
\end{aligned}$$

$$= 2 \sum_{i \in \mathcal{N}} (\mathbb{I}[y_i = h(x_i)] - \frac{1}{2})$$

Algorithm 6 irBoost.SH

Input: $T; K; \sigma > 0; \gamma \in (0, 1]; S = \{(\mathbf{x}_i, y_i)\}_{i \in \mathcal{N}}$, where $\mathcal{N} = \bigcup_{j=1}^{K} \mathcal{N}_j$, and $\mathcal{N}_j$ is a set of samples in jth view; $y_i \in \mathcal{Y} \subset \mathbb{N}$

1: Initialize $w_i^1 = \frac{1}{|\mathcal{N}|}$ for $i \in \mathcal{N}$; $p_j^1 = e^{\frac{\sigma\gamma}{3}\sqrt{\frac{T}{K}}}$ for $j = 1, ..., K$

2: **For each** $t = 1, 2, ..., T$ **do**

3: $q_j^t = (1 - \gamma)\frac{p_j^t}{\sum_{k=1}^{K} p_k^t} + \frac{\gamma}{K}, j = 1, ..., K$

4: Let j^* be the view chosen according to $\mathbf{q}_t$

5: Compute base classifier prediction $h_t^{j^*}(x_i^{j^*})$ and class probabilities $\mathbf{h}_t^{j^*}(x_i^{j^*})$ with w_i^t, for $i \in \mathcal{N}_{j^*}$

6: Set $\hat{h}_t^{j^*}(x_i^{j^*})$ with 2.1 and $\hat{\mathbf{h}}_t^{j^*}(x_i^{j^*})$ with 2.2

7: Compute edge $\theta_t^{j^*} = 2 \sum_{i \in \mathcal{N}} w_i^t (\mathbb{I}[y_i = \hat{h}_t^{j^*}(x_i^{j^*})] - 0.5)$ where $\mathbb{I}[\cdot = \cdot]$ is given in 2.3

8: Compute reward $r_t^{j^*} = 1 - \sqrt{1 - (\theta_t^{j^*})^2}$

9: For $k = 1, ..., K$ set

$$\text{i): } \hat{r}_t^k = \begin{cases} r_t^k / q_k^t & \text{if } k = j^* \\ 0 & \text{otherwise} \end{cases} \qquad \text{ii): } p_{t+1}^k = p_t^k e^{(\frac{\gamma}{3K}(\hat{r}_t^k + \frac{\sigma}{q_t^k \sqrt{KT}}))}$$

10: $\alpha_t = \frac{1}{2} \ln(\frac{1 + \theta_t^{j^*}}{1 - \theta_t^{j^*}})$

11: $w_i^{(t+1)} = \frac{w_i^t}{Z_t^{j^*)}} \cdot e^{-2\alpha_t (\mathbb{I}[y_i = \hat{h}^{j^*}(x_i^{j^*})] - 0.5)}$

where $Z_t^{j^*}$ is chosen such that $\sum_{i \in \mathcal{N}} w_i^{(t+1)} = 1$

Output: $H(\hat{\mathbf{x}}) = \underset{l \in \mathcal{Y}}{\operatorname{argmax}} \frac{\sum_{t=1}^{T} \alpha_t \hat{\mathbf{h}}_{t,l}^{j^*}(\hat{\mathbf{x}}^{j^*})}{\mathcal{C}}$,

where $\mathcal{C}$ is chosen such that $\sum_{l \in \mathcal{Y}} \sum_{t=1}^{T} \alpha_t \hat{\mathbf{h}}_{t,l}^{j^*}(\hat{\mathbf{x}}^{j^*}) = 1$

This is almost exactly the same term used in edge formula (line 7), just weighted. Since $\mathbb{I}$ compares real and predicted values, and always outputs value in $\{0, 1\}$, now class label can be any natural number (without zero). In incomplete view case, the predicted and real class label for sample without representation in the chosen view would never be the same, because of the definition given in Equation 2.1 and the assumption that class labels are from subset of natural numbers. It contributes with zero to the total sum or in other words do not contribute at all. In total, we have

$$\sum_{i \in \mathcal{N}} y_i \hat{h}(x_i) = 2 \sum_{i \in \mathcal{N}} (\mathbb{I}[y_i = \hat{h}(x_i)] - 0.5)$$

Steps after edge calculation are same as in rBoost.SH (line 8,9 and 10 in irBoost.SH), except that in weight update a multiplication of real and predicted label is changed and derived from previous calculations: $y_i\hat{h}(x_i) = 2(\mathbb{I}[y_i = h(x_i)] - 0.5)$.

The predicted class is the one with highest weighted mean class probability estimate across iterations. After averaging weak classifier probabilities, we normalize it with normalization constant $\mathcal{C}$. If test sample does not have a representation in the chosen view at time t, then by Equation 2.2, all components l of probability vector $\hat{\mathbf{h}}$ would be zero. However, since rBoost.SH, and hence irBoost.SH, guarantees diversity, after enough iterations it would choose at least one view for which representation of that sample exists.

Data

Data from several microbiome studies were used. From different datasets obtained from NCBI SRA database and other sources, we created two main datasets. They span different domains (environments and disease tasks) so we could not mix everything in one. Also, we created different experimental settings, to test different situations with incomplete views. Further, mentioned datasets will be explained in details.

3.1 ASD dataset

Autism Spectrum Disorder (ASD) is a severe neurodevelopmental disorder that is primarily characterized by abnormal behavioral symptoms: social interaction impairment, stereotyped behavior, and restricted interests. Recent studies have shown great association between this disease and gut microbiome community through the gut-microbiome-brain axis [2, 26, 27].

Data published in Dan *et al.* [2] is used as an experimental dataset. Data associated with this study come from a cohort of 143 children with clinical diagnosis of ASD, aged 2–13 years old and 143 age and sex-matched typically developing (TD) individuals (average age 5.189 ± 0.170; sex, male: female 127:16) who attended annual physical examination [2]. 16S rRNA sequencing of feces samples (abbrv. 16S) was performed for all 286 individuals, while shotgun metagenomic sequencing (abbrv. shotgun) only for 30 ASD and 30 TD. Two multi-view datasets are created (ASD and ASD-16S) by changing different view types. First one or ASD has already been processed and obtained from Kaggle Website. Two provided feature tables or OTU tables[1] are directly used in multi-view classification as views. They are microbiome abundance profiles of ASD and TD samples, obtained by preprocessing 16S rRNA and shotgun sequences. Total number of samples and features in ASD dataset is shown in Table 3.1. Some of the samples didn't pass preprocessing step, so the total number of samples per view (16S or shotgun) or per class vary compared to studied data in [2, 27].

On the other hand, raw 16S rRNA sequences used in [2, 27] are downloaded with SRA toolkit from NCBI SRA archive [29] and preprocessed in order to create views obtained using different 16S preprocessing steps (ASD-16S dataset). We established pipeline for sequence analysis and used it for preprocessing CRC dataset (Section 3.2), too. It is explained in details in Section 3.3. For ASD-16s dataset 40 views are created and samples are removed only if the pipeline specifies that.

[1] From Operational Taxonomic Unit (OTU) as a way of defining features in biological feature tables.

Table 3.1: Overview of samples and features in ASD dataset

Data	Number of samples			Number of features
	ASD	TD	Total	
16S	143	111	254	1322
Shotgun	30	30	60	5619
16S Only	113	85	198	1322
Shotgun Only	0	4	4	5619
16S + Shotgun	30	26	56	6941

3.2 CRC dataset

Colorectal cancer (CRC) is the second leading cause of death among cancers in the United States [5]. Numerous studies suggests that the gut microbiota may represent a reservoir of biomarkers that would complement existing non-invasive methods such as the widely used fecal immunochemical test (FIT) [3, 4, 5]. The third dataset or CRC dataset consists of OTU tables created from 16S rRNA paired-end sequences from three studies as explained in Table 3.2 for prediction of CRC state: CRC versus Adenoma[2] versus Healthy.

Table 3.2: Number of individuals per class and per study in the CRC dataset

Dataset	Control	Adenoma	CRC	Available metadata
Baxter	172	198	120	Gender, age, weight, height, BMI, country, race
Zackular	30	30	30	Gender, age, weight, height, BMI, country, race, FOBT, medication
Zeller	75	13	41	Gender, age, BMI, country, FOBT
TOTAL	277	241	191	All of the above

Sequence data is chosen based on guidelines produced within COST Action 18131 *Statistical and machine learning techniques in human microbiome studies*. Sequences were downloaded with SRA toolkit [29]. We analyzed sequence data as explained in Section 3.3 and created 44 views for CRC dataset.

[2] A tumor that is not cancer. It starts in gland-like cells of the epithelial tissue (thin layer of tissue that covers organs, glands, and other structures within the body) [28].

3.3 Data preprocessing

Sequence data preprocessing was done using software QIIME 2 [30], version 2021.2, for both ASD-16S and CRC datasets. We created pipeline presented in Figures 3.1, 3.2. After importing FASTQ files with sequences, they were denoised with DADA2 [31], truncated at 240 bp (Figure 3.1) and closed-reference clustered against SILVA database [32] with varying similarity threshold (90%, 94%, 97% and 99%; Figure 3.2). Each of produced OTU tables after clustering are than subjected to *feature filtering*. To supplement our classification task with information from both, filtered and unfiltered OTU tables are used as views. Additionally, in CRC case we produced four OTU tables (for each similarity threshold in clustering) obtained from sequences without denoising step (without DADA2). This is the reason why CRC dataset has 4 views more compared to ASD-16S.

Feature filtration has been done in the following two directions:

1. Filtration by frequency - setting minimal frequency as 2, 5, 10 and 50
2. Filtration by samples - setting minimal percent of samples that the feature must be present in, as 2%, 5%, 10% and 50% of total number of samples

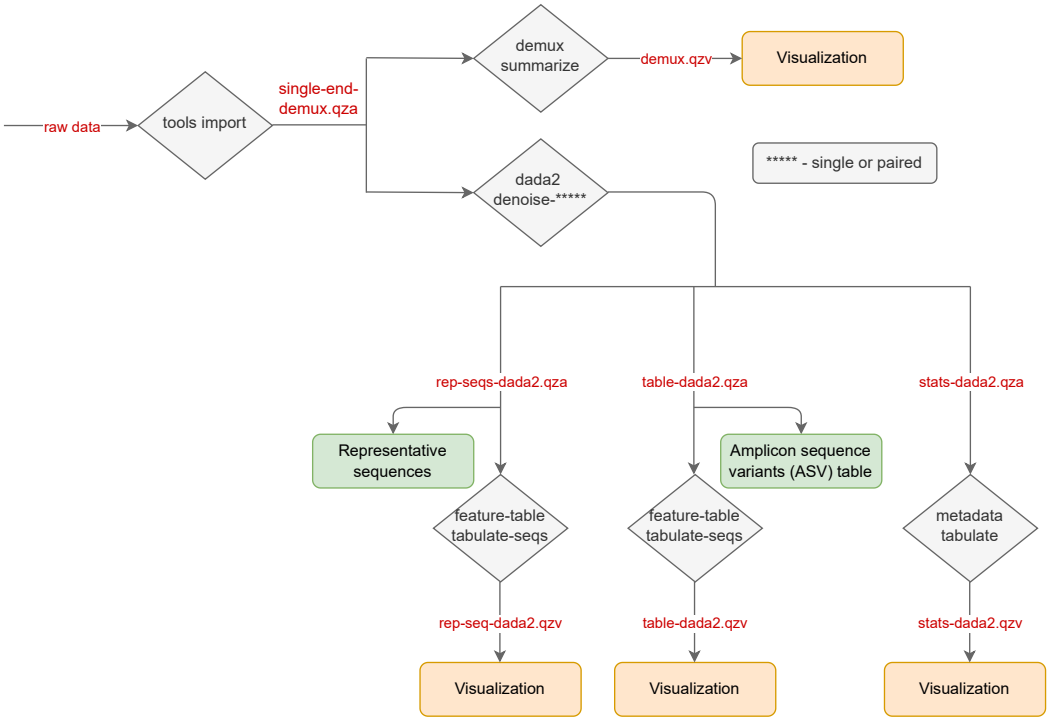


Figure 3.1: QIIME 2 pipeline: Import and quality control

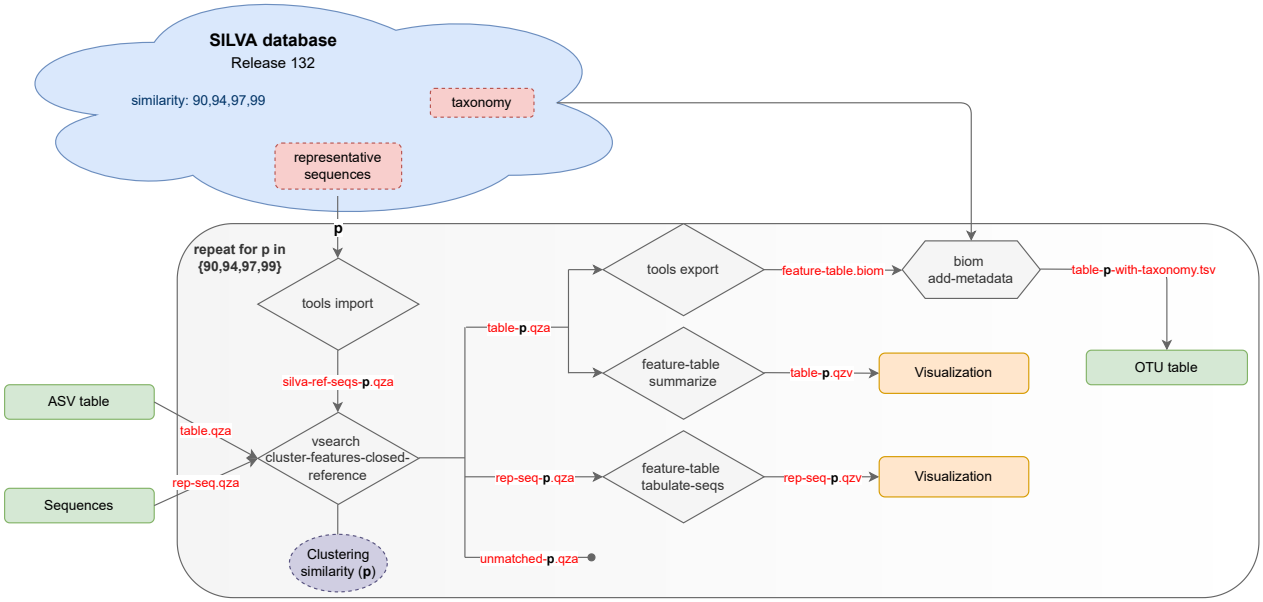


Figure 3.2: QIIME 2 pipeline: OTU clustering

Further, we provide details on sequence analysis in both 16S datasets. Only forward sequences were analyzed and thus, denoise single option was used in DADA2. In ASD-16S, per-sample read count before denoising was in the range 32484-70833 (min-max). Average reads quality scores were very good, more than 30, except one point in the region between 240 and 260 bp. This is the reason for truncating at 240bp in DADA2 denoising. In average, 88% of input ASD-16S sequences passed denosing and in total 2517 of features were identified in ASV[3] table used subsequently in different clustering tasks.

Regarding the CRC sequence analysis, we started with paired-end sequences (both forward and reverse DNA strand) of not so good quality especially after 240 bp. Around 70% of input sequences passed DADA2 denoising and merging for paired-end reads, and sequences were cut at 240bp. After denoising, 14055 ASV features were found, but this number was greatly reduced by clustering into OTUs. DADA2 option for denoising paired-end sequences was used.

[3]From Amplicon Sequence Variant. A table create after DADA2.

Experiments and results

Three datasets (ASD, ASD-16S and CRC) were used, two of which (ASD and ASD-16S) with binary labels and one (CRC) with three class labels. ASD dataset has 2 views and 254 samples in total, ASD-16S 40 views and 286 samples, and CRC dataset 44 views and 708 samples.

For irBoost.SH parameters setting, the guidelines on choosing the right one are used [6]: $\sigma = 0.15$ and $\gamma = 0.3$. Our focus was to evaluate model on microbiome data given in Section 3. We compared results for two base classifiers: Decision Tree (default) and Random Forest (with 50 estimators). The results for the two base classifiers were nearly the same, after enough iterations (more than 50). We have chosen to continue presenting only results for Random Forest as base classifier, since they were slightly better.

For estimating the performance of classifiers, 10-fold cross-validation was used. In multi-view case, cross-validation was adapted such that each view was split into 10-folds, and then the final multi-view training set consisted of training sets for each view, and multi-view test set was constructed from test sets for each view. Different experimental settings has been created for different datasets and further are explained in more details.

4.1 ASD results

For first part of ASD data, ASD dataset with 16S and shotgun data views, we created special experimental setting since there were found incomplete views. Primarily, incompleteness comes from a much smaller shotgun view. Furthermore, during preprocessing of 16S and shotgun OTU tables, some of the samples were discarded, such that we ended up with: *(i)* only 16S samples, *(ii)* only shotgun samples, and *(iii)* samples in the intersection. Therefore, we distinguish three types of input schemes as shown in Table 4.1.

First scheme included only samples from 16S view. In that case, representation data from shotgun was used only for those samples that also have representation in 16S view. For comparison, single-view classifier was evaluated along with irBoost.SH (as base classifier: Random Forest Classifier with 50 estimators). Single-view classifier is standard classifier applied on single-view input, i.e., not multi-view. Same classifier settings were used for single-view classifier and for base classifier in irBoost.SH. Also, as a baseline for multi-view, feature concatenation with filling missing values (due to incomplete shotgun view) with feature average was used for estimating single-view model performance with same setting as earlier mentioned, but now on concatenated table. We will refer to it as baseline multi-view classifier. Finally, our irBoost.SH was trained with multi-view dataset (both 16S and shotgun view data, but with sample representation in 16S).

Second scheme was almost the same as the first one, except it encompassed only samples in shotgun view. There were also two single-view classifiers for comparison, one for only shotgun data and one for concatenated views (with average feature imputation) - baseline multi-view.

For the third scheme, only samples (56) in the intersection of 16S and shotgun views were included. Here we only have one single-view classifier cross-validated on concatenated 16S and shotgun features (baseline multi-view), and our irBoost.SH cross-validated on same two views, but only with samples in the intersection.

Table 4.1: Experimental settings for ASD dataset

Input	Samples	Features	Classifier
1 (incomplete scheme)	254	16S	Single-View Classifier
	254	16s+shotgun	Baseline Multi-View Classifier (concatenation + average shotgun features filling)
	254	16S+shotgun	irBoost.SH
2 (incomplete scheme)	60	shotgun	Single-View Classifier
	60	shotgun+16S	Baseline Multi-View Classifier (concatenation + average 16s features filling)
	60	shotgun+16S	irBoost.SH
3 (full scheme)	56	16S + Shotgun	Baseline Multi-View Classifier (concatenation)
	56	16S + Shotgun	irBoost.SH

Figure 4.1 shows average F1-score performance on ASD dataset for all three schemes (ASD-1, ASD-2 and ASD-3). Confusion matrices for them can be found in the Appendix (Figures A.1-A.8). Model irBoost.SH achieved better results than two other baseline models. Results showed improvement of 7% in average for average F1 score if using Random Forest Classifier as base estimator for our irBoostSH model on ASD dataset, compared to the baseline models. Notice that the first scheme has a larger number of samples compared to the other two.

4.2 CRC and ASD-16S results

To examine how different 16s preprocessing techniques behave in multi-view classification, we used ASD-16s and CRC datasets (Section 3). Additionally, CRC dataset was chosen to examine multi-class case. There are three classes: CRC, Adenoma (merged classes Large Adenoma, Small Adenoma and Adenoma) and Healthy. Average F1-score performance for ASD-16s and CRC-16s is also presented in the Figure (4.1). Here, it would be pointless to concatenate views, since each dataset consists of around 40 views.

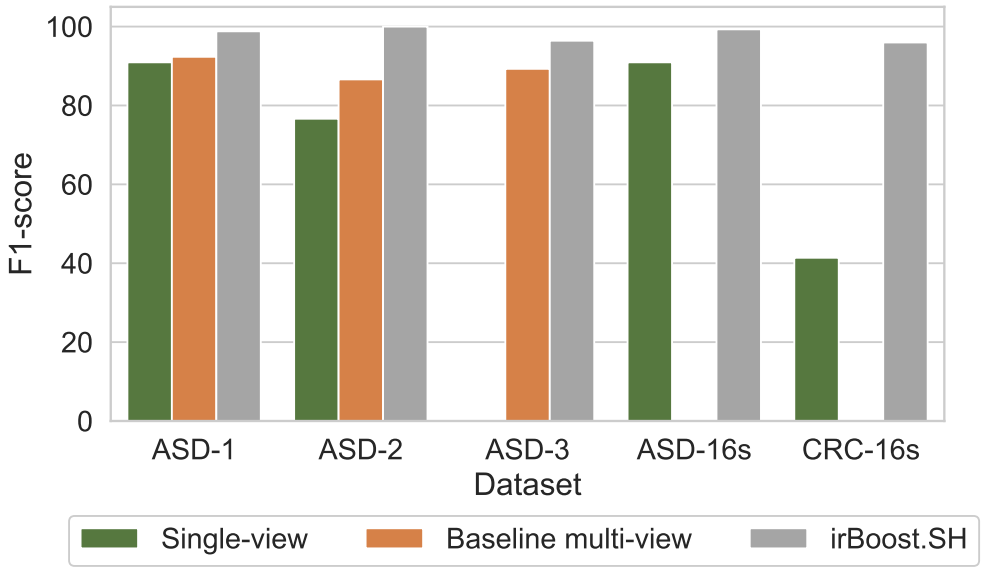
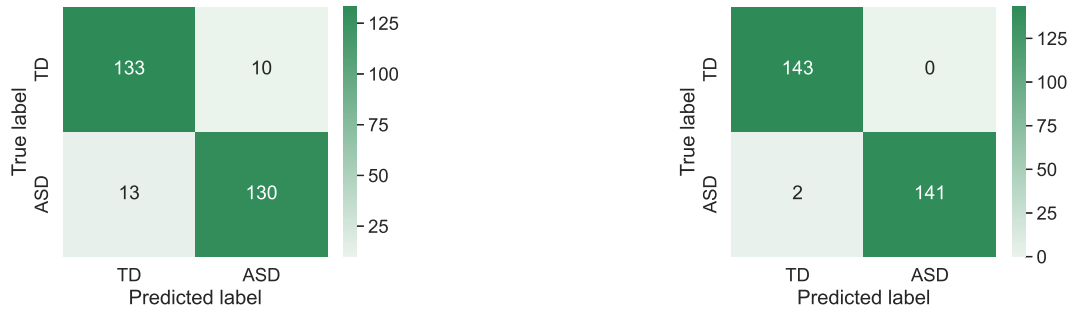


Figure 4.1: Performance

We ran irBoost.sH (Figure 4.3) and exported the winning views at each iteration step, to examine the algorithm diversity with both ASD-16s and CRC dataset. In 50 iterations and with 10-fold cross-validation, every view was chosen at least one time. The most frequently chosen view was used to train single-view classifier for comparison with the proposed model.

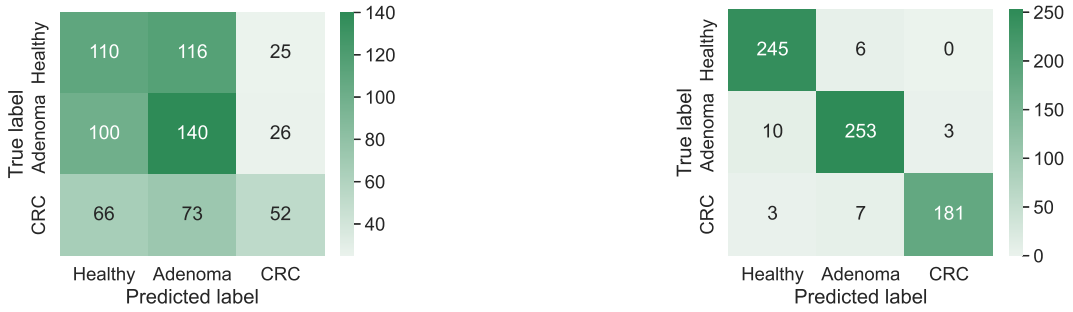
In ASD-16s dataset, the classes are more balanced and sample size is greater (286 samples) compared to ASD dataset. This difference had emerged due to preprocessing in ASD dataset which led to sample reduction. The results for irBoost.SH and ASD-16s show only two missed samples from ASD class, which are predicted to be in TD class (Figure 4.2b). The most frequently used view was OTU table with 90% for similarity threshold in clustering OTUs and which features are filtered if their frequency is less than 10 (Figure 4.2a). Compared to our model, single-view classifier on that view showed slightly worse results.



(a) *(b)*

Figure 4.2: Confusion matrices for irBoost.SH (Random Forest base classifier) trained on (a) OTU table from CRC dataset created with 90% for similarity threshold and by filtering features which frequency is less than 10; and (b) ASD-16S multiview dataset - different preprocessing views

In CRC dataset, the situation with single-view classifier was a bit chaotic (Figure 4.3a). Most frequent view was the OTU table created with 90% for similarity threshold and by



(a) *(b)*

Figure 4.3: Confusion matrices for irBoost.SH (Random Forest base classifier) trained on (a) OTU table from CRC dataset created with 90% for similarity threshold and by filtering features if their frequency is less than 2; and (b) CRC multiview dataset - different preprocessing views

filtering features if their frequency is less than 2. However, it achieved 40% of F1-score. The proposed model achieved a way better results compared to the single-view one. Despite very good results for CRC class, separation of Healthy and Adenoma class remained still hard for the classifier. Overall, F1-score was increased by more than 50%.

Conclusion

In this paper, we investigated the multi-class and incomplete-view problem in multi-view classification. Multi-class refers to the classification task with multiple possible discrete value for target, while incomplete views imply that some of the samples does not have representation in all views. We did not tackle the case when sample variable values are missing. However, this is one of the points that can be improved in future.

Instead of leaving out the incomplete views or filling them with the average information, exploiting the connections between multiple views is the key solution for handling incomplete views. In that manner, we introduced irBoost.SH, as upgraded version of randomized boosting algorithm with adversarial multi-arm bandits [6]. By assuming that different views are generated from a shared subspace, it is possible to estimate the incomplete views by integrating the information from the other observed views through this subspace [17]. Hence, the shared sample weight is defined and updated through our algorithm.

The concepts of game, player, adversarial and reward were introduced as an adversarial multi-arm bandits setting. This was used in a boosting process to probabilistically choose a view at each iteration step. We redefined edge, weight update and prediction formulas, to account for multi-class and incomplete view(s) case. We also found numerous multi-view algorithms for multi-class and incomplete view classification, but they were different to some extent compared to irBoost.SH. One of additional benefits of irBoost.SH is the possibility to explore the best views chosen by the algorithm. However, the experiments showed that information from single view does not have as good predictive power as integrated information (multi-view). More investigation can help us to understand the difference between them.

Proposed algorithm is a good starting point for further investigation. It can be extended to solve a regression task or to encompass semi-supervised setting. Also, the need for good imputation method is great, and this looks like a promising method for dealing with incomplete data in general.

Bibliography

[1] Cheryl S. Rosenfeld. “Microbiome Disturbances and Autism Spectrum Disorders”. In: *Drug Metabolism and Disposition* 43.10 (Apr. 2015), pp. 1557–1571. DOI: 10.1124/dmd.115.063826.

[2] Zhou Dan et al. “Altered gut microbial profile is associated with abnormal metabolism activity of Autism Spectrum Disorder”. In: *Gut Microbes* 11.5 (Apr. 2020), pp. 1246–1267. DOI: 10.1080/19490976.2020.1747329.

[3] Georg Zeller et al. “Potential of fecal microbiota for early-stage detection of colorectal cancer”. In: *Molecular Systems Biology* 10.11 (Nov. 2014), p. 766. DOI: 10.15252/msb.20145645.

[4] Joseph P. Zackular et al. “The Human Gut Microbiome as a Screening Tool for Colorectal Cancer”. In: *Cancer Prevention Research* 7.11 (Nov. 2014), pp. 1112–1121. DOI: 10.1158/1940-6207.capr-14-0129.

[5] Nielson T. Baxter et al. “Microbiota-based model improves the sensitivity of fecal immunochemical test for detecting colonic lesions”. In: *Genome Medicine* 8.1 (Apr. 2016). DOI: 10.1186/s13073-016-0290-3.

[6] Jing Peng et al. “Multiview Boosting With Information Propagation for Classification”. In: *IEEE Transactions on Neural Networks and Learning Systems* 29.3 (Mar. 2018), pp. 657–669. DOI: 10.1109/tnnls.2016.2637881.

[7] Avrim Blum and Tom Mitchell. “Combining labeled and unlabeled data with co-training”. In: *Proceedings of the eleventh annual conference on Computational learning theory - COLT’ 98*. ACM Press, 1998. DOI: 10.1145/279943.279962.

[8] Jiao Wang, Siwei Luo, and Yan Li. “A Multi-view Regularization Method for Semi-supervised Learning”. In: *Advances in Neural Networks - ISNN 2010*. Springer Berlin Heidelberg, 2010, pp. 444–449. DOI: 10.1007/978-3-642-13278-0_57.

[9] Vikas Sindhwani, Partha Niyogi, and Mikhail Belkin. “A Co-Regularization Approach to Semi-supervised Learning with Multiple Views”. In: *Proceedings of the Workshop on Learning withMultiple Views, 22nd*. ICML, Bonn, Germany, 2005.

[10] Yoav Freund and Robert E. Schapire. “A desicion-theoretic generalization of on-line learning and an application to boosting”. In: *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 1995, pp. 23–37. DOI: 10.1007/3-540-59119-2_166.

[11] Trevor Hastie et al. “Multi-class AdaBoost”. In: *Statistics and Its Interface* 2.3 (2009), pp. 349–360. DOI: 10.4310/sii.2009.v2.n3.a8.

[12] Zhijie Xu and Shiliang Sun. “An Algorithm on Multi-View Adaboost”. In: *Neural Information Processing. Theory and Algorithms*. Springer Berlin Heidelberg, 2010, pp. 355–362. DOI: 10.1007/978-3-642-17537-4_44.

[13] Robert Busa-Fekete and Balazs Kegl. “Fast boosting using adversarial bandits”. In: *Proceedings of the 27th International Conference on Machine Learning, Haifa, Israel*. 2010. URL: https://core.ac.uk/download/pdf/46767238.pdf.

[14] Minh Hà Quang, Loris Bazzani, and Vittorio Murino. “A unifying framework for vector-valued manifold regularization and multi-view learning”. In: *Proceedings of the 30th International Conference on Machine Learning*. Vol. 28. 2. June 2013, pp. 100–108.

[15] Xinxing Xu et al. “Co-Labeling for Multi-View Weakly Labeled Learning”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.6 (June 2016), pp. 1113–1125. DOI: 10.1109/tpami.2015.2476813.

[16] Sokol Koço and Cécile Capponi. “A Boosting Approach to Multiview Classification with Cooperation”. In: *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2011, pp. 209–228. DOI: 10.1007/978-3-642-23783-6_14.

[17] Chang Xu, Dacheng Tao, and Chao Xu. “Multi-View Learning With Incomplete Views”. In: *IEEE Transactions on Image Processing* 24.12 (Dec. 2015), pp. 5812–5825. DOI: 10.1109/tip.2015.2490539.

[18] Mouxing Yang et al. “Robust Multi-view Clustering with Incomplete Information”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* (2022), pp. 1–1. DOI: 10.1109/tpami.2022.3155499.

[19] Pengfei Zhu et al. “Latent Heterogeneous Graph Network for Incomplete Multi-View Learning”. In: *IEEE Transactions on Multimedia* (2022), pp. 1–1. DOI: 10.1109/tmm.2022.3154592.

[20] Peter Auer et al. “The Nonstochastic Multiarmed Bandit Problem”. In: *SIAM Journal on Computing* 32.1 (Jan. 2002), pp. 48–77. DOI: 10.1137/s0097539701398375.

[21] H.E. Robbins. “Some aspects of the sequential design of experiments”. In: *Bull. Amer. Math. Soc.* 58.5 (1952), pp. 527–535.

[22] Yoav Freund Peter Auer Nicolo Cesa-Bianchi and Robert E. Schapire. *Gambling in a rigged casino:The adversarial multi-armed bandit problem*. Tech. rep. University of Technology Graz/Universita di Milano/AT&T Labs, Florham Park, NJ, 1998. URL: http://www.dklevine.com/archive/refs4462.pdf.

[23] P. Auer et al. “Gambling in a rigged casino: The adversarial multi-armed bandit problem”. In: *Proceedings of IEEE 36th Annual Foundations of Computer Science*. IEEE Comput. Soc. Press, 1995. DOI: 10.1109/sfcs.1995.492488.

[24] Jing Peng et al. “ShareBoost: Boosting for Multi-view Learning with Performance Guarantees”. In: *Machine Learning and Knowledge Discovery in Databases*. Springer Berlin Heidelberg, 2011, pp. 597–612. DOI: 10.1007/978-3-642-23783-6_38.

[25] Nicolò Cesa-Bianchi. *Prediction, Learning, and Games*. Cambridge University Press, Nov. 2010. 408 pp. ISBN: 0521841089. URL: https://www.ebook.de/de/product/5356845/nicolo_cesa_bianchi_prediction_learning_and_games.html.

[26] Joby Pulikkan, Agnisrota Mazumder, and Tony Grace. “Role of the Gut Microbiome in Autism Spectrum Disorders”. In: *Advances in Experimental Medicine and Biology*. Springer International Publishing, 2019, pp. 253–269. DOI: 10.1007/978-3-030-05542-4_13.

[27] Zhi Liu et al. “Gene variations in Autism Spectrum Disorder are associated with alternation of gut microbiota, metabolites and cytokines”. In: *Gut Microbes* 13.1 (Jan. 2021). DOI: 10.1080/19490976.2020.1854967.

[28] National Cancer Institute. *NCI Dictionaries: adenoma*. URL: https://www.cancer.gov/publications/dictionaries/cancer-terms/def/adenoma.

[29] R. Leinonen, H. Sugawara, and M. Shumway and. “The Sequence Read Archive”. In: *Nucleic Acids Research* 39.Database (Nov. 2010), pp. D19–D21. DOI: 10.1093/nar/gkq1019.

[30] Evan Bolyen et al. “Reproducible, interactive, scalable and extensible microbiome data science using QIIME 2”. In: *Nature Biotechnology* 37.8 (July 2019), pp. 852–857. DOI: 10.1038/s41587-019-0209-9.

[31] Benjamin J Callahan et al. “DADA2: High-resolution sample inference from Illumina amplicon data”. In: *Nature Methods* 13.7 (May 2016), pp. 581–583. DOI: 10.1038/nmeth.3869.

[32] Christian Quast et al. “The SILVA ribosomal RNA gene database project: improved data processing and web-based tools”. In: *Nucleic Acids Research* 41.D1 (Nov. 2012), pp. D590–D596. DOI: 10.1093/nar/gks1219.

Appendix

A.1 ASD-16s features

Table A.1: Number of features in differently preprocessed ASD-16s views

Filtration	**Similarity threshold**			
	90%	**94%**	**97%**	**99%**
Number of samples	286	286	286	286
No	1083	1444	1649	1736
Min frequency 2	1084	1445	1650	1737
Min frequency 5	957	1282	1481	1586
Min frequency 10	859	1135	1321	1431
Min frequency 50	621	815	962	1053
In >2% of samples	632	779	873	907
In >5% of samples	510	602	665	687
In >10% of samples	399	466	501	515
In >20% of samples	327	374	377	376
In >50% of samples	219	237	233	233

A.2 CRC features

Table A.2: Number of features in differently preprocessed CRC views

Filtration	Similarity threshold			
	90%	**94%**	**97%**	**99%**
Number of samples	708	708	708	708
No	2616	3870	4245	4402
Min frequency 2	2616	3870	4245	4402
Min frequency 5	2305	3473	4107	4228
Min frequency 10	2071	3087	3684	3956
Min frequency 50	1565	2377	2951	3247
In >2% of samples	899	1107	1178	1169
In >5% of samples	611	721	751	727
In >10% of samples	417	481	457	437
In >20% of samples	258	254	244	239
In >50% of samples	85	75	70	69

A.3 Scheme 1 results

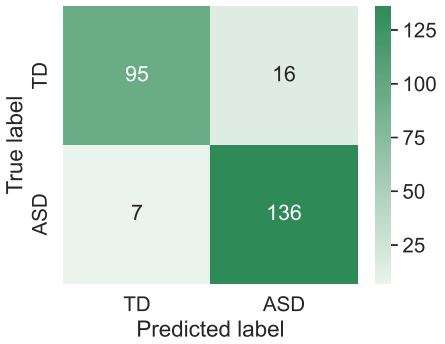


Figure A.1: Confusion matrices for Random Forest trained on ASD dataset with only 16S samples

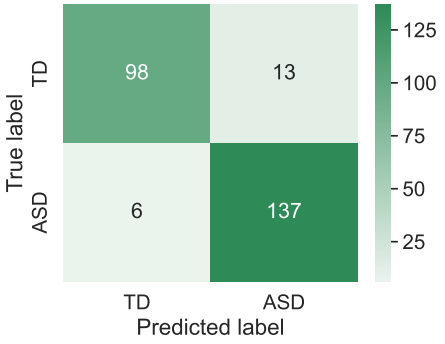


Figure A.2: Confusion matrix for Random Forest trained on ASD dataset with concatenated features from the two views (only 16S samples) and imputed missing shotgun samples

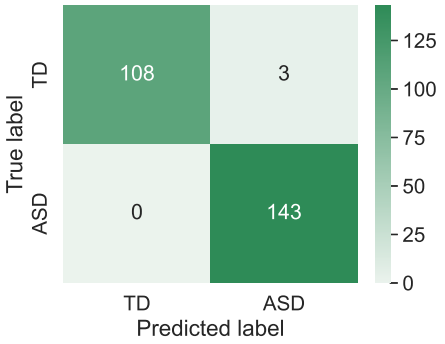


Figure A.3: Confusion matrix for irBoost.SH (Random Forest base classifier) trained on ASD multiview dataset (16S and shotgun views), but only with samples in 16S view

A.4 Scheme 2 results

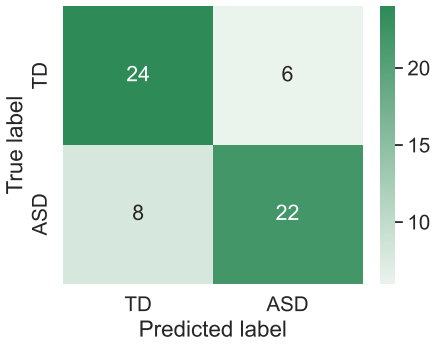


Figure A.4: Confusion matrix for Random Forest trained on ASD dataset with only shotgun samples

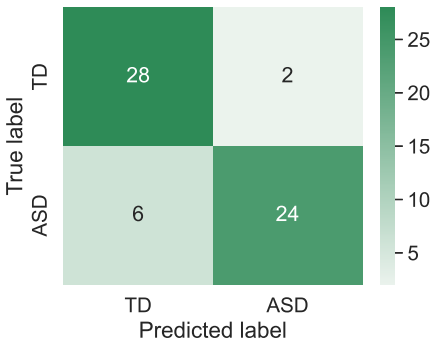


Figure A.5: Confusion matrix for Random Forest trained on ASD dataset with concatenated features from the two views (only shotgun samples) and imputed missing 16S samples

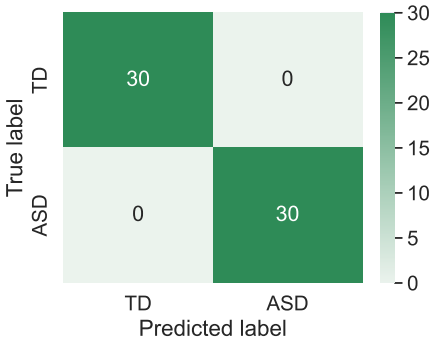


Figure A.6: Confusion matrix for irBoost.SH (Random Forest base classifier) trained on ASD multiview dataset (16S and shotgun views), but only with samples in shotgun view

A.5 Scheme 3 results

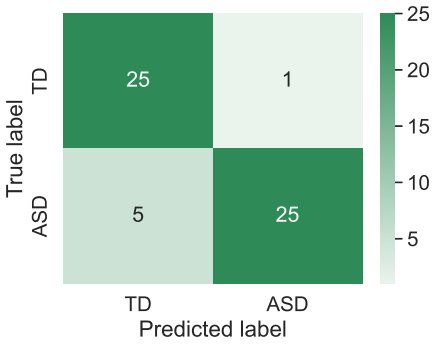


Figure A.7: Confusion matrix for Random Forest trained on ASD dataset with concatenated features from the two views and just samples in the intersection of these views

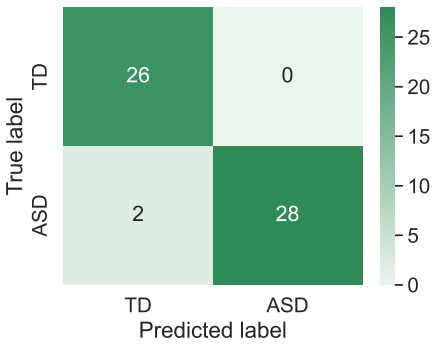


Figure A.8: Confusion matrix for irBoost.SH (Random Forest base classifier) trained on ASD multiview dataset (16S and shotgun views), but only with samples in the intersection