Identification and validation of microbial biomarkers from cross-cohort datasets using xMarkerFinder

Wenxing Gao
Wanning Chen
Wenjing Yin
Xinyue Zhu
Sheng Gao
Lei Liu
Dingfeng Wu
Ruixin Zhu
Na Jiao (jiaona@zju.edu.cn)

Method Article

Keywords: cross-cohort datasets, microbial biomarkers, biomarker identification

Posted Date: August 26th, 2022

DOI: https://doi.org/10.21203/rs.3.pex-1984/v1

Abstract

Microbial signatures have emerged as promising biomarkers and targets for disease diagnosis, prognosis, and remission. However, these biomarkers exhibit contradictory results in different studies, necessitating the identification of universally robust microbial biomarkers. Therefore, we introduce xMarkerFinder, a four-stage computational framework for microbial biomarker identification with comprehensive validations from cross-cohort datasets, including differential signature identification, model construction, model validation, and biomarker interpretation. xMarkerFinder enables the identification and validation of reproducible biomarkers for cross-cohort studies, along with the establishment of classification models and potential microbiome-induced mechanisms. Although xMarkerFinder is initially developed for gut microbiome study, it is generalizable to different omics layers, as well as other habitats. Execution time varies depending on the sample size, selected algorithm, and computational resource. xMarkerFinder can be accessed at https://github.com/tjcadd2020/xMarkerFinder. FAQs, detailed tutorials, issue boards, and a ready-to-use Docker image are provided for users with no prior bioinformatics or statistics training.

Introduction

Microbiome, the complex and dynamic ecosystem colonizing the human body, coordinates with multiple host physiological processes, the perturbation of which has been increasingly suggested to associate with a wide spectrum of diseases[1-3]. Particularly, microbial signatures derived from fecal samples are now emerging as novel non-invasive biomarkers for disease susceptibility[4], progression[5], prognosis[6], and treatment response[7], yet these biomarkers exhibit inconsistency across different studies[8, 9]. Recent advances in high-throughput technologies catalyze the exponential accumulation of publicly available microbial sequencing data, thereby enabling meta-analyses, the integration and exploration of cross-cohort datasets, to increase statistical power in the identification of globally robust disease-related microbial signatures[10, 11]. Machine learning (ML) algorithms have been widely used in data-driven meta-analyses to establish candidate biomarkers and classification models based on such reproducible microbial signatures. Considering the heterogeneity across cohorts, sufficient validations are needed to determine biomarkers that remain effective among different populations. Further, to explore the clinical significance of identified biomarkers, validations of their specificity in a broader context (e.g., other microbiome-linked diseases with similar clinical symptoms) are also recommended. However, the absence of a standardized workflow for such complicated analyses remains a challenge, especially for researchers who are not specialized in bioinformatics.

To address this issue, we introduce xMarkerFinder, a comprehensive and user-friendly workflow for biomarker identification across large-scale populations while accounting for inter-cohort variation in microbiome data. xMarkerFinder was developed by integrating meta-analysis methods, ML models and biomarker interpretation tools, with a focus on biomarker specificity and generalizability evaluation.

Development of the protocol

The xMarkerFinder workflow was initially developed to characterize multi-population microbiome datasets and identify a panel of universally robust microbial biomarkers for colorectal cancer (CRC) screening. To achieve such object, a pipeline for biomarker identification and validation was established (Fig. 1).

Biomarker identification heavily depends on identifying concordant disease-specific signatures across cohorts as the candidate biomarker pool. Since the integration of different cohorts inevitably brings about confounding effects caused by distinct cohort characteristics and technical differences, we employed Meta-analysis Methods with Uniform Pipeline for Heterogeneity in Microbiome Studies (MMUPHin), a widely used meta-analysis method, allowing joint confounding effect correction and differential signature identification between groups in cross-cohort datasets[12]. Further, to determine a panel of signatures with predictive power of host phenotype as *candidate biomarkers*, Triple-E, a curated three-step feature selection procedure dedicated to reducing irrelevant and redundant features while accounting for collinearity issues, was implemented.

The integration of multiple cohorts enables multi-perspective validations to assess the robustness and reproducibility of identified *candidate biomarkers*, which is also the advantage of multi-cohort meta-analysis over common single-cohort studies. Firstly, internal validations within the discovery cohorts including intra-cohort, cohort-to-cohort, and LOCO validations were performed to evaluate the generalizability of *candidate biomarkers* on a cross-cohort level. Next, validations with independent cohorts externally assessed the robustness of *candidate biomarkers*. Considering shared microbial dysregulation between multiple diseases, we further evaluated *candidate biomarkers*' specificity for the disease of interest, aiming to avoid clinical false positives. Such validations simulated the examination of *candidate biomarkers* in a prospective setting and could be used to determine the final *biomarkers*.

In a recent example of this pipeline, we performed an integrated analysis on 16S rRNA gene sequencing data of 1056 fecal samples from four publicly available cohorts to identify adenoma-specific microbial biomarkers[13]. After adjusting potential confounders, Random Forest (RF) classifiers were constructed based on differentially abundant microbial signatures and optimized by recursive feature elimination. A core set of 11 *biomarkers* reached an area under the receiver operating characteristic curve (AUC) of 0.80 for distinguishing between control subjects and adenoma patients, and were validated in two independent cohorts (AUCs=0.78 and 0.84). Further evaluation of the biomarker panel in other microbiome-linked studies revealed its high specificity for adenoma. In addition, we investigated CRC whole metagenomic sequencing (WMS) datasets collected from eight distinct cohorts via a multi-kingdom perspective[14]. Specifically, we systemically assessed microbial single-kingdom and multi-kingdom RF diagnostic models based on an integrated analysis of CRC-associated microbial species, and the best-performing panel of *biomarkers* achieved an AUC of 0.83. Apart from the microbial compositional layer, we further constructed diagnostic models based on microbial functional genes and pathways which also exhibited superior diagnostic capabilities.

Based on above endeavors, xMarkerFinder is developed, and we have optimized it by providing additional ML classifiers and feature selection methods, allowing users to choose suitable algorithms for their own research purposes. Further, comprehensive biomarker interpretation methods are implemented, providing deeper insights into microbiome-mediated mechanisms and potential intervention strategies for restoring dysregulated microbial communities. To better illustrate our protocol, the original datasets used in aforementioned studies are provided[14].

Experimental design

This section outlines the data preparing process and four major stages of xMarkerFinder: differential signature identification, model construction, model validation, and biomarker interpretation, with visualization steps provided whenever possible (Fig. 2). Notably, each step of xMarkerFinder can be conducted independently, and users could skip any one of them according to specific study designs. Detailed instructions of xMarkerFinder are presented in the Procedure part below.

Data preparation

Implementation of xMarkerFinder requires pre-processed microbial profiles from multiple cohorts (here a cohort is defined as dataset from an individual study). Taxonomic, functional, and genetic profiles derived from amplicon sequencing and whole-genome sequencing are all allowed. We encourage the re-use and exploration of public datasets, as well as generating new microbiome datasets via processing sequencing data with consistent pipelines to reduce technical heterogeneity. Processing approaches for microbial sequencing data have been reviewed in refs[15, 16]. To better account for inter-cohort and inter-individual differences and correct for confounding effects, detailed metadata information is encouraged, including but not limited to demographic and biogeographic variables. Users are recommended to separate input microbial profiles into discovery set and external test set (details in Materials part and Box 1), and xMarkerFinder requires at least three cohorts in the discovery set to increase *biomarkers*' robustness and generalizability with sufficient internal validations. To increase statistical power, sufficient sample size in each cohort is strongly recommended, and the necessary sample sizes could be estimated according to refs[17-20]. Further, as microbial dysregulation is associated to a wide spectrum of diseases, the clinical application of *biomarkers* requires a high degree of disease specificity. Here, to avoid false positives in clinical, microbial profiles of other microbiome-linked diseases and corresponding metadata are also needed for validation.

To help users better understand and explore the protocol, we provide example microbial datasets from our previously published research[14], as well as example result files, in our GitHub repository (https://github.com/tjcadd2020/xMarkerFinder).

Stage 1 Differential signature identification (Steps 1-4)

Data normalization and filtering (Steps 1-2): To mitigate challenges induced by different number of sequencing (e.g., library size), microbial count matrices are often normalized by various computational strategies prior to downstream analyses[21]. Here, xMarkerFinder takes the relative abundance as default, other normalization methods were reviewed in ref[21] (Step 1). In addition, rare signatures, those with low occurrence rates across cohorts are discarded (default: prevalence below 20% of samples) to ensure that identified *biomarkers* are reproducible and could be applied to prospective cohorts (Step 2).

Confounder analysis (Step 3): Inter-cohort heterogeneity caused by variance in confounders is inevitable in meta-analyses, strongly affecting downstream differential signature identification. Permutational multivariate analysis of variance (PERMANOVA) test, one of the most widely used nonparametric methods to fit multivariate models based on dissimilarity metric in microbial studies[22], quantifies microbial variations attributable to each metadata variable, thus assigning a delegate to evaluate confounding effects. Commonly used microbial community diversity measures include Bray-Curtis dissimilarity, Jaccard distance, Aitchison and UniFrac distance. PERMANOVA test here is performed on Bray-Curtis matrices generated using the vegan package as default. For each metadata variable, coefficient of determination (R^2) value and *p* value are calculated to explain how variation is attributed. The variable with the most predominant impact on microbial profiles is treated as major batch, and other confounders are subsequently used as covariates. Principal coordinate analysis (PCoA) plot with Bray-Curtis dissimilarity is also provided.

Differential analysis (Step 4): To identify disease or trait-associated microbial signatures across cohorts, MMUPHin is employed. Regression analyses in individual cohorts are performed using the well-validated Microbiome Multivariable Association with Linear Models (MaAsLin2) package, where multivariable associations between phenotypes, experimental groups or other metadata factors and microbial profiles are determined. These results are then aggregated with established fixed effects models to test for consistently *differential signatures* between groups with the major confounder (determined in Step 3) set as the main batch and other minor confounders (e.g., demographic indices, technical differences) as covariates. Signatures with consistently significant differences in meta-analysis (default *p* values less than 0.05) are identified as cross-cohort *differential signatures* and used for further feature selection in subsequent stages. Volcano plot of *differential signatures* is provided.

Stage 2 Model construction (Steps 5-7)

Classifier selection (Step 5): xMarkerFinder integrates multiple widely used ML algorithms, including Logistic Regression (LR, L1 and L2 regularization), Support Vector classifier (SVC) with the Radial Basis Function kernel, K-nearest Neighbors (KNN) classifier, Decision Tree (DT) classifier, RF classifier, and Gradient Boosting (GB) classifier. All classifiers are assessed by constructing classification models with default parameters based on *differential signatures*. Throughout this whole workflow, multiple model

evaluation metrics are used to evaluate model performances (Table 1). AUC is one of the most widely used metrics with an aggregate measure of performance, thus being used as the major evaluation metric in subsequent steps, including feature selection, hyperparameter tuning, and model validation. Additionally, other metrics, such as accuracy, precision, specificity, sensitivity, and F1 score are also crucial in certain scenarios providing supplemental information, and could be used together with AUC to determine the most suitable classifier according to specific study design (Table 1). Despite various ML algorithms provided in xMarkerFinder, RF has been shown to outperform, on average, other learning tools for microbiome data[23-25], and is recommended as the default classifier. Step 6 is not compulsory and other algorithms are encouraged if RF models underperform.

Feature selection (Step 6): Based on *differential signatures* from all cohorts in the training set and the selected ML classifier, xMarkerFinder presents the Triple-E feature selection procedure to identify an optimal panel of *candidate biomarkers*. Firstly, "feature effectiveness evaluation" is performed to select features with discriminative power (Step 6a). In detail, each individual *differential signature* is used to build a classification model with default parameters. To ensure that characteristic of each single cohort is used to assess feature effectiveness, GroupKfold cross-validation is utilized. Since a perfect classifier gives an AUC of 1 while a simple classifier that makes completely random guesses gives an AUC of 0.5, signatures with AUC values above 0.5 (as default) are considered predictive and preserved as *effective features*.

Due to extensive interplays within the microbial community, microbial features tend to tightly correlate with each other. However, such collinearity issues often result in false identification of relevant features and incorrect inferences, compromising the accuracy and interpretability of the model, especially when a model is trained on and predicted to datasets from different sources[26]. Therefore, xMarkerFinder designs the "collinear feature exclusion" as the second step of Tripe-E to avoid collinearity by excluding highly correlated features. To be specific, associations between *effective features* are estimated with the Spearman's rank correlation coefficient, and the *uncorrelated-effective features* are determined where the absolute values of correlation coefficients between any feature pair are less than 0.7 (as default, Step 6b).

The last step of Triple-E is "recursive feature elimination" for defining the optimal panel of features with the greatest predictive capability (Step 6c). This step (Fig. 3) is achieved by recursively removing the weakest feature per loop and calculate corresponding AUCs until only one is left, and the best panel of features with the highest AUC value are then treated as *candidate biomarkers*. During this process, input *uncorrelated-effective features* are ranked according to their permutation importances (Details in instructions of Step 10).

In addition, xMarkerFinder provides an alternative algorithm, Boruta feature selection (Step 6*). In Boruta, "shadow features" are generated by creating randomly shuffled copies of all original features. For each iteration, shadow features and original features are merged together to train a RF classifier where feature importances are calculated. The original features which are less important than any shadow feature are

regarded as unimportant and removed for next iteration. The iteration stops when all features are determined as either important or unimportant, and only important features are reserved as *candidate biomarkers*. Step 6* is optional and users are recommended to try different feature selection procedures.

Hyperparameter tuning (Step 7): Based on the selected classifier (from Step 5), *candidate biomarkers* (from Step 6) are used to construct a classification model with stratified five-fold cross-validation to exhibit their predictive abilities. To optimize this classification model, xMarkerFinder tunes hyperparameters (e.g., the number of estimated trees, the maximum depth of the trees, and the numbers of features per tree of RF classifier) via bayesian optimization method, a global optimization tool built upon bayesian inference and gaussian process. The combination set with the highest AUC is then determined as the final optimal hyperparameters, which are used to build the best-performing classification model. The general performance of the best-performing model is reported with several metrics mentioned above (Table 1) and its cross-validation AUC plot is provided (Fig. 4a).

Stage 3 Model validation (Steps 8-9)

To ensure the robustness and generalization of established *candidate biomarkers* and the best-performing model among geographically distinct cohorts, xMarkerFinder provides multi-perspective internal and external validations (Box 1).

Internal validations (Step 8): xMarkerFinder assesses *candidate biomarkers*' robustness and generalizability at different level within the discovery dataset, including 8a, intra-cohort, 8b, cohort-to-cohort, and 8c, LOCO validations. The intra-cohort validation constructs stratified five-fold cross-validation classification models using profiles of individual cohorts. The performance metrics are calculated against each intra-cohort model (Table 1), ascertaining that established *candidate biomarkers* are capable not only in merged cross-cohort discovery dataset but also in individual cohorts. For cohort-to-cohort transfer, classification model is fit with the profile of one cohort and validated on each profile of the remaining cohorts, respectively. Cohort-to-cohort transfer ensures that models trained on one cohort are applicable on any of the other cohorts. For LOCO validation, one cohort is ruled out for test, and all other cohorts are pooled together to train the classification model. Both cohort-to-cohort transfer and LOCO validation assess the generalizability of *candidate biomarkers* in distinct cohorts and could be seen as a simulation of predicting disease-positive samples in a newly established cohort. Heatmap plot of these internal validation results with multiple metrics are provided (Fig. 4b).

External validations (Step 9): xMarkerFinder provides an independent test and two specificity assessment tools as external validations (Box 1). The independent test of the classification model and identified *candidate biomarkers* is to test their performance with independent cohort(s) not involved in any discovery steps mentioned above, which ensures their potential applicability in future prospective settings. The best-performing classification model constructed in Step 7 (with the *candidate biomarkers*

and optimal hyperparameters) is used to predict the disease or trait type of the external test cohort(s), and similarly, multiple performance metrics are employed (Step 9a).

Previous researches suggested shared microbial dysbiosis patterns among different microbiome-linked diseases[27]. Therefore, the existence of clinical false positives remains a major concern of using microbiome-derived biomarkers, necessitating biomarkers' specificity for the disease of interest. Thus, in addition to the independent test, Step 9 integrates two biomarker specificity assessments. First, *candidate biomarkers* are used to construct similar classification models in other microbiome-related diseases and those with relatively worse performances in other diseases are deemed specific (Step 9b)[13]. Besides this straightforward assessment, xMarkerFinder provides an alternative approach where samples randomly selected from each case and control condition of other diseases are labelled as "control" and added into the classification model, and the variations of corresponding AUCs are calculated. The difference between the AUCs of adding external cases and the AUCs of adding external controls could be seen as a proxy of biomarkers' specificity[11] (Step 9b*), and those with similar AUCs are considered specific. Visualization tools are provided within each step.

Above external validations could be seen as a simulation of *candidate biomarkers*' application in future prospective cohorts, and those with statistically rigorous overall performances are then determined as final *biomarkers* with potential. Users are encouraged to fulfill these validation tools and re-determine the *biomarker* panel via adjusting the combination of discovery datasets or tuning parameters in Stage 2.

Stage 4 Biomarker Interpretation (Steps 10-12)

In addition to biomarker identification and model construction, xMarkerFinder provides multiple biomarker interpretation steps to better characterize microbial *biomarkers*' contributions to the classification model and their potential roles in trait stratification or disease initiation and progression.

Biomarker importance (Step 10): The contribution of each *biomarker* for the best-performing classification model could assist in determining candidate targets for potential interventional therapy. Here, xMarkerFinder adopts permutation feature importance, a model inspection technique implemented in scikit-learn. In detail, the decrease of the model AUC is calculated when a single *biomarker*'s value is randomly shuffled[28]. Therefore, a feature would be defined as important if shuffling its values increases the model error, as the model heavily relies on it. Important *biomarkers* are thus identified and recommended for further researches. Biomarker importance visualization is also provided with boxplot.

Microbial co-occurrence network (Step 11): As microorganisms constitutes a highly interactive community, deciphering complex microbial interaction patterns and reconstructing the community structure provides in-depth insights for unravelling *biomarkers*' mechanistic contributions. Here, we adopted FastSpar, a fast and parallelizable implementation of the well-established SparCC algorithm, to construct microbial co-occurrence networks[29]. Spearman correlations between log-transformed microbial

profiles are calculated to approximate the microbial co-occurrence network[30]. Similarly, hub microbial signatures which are significantly correlated with a large number of other signatures thus more important in the microbial community hold great potential for further exploration. As highly correlated features are excluded in Step 6b, *differential signatures* are recommended as input for constructing microbial co-occurrence networks where the roles of *biomarkers* in the microbial community could be estimated via multiple network topology metrics. Visualization of microbial co-occurrence network is conducted via Gephi.

Multi-omics correlation (Step 12): Integration of multiple omics layers might lead to novel perspectives in understanding the roles of microbial signatures and biomarkers. Since multi-omics and multidimensional microbial data, such as microbial species composition and functional gene composition, is allowed in xMarkerFinder, resultant multiple layers of *differential signatures* and *biomarkers* would be generated. To capture the correlation between these paired datasets and further explore the underlying mechanisms, Hierarchical All-against-All association testing (HAllA) is employed. HAllA integrates hierarchical hypothesis testing with false discovery rate correction to reveal significant block-wise relationships among various data layers[31], in order to unravel hidden multi-omics mysteries. Visualization of multi-omics correlation is also provided in HAllA.

Altogether, within xMarkerFinder, a collection of parameters is used. We provide default parameter values which have been tested in our example datasets. However, we encourage users to tune these parameters for finer results.

Applications of the method

We have previously demonstrated the basic application of xMarkerFinder in two CRC microbial biomarker studies where consistent cross-cohort microbial *biomarkers* were identified for adenoma and CRC[13, 14]. As shown in these researches, xMarkerFinder is well adapted to amplicon sequencing and WMS datasets, suitable for exploring taxonomic and functional profiles derived from both sequencing technologies. Apart from genome layer, xMarkerFinder is applicable to multi-omics approaches, including metatranscriptomics, metaproteomics, and metabolomics, providing a comprehensive picture of novel microbial biomarkers[32].

In addition to diagnostic purposes as shown in our and others' previous studies[11, 13, 14, 33], profiling the microbial community with human intestines and determining key microbiome signatures in different disease or phenotype groups offer great potential for prognosis and treatment stratification, metastasis surveillance and adverse reactions anticipation[1, 3, 34, 35]. xMarkerFinder could be applied to all aforementioned scenarios where the classification process is involved.

Besides gut, the most frequently studied microbial habitat in human beings, other body sites, such as skin, oral cavity, and airways, also harbor trillions of microorganisms, together constituting the complex

human microbiome[35]. The microbial communities colonizing these various ecological zones are associated with site-specific and whole-body systematic diseases and are increasingly used as biomarkers[36-38]. Although xMarkerFinder is not primarily designed for other habitats, its application for unravelling the microbial potential in distinct communities is highly expected.

Comparison with other methods

xMarkerFinder is the first computational framework aggregating meta-analyses and ML models for the establishment and validation of universally robust microbial biomarkers across multiple cohorts. In this section, we briefly describe and compare alternative methods that are commonly used to achieve similar results as xMarkerFinder.

For meta-analysis, xMarkerFinder advocates the use of the well-established MMUPHin package. MMUPHin performs multivariate linear regression within individual cohorts using MaAsLin2, while accounting for both technical and biological variables, the resultant cohort-specific differential abundance effects estimations are aggregated into meta-analysis effect sizes with fixed effects modelling. The application of this method in joint datasets supported many of the microbial taxonomic and functional associations previously ascribed to inflammatory bowel disease and CRC, while uncovering novel microbial determinants[12]. Other widely used tools for microbial meta-analysis include Bayesian Dirichlet-multinomial regression meta-analysis (BDMMA) and blocked Wilcoxon rank-sum test[11, 13, 39]. However, BDMMA is computationally expensive, and blocked Wilcoxon rank-sum test only blocks for the major batch but lacks the consideration for other biological covariates and the integration of different blocks, therefore should be used with caution.

xMarkerFinder implements ML algorithms nested in scikit-learn, which provides state-of-the-art implementations of various ML algorithms with an easy-to-use interface, thus being one of the most suitable tools for data mining[40]. Besides scikit-learn, there exists other general-purpose ML toolboxes, such as mlr, caret, tidymodels, and SIAMCAT, a R toolbox specially designed for ML-based comparative metagenomics, as well as deep learning frameworks, such as PyTorch, Keras, and TensorFlow[41-45]. These toolboxes provide similar functions as scikit-learn, and scikit-learn is chosen here for its simplicity, efficiency, and easy implementation. To optimize the classification model, xMarkerFinder utilizes bayesian optimization, a sequential model-based optimization algorithm, where the results of previous iterations are used to improve the next experiment[46]. Other popular hyperparameter tuning techniques include manual search, random search, and grid search, all of which require time-consuming procedures, thus not suitable here.

Finally, to better interpret the *biomarker* panel and the classification model, we used several interpretation tools. Permutation feature importance, for instance, measures the increase in the prediction error of the model after permutating feature's values, breaking the relationship between the feature and the true

outcome, and is applicable to various ML algorithms. Alternative methods include model-specific methods, such as mean decrease impurity (MDI) importance for RF classifiers and standardized regression coefficients for regression models, and model-agnostic methods, such as Sobol's indices, functional ANOVA and SHAP importances[24, 47]. Model-specific methods without a broad range of application scenario are dismissed since various ML algorithms might be selected in xMarkerFinder. Meanwhile, permutation-based feature importance provides a global insight and straightforward interpretation result and could avoid the issue caused by MDI that high importance might be assigned to features with poor predictive capability on unseen data when the model is overfitting. Further, to better characterize the highly-correlated microbial community and construct the co-occurrence network, we adopted a parallelizable implementation of the well-established SparCC algorithm, FastSpar, preserving equivalent robustness and accuracy as SparCC while substantially reducing required computational time and resource[29, 30]. To optimize our workflow, more in-depth tools for better exploring *biomarkers* and the integral microbial community will be provided in future versions of xMarkerFinder.

Expertise needed to implement the protocol

Preliminary understanding of R and python would facilitate users to comprehend all stages of this workflow, but intermediate experience of both programming languages is required if users want to make modifications or refinements of xMarkerFinder according to their specific study design. For users with no prior bioinformatics training, we provide detailed scripts and step-to-step instructions of input/output files and parameters, allowing easier implementation.

Limitations

Biomarker identification relies heavily on prior determined differential signatures. However, the critical challenge lying in the inherent nature of meta-analyses is that unwanted sources of confounders between cohorts might be large enough to veil true associations with fallacious results even after our endeavor of correction. Current approaches for accounting for and correcting batch effects for microbiome data could not perfectly solve the challenge brought by considerable individual variations and the sparse and compositional characteristics of microbiome data[48]. Further, highly imbalanced class and cohort sizes that are often the case might cause biased results with untrustworthy signatures and spurious biomarkers. Therefore, xMarkerFinder could not guarantee the performances of identified *candidate biomarkers* and users are encouraged to manually adjust input datasets (e.g., undersampling or oversampling to get balanced datasets) and parameters and re-identify *biomarkers* based on priori knowledge. The adjustment process is not automatic thus providing more options. Additionally, as an integrated workflow, xMarkerFinder does not provide an interactive interface, which might be optimized in future versions. Limitations notwithstanding, xMarkerFinder provides a comprehensive guide on cross-

cohort biomarker identification, validation, and interpretation in cross-cohort microbial datasets, facilitating future researches.

Reagents

Equipment

Hardware

The protocol can be executed on standard computational hardware, and greater computational resources would allow for faster execution.

Software

● R v.3.6.1 or newer (https://www.r-project.org)

● Python3 v3.7 or newer (https://www.python.org)

● HAllA (https://huttenhower.sph.harvard.edu/halla)

● FastSpar (https://github.com/scwatts/FastSpar)

● Gephi (https://gephi.org)

R packages

● BiocManager (https://cran.r-project.org/web/packages/BiocManager) to ensure the installation of the following packages and their dependencies.

● MMUPHin (https://huttenhower.sph.harvard.edu/mmuphin)

● dplyr (https://cran.r-project.org/web/packages/dplyr)

● vegan (https://cran.r-project.org/web/packages/vegan/)

● XICOR (https://cran.r-project.org/web/packages/XICOR/)

● eva (https://cran.r-project.org/web/packages/eva/)

● labdsv (https://cran.r-project.org/web/packages/labdsv/)

● Boruta (https://cran.r-project.org/web/packages/Boruta/, optional)

python packages

● pandas (https://pandas.pydata.org)

● NumPy (https://numpy.org/)

● scikit-learn (https://scikit-learn.org)

● bioinfokit (https://github.com/reneshbedre/bioinfokit)

● Bayesian Optimization (https://github.com/fmfn/BayesianOptimization)

● Matplotlib (https://matplotlib.org/)

● seaborn (https://seaborn.pydata.org/)

Docker image

Above software list provides the minimal requirements for the complete execution of xMarkerFinder locally. Alternatively, we provide a ready-to-use Docker image (https://hub.docker.com/r/tjcadd2022/xmarkerfinder), as well as an interactive JupyterHub server (https://mybinder.org/v2/gh/tjcadd2020/xMarkerFinder/HEAD), enabling users to skip the software installation and environment setup.

Input data

Processed microbial count matrices and corresponding metadata are required. For cross-cohort analysis, we require merged datasets from at least three cohorts in the discovery set to accomplish the full protocol with internal validations. Standard data cleansing procedure includes data normalization and the removal of rare signatures. Input data are split to training and test set using either method as follows: a) one or two cohorts are singled out as test dataset while other cohorts are pooled together as the training set; b) randomly select 10-30% samples of each cohort to build a mixed test set and the remaining samples are set as the training dataset. For specificity assessments, microbial profiles and corresponding metadata of other microbiome-linked diseases are also needed. All input files should be in tab-delimited text formats (.txt) where each row describes a sample and each column represents a metadata index or microbial signature. Example input files are as follows and can be downloaded from our GitHub repository (https://github.com/tjcadd2020/xMarkerFinder):

Input files for xMarkerFinder

● train_metadata.txt: clinical metadata of the training dataset, including experimental group, cohort, and other metadata indices (e.g., age, gender, BMI) as columns and samples as rows.

● train_abundance.txt: microbial abundance profile of the training dataset where each row describes a sample and each column represents a microbial signature.

● test_metadata.txt: clinical metadata of the external test dataset.

● test_abundance.txt: microbial abundance profile of the external test dataset.

● other_metadata.txt: clinical metadata of case and control samples with other diseases, used to assess the disease specificity of identified *candidate biomarkers*.

● other_abundance.txt: the microbial abundance profile of case and control samples with other diseases.

Equipment setup

Most of the commands provided in this protocol run in a Linux or Mac shell prompt (commands are prefixed with a “$” character), and all commands should be executed in the same directory except where otherwise set. The protocol also contains R scripts prefixed with a “>” character.

Installation of R and R packages:

Installation of R on different platforms can be conducted following the instructions on the official website (https://www.r-project.org/). All R packages used in this protocol can be installed following given commands.

> install.packages(Package_Name)

Or:

> if (!requireNamespace(“BiocManager”, quietly = TRUE))

> install.packages(“BiocManager”)

> BiocManager::install(Package_Name)

Installation of python and python packages

Python can be downloaded and installed from its official website (https://www.python.org/), and all python packages could be installed using pip.

$ pip install Package_Name

Installation of HAllA

HAllA can be installed according to its website (https://huttenhower.sph.harvard.edu/halla/) with the following command.

$ pip install halla

Installation of FastSpar

FastSpar can be installed following its GitHub repository (https://github.com/scwatts/fastspar).

Installation through conda:

$ conda install -c bioconda -c conda-forge fastspar

Or compiling from source code:

$ git clone https://github.com/scwatts/fastspar.git

$ cd fastspar

$./autogen.sh

$./configure –prefix=/usr/

$ make

$ make install

Installation of Gephi

Gephi could be freely downloaded and installed from its website (https://gephi.org/).

Docker image setup

Instead of above environment setup process, we provide a Docker image for easier use to replace all Equipment setup steps excluding Gephi. Firstly, users should download and install Docker (https://docs.docker.com/engine/install/) and then setup xMarkerFinder computational environment. All scripts in the Procedure part below should be executed within the Docker container created from the xMarkerFinder Docker image (https://hub.docker.com/r/tjcadd2022/xmarkerfinder/).

(i) Make sure that Docker has been properly installed.

$ docker run hello-world

(ii) Pull our xMarkerFinder image from Docker Hub.

$ docker pull tjcadd2022/xmarkerfinder:1.0.14

(iii) Run xMarkerFinder Docker image

$ docker run -it -v $(pwd):/work tjcadd2022/xmarkerfinder:1.0.14 /bin/bash

-it Run containers in an interactive mode, allowing users to execute commands and access files within the Docker container.

-v Mounts a volume between present working directory in your local machine to the /work directory in the Docker container.

Procedure

For each step of the four stages of xMarkerFinder, detailed instructions of execution commands and input/output files, as well as default parameters are provided here, as well as in our GitHub repository (https://github.com/tjcadd2020/xMarkerFinder). Further characterization and elaboration of output files are in the Anticipated results part.

Stage 1: Differential signature identification

For Stage 1, input files include merged metadata and microbial count matrices. First, data cleansing is conducted prior to all analyses aiming to convert observed counts to compositional relative abundances, and to exclude rare signatures in the discovery set. Differential analyses are then performed, in each cohort, respectively, and across cohorts. The resultant differential significance file is served for visualization and the identified *differential signatures* are used later in Stage 2 for further feature selection and model construction.

1 Data normalization. Convert microbial counts to relative abundance profiles of all datasets involved, including discovery and test set, as well as datasets of other diseases.

$ Rscript 1_Normalization.R -W /workplace/ -p abundance.txt -o TEST

Users should specify these parameters or enter the default values, subsequent repetitions of which are not listed.

-W the Workplace of this whole protocol

-p the input microbial count profile

-o prefix of output files

Input files:

abundance.txt: merged microbial count profile of all datasets.

Output files:

relative_abundance.txt: normalized relative abundance profile of input dataset. Relative abundance profiles are used as input files for all subsequent analyses, except for Step 11, which requires raw count matrices.

2 Data filtering. Filter microbial signatures with low occurrence rates across cohorts.

$ Rscript 2_Filtering.R -W /workplace/ -m train_metadata.txt -p relative_abundance.txt -b Cohort -t 2 -o TEST

-m the input metadata file

-p the input microbial relative abundance file (output file of Step 1)

-b the column name of batch(cohort) in metadata (in example file: Cohort)

-t the minimum number of cohorts where features have to occur (default: 2)

-O prefix of output files

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

relative_abundance.txt: normalized relative abundance profile of the training dataset.

Output files:

filtered_abundance.txt: filtered relative abundance profile of the training dataset, used as the input file for following steps.

3 Confounder analysis. PERMANOVA test based on Bray-Curtis dissimilarity is performed to assess the correlation between metadata and microbial profiles and returns the coefficient of determination (R^2) value and *p* value of each metadata index. Whichever index that contributes the most is considered as the major confounder used later in Step 4. PCoA plot with Bray-Curtis dissimilarity is provided.

$ Rscript 3_Confounder_analysis.R -W /workplace/ -m train_metadata.txt -p filtered_abundance.txt -g Group -o TEST

-m input metadata file

-p input filtered microbial relative abundance file

-g the column name of experimental interest(group) in metadata (in example file: Group)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

filtered_abundance.txt: filtered relative abundance profile after preprocessing.

Output files:

metadata_microbiota.txt: the confounding effects caused by clinical information, used to determine the major batch and covariates.

pcoa_plot.pdf: the PCoA plot with Bray-Curtis dissimilarity between groups.

4 Differential analysis. Based on the major confounder and covariates found in Step 3, cross-cohort differential signature analysis is conducted. Volcano plot of identified *differential signatures* is also provided.

$ Rscript 4_Differential_analysis.R -W /workplace/ -m train_metadata.txt -p filtered_abundance.txt -g Group -b Cohort -c covariates.txt -t 0.05 -o TEST

-g the column name of experimental interest(group) in metadata (in example file: Group)

-b the column name of major confounder in metadata (in example file: Cohort)

-c input covariates file (tab-delimited format containing all covariates)

-t the threshold of *p* value for plotting (default: 0.05)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

filtered_abundance.txt: filtered relative abundance profile after preprocessing.

covariates.txt: covariates identified in Step 3 (newly generated tab-delimited file where each row is a covariate, example file is provided).

Output files:

differential_significance_single_cohort.txt: the differential significance result in individual cohorts.

differential_significance.txt: meta-analytic testing results aggregating differential testing results in individual cohorts, used for visualization.

differential_signature.txt: significantly *differential signatures* between groups derived from input filtered microbial profiles, used as input files for feature selection.

differential_plot.pdf: the volcano plot of input differential significance file.

Stage 2: Model construction

5 Classifier selection. This step provides optional classifier selection for subsequent steps where the performances of each ML algorithm are generally assessed using all *differential signatures*. The output file contains the cross-validation AUC, specificity, sensitivity, accuracy, precision and F1 score of all

classification models built with these various algorithms. Users should specify a selected classifier in all following steps.

$ python 5_Classifier_selection.py -W /workplace/ -m train_metadata.txt -p differential_signature.txt -g Group -e exposure -s 0 -o TEST

-p input differential signature file (output file of Step 4)

-g the column name of experimental interest(group) in metadata (in example file: Group)

-e the experiment group(exposure) of interest (in example data: CRC)

-s random seed (default:0)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

differential_signature.txt: significantly *differential signatures* between groups.

Output files:

classifier_selection.txt: the overall cross-validation performance of all classifiers using *differential signatures*, used to determine the most suitable classifier.

6 Feature selection. xMarkerFinder introduces Triple-E, a curated three-step feature selection procedure.

6a) Feature effectiveness evaluation.

The first step of Triple-E evaluates the predictive capability of each feature via constructing individual classification models, respectively. Users should specify an algorithm here and in every future step as the overall classifier for the whole protocol from the following options: *LRl1*, *LRl2*, *SVC*, *KNN*, *DT*, *RF*, and *GB*. Features with cross-validation AUC above the threshold (default: 0.5) are defined as *effective features* and are returned in the output file.

$ python 6a_Feature_effectiveness_evaluation.py -W /workplace/ -m train_metadata.txt -p differential_signature.txt -g Group -e exposure -b Cohort -c classifier -s 0 -t 0.5 -o TEST

-p input differential signature file (output file of Step 4)

-b the column name of batch(cohort) in metadata (default: Cohort)

-c selected classifier

-t AUC threshold for defining if a feature is capable of prediction (default:0.5)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

differential_signature.txt: significantly *differential signatures* between groups.

Output files:

feature_auc.txt: cross-validation AUC values of individual features.

effective_feature.txt: features derived from *differential signatures* that are capable of predicting disease states, used as input file of the following step.

6b) Collinear feature exclusion.

The second step of Triple-E aims to exclude collinear issue caused by highly correlated features based on the result of Step 6a and returns the *uncorrelated-effective features*.

$ python 6b_Collinear_feature_exclusion.py -W /workplace/ -p effective_feature.txt -t 0.7 -o TEST

-p input effective feature file (output file of Step 6a)

-t correlation threshold for collinear feature exclusion (default:0.7)

Input files:

effective_feature.txt: features with classification capability.

Output files:

feature_correlation.txt: spearman correlation coefficients of each feature pair.

uncorrelated_effective_feature.txt: features derived from input *effective features* excluding highly collinear ones, used as input file of the following step.

6c) Recursive feature elimination.

The last step of Triple-E recursively eliminates the weakest feature per loop to sort out the minimal panel of *candidate biomarkers*.

$ python 6c_Recursive_feature_elimination.py -W /workplace/ -m train_metadata.txt -p uncorrelated_effective_feature.txt -g Group -e exposure -c classifier -s 0 -o TEST

-p input uncorrelated-effective feature file (output file of Step 6b)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

uncorrelated_effective_feature.txt: independent features derived from *effective features*.

Output files:

candidate_biomarker.txt: identified optimal panel of *candidate biomarkers*, used as model input for all subsequent steps.

6* Boruta feature selection.

Besides Triple-E feature selection procedure, we provide an alternative method, feature selection with the Boruta algorithm.

$ Rscript alt_6_Boruta_feature_selection.R -W /workplace/ -m train_metadata.txt -p differential_signature.txt -g Group -s 0 -o TEST

-p input differential signature profile (output file of Step 4) or uncorrelated-effective feature file (output file of Step 6b)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

differential_signature.txt: *differential signatures* used for feature selection (could also be *uncorrelated-effective features* from Step 6b).

Output files:

boruta_feature_imp.txt: confirmed feature importances via Boruta algorithm.

boruta_selected_feature.txt: selected feature profile, which could also be used as input *candidate biomarkers* for all subsequent steps.

7 Hyperparameter tuning.

Based on the selected classifier and *candidate biomarkers*, the hyperparameters of the classification model are adjusted via bayesian optimization method based on cross-validation AUC. The output files contain the tuned hyperparameters and the multiple performance metric values of the constructed best-performing model, as well as its cross-validation AUC plot.

$ python 7_Hyperparameter_tuning.py -W /workplace/ -m train_metadata.txt -p candidate_biomarker.txt -g Group -e exposure -c classifier -s 0 -o TEST

-p input candidate biomarker profile (output file of Step 6)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

candidate_biomarker.txt (or boruta_selected_feature.txt for all subsequent steps): the optimal panel of *candidate biomarkers*.

Output files:

best_param.txt: the best hyperparameter combination of the classification model.

optimal_cross_validation.txt: the overall cross-validation performance of the best-performing model.

cross_validation_auc.pdf: the visualization of the cross-validation AUC of the best-performing model.

Stage 3: Model validation

8 Internal validations.

As stated above, this step provides extensive internal validations to assess the robustness and reproducibility of identified *candidate biomarkers* in different cohorts via intra-cohort, cohort-to-cohort, and LOCO validations. Output files contain multiple performance metrics used to assess these *candidate biomarkers* internally (Table 1). Corresponding grid plots are provided.

$ python 8_Validation.py -W /workplace/ -m train_metadata.txt -p candidate_biomarker.txt -g Group -e exposure -b Cohort -c classifier -s 0 -o TEST

-p input optimal candidate biomarker file (output file of Step 6)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

candidate_biomarker.txt: the optimal panel of *candidate biomarkers*.

Output files:

validation_metric.txt: the performance of *candidate biomarkers* in internal validations via each model performance metric (Table 1).

validation_metric.pdf: the visualization of input validation_metric.txt.

9 External validations.

9a) Independent test.

As the best-performing *candidate biomarkers* and classification model are established, the test dataset is used to externally validate their robustness and generalizability. The input external metadata and microbial relative profiles need to be in the same format as initial input files for the training dataset. This step returns the overall performance of the model and its AUC plot.

$ python 9a_Test.py -W /workplace/ -m train_metadata.txt -p candidate_biomarker.txt -a test_metadata.txt -x test_relative_abundance.txt -g Group -e exposure -c classifier -r best_param.txt -s 0 -o TEST

-a input external metadata file for the test dataset

-x input external microbial relative abundance file as the test dataset

-r input optimal hyperparameter file (output file of Step 7)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

candidate_biomarker.txt: the optimal panel of *candidate biomarkers*.

test_metadata.txt: the clinical metadata of the external test dataset.

test_relative_abundance.txt: the relative abundance matrix of the external test dataset.

best_param.txt: the hyperparameter combination of the best-performing classification model.

Output files:

test_result.txt: the overall performance of model in external test dataset.

test_auc.pdf: the visualization of the AUC value in test_result.txt.

9b) Specificity assessment.

To further assess biomarkers' disease specificity, they are used to construct classification models to discriminate between other microbiome-related diseases and corresponding controls. Cross-validation AUC values of these classification models and corresponding visualization are returned.

$ python 9b_Specificity.py -W /workplace/ -p candidate_biomarker.txt -a other_metadata.txt -x other_relative_abundance.txt -g Group -e CTR -b Cohort -c classifier -r best_param.txt -s 0 -o TEST

-a input metadata file of samples from other diseases

-x input microbial relative abundance file of samples from other diseases

-e the control group name (in example file: CTR)

-b the column name of cohort (in example file: Cohort)

Input files:

candidate_biomarker.txt: the optimal panel of *candidate biomarkers*.

other_metadata.txt: the clinical metadata of samples for other diseases.

other_relative_abundance.txt: the microbial relative abundance matrix of other diseases.

Output files:

specificity_result.txt: AUC values of models constructed with *candidate biomarkers* in other microbiome-related diseases.

specificity_auc.pdf: the visualization of the specificity_result.txt.

9b*) Alternative specificity assessment.

Random samples of case and control class of other diseases are added into the classification model, respectively, both labelled as “control”, the variations of corresponding AUCs of which are calculated and used for visualization.

$ python alt_9b_Specificity.py -W /workplace/ -m train_metadata.txt -p candidate_biomarker.txt -q test_metadata.txt -l test_relative_abundance.txt -a other_metadata.txt -x other_relative_abundance.txt -g Group -e CTR -b Cohort -c classifier -r best_param.txt -n 5 -s 0 -o TEST

-q input external metadata file for the test dataset

-l input external microbial relative abundance file as the test dataset

-a input metadata file of samples from other diseases

-x input microbial relative abundance file of samples from other diseases

-e the control group name (in example file: CTR)

-b the column name of cohort (in example file: Cohort)

-n the number of samples to add into the model each time

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

candidate_biomarker.txt: the optimal panel of *candidate biomarkers*.

test_metadata.txt: the clinical metadata of the external test dataset.

test_relative_abundance.txt: the relative abundance matrix of the external test dataset.

other_metadata.txt: the clinical metadata of samples for other diseases.

other_relative_abundance.txt: the relative abundance matrix of other diseases.

Output files:

specificity_add_result.txt: AUC values of the best-performing model applied in the external test dataset with additional case or control samples from other diseases.

specificity_add_auc.pdf: the visualization of the specificity_add_result.txt.

Stage 4: Biomarker Interpretation

10 Biomarker importance. Permutation feature importance is employed here to evaluate biomarkers' contributions in the best-performing classification model.

$ python 10_Biomarker_importance.py -W /workplace/ -m train_metadata.txt -p candidate_biomarker.txt -g Group -e exposure -c classifier -r best_param.txt -s 0 -o TEST

-p input candidate biomarkers (output file of Step 6)

-r input optimal hyperparameter file (output file of Step 7)

Input files:

train_metadata.txt: the clinical metadata of the training dataset.

candidate_biomarker.txt: the optimal panel of selected *candidate biomarkers*.

best_param.txt: the hyperparameter combination of the best-performing classification model.

Output files:

biomarker_importance.txt: permutation feature importance of *biomarkers* via ten permutations.

biomarker_importance.pdf: the visualization of biomarker importance file.

11 Microbial co-occurrence network.

11a) Convert.

As the input file for microbial co-occurrence network construction needs to be microbial count profile in .tsv format where each row describes a microbial signature and each column represents a sample (could be converted profiles of all features, *differential signatures*, or *candidate biomarkers* according to users' need, and null values needed to be set as 0) and header needs to start with "#OTU ID", an additional file conversion script is provided for easier implementation.

$ python 11a_Convert.py -W /workplace/ -p abundance.txt -s selected_feature.txt -o TEST

-p input feature raw count file before normalization.

-s selected features for constructing microbial co-occurrence network (could be differential signatures or candidate biomarkers, output file of Step 4 or 6).

Input files:

abundance.txt: microbial raw count profile before normalization.

selected_feature.txt: selected features for constructing microbial co-occurrence network (output file of Step 4 or 6)

Output files:

convert.tsv: the converted file appropriate for constructing microbial co-occurrence network.

11b) Microbial co-occurrence network.

Microbial co-occurrence network is constructed using FastSpar with 50 iterations and the output files contain the correlation coefficients and *p* values between each signature pair.

$ ./11b_Microbial_network.sh -W /workplace/ –i convert.tsv -o TEST -t 4

-i input feature count file

-t threads of available computational source

Input files:

convert.tsv: microbial count profile in .tsv format where each row describes a microbial signature and each column represents a sample and the header needs to start with "#OTU ID". Example input file is provided and users are recommended to use Step 11a to convert files into appropriate formats.

-t the threads of available computational source when running.

Output files:

median_correlation.tsv: the correlation coefficients between each input signature pair.

pvalues.tsv: the statistical significance of median_correlation.tsv.

11c) Microbial co-occurrence network plot.

The visualization of microbial co-occurrence network is performed using Gephi.

(i) Preprocess of the results of Step 11b to ensure that Step 11c only draws significant correlations ($p < 0.05$) with absolute correlation coefficients above 0.5 (as default).

$ python 11c_Microbial_network_plot.py -W /workplace/ –c median_correlation.tsv -p pvalues.tsv -t 0.5 -o TEST

-c input correlation profile (output file of Step 11b)

-p input *p* value profile (output file of Step 11b)

-t input correlation coefficient threshold (default: 0.5)

Input files:

median_correlation.tsv: the correlation coefficients profile (output file of Step 11b).

pvalues.tsv: the statistical significance of median_correlation.tsv (output file of Step 11b).

Output files:

microbial_network.csv: adjusted network profile for Gephi, only significant correlations reserved.

(ii) Open Gephi and click the “File” button, choose the “Import spreadsheet” option, and then choose the adjusted network profile.

(iii) Import the prepared file as undirected network.

(iv) Choose a preferable layout type to form the basic network and press the “stop” button when the network becomes stable (Fruchterman Reingold style is recommended).

(v) For further optimization of the network, appearances of nodes and edges could be adjusted according to users’ need, as well as the labels of nodes. Fig. 9 provides an optimized network example[14].

12 Multi-omics correlation. If users have multi-omics or multidimensional microbial profiles of the same dataset, the correlation between different omics or dimensions are calculated via HAllA.

$ ./12_Multi_omics_correlation.sh -W /workplace/ -i microbial_abundance_1.txt -d microbial_abundance_2.txt -o TEST

-i input microbial abundance file 1

-d input microbial abundance file 2

Input files:

microbial_abundance_1.txt: microbial abundance profile 1.

microbial_abundance_2.txt: microbial abundance profile 2. These two input files should have the same samples (columns) but different features (rows).

Output files:

results/all_associations.txt: associations between different omics or dimensions.

results/hallagram.png: the visualization of all_associations.txt with only significant associations highlighted.

Troubleshooting

Troubleshooting advice can be found in (https://github.com/tjcadd2020/xMarkerFinder/). Most problems would be solved by simply using the Docker image we provide instead of running all scripts locally. Additionally, we provide a FAQ page (https://github.com/tjcadd2020/xMarkerFinder/) as well as an issue board (https://github.com/tjcadd2020/xMarkerFinder/issues) in our GitHub repository for users to bring up their own problems or suggestions regarding our workflow.

Time Taken

Time required for completing this protocol depends on the cohort size, selected classifier, and computational resources available. The most time-consuming parts of this protocol are Step 6c, Recursive feature elimination, Step 6* Boruta feature selection, and Step 8 Internal validations. The time estimates at https://github.com/tjcadd2020/xMarkerFinder/ are based on execution of our protocol on provided example datasets (in total: 1099 samples and 5331 features) with all available classifiers using the xMarkerFinder Docker image on a MacBook Pro (2.4-GHz quad-core eighth-generation Intel Core i5 processor, 16-GB 2133-MHz LPDDR3 memory).

Anticipated Results

xMarkerFinder identifies *differential signatures* across cohorts and determines the best panel of *biomarkers* for model construction. Corresponding results of each stage are described as follows. Example result files are provided at https://github.com/tjcadd2020/xMarkerFinder.

Stage1 Differential signature identification (Step 1-4)

● relative_abundance.txt (Step 1): normalized relative abundance profile.

● filtered_abundance.txt (Step 2): the filtered relative abundance file after preprocess.

● metadata_microbiota.txt (Step 3): the coefficient of determination (R^2) value and *p* value of each metadata index with the microbial profile. Higher R^2 with *p* value less than 0.05 indicates that more variance of microbial profile is attributed to the specific metadata variable thus representing more extensive metadata-microbiota correlation.

● pcoa_plot.pdf (Step 3): the PCoA visualization of Bray-Curtis similarity between groups.

● differential_significance_single_cohort.txt (Step 4): the feature significance between groups in single cohorts. If “cohort.pval” is less than 0.05, then this feature is significantly different between groups in this specified cohort where positive “cohort.coef” value indicates upregulation in interested experimental group and negative value represents otherwise.

● differential_significance.txt (Step 4): the feature significance between groups in cross-cohort analysis. This file integrates individual cohorts’ results and provide a meta-analysis result. If “pval” is less than 0.05, then this feature is significantly different between groups in meta-analysis where positive “coef” values indicate upregulation in interested experimental group and negative values represent otherwise.

● differential_signature.txt (Step 4): the *differential signatures* derived from input feature abundance profiles which are used for further visualization and feature selection.

● differential_plot.pdf (Step 4): the volcano plot of input differential significance file where identified *differential signatures* are highlighted. Red dots represent features significantly more abundant in interested experiment group and blue dots represent otherwise.

Stage 2: Model construction (Steps 5-7)

● classifier_selection.txt (Step 5): the cross-validation AUC, specificity, sensitivity, accuracy, precision and F1 score of each ML algorithm are provided here. We recommend RF classifier as default classifier, but encourage users to try all algorithms and select what’s most suitable for their specific study design.

● feature_auc.txt (Step 6a): the cross-validation AUCs of individual feature in respective classification models are provided here. As default, features with AUCs above 0.5 are deemed as capable for prediction and reserved as *effective features*.

● effective_feature.txt (Step 6a): *effective features* derived from all differential signatures.

● feature_correlation.txt (Step 6b): spearman correlation coefficients of each feature pair of *effective features* are provided here. As default, features with absolute correlation coefficients less than 0.7 are

regarded as *uncorrelated-effective features*.

● uncorrelated_effective_feature.txt (Step 6b): *uncorrelated-effective features* derived from all *effective features*.

● candidate_biomarker.txt (Step 6c): the optimal *candidate biomarker* panel established from *uncorrelated-effective features* which is used as model input for all subsequent steps.

● boruta_feature_imp.txt (Step 6*): confirmed feature importances via Boruta algorithm.

● boruta_selected_feature.txt (Step 6*): selected feature profile, could also be used as input *candidate biomarkers* for subsequent steps.

● best_param.txt (Step 7): the best hyperparameter combination of the classification model.

● optimal_cross_validation.txt (Step 7): the cross-validation AUC, specificity, sensitivity, accuracy, precision and F1 score of the best-performing model constructed with selected classifier, *candidate biomarkers* and tuned hyperparameter combination.

● cross_validation_auc.pdf (Step 7): ROC curve of the best-performing model with mean AUC and standard deviation of stratified five-fold cross-validation provided.

Stage 3: Model validation (Steps 8-9)

● validation_metric.txt (Step 8): the AUC, specificity, sensitivity, accuracy, precision and F1 score of the intra-cohort, cohort-to-cohort, and LOCO validations based on *candidate biomarkers*. Values on the diagonal refer to the average AUC values of intra-cohort five-fold cross-validation. Off-diagonal values refer to the AUC values obtained by training the classifier on the profile of cohort on the corresponding row and applying it to the profile of cohort on the corresponding column. The LOCO row refers to the performances obtained by training the model using all but the profile of cohort on the corresponding column and applying it to the profile of cohort on the corresponding column.

● validation_metric.pdf (Step 8): the visualization of validation_metric.txt.

● test_result.txt (Step 9a): the AUC, specificity, sensitivity, accuracy, precision and F1 score of applying the best-performing model to the external test dataset.

● test_auc.pdf (Step 9a): the visualization of the AUC of external test set.

● specificity_result.txt (Step 9b): the cross-validation AUC values of models constructed with *candidate biomarkers* in other microbiome-related diseases.

● specificity_auc.pdf (Step 9b): the visualization of the specificity_result.txt.

● specificity_add_result.txt (Step 9b*): the cross-validation AUC values of models based on external test dataset with additional control samples or additional case samples from other microbiome-related diseases.

● specificity_add_auc.pdf (Step 9b*): the visualization of the specificity_add_result.txt.

Stage 4: Biomarker Interpretation (Steps 10-12)

● biomarker_importance.txt (Step 10): permutation feature importances of *biomarkers* in the best-performing model.

● biomarker_importance.pdf (Step 10): the visualization of biomarker_importance.txt where *biomarkers* are sorted by importances.

● convert.tsv (Step 11a): the converted file appropriate for Step 11b.

● median_correlation.tsv (Step 11b): the correlation coefficients between each signature pair of the microbial profiles.

● pvalues.tsv (Step 11b): the *p* values of each correlation coefficient in median_correlation.tsv.

● microbial_network.csv (Step 11c): processed correlation coefficients file for co-occurrence network visualization, with only significant correlations reserved.

● microbial_network.pdf (Step 11c): the microbial co-occurrence network, visualization of microbial_network.csv from Gephi.

● results/all_associations.txt (Step 12): the correlation coefficients between each signature pair of different layers of microbial profiles.

● results/hallagram.png (Step 12): the visualization of all_associations.txt with only significant associations highlighted.

In conclusion, Steps 1-4 provide filtered microbial profiles and consistent *differential signatures* across cohorts. Step 5 encourages users to try different classifiers on their own data and select the most appropriate one. Step 6 identify a *candidate biomarker* panel. Step 7 construct and optimize the best-performing model. Steps 8-9 internally and externally validate the *candidate biomarkers* and the optimized classification model to establish the final *biomarkers*. Lastly, Steps 10-12 yield multiple biomarker interpretations for potentially unravelling the underlying the microbiome-driven mechanisms.

References

1. Cullin, N., Azevedo Antunes, C., Straussman, R., Stein-Thoeringer, C.K. & Elinav, E. Microbiome and cancer. *Cancer Cell* **39**, 1317-1341 (2021).

2. LaCourse, K.D., Johnston, C.D. & Bullman, S. The relationship between gastrointestinal cancers and the microbiota. *Lancet Gastroenterol. Hepato.* **6**, 498-509 (2021).

3. Poore, G.D. et al. Microbiome analyses of blood and tissues suggest cancer diagnostic approach. *Nature* **579**, 567-574 (2020).

4. Fan, Y. & Pedersen, O. Gut microbiota in human metabolic health and disease. *Nat. Rev. Microbiol.* **19**, 55-71 (2021).

5. Britton, G.J. et al. Microbiotas from humans with inflammatory bowel disease alter the balance of gut Th17 and RORγt+ regulatory T cells and exacerbate colitis in mice. *Immunity* **50**, 212-224. e214 (2019).

6. Mima, K. et al. Fusobacterium nucleatum in colorectal carcinoma tissue and patient prognosis. *Gut* **65**, 1973-1980 (2016).

7. McQuade, J.L., Daniel, C.R., Helmink, B.A. & Wargo, J.A. Modulating the microbiome to improve therapeutic response in cancer. *Lancet. Oncol.* **20**, e77-e91 (2019).

8. Tierney, B.T. et al. Systematically assessing microbiome-disease associations identifies drivers of inconsistency in metagenomic research. *PLoS. Biol.* **20**, e3001556 (2022).

9. Vujkovic-Cvijin, I. et al. Host variables confound gut microbiota studies of human disease. *Nature* **587**, 448-454 (2020).

10. Gonzalez, A. et al. Qiita: rapid, web-enabled microbiome meta-analysis. *Nat. Methods* **15**, 796-798 (2018).

11. Thomas, A.M. et al. Metagenomic analysis of colorectal cancer datasets identifies cross-cohort microbial diagnostic signatures and a link with choline degradation. *Nat. Med.* **25**, 667-678 (2019).

12. Ma, S (2019). MMUPHin: Meta-analysis Methods with Uniform Pipeline for Heterogeneity in Microbiome Studies. R package version 1.2.0.

13. Wu, Y. et al. Identification of microbial markers across populations in early detection of colorectal cancer. *Nat. Commun.* **12**, 3063 (2021).

14. Liu, N.-N. et al. Multi-kingdom microbiota analyses identify bacterial–fungal interactions and biomarkers of colorectal cancer across cohorts. *Nat. Microbiol.* **7**, 238-250 (2022).

15. Knight, R. et al. Best practices for analysing microbiomes. *Nat. Rev. Microbiol.* **16**, 410-422 (2018).

16. Coelho, L.P. et al. NG-meta-profiler: fast processing of metagenomes using NGLess, a domain-specific language. *Microbiome* **7**, 84 (2019).

17. Faul, F., Erdfelder, E., Buchner, A. & Lang, A.-G. Statistical power analyses using G*Power 3.1: Tests for correlation and regression analyses. *Behav. Res. Methods* **41**, 1149-1160 (2009).

18. Faul, F., Erdfelder, E., Lang, A.G. & Buchner, A. G*Power 3: a flexible statistical power analysis program for the social, behavioral, and biomedical sciences. *Behav. Res. Methods* **39**, 175-191 (2007).

19. Kelly, B.J. et al. Power and sample-size estimation for microbiome studies using pairwise distances and PERMANOVA. *Bioinformatics* **31**, 2461-2468 (2015).

20. Casals-Pascual, C. et al. Microbial Diversity in Clinical Microbiome Studies: Sample Size and Statistical Power Considerations. *Gastroenterology* **158**, 1524-1528 (2020).

21. Weiss, S. et al. Normalization and microbial differential abundance strategies depend upon data characteristics. *Microbiome* **5**, 27 (2017).

22. Anderson, M.J. Permutational multivariate analysis of variance (PERMANOVA). *Wiley statsref: statistics reference online*, 1-15 (2014).

23. Ren, Z. et al. Gut microbiome analysis as a tool towards targeted non-invasive biomarkers for early hepatocellular carcinoma. *Gut* **68**, 1014-1023 (2019).

24. Marcos-Zambrano, L.J. et al. Applications of Machine Learning in Human Microbiome Studies: A Review on Feature Selection, Biomarker Identification, Disease Prediction and Treatment. *Front. Microbiol.* **12** (2021).

25. Pasolli, E., Truong, D.T., Malik, F., Waldron, L. & Segata, N. Machine Learning Meta-analysis of Large Metagenomic Datasets: Tools and Biological Insights. *PLoS Comput. Biol.* **12**, e1004977 (2016).

26. Dormann, C.F. et al. Collinearity: a review of methods to deal with it and a simulation study evaluating their performance. *Ecography* **36**, 27-46 (2013).

27. Duvallet, C., Gibbons, S.M., Gurry, T., Irizarry, R.A. & Alm, E.J. Meta-analysis of gut microbiome studies identifies disease-specific and shared responses. *Nat. Commun.* **8**, 1784 (2017).

28. Fisher, A., Rudin, C. & Dominici, F. All Models are Wrong, but Many are Useful: Learning a Variable's Importance by Studying an Entire Class of Prediction Models Simultaneously. *J. Mach. Learn. Res.* **20**, 1-81 (2019).

29. Watts, S.C., Ritchie, S.C., Inouye, M. & Holt, K.E. FastSpar: rapid and scalable correlation estimation for compositional data. *Bioinformatics* **35**, 1064-1066 (2018).

30. Friedman, J. & Alm, E.J. Inferring Correlation Networks from Genomic Survey Data. *PLoS Comput. Biol.* **8**, e1002687 (2012).

31. Ghazi, A.R. et al. “High-sensitivity pattern discovery in large, paired multi-omic datasets”.

32. Zhang, X., Li, L., Butcher, J., Stintzi, A. & Figeys, D. Advancing functional and translational microbiome research using meta-omics approaches. *Microbiome* **7**, 154-154 (2019).

33. Wirbel, J. et al. Meta-analysis of fecal metagenomes reveals global microbial signatures that are specific for colorectal cancer. *Nat. Med.* **25**, 679-689 (2019).

34. Lee, K.A. et al. Cross-cohort gut microbiome associations with immune checkpoint inhibitor response in advanced melanoma. *Nat. Med.* **28**, 535–544 (2022).

35. Relman, D.A. The Human Microbiome and the Future Practice of Medicine. *JAMA* **314**, 1127-1128 (2015).

36. Gao, L. et al. Oral microbiomes: more and more importance in oral cavity and whole body. *Protein Cell* **9**, 488-500 (2018).

37. Byrd, A.L., Belkaid, Y. & Segre, J.A. The human skin microbiome. *Nat. Rev. Microbiol.* **16**, 143-155 (2018).

38. Wang, Z. et al. Inflammatory Endotype-associated Airway Microbiome in Chronic Obstructive Pulmonary Disease Clinical Stability and Exacerbations: A Multicohort Longitudinal Analysis. *Am. J. Respir. Crit. Care Med.* **203**, 1488-1502 (2021).

39. Dai, Z., Wong, S.H., Yu, J. & Wei, Y. Batch effects correction for microbiome data with Dirichlet-multinomial regression. *Bioinformatics* **35**, 807-814 (2018).

40. Pedregosa, F. et al. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.* **12**, 2825-2830 (2011).

41. Wirbel, J. et al. Microbiome meta-analysis and cross-disease comparison enabled by the SIAMCAT machine learning toolbox. *Genome Biol.* **22**, 93 (2021).

42. Bischl, B. et al. mlr: Machine Learning in R. *J. Mach. Learn. Res.* **17**, 5938-5942 (2016).

43. Lang, M. et al. mlr3: A modern object-oriented machine learning framework in R. *J. Open Source Softw.* **4**, 1903 (2019).

44. Kuhn, M. Building Predictive Models in R Using the caret Package. *J. Stat. Softw.* **28**, 1 - 26 (2008).

45. Paszke, A. et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *Adv. Neural. Inf. Process. Syst.* **32**, 8024-8035 (2019).

46. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P. & Freitas, N.d. Taking the Human Out of the Loop: A Review of Bayesian Optimization. *Proc. IEEE* **104**, 148-175 (2016).

47. Wei, P., Lu, Z. & Song, J. Variable importance analysis: a comprehensive review. *Reliab. Eng. Syst.* **142**, 399-432 (2015).

48. Wang, Y. & LeCao, K.A. Managing batch effects in microbiome data. *Brief. Bioinform.* **21**, 1954-1970 (2020).

Acknowledgements

This work was supported by the National Natural Science Foundation of China (grant number 82170542 to RZ, 82000536 to NJ), and the National Key Research and Development Program of China (grant number 2021YFF0703700/2021YFF0703702 to RZ). We thank X. Huang, K. Chen, and Y. Huang for testing xMarkerFinder and providing their constructive feedbacks.

Figures

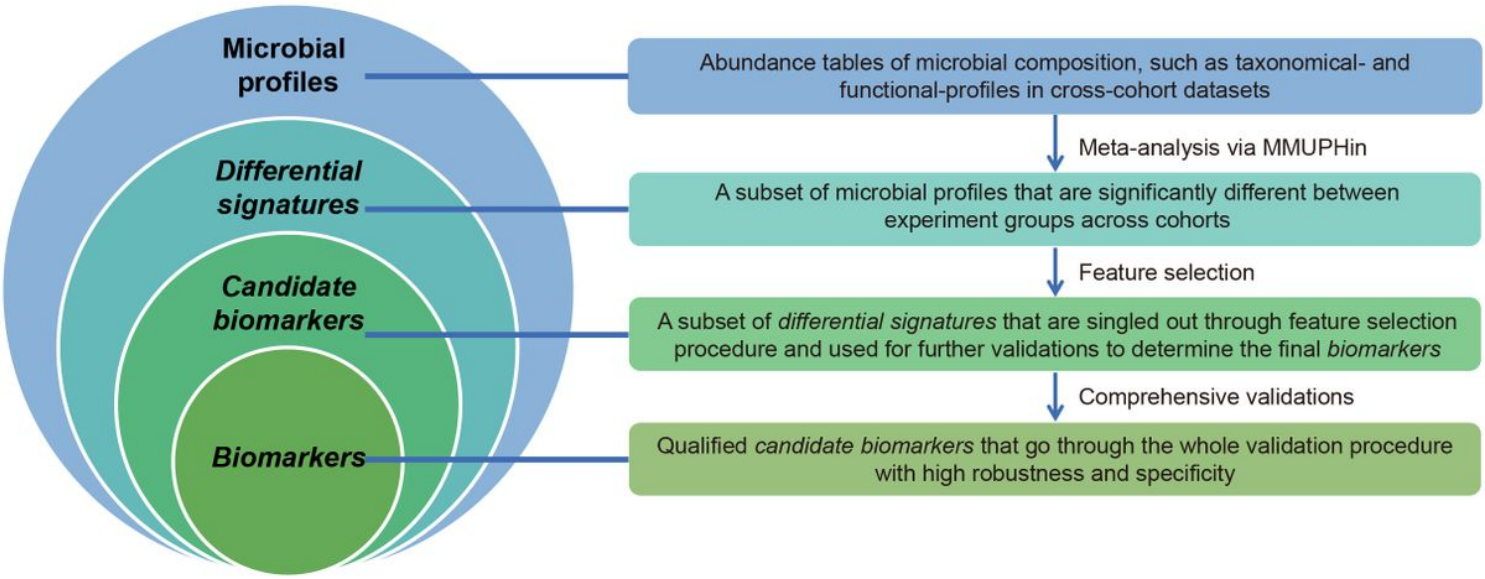

Figure 1

The procedure for selecting *biomarkers* from cross-cohort microbial profiles. From cross-cohort microbial profiles, *differential signatures* were determined by meta-analysis via MMUPHin. *Candidate biomarkers* were derived from *differential signatures* with feature selection procedure and further validated to establish the final *biomarkers*.

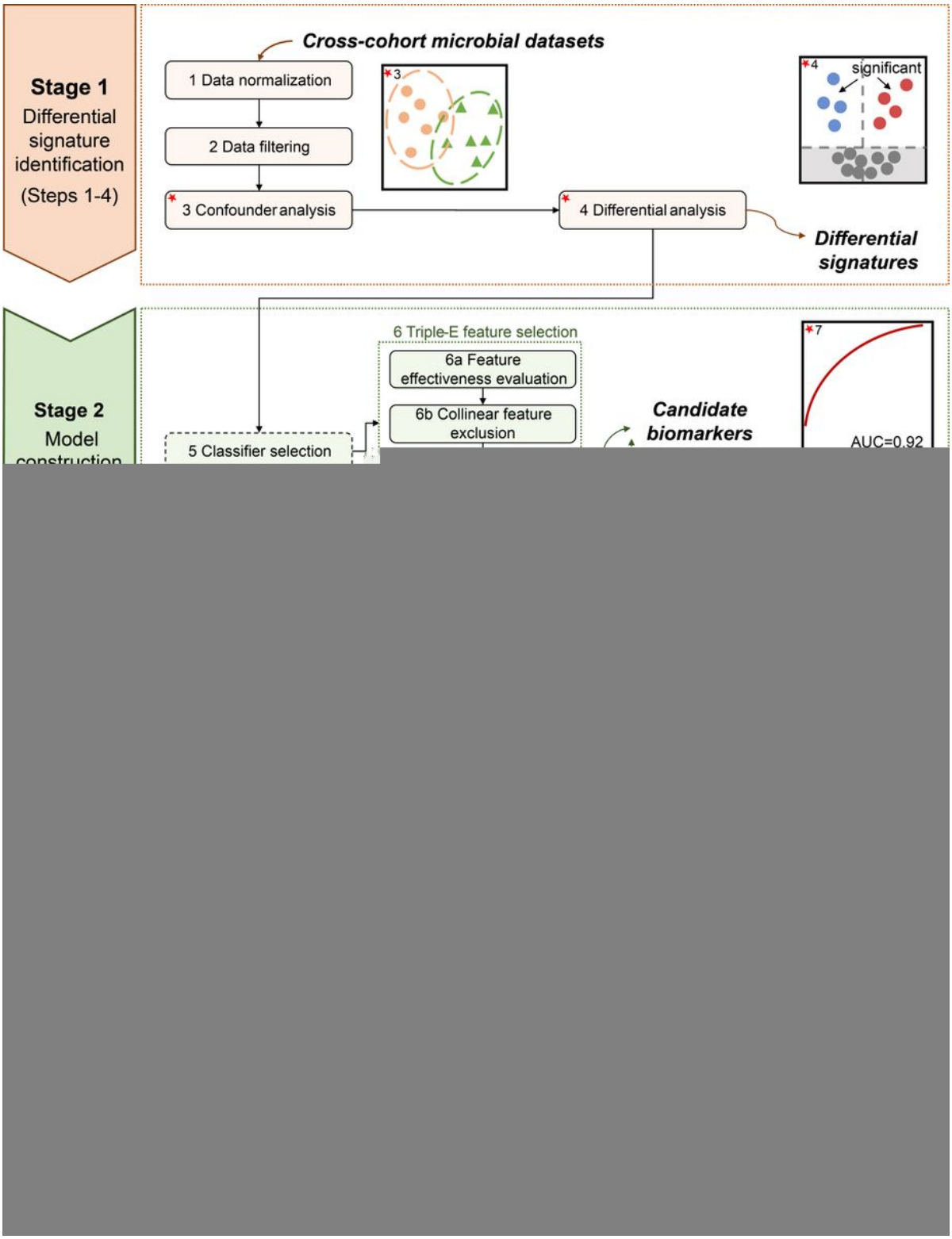

Figure 2

Overview of the xMarkerFinder workflow. There are four main stages: differential signature identification (steps 1-4), model construction (steps 5-7), model validation (Steps 8-9), and biomarker interpretation (steps 10-12). Stage 1 provides data normalization and filtering for all subsequent steps. Confounder analysis enables the identification of major confounder and covariates, which are then accounted for in differential analysis to identify concordant *differential signatures* in cross-cohort datasets. Stage 2

provides classifier selection for users to choose the most suitable ML classifier for the whole workflow. *Candidate biomarkers* are then determined using either Triple-E or Boruta feature selection procedure. The best-performing model is then constructed with *candidate biomarkers* and optimal hyperparameters. Stage 3 provides sufficient internal and external validations to establish the final *biomarkers*. Stage 4 provides additional biomarker interpretation tools including biomarker importance, microbial co-occurrence network, and multi-omics correlation. Scripts for alternative steps are named starting with *alt*.

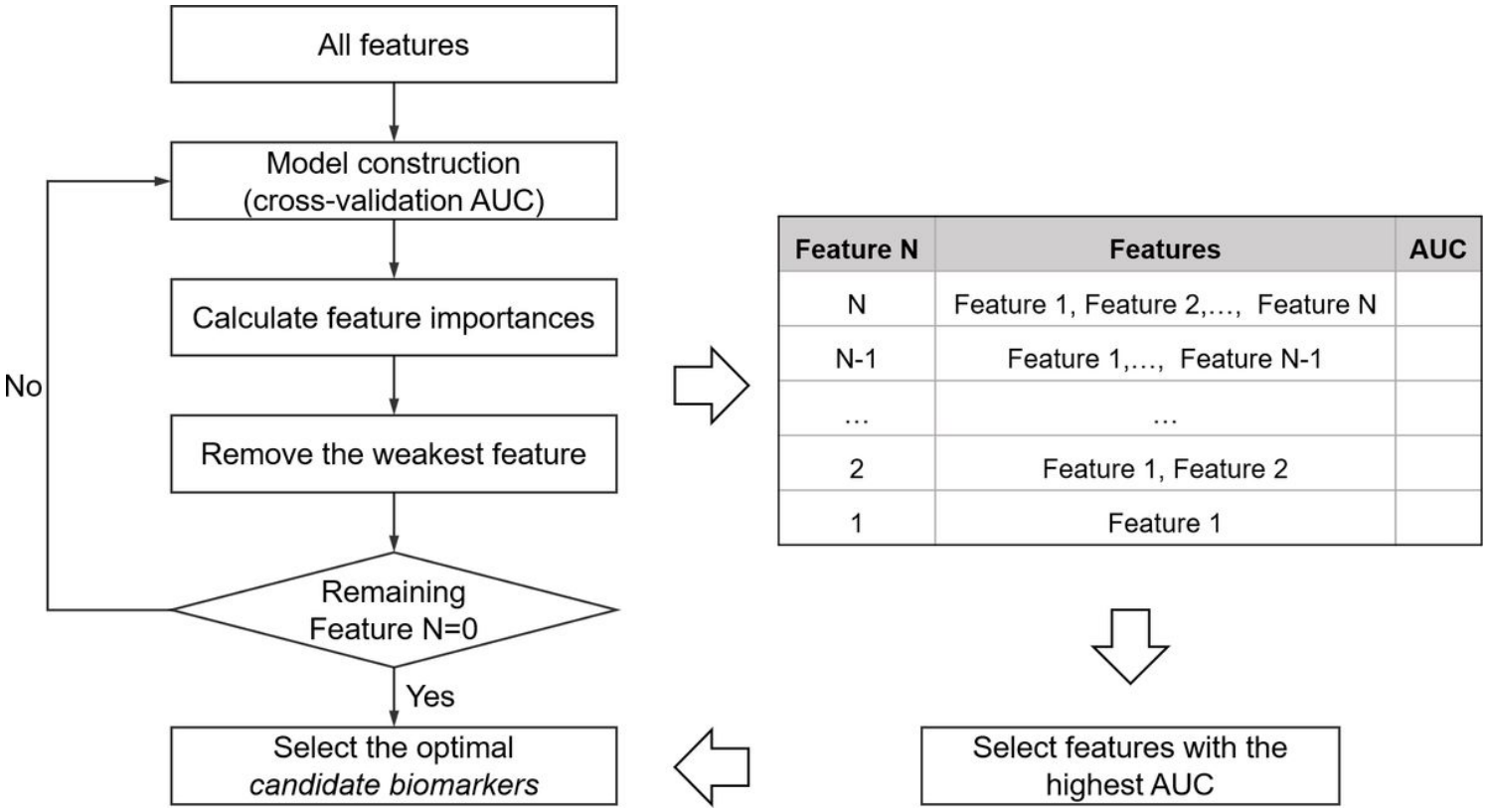

Figure 3

Recursive feature elimination. The third step of Triple-E feature selection procedure aiming at determining the minimal panel of *candidate biomarkers*, achieved by recursively removing the weakest feature for model construction per loop to obtain the best panel of features with the highest cross-validation AUC value.

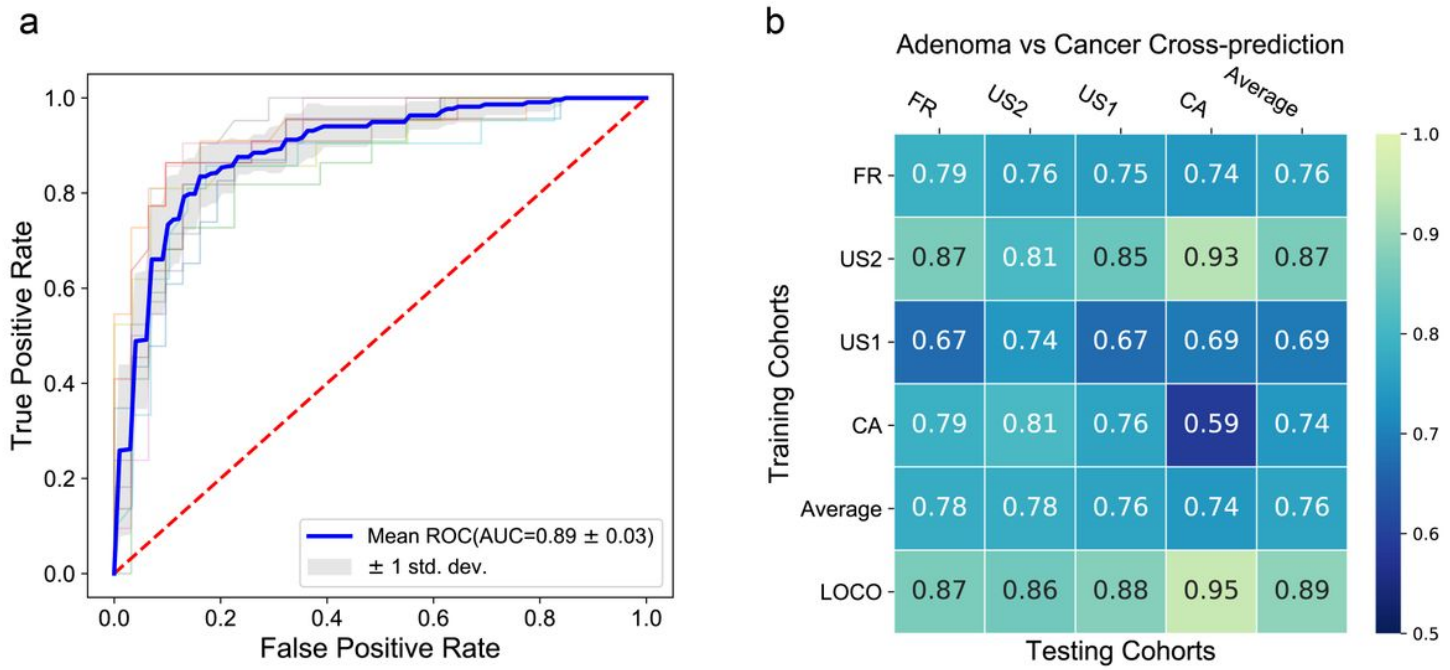

Figure 4

Example output files for model construction and validation. a, Example ROC curve[13]. Mean AUC and standard deviation of stratified five-fold cross-validation are shown. **b**, Example internal validation AUC matrix[13]. Values on the diagonal refer to the average AUC values of stratified five-fold cross-validation within profiles of each cohort. Off-diagonal values refer to the AUC values obtained by training the classifier on the profile of the corresponding row and applying it to the profile of the corresponding column. The LOCO row refers to the performances obtained by training the model using all but the profile of the corresponding column and applying it to the profile of the corresponding column.

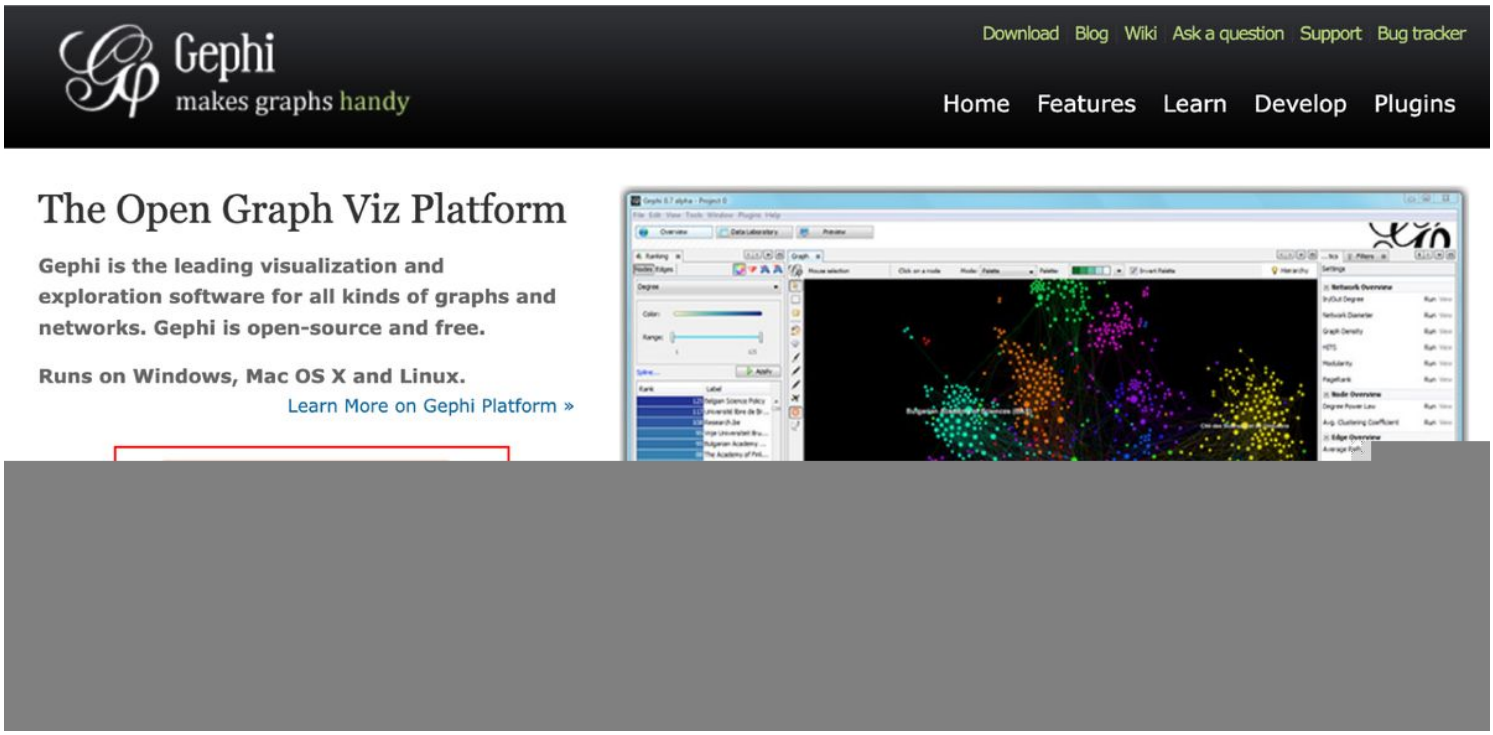

Figure 5

Gephi installation interface. Gephi provides free downloading and easy installation from its official website.

Figure 6

The initial interface of Gephi. Users should (1) click the “File” button, (2) click “Import spreadsheet…” option, (3) choose the prepared microbial co-occurrence network file for Gephi, and (4) click the “Open” button.

Figure 7

Gephi import report. Users should (5) specify the appropriate separator (“comma”), (6) click the “Next” button, (7) import prepared network file as “undirected” network and then (8) click the “OK” button.

Figure 8

The interface for network construction. Users could perform the network construction within Gephi and press the “stop” button to get a stable network for further optimization.

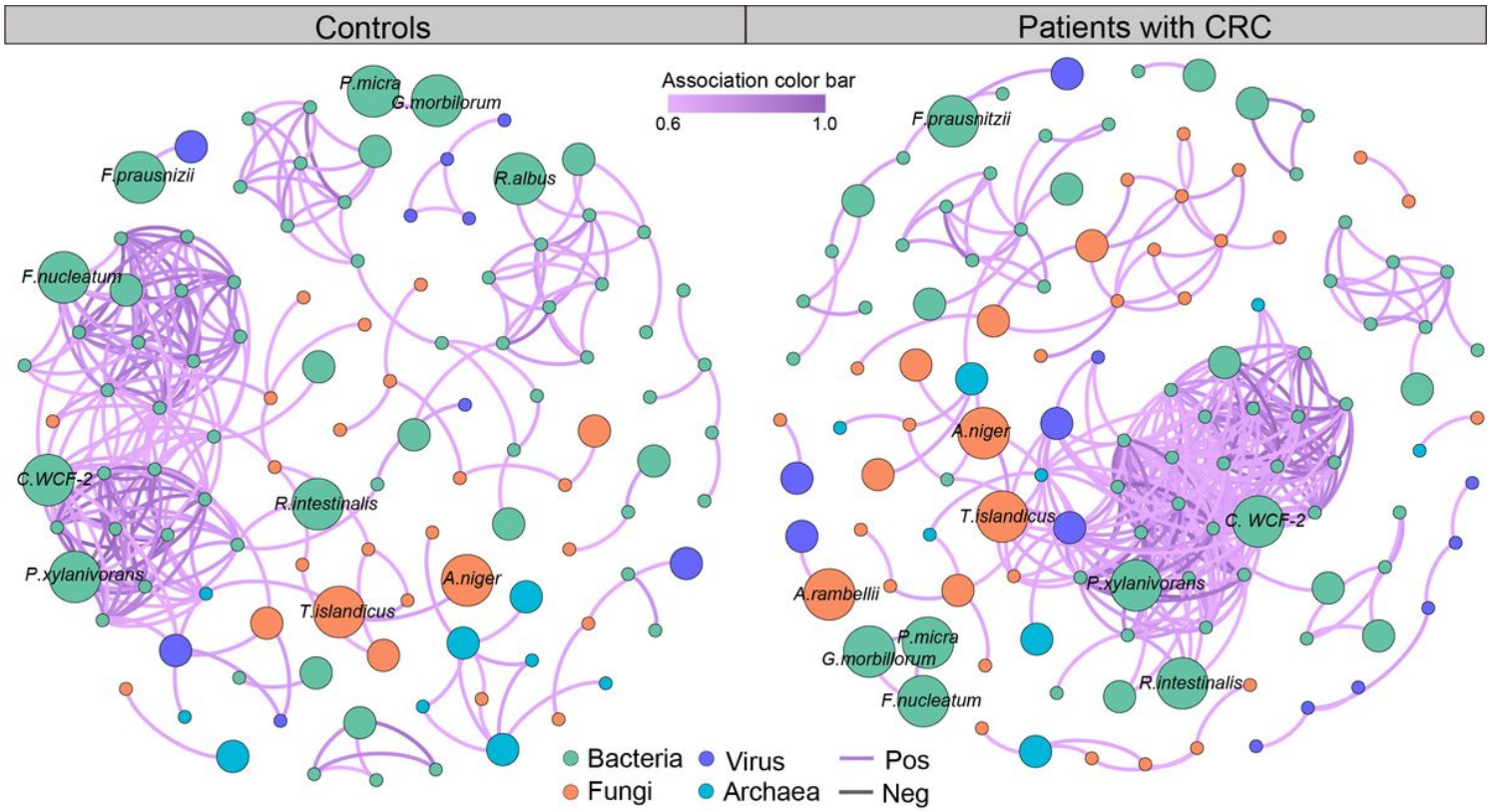

Figure 9

An example microbial network[14]. Microbial network generated using Gephi where each node represents a microbial signature and each edge represents a correlation between microbial signatures. The edges are colored according to the magnitude of the association in the networks as shown by the color bar.