Package: iimi (via r-universe)

November 11, 2026

Title Identifying Infection with Machine Intelligence

Version 1.2.2

Description A novel machine learning method for plant viruses diagnostic using genome sequencing data. This package includes three different machine learning models, random forest, XGBoost, and elastic net, to train and predict mapped genome samples. Mappability profile and unreliable regions are introduced to the algorithm, and users can build a mappability profile from scratch with functions included in the package. Plotting mapped sample coverage information is provided.

Encoding UTF-8

RoxygenNote 7.3.2

LazyData true

VignetteBuilder knitr

Imports Biostrings, caret, data.table, dplyr, GenomicAlignments, IRanges, mltools, randomForest, Rsamtools, stats, xgboost, MTPS, stringr, R.utils, Rdpack

Depends R (>= 3.5.0)

Suggests rmarkdown, testthat (>= 3.0.0), knitr, httr

RdMacros Rdpack

License MIT + file LICENSE

LazyDataCompression xz

NeedsCompilation no

Author Haochen Ning [aut], Ian Boyes [aut], Ibrahim Numanagić [aut] (ORCID: <https://orcid.org/0000-0002-2970-7937>), Michael Rott [aut], Li Xing [aut] (ORCID: <https://orcid.org/0000-0002-4186-7909>), Xuekui Zhang [aut, cre] (ORCID: <https://orcid.org/0000-0003-4728-2343>)

Maintainer Xuekui Zhang <xuekui@uvic.ca>

Config/pak/sysreqs make libbz2-dev libicu-dev liblzma-dev xz-utils zlib1g-dev

Repository https://cranhaven.r-universe.dev

Date/Publication 2026-11-11 03:03:00 UTC

RemoteUrl https://github.com/cranhaven/cranhaven.r-universe.dev

RemoteRef package/iimi

RemoteSha 471e3bc3aaa5a12498b4a39aab6de2bb5d36c728

RemoteSubdir iimi

Contents

convert_bam_to_rle 2
convert_rle_to_df 3
create_high_nucleotide_content 4
create_mappability_profile 5
example_cov 6
example_diag 6
nucleotide_info 7
plot_cov 7
predict_iimi 8
train_iimi 9
trained_en 10
trained_rf 11
trained_xgb 11
unreliable_regions 11

Index **13**

`convert_bam_to_rle` *convert_bam_to_rle*

Description

Converts one or more indexed and sorted BAM files (ending in `*.sorted.bam` and `*.bai`) into a run-length encodings (RLEs) list.

Usage

```
convert_bam_to_rle(bam_file, paired = FALSE)
```

Arguments

`bam_file`	path to BAM file(s).
`paired`	Indicate if the sequencing paired is single-end or paired-end reads. `TRUE` if paired-end. `FALSE` if single-end.

Value

A list of coverage profile(s) in RLE format with one or more samples.

Examples

```
## Not run:
## Please change the path to your folder where you
## store sorted and indexed BAM files of mapped samples

rles <- convert_bam_to_rle("path/to/bam/file")

## End(Not run)
```

`convert_rle_to_df` *Convert run-length encodings (RLEs) to a data frame.*

Description

Converts a list of run-length encodings (RLEs) into a data frame with 16 features after mappability profiling and nucleotide filtering.

Usage

```
convert_rle_to_df(
  covs,
  unreliable_region_version = "1_4_0",
  unreliable_region_enabled = TRUE,
  additional_nucleotide_info = data.frame()
)
```

Arguments

`covs`	A list of Coverage profile(s) in RLE format. Can be one or more samples.
`unreliable_region_version`	The version number (character string) of unreliable regions of the virus segments. Default is `1_4_0`. It includes the mappability profile from a host genome (we only have Arabidopsis thaliana right now) and virus references, and the regions that have CG% and A% over 60% and 45% respectively.
`unreliable_region_enabled`	Default is `TRUE`. If `TRUE`, the input will be checked against `unreliable_region_df`. If `FALSE`, this step will be skipped.
`additional_nucleotide_info`	Additional nucleotide information for virus segments that are not included in `nucleotide_info`. The information provided must be a data frame that follows the format of `nucleotide_info`. Default is an empty data frame.

Details

Converts a list of run-length encodings (RLEs) into a data frame.

The returned dataframe contains 16 features for training a machine learning model. after mappability profiling and nucleotide filtering.

Value

A data frame object that contains the mapping result for each virus segment that the plant sample reads are aligned to and a RLE list of coverage information.

Examples

```
## Not run:
df <- convert_rle_to_df(example_cov)

## End(Not run)
```

`create_high_nucleotide_content` *create_high_nucleotide_content*

Description

Creates a data frame of the start and end positions of the regions_a that are considered high in A% and GC%.

Usage

```
create_high_nucleotide_content(gc = 0.6, a = 0.45, window = 75, virus_info)
```

Arguments

`gc`	The threshold for GC content. It is the proportion of G and C nucleotides in a sliding window. Default is 0.6.
`a`	The threshold for A nucleotide. It is the proportion of A nucleotide in a sliding window. Default is 0.45.
`window`	The sliding window size of your choice. Default is 75.
`virus_info`	A DNAStringSet of virus segments. The format should be similar to `virus_segments`.

Value

A data frame of the start and end positions of the regions_a that are considered high in A% and GC%.

Examples

```
## Not run: high_nucleotides_regions <- create_high_nucleotide_content()
```

create_mappability_profile
create_mappability_profile

Description

Creates a data frame of start and end positions of the regions that are considered unmappable. Unmappable areas indicate that they can be mapped to another virus segment or a host genome. Note that we only have Arabidopsis Thaliana as a host.

Usage

```
create_mappability_profile(
  path_to_bam_files,
  category,
  window = 75,
  virus_info
)
```

Arguments

path_to_bam_files	Path to the folder that stores the indexed and sorted BAM file(s) (ending in `*.sorted.bam` and `*.bai`).
category	Type of unreliable region you are creating. You can use categories in the provided `unreliable_regions` data frame or customize in your own way.
window	The sliding window size of your choice. Default is 75.
virus_info	A DNAStringSet of virus segments. The format should be similar to `virus_segments`.

Value

A data frame of start and end positions of the regions that are considered unmappable.

Examples

```
## Not run:
## Please change the path to your folder where you store the mapped viruses
mappability_profile <- create_mappability_profile("path/to/folder",
  category = "Unmappable regions")

## End(Not run)
```

example_cov *Coverage profiles of three plant samples.*

Description

A list of coverage profiles for three plant samples. This is only a toy sample. You can use it for running the examples in the vignette. We recommend using more data to train the model, the more the better.

Usage

```
example_cov
```

Format

A list of 3 run length encoding (RLE) lists for 3 plant samples. Each RLE list has the RLE vector of a virus segment

example_diag *Known diagnostics result of virus segments*

Description

A matrix containing the known truth about the diagnostics result (using virus database version 1.4.0) for each plant sample for the example data. It records whether the sample is infected with a virus segment. Each column is a sample, and each row is a virus segment's diagnostics status for three samples.

Usage

```
example_diag
```

Format

A matrix with 3 columns:

S1 Sample one

S2 Sample two

S3 Sample three

nucleotide_info *Nucleotide information of virus segments*

Description

A data set containing the GC content and other information about the virus segments from the official Virtool virus data base (version 1.4.0). The variables are as follows:

Usage

```
nucleotide_info
```

Format

A data frame with 7 variables:

virus_name The virus name

iso_id The virus isolate ID

seg_id The virus segment ID

A_percent The percentage of A nucleotides in the virus segment

C_percent The percentage of C nucleotides in the virus segment

T_percent The percentage of T nucleotides in the virus segment

GC_percent The percentage of G and C nucleotides in the virus segment (GC content)

seg_len The length of the virus segment

1_4_0 The version number of the virus database

1_5_0 The version number of the virus database

plot_cov *plot_cov()*

Description

Plots the coverage profile of the mapped plant sample.

Usage

```
plot_cov(
  covs,
  legend_status = TRUE,
  nucleotide_status = TRUE,
  window = 75,
  nucleotide_info_version = "1_4_0",
  virus_info,
  ...
)
```

Arguments

covs	An RLE list of coverage information of one or more plant samples.
legend_status	Whether display legend. Default is TRUE.
nucleotide_status	Whether display a sliding window of A percentage and CG content. Default is TRUE.
window	The sliding window size. Default is 75.
nucleotide_info_version	The version number (character string) of the nucleotide information of the virus segments. Default is 1_4_0.
virus_info	A DNAStringSet of virus segments. The format should be similar to virus_segments.
...	Other arguments that can be passed to plot, lines, or legend.

Value

The coverage profile of the mapped plant sample.

Examples

```
plot_cov(example_cov$S1)
```

---

predict_iimi *predict_iimi()*

Description

Uses a machine learning model to predict the infection status for the plant sample(s). User can use their own model if needed.

Usage

```
predict_iimi(newdata, method = "xgb", trained_model, report_virus_level = TRUE)
```

Arguments

newdata	A matrix or data frame that contains the features extracted from the coverage profile using convert_bam_to_cov().
method	The machine learning method of choice, rf, xgb, or en. rf stands for random forest model; xgb stands for XGBoost model; and en stands for elastic net model.
trained_model	The trained model. If not provided, default model is used.

report_virus_level
: If TRUE, the function returns the aggregated results based on the virus. If FALSE, the function returns the unaggregated results based on segment level with each decision's probability decided by the model. We do not recommended to set this to FALSE.

Value

A data frame of diagnostics result for each sample

Examples

```
## Not run: df <- convert_rle_to_df(example_cov)
predictions <- predict_iimi(df)

## End(Not run)
```

---

train_iimi *train_iimi()*

Description

Trains a XGBoost (default), Random Forest, or Elastic Net model using user-provided data.

Usage

```
train_iimi(
  train_x,
  train_y,
  method = "xgb",
  nrounds = 100,
  min_child_weight = 10,
  gamma = 20,
  ntree = 200,
  mtry = 10,
  k = 5,
  ...
)
```

Arguments

train_x	A data frame or a matrix of predictors.
train_y	A response vector of labels (needs to be a factor).
method	The machine learning method of choice, Random Forest or XGBoost, or Elastic Net model. Default is XGBoost model.

nrounds	Max number of boosting iterations for XGBoost model. Default is 100.
min_child_weight	Default is 10.
gamma	Minimum loss reduction required in XGBoost model. Default is 20.
ntree	Number of trees in Random Forest model. Default is 100.
mtry	Default is 10.
k	Number of folds. Default is 5.
...	Other arguments that can be passed to randomForest, xgboost, or glmnet.

Value

A Random Forest, XGBoost, Elastic Net model

Examples

```
## Not run:
df <- convert_rle_to_df(example_cov)
train_x <- df[,-c(1:4)]
train_y = c()
for (ii in 1:nrow(df)) {
  seg_id = df$seg_id[ii]
  sample_id = df$sample_id[ii]
  train_y = c(train_y, example_diag[seg_id, sample_id])
}
trained_model <- train_iimi(train_x = train_x, train_y = train_y)

## End(Not run)
```

---

trained_en *A trained model using the default Elastic Net settings*

Description

A trained model using the default Elastic Net settings

Usage

```
trained_en
```

Format

An object of class cv.glmnet of length 13.

trained_rf *A trained model using the default Random Forest settings*

Description

A trained model using the default Random Forest settings

Usage

```
trained_rf
```

Format

An object of class `randomForest.formula` (inherits from `randomForest`) of length 19.

trained_xgb *A trained model using the default XGBoost settings*

Description

A trained model using the default XGBoost settings

Usage

```
trained_xgb
```

Format

An object of class `raw` of length 130645.

unreliable_regions *The unreliable regions of the virus segments*

Description

A data frame of unmappable regions and regions of CG% and A% over 60% and 45% respectively for the virus segments. It is worth to note that if a virus segment does not have any unreliable regions, that virus segment is not shown in this data frame.

Usage

```
unreliable_regions
```

Format

A data frame of unreliable regions in the run-length encoding format for virus segments.

Start The start position of the region that is considered unreliable

End The end position of the region that is considered unreliable

Virus segment The virus segment ID

Categories The category that this unrelible region belong to, which are Unmappable regions (host), Unmappable regions (virus), CG% > 60%, A% > 45%

1_4_0 The version number of the virus database

1_5_0 The version number of the virus database

Index

∗ **datasets**
example_cov, 6
example_diag, 6
nucleotide_info, 7
trained_en, 10
trained_rf, 11
trained_xgb, 11
unreliable_regions, 11

convert_bam_to_rle, 2
convert_rle_to_df, 3
create_high_nucleotide_content, 4
create_mappability_profile, 5

example_cov, 6
example_diag, 6

nucleotide_info, 7

plot_cov, 7
predict_iimi, 8

train_iimi, 9
trained_en, 10
trained_rf, 11
trained_xgb, 11

unreliable_regions, 11