Package ‘snifter’

May 11, 2024

Type Package
Title R wrapper for the python openTSNE library
Version 1.14.0
Date 2023-09-03
Depends R (>= 4.0.0)
Imports basilisk, reticulate, irlba, stats, assertthat
License GPL-3
Encoding UTF-8
LazyData true
RoxygenNote 7.2.1
VignetteBuilder knitr
biocViews DimensionReduction, Visualization, Software, SingleCell, Sequencing
StagedInstall no
BugReports https://github.com/Alanocallaghan/snifter/issues
URL https://bioconductor.org/packages/snifter
Suggests knitr, rmarkdown, BiocStyle, ggplot2, testthat (>= 3.0.0)
Config/testthat/edition 3
git_url https://git.bioconductor.org/packages/snifter
git_branch RELEASE_3_19
git_last_commit 8e5a011
git_last_commit_date 2024-04-30
Repository Bioconductor 3.19
Date/Publication 2024-05-10
Author Alan O'Callaghan [aut, cre],
 Aaron Lun [aut]
Maintainer Alan O'Callaghan <alan.ocallaghan@outlook.com>
Description

Project new data into an existing t-SNE embedding object.

Usage

```r
project(x, new, old, perplexity = 5,
       initialization = c("median", "weighted", "random"),
       k = 25L,
       learning_rate = 0.1,
       early_exaggeration = 4,
       early_exaggeration_iter = 0L,
       exaggeration = 1.5,
       n_iter = 250L,
       initial_momentum = 0.5,
       final_momentum = 0.8,
       max_grad_norm = 0.25,
       tolerance = 1e-04)
```

Arguments

- `x`  
  t-SNE embedding created with `fitsne`.

- `new`  
  New data to project into existing embedding

- `old`  
  Data used to create the original embedding.

- `perplexity`  
  Numeric scalar. Perplexity can be thought of as the continuous number of nearest neighbors, for which t-SNE will attempt to preserve distances. However, when projecting, we only consider neighbors in the existing embedding i.e. each data point is placed into the embedding, independently of other new data points.

- `initialization`  
  Character scalar specifying the method used to compute the initial point positions to be used in the embedding space. Can be "median", "weighted" or "random". In all cases, "median" or "weighted" should be preferred.
project

k Integer scalar specifying the number of nearest neighbors to consider when initially placing the point onto the embedding. This is different from "perplexity" because perplexity affects optimization while this only affects the initial point positions.

learning_rate The learning rate for t-SNE optimization. When learning_rate="auto" the appropriate learning rate is selected according to max(200, N / 12), as determined in Belkina et al. Otherwise, a numeric scalar.

early_exaggeration Numeric scalar; the exaggeration factor to use during the *early exaggeration* phase. Typical values range from 12 to 32.

early_exaggeration_iter The number of iterations to run in the *early exaggeration* phase.

exaggeration The exaggeration factor to use during the normal optimization phase. This can be used to form more densely packed clusters and is useful for large data sets.

n_iter The number of iterations to run in the normal optimization regime.

initial_momentum The momentum to use during the *early exaggeration* phase.

final_momentum The momentum to use during the normal optimization phase.

max_grad_norm Maximum gradient norm. If the norm exceeds this value, it will be clipped. When adding points into an existing embedding, and the new points overlap with the reference points, this may lead to large gradients. This can make points "shoot off" from the embedding, causing the interpolation method to compute a very large grid, and leads to worse results.

tolerance Numeric scalar specifying the numeric tolerance used to ensure the affinities calculated on the old data match those of the original embedding.

Value

Numeric matrix of t-SNE co-ordinates resulting from embedding new into the t-SNE embedding x.

References


Examples

```r
set.seed(42)
m <- matrix(rnorm(2000), ncol=20)
out_binding <- fitsne(m[-(1:2), ], random_state = 42L)
new_points <- project(out_binding, new = m[1:2, ], old = m[-(1:2), ])
plot(as.matrix(out_binding), col = "black", pch = 19,
     xlab = "t-SNE 1", ylab = "t-SNE 2")
points(new_points, col = "red", pch = 19)
```
Description
An R package for running openTSNE’s implementation of fast interpolated t-SNE.
See the openTSNE documentation for further details on these arguments and the general usage of this algorithm.

Usage
fitsne(
  x,
  simplified = FALSE,
  n_components = 2L,
  n_jobs = 1L,
  perplexity = 30,
  n_iter = 500L,
  initialization = c("pca", "spectral", "random"),
  pca = FALSE,
  pca_dims = 50L,
  partial_pca = FALSE,
  pca_center = TRUE,
  pca_scale = TRUE,
  neighbors = c("auto", "exact", "anony", "pynndescent", "approx"),
  negative_gradient_method = c("fft", "bh"),
  learning_rate = "auto",
  early_exaggeration = 12,
  early_exaggeration_iter = 250L,
  exaggeration = NULL,
  dof = 1,
  theta = 0.5,
  n_interpolation_points = 3L,
  min_num_intervals = 50L,
  ints_in_interval = 1,
  metric = "euclidean",
  metric_params = NULL,
  initial_momentum = 0.5,
  final_momentum = 0.8,
  max_grad_norm = NULL,
  random_state = NULL,
  verbose = FALSE
)

Arguments
x Input data matrix.
simplified Logical scalar. When FALSE, the function returns an object of class snifter. This contains all information necessary to project new data into the embedding using project. If TRUE, all extra attributes will be omitted, and the return value is a base matrix.

n_components Number of t-SNE components to be produced.

n_jobs Integer scalar specifying the number of cores to be used.

perplexity Numeric scalar controlling the neighborhood used when estimating the embedding.

n_iter Integer scalar specifying the number of iterations to complete.

initialization Character scalar specifying the initialization to use. “pca” may preserve global distance better than other options.

pca Logical scalar specifying whether PCA should be run on the data before creating the embedding.

pca_dims Integer scalar specifying the number of principal components to be calculated in the initial PCA step if pca=TRUE.

partial_pca Logical scalar specifying whether prcomp_irlba should be used if pca=TRUE. This is useful for very large data matrices.

dof Numeric scalar specifying the degrees of freedom, as described in Kobak et al. (2019).

theta Numeric scalar, only used when negative_gradient_method=“bh”. This is the trade-off parameter between speed and accuracy of the tree approximation method. Typical values range from 0.2 to 0.8. The value 0 indicates that no approximation is to be made and produces exact results also producing longer runtime.
n_interpolation_points

Integer scalar, only used when negative_gradient_method="fft". The number of interpolation points to use within each grid cell for interpolation based t-SNE. It is highly recommended leaving this value at the default 3.

min_num_intervals

Integer scalar, only used when negative_gradient_method="fft". The minimum number of grid cells to use, regardless of the ints_in_interval parameter. Higher values provide more accurate gradient estimations.

ints_in_interval

Numeric scalar, only used when negative_gradient_method="fft". Indicates how large a grid cell should be e.g. a value of 3 indicates a grid side length of 3. Lower values provide more accurate gradient estimations.

metric

Character scalar specifying the metric to be used to compute affinities between points in the original space.

metric_params

Named list of additional keyword arguments for the metric function.

initial_momentum

Numeric scalar specifying the momentum to use during the early exaggeration phase.

final_momentum

Numeric scalar specifying the momentum to use during the normal optimization phase.

max_grad_norm

Numeric scalar specifying the maximum gradient norm. If the norm exceeds this value, it will be clipped.

random_state

Integer scalar specifying the seed used by the random number generator.

verbose

Logical scalar controlling verbosity.

Value

A matrix of t-SNE embeddings.

References


Examples

```r
set.seed(42)
m <- matrix(rnorm(2000), ncol=20)
out <- fitsne(m, random_state = 42L)
plot(out, pch = 19, xlab = "t-SNE 1", ylab = "t-SNE 2")

## openTSNE allows us to project new points into the existing
## embedding - useful for extremely large data.
## see https://opentsne.readthedocs.io/en/latest/api/index.html

out_binding <- fitsne(m[-(1:2), ], random_state = 42L)
new_points <- project(out_binding, new = m[1:2, ], old = m[-(1:2), ])
plot(as.matrix(out_binding), col = "black", pch = 19,
    xlab = "t-SNE 1", ylab = "t-SNE 2")
points(new_points, col = "red", pch = 19)
```
Index

fitsne, 2
fitsne (snifter), 4

prcomp_irlba, 5
project, 2, 5

snifter, 4