R documentation

of ‘trainDigitalDLSorterModel.Rd’

September 3, 2020

---

trainDigitalDLSorterModel
Train DigitalDLSorter *Deep Neural Network model.*

Description

Train DigitalDLSorter Deep Neural Network model with data store in final.data slot. Moreover, model is evaluated on test data and prediction results are produced.

Usage

```
trainDigitalDLSorterModel(
  object,
  batch.size = 128,
  num.epochs = 20,
  val = FALSE,
  freq.val = 0.1,
  loss = "kullback_leibler_divergence",
  metrics = c("accuracy", "mean_absolute_error", "categorical_accuracy"),
  view.metrics.plot = TRUE,
  verbose = TRUE
)
```

Arguments

object DigitalDLSorter object with final.data slot.

batch.size Number of samples per gradient update. If unspecified, batch.size will default to 128.

num.epochs Number of epochs to train the model.

val Boolean that determines if a validation subset is used during training (FALSE by default).

freq.val Number between 0.1 and 0.5 that determines the number of samples from training data that will be used as validation subset.

loss Character indicating loss function selected for training the model (Kullback-Leibler divergence by default). Look at keras documentation to see available loss functions.

`metrics` Vector of metrics used to evaluate the performance of the model during training and on test data (`c("accuracy","mean_absolute_error","categorical_accuracy` by default)

`verbose` Boolean indicating if show the progression of the model during training. Besides, it is shown information about the architecture of the model (`TRUE` by default).

`view.metrics.plots`
Boolean indicating if show progression plots of loss and metrics during training (`TRUE` by default). `keras` for R allows to see the progression of the model during training if you are working on RStudio.

Details

All steps related with Deep Neural Network in `digitalDLSorteR` package are performed by using `keras` package, an API in R for `keras` in Python available from CRAN. We recommend use the guide of installation available on `https://keras.rstudio.com/` in order to set a custom configuration (type of back-end used, CPU or GPU, etc.).

Although `trainDigitalDLSorterModel` allows to select a custom loss function used during training, we recommend using Kullback-Leibler divergence because its better results. If you want to know more details about the architecture of the DNN and its construction, see Torroja and Sanchez-Cabo, 2019.

Value

A `DigitalDLSorter` object with `trained.model` slot containing a `DigitalDLSorterDNN` object. For more information about the structure of this class, see `DigitalDLSorterDNN`.

References

Torroja, C. y Sánchez-Cabo, F. (2019). digitalDLSorter: A Deep Learning algorithm to quantify immune cell populations based on scRNA-Seq data. Frontiers in Genetics 10, 978. doi: `10.3389/fgene.2019.00978`

See Also

`plotTrainingHistory` `deconvDigitalDLSorter` `deconvDigitalDLSorterObj`

Examples

```
## Not run:
DDLSChung <- trainDigitalDLSorterModel(
  object = DDLSChung,
  batch.size = 128,
  num.epochs = 20
)

## End(Not run)
```

Index