Package 'resautonet'

March 17, 2019

Type Package

Title Autoencoder-based Residual Deep Network with Keras Support

Version 1.0

Date 2019-03-13

Author Lianfa Li

Maintainer Lianfa Li <lspatial@gmail.com>

Description This package is the R implementation of the Autoencoder-based Residual Deep Network that is based on this paper (https://arxiv.org/abs/1812.11262).

Depends R (>= 3.1)

Imports Rcpp (>= 1.0.0),keras,rstack,dplyr,magrittr

LinkingTo Rcpp, RcppArmadillo

SystemRequirements C++11

License GPL

Encoding UTF-8

LazyData true

NeedsCompilation yes

RoxygenNote 6.1.1

Repository CRAN

Date/Publication 2019-03-16 23:00:08 UTC

R topics documented:

resautonet-package *Autoencoder-based Residual Deep Network with Keras Support*

Description

This package is the R implementation of the Autoencoder-based Residual Deep Network that is based on this paper (https://arxiv.org/abs/1812.11262).

Details

The DESCRIPTION file:

Package:	resautonet
Type:	Package
Title:	Autoencoder-based Residual Deep Network with Keras Support
Version:	1.0
Date:	2019-03-13
Author:	Lianfa Li
Maintainer:	Lianfa Li <lspatial@gmail.com>
Description:	This package is the R implementation of the Autoencoder-based Residual Deep Network that is based
Depends:	R (>= 3.1)
Imports:	Rcpp (>= 1.0.0),keras,rstack,dplyr,magrittr
LinkingTo:	Rcpp, RcppArmadillo
SystemRequirements:	C++11
License:	GPL
Encoding:	UTF-8
LazyData:	true
NeedsCompilation:	yes
RoxygenNote:	6.1.1

Index of help topics:

Autoencoder-based residual deep network, Just call the function, AutoEncoderModel to get the model in run in R

Author(s)

Lianfa Li

Maintainer: Lianfa Li <lspatial@gmail.com>

References

Lianfa Li et al, 2018, Autoencoder Based Residual Deep Networks for Robust Regression Prediction and Spatiotemporal Estimation, preprint, arXiv:1812.11262 [cs.LG]

Examples

```
###### Generate the data
#Sample size
library(dplyr)

n=4000
#Number of features
nfea=4
#Generate the dataset with normalization of the covariates
cmdStr="data.frame("
for(i in c(1:nfea)){
  var=runif(n,runif(1,1,5),runif(1,100,200))
  varName=paste("x",i,sep="")
  assign(varName,var)
  cmdStr=paste(cmdStr,varName,"n=(",varName,"-mean(",varName,"))/sd(",varName,")",sep="")
  if(i<nfea)
    cmdStr=paste(cmdStr,",",sep="")
}
cmdStr=paste(cmdStr,")",sep="")
dataset=eval(expr=parse(text=cmdStr))
y=sin(x1)+2*cos(x2)+x3^2+sqrt(x4)+rnorm(n)
#Normalization of y
yn=(y-mean(y))/sd(y)
#Obtain the index of training and test samples
prop=0.2
test_index=sample(c(1:n),size=ceiling(n*prop))
train_index=setdiff(c(1:n),test_index)

#Obtain the training and test dataset
x_train=as.matrix(dataset[train_index,])
y_train=as.vector(yn[train_index])

x_test=as.matrix(dataset[test_index,])
y_test=as.vector(yn[test_index])

#Define the metric, r2 in keras
metric_r2= keras::custom_metric("rsquared", function(y_true, y_pred) {
  SS_res =keras::k_sum(keras::k_square(y_true-y_pred ))
  SS_tot =keras::k_sum(keras::k_square(( y_true - keras::k_mean(y_true))))
  return ( 1 - SS_res/(SS_tot + keras::k_epsilon()))
```

```
})

#Define the autoencoder-based deep network
nout=1;nodes=c(16,8,4,2);mdropout=0.2;isres=TRUE;outtype=0;fact="linear"
acts=rep("relu",length(nodes));fact="linear";reg=NULL;batchnorm=TRUE
autoresmodel=resautonet::AutoEncoderModel(nfea,nout,nodes,
      acts,mdropout,reg,batchnorm,isres,outtype,fact=fact)
autoresmodel %>% keras::compile(
  loss = "mean_squared_error",
  optimizer = keras::optimizer_rmsprop(),
  metrics = c("mean_squared_error",metric_r2)
)
#Show the network
summary(autoresmodel)

early_stopping = keras::callback_early_stopping(monitor ='loss', min_delta=0.0000001,patience=10)
reduce=keras::callback_reduce_lr_on_plateau(patience=10)

nepoch=10
#Train the network
history <- autoresmodel %>% keras::fit(
  x_train, y_train,
  epochs = nepoch, batch_size = 60, callbacks=list(early_stopping,reduce),
  validation_split = 0.2,verbose=0
)
#Predict the test samples
pred = autoresmodel %>% predict(x_test,batch_size = 30)
pred=pred*sd(y)+mean(y)
y_test=y_test*sd(y)+mean(y)

#Test results
testr2=r2_squ(y_test,y_test-pred)
testrmse=rmse(y_test,pred)

#Print the results
print(paste("R2: regular ",round(max(history$metrics$rsquared),3),
            ";validation ",round(max(history$metrics$val_rsquared),3),
            ";test ",round(testr2,3),sep=""))
print(paste("RMSE: regular ",round(min(history$metrics$mean_squared_error),3),
            ";validation ",round(min(history$metrics$val_mean_squared_error),3),
            ";test ",round(testrmse,3),sep=""))
```

AutoEncoderModel *AutoEncoderModel*

Description

This function is to construct a residual autoencoder-based deep network.

Usage

```
AutoEncoderModel(nfea, nout, nodes, acts, mdropout = 0, reg = NULL,
  batchnorm = TRUE, isres = TRUE, outtype = 0, fact = "linear")
```

Arguments

nfea	integer, Number of features
nout	integer, Number of output units
nodes	list of integers, list of the number of nodes for the hidden layers in encoding component
acts	list of strings, list of activation function names
mdropout	double, dropout rate of the coding (middle) layer (default:0)
reg	string, regularization string (default: NULL)
batchnorm	bool, flag to conduct batch normalization (default:TRUE)
isres	bool, flag to conduct the residual connections (default: TRUE)
outtype	integer, output type, 0 indicating nout outputs and 1 nout+nfea outputs (default: 0)
fact	string, activation for output layer (default:"linear")

Value

keras model, model of (residual) autoencoder-based deep network

---

hiddenBlock *hiddenBlock*

Description

This function is to construct the hidden block .

Usage

```
hiddenBlock(inlayer, nodes, acts, idepth, orginlayer = NULL,
  reg = NULL, dropout = 0, batchnorm = TRUE)
```

Arguments

inlayer	input layer, keras layer
nodes	list of integers, list of the number of nodes for all the hidden layers
acts	list of strings, list of activations for hidden layers
idepth	integer, index of the hidden layer
orginlayer	keras layer, original layer to be added to decoding layer (default: NULL)
reg	string, regularization (default: NULL)
dropout	double, dropout rate for the target hidden layer (default 0)
batchnorm	bool, flag to conduct batch normalization (default: TRUE)

Value

keras layer, block of a hidden layer (with addtion of actionvation or/and batch normalization)

`keras_r2` *keras_r2*

Description

This function is to calculate rsquared value for regression models.

Usage

```
keras_r2(y_true, y_pred)
```

Arguments

`y_true`	tensor of ground truth
`y_pred`	tensor of predicted values

Value

keras tensor, double for rsquared

`r2_squ` *r2_squ*

Description

This function is to calculate rsquared value for regression models.

Usage

```
r2_squ(obs, res)
```

Arguments

`obs`	: vector, observed values
`res`	residual (observed values-predicted values)

Value

double for rsquared

RcppArmadillo-Functions

Set of functions in example RcppArmadillo package

Description

These four functions are created when `RcppArmadillo.package.skeleton()` is invoked to create a skeleton packages.

Usage

```
rcpparma_hello_world()
rcpparma_outerproduct(x)
rcpparma_innerproduct(x)
rcpparma_bothproducts(x)
```

Arguments

`x`	a numeric vector

Details

These are example functions which should be largely self-explanatory. Their main benefit is to demonstrate how to write a function using the Armadillo C++ classes, and to have to such a function accessible from R.

Value

`rcpparma_hello_world()` does not return a value, but displays a message to the console.

`rcpparma_outerproduct()` returns a numeric matrix computed as the outer (vector) product of `x`.

`rcpparma_innerproduct()` returns a double computer as the inner (vector) product of `x`.

`rcpparma_bothproducts()` returns a list with both the outer and inner products.

Author(s)

Dirk Eddelbuettel

References

See the documentation for Armadillo, and RcppArmadillo, for more details.

Examples

```
x <- sqrt(1:4)
rcpparma_innerproduct(x)
rcpparma_outerproduct(x)
```

rmse *rmse*

Description

This function is to calculate rsquared value for regression models.

Usage

```
rmse(obs, pre)
```

Arguments

obs	vector, observed values
pre	vector, (predicted values)

Value

double for rmse

Index