Package: communication (via r-universe)

October 22, 2024

Type Package

Title Feature Extraction and Model Estimation for Audio of Human Speech

Version 0.1

Date 2021-02-12

Maintainer Christopher Lucas <christopher.lucas@wustl.edu>

Description Provides fast, easy feature extraction of human speech and model estimation with hidden Markov models. Flexible extraction of phonetic features and their derivatives, with necessary preprocessing options like feature standardization. Communication can estimate supervised and unsupervised hidden Markov models with these features, with cross validation and corrections for auto-correlation in features. Methods developed in Knox and Lucas (2021) <doi:10.7910/DVN.8BTOHQ>.

Depends R (>= 3.5.0)

License GPL (>= 2)

Imports Rcpp (>= 1.0.2), purrr, magrittr, diagram, GGally, grid, useful, ggplot2, reshape2, tuneR, wrassp, gtools, signal, plyr, RColorBrewer, scales, abind, igraph, gtable

LinkingTo Rcpp, RcppArmadillo (>= 0.9.700.2.0)

RoxygenNote 7.1.1

Encoding UTF-8

Suggests knitr, qpdf, rmarkdown, testthat

VignetteBuilder knitr

NeedsCompilation yes

Author Dean Knox [aut], Christopher Lucas [aut, cre], Guilherme Duarte [ctb], Alex Shmuley [ctb], Vineet Bansal [ctb], Vadym Vashchenko [ctb]

Repository CRAN

Date/Publication 2021-02-25 09:20:02 UTC

Contents

`audio` *Audio features from Stephen Breyer*

Description

Features extracted from 100 randomly selected utterances by Stephen Breyer in Supreme Court Oral Arguments.

Usage

```
data(audio)
```

Format

An object of class preppedAudio

`extractAudioFeatures` *Title*

Description

Title

Usage

```
extractAudioFeatures(
  wav.dir = getwd(),
  wav.fnames = NULL,
  windowSize = 25,
  windowShift = 12.5,
  windowType = "HAMMING",
  derivatives = 2,
  verbose = 1,
  recursive = FALSE
)
```

Arguments

wav.dir	Directory of wav files for featurization
wav.fnames	If wav.dir = NULL, a list of wav files for featurization
windowSize	Size of window in milliseconds
windowShift	Amoung to shift window in milliseconds
windowType	Window type
derivatives	Include no (0), first (1), or first and second (2) derivatives of features
verbose	Verbose printing
recursive	Recursively traverse directory for wav files

Value

An object of class preppedAudio, which consists of a list of ‘data‘, ‘files‘, and ‘control‘. ‘data‘ is a list with elements corresponding to audio features for each of the input wav files, where each element is the audio features for the respective wav file. ‘files‘ contains metadata about each wav file for which audio features were extracted. ‘control‘ records arguments passed to extractAudioFeatures().

Examples

```
## Not run:
wav.fnames = list.files(file.path('PATH/TO/WAV/FILES'),
                        pattern = 'wav$',
                        recursive = TRUE,
                        full.names = TRUE
                        )
audio <- extractAudioFeatures(wav.fnames = wav.fnames,
                              derivatives = 0
                              )

## End(Not run)
```

---

hmm *Train a hidden Markov model with multivariate normal state distributions.*

Description

Train a hidden Markov model with multivariate normal state distributions.

Usage

```
hmm(
  Xs,
  weights = NULL,
  nstates,
  par = list(),
  control = list(),
  labels = list()
)
```

Arguments

Xs	List of nsequences matrices; each matrix represents one observation sequence and is of dimension nobs x nfeatures. For a single observation sequence, a single matrix can be provided
weights	Optional vector of weights, one for each observation sequence
nstates	Integer; number of states
par	List of initialization parameters; see 'Details'
control	List of control parameters for EM steps
labels	List of observation labels for supervised training, with each element corresponding to an observation sequence. Element i can either be an vector of integer state labels in `1:nstates` or a matrix of dimension `nstates` x `nrow(Xs[[i]])` with columns summing to 1. If labels are supplied, E-step is suppressed.

Details

The `par` argument is a list of initialization parameters. Can supply any of the following components:

- `method` Name of method used to automatically initialize EM run. Currently only `'dirichlet'` and `'random-spherical'` are implemented. If provided, user-specified state distributions are ignored. `'dirichlet'` randomly generates responsibilities which are in turn used to calculate starting distributions. `'random-spherical'` randomly draws nstates observations and uses their features as state means; all state covariance matrices are set to a diagonal matrix with entries `method_arg` (default=1).
- `method_arg` Argument to supply to `method`. For `method='dirichlet'`, this is a scalar concentration `alpha` (same value used for all states). For `method='random-spherical'`, this is a scalar for diagonal entries of the spherical covariance matrices of the starting distributions (after features are standardized). `'dirichlet'` is implemented. If provided, all other arguments are ignored.
- `resp` Matrix or list of nsequences matrices with rows summing to 1; each matrix represents one observation sequence and is of dimension nobs x nstates, with the (t,k)-th entry giving the initial probability that the t-th observation belongs to state k. If either `resp` or both `mus` and `Sigmas` are not provided, responsibilities are randomly initialized using `rdirichlet` with all shape parameters set to 10.
- `mus` List of nstates vectors with length nfeatures, each corresponding to the mean of a state distribution

- Sigmas List of nstates matrices with dimension nfeatures x nfeatures, each corresponding to the covariance matrix of a state distribution
- Gamma Matrix of transition probabilities with dimension nstates x nstates, with row k representing the probabilities of each transition out of k and summing to 1. If not supplied, each row is randomly drawn from rdirichlet with all shape parameters set to 10.
- delta Vector of initial state probabilities, of length nstates and summing to 1. If not supplied, delta is set to the stationary distribution of Gamma, i.e. the normalized first left eigenvector.

The control argument is a list of EM control parameters that can supply any of the following components

- lambda Ridge-like regularization parameter. lambda is added to each diag(Sigmas[[k]]) to stabilize each state's covariance matrix, which might otherwise be numerically singular, before inverting to calculate multivariate normal densities. Note that regularization is applied after all features are standardized, so diag(Sigmas[[k]]) is unlikely to contain elements greater than 1. This parameter should be selected through cross-validation.
- tol EM terminates when the improvement in the log-likelihood between successive steps is < tol. Defaults to 1e-6.
- maxiter EM terminates with a warning if maxiter iterations are reached without convergence as defined by tol. Defaults to 100.
- uncollapse Threshold for detecting and resetting state distribution when they collapse on a single point. State distributions are uncollapsed by re-drawing mus[[k]] from a standard multivariate normal and setting Sigmas[[k]] to the nfeatures-dimensional identity matrix. Note that this distribution is with respect to the standardized features.
- standardize Whether features should be standardized. Defaults to TRUE. This option also adds a small amount of noise , equal to .01 x feature standard deviation, to observation-features that have been zeroed out (e.g. f0 during unvoiced periods). If set to FALSE, it is assumed that features have been externally standardized and zeroed-out values handled. scaling$feature_means and scaling$feature_sds are set to 0s and 1s, respectively, and no check is done to ensure this is correct. If features are in fact not standardized and zeroes handled, bad things will happen and nobody will feel sorry for you.
- verbose Integer in 0:1. If 1, information on the EM process is reported. Defaults to 1.

Value

An object of class hmm. Contains fitted values of model parameters, along with input values for hyperparameters and features.

Examples

```
data('audio')
## Not run:
mod <- hmm(audio$data, nstates = 2, control = list(verbose = TRUE))

## End(Not run)
```

llh *Title*

Description

Title

Usage

```
llh(Xs, mod, control = list())
```

Arguments

Xs	List of nsequences matrices; each matrix represents one observation sequence and is of dimension nobs x nfeatures. For a single observation sequence, a single matrix can be provided
mod	Model object of class 'feelr.hmm', as output by hmm
control	List of control parameters

Value

List with two components. llhs is a numeric vector of log-likelihoods of each observation sequence in Xs. llh_total is the log-likelihood of all observation sequences together, i.e. sum(llhs). If Xs is the same data that generated mod, the values calculated here will be slightly lower than those output in mod$llhs. This is because hmm estimates the starting state of each sequence, whereas here it is assumed that the starting state is drawn from the stationary distribution mod$delta.

standardizeFeatures *Title*

Description

Title

Usage

```
standardizeFeatures(Xs, feature_means = NULL, feature_sds = NULL, verbose = 1)
```

Arguments

Xs	Data
feature_means	Numeric vector corresponding to columns of elements in Xs
feature_sds	Numeric vector corresponding to columns of elements in Xs. If not supplied, will be computed from Xs.
verbose	Verbose printing

Details

`feature_means` and `feature_sds` are provided to allow alignment of new datasets. For example, after a model is trained, new data for prediction must be transformed in the same way as the training data to ensure predictions are valid. If either is `NULL`, both will be computed from `Xs` and the output will be internally standardized (i.e., columns of `do.call(rbind, standardizeFeatures(Xs))` will be have a mean of 0 and a standard deviation of 1).

Value

Standardizes 'data' of objects of class 'preppedAudio'. Maintains structure of original object otherwise. Is used to standardize data where the recording environment systematically shifts audio features.

Examples

```
data('audio')
audio$data <- standardizeFeatures(
    lapply(audio$data, function(x) na.omit(x))
)
```

Index