Package: text2speech (via r-universe)

November 14, 2024

Type Package

Title Text to Speech Conversion

Description Converts text into speech using various text-to-speech (TTS) engines and provides an unified interface for accessing their functionality. With this package, users can easily generate audio files of spoken words, phrases, or sentences from plain text data. The package supports multiple TTS engines, including Google's 'Cloud Text-to-Speech API', 'Amazon Polly', Microsoft's 'Cognitive Services Text to Speech REST API', and a free TTS engine called 'Coqui TTS'.

Version 1.0.0

License GPL-3

Suggests aws.polly, covr, patrick, rmarkdown, stringi, testthat (>= 3.0.0)

Encoding UTF-8

VignetteBuilder knitr

RoxygenNote 7.2.3

URL https://github.com/jhudsl/text2speech

BugReports https://github.com/jhudsl/text2speech/issues

Imports aws.signature, cli, dplyr, googleAuthR, googleLanguageR, knitr, magrittr, conrad, tidyr, tuneR, utils, withr

Roxygen list(markdown = TRUE)

Config/testthat/edition 3

Config/pak/sysreqs make libicu-dev libssl-dev

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

RemoteUrl https://github.com/jhudsl/text2speech

RemoteRef HEAD

RemoteSha f3ab315e7a095cdce63809fbd91f227c48e2eb3a

Contents

`pcm_to_wav` *Convert PCM to WAV*

Description

Accepts PCM audio data as input and generates a corresponding WAV file

Usage

```
pcm_to_wav(
  input,
  output = tempfile(fileext = ".wav"),
  sample_rate = 16000,
  extensible = FALSE
)
```

Arguments

`input`	output from ‘get_synthesis“ from `aws.polly` or PCM filename
`output`	output file for Wav file
`sample_rate`	Sampling rate for tuneR::Wave
`extensible`	passed to tuneR::writeWave

Value

A filename of the output

Examples

```
## Not run:
fname = system.file("extdata", "pcm_file.wav", package = "text2speech")
res = pcm_to_wav(fname)
testthat::expect_error(tuneR::readWave(fname))
testthat::expect_is(tuneR::readWave(res), "Wave")

## End(Not run)
## Not run:
if (requireNamespace("aws.polly", quietly = TRUE)) {
text = "hey, ho, let's go!"
if (tts_amazon_auth()) {
   res = tts_amazon(text, output_format = "wav")
}
}

## End(Not run)
```

`play_audio` *Play audio in a browser*

Description

This uses HTML5 audio tags to play audio in your browser.

Usage

```
play_audio(audio = "output.wav", html = "player.html")
```

Arguments

`audio`	The file location of the audio file. Must be supported by HTML5.
`html`	The html file location that will be created to host the audio file.

Details

Borrowed from `googleLanguageR::gl_talk_player()`

Examples

```
## Not run:
  play_audio(audio = "audio.wav", html = "player.html")

## End(Not run)
```

set_coqui_path *Point to local coqui tts Executable File*

Description

Function to set an option that points to the local coqui tts Executable File `tts`.

Usage

```
set_coqui_path(path)
```

Arguments

`path`	path to the local coqui tts Executable File

Details

List of possible file path locations for the local coqui tts Executable File

Linux /usr/bin/tts, /usr/local/bin/tts

Mac /opt/homebrew/Caskroom/miniforge/base/bin/tts

Windows C:\Program Files\tts

Value

Returns nothing, function sets the option variable `path_to_coqui`.

Examples

```
set_coqui_path("~/path/to/tts")
```

tts *Text-to-Speech (Speech Synthesis)*

Description

Convert text-to-speech using various engines, including Amazon Polly, Coqui TTS, Google Cloud Text-to-Speech API, and Microsoft Cognitive Services Text to Speech REST API.

With the exception of Coqui TTS, all these engines are accessible as R packages:

- aws.polly is a client for Amazon Polly.
- googleLanguageR is a client to the Google Cloud Text-to-Speech API.
- conrad is a client to the Microsoft Cognitive Services Text to Speech REST API

Usage

```
tts(
  text,
  output_format = c("mp3", "wav"),
  service = c("amazon", "google", "microsoft", "coqui"),
  bind_audio = TRUE,
  ...
)

tts_amazon(
  text,
  output_format = c("mp3", "wav"),
  voice = "Joanna",
  bind_audio = TRUE,
  save_local = FALSE,
  save_local_dest = NULL,
  ...
)

tts_google(
  text,
  output_format = c("mp3", "wav"),
  voice = "en-US-Standard-C",
  bind_audio = TRUE,
  save_local = FALSE,
  save_local_dest = NULL,
  ...
)

tts_microsoft(
  text,
  output_format = c("mp3", "wav"),
  voice = NULL,
  bind_audio = TRUE,
  save_local = FALSE,
  save_local_dest = NULL,
  ...
)

tts_coqui(
  text,
  exec_path,
  output_format = c("wav", "mp3"),
  model_name = "tacotron2-DDC_ph",
  vocoder_name = "ljspeech/univnet",
  bind_audio = TRUE,
  save_local = FALSE,
  save_local_dest = NULL,
```

```
  ...
)
```

Arguments

text	A character vector of text to be spoken
output_format	Format of output files: "mp3" or "wav"
service	Service to use (Amazon, Google, Microsoft, or Coqui)
bind_audio	Should the tts_bind_wav() be run on after the audio has been created, to ensure that the length of text and the number of rows is consistent?
...	Additional arguments
voice	Full voice name
save_local	Should the audio file be saved locally?
save_local_dest	If to be saved locally, destination where output file will be saved
exec_path	System path to Coqui TTS executable
model_name	(Coqui TTS only) Deep Learning model for Text-to-Speech Conversion
vocoder_name	(Coqui TTS only) Voice coder used for speech coding and transmission

Value

A standardized tibble featuring the following columns:

- index : Sequential identifier number
- original_text : The text input provided by the user
- text : In case original_text exceeds the character limit, text represents the outcome of splitting original_text. Otherwise, text remains the same as original_text.
- wav : Wave object (S4 class)
- file : File path to the audio file
- audio_type : The audio format, either mp3 or wav
- duration : The duration of the audio file
- service : The text-to-speech engine used

Examples

```
## Not run:
# Amazon Polly
tts("Hello world! This is Amazon Polly", service = "amazon")

tts("Hello world! This is Coqui TTS", service = "coqui")

tts("Hello world! This is Google Cloud", service = "google")

tts("Hello world! This is Microsoft", service = "microsoft")

## End(Not run)
```

tts_auth *Authentication for Text-to-Speech (Speech Synthesis) Engines*

Description

Verify the authentication status of different text-to-speech engines, including Amazon Polly, Coqui TTS, Google Cloud Text-to-Speech API, and Microsoft Cognitive Services Text to Speech REST API.

Usage

```
tts_auth(
  service = c("amazon", "google", "microsoft", "coqui"),
  key_or_json_file = NULL,
  ...
)

tts_amazon_auth(key_or_json_file = NULL, ...)

tts_google_auth(key_or_json_file = NULL, ...)

tts_microsoft_auth(key_or_json_file = NULL, ...)

tts_coqui_auth()
```

Arguments

service	Service to use (Amazon, Google, Microsoft, or Coqui)
key_or_json_file	Either an API key (for Microsoft) or JSON file (for Google)
...	Additional arguments

Details

To determine the availability of Coqui TTS, `tts_auth()` examines whether the tts executable exists on local system.

Value

A logical indicator of authorization

Examples

```
# Amazon Polly
tts_auth("amazon")

# Google Cloud Text-to-Speech API
```

```
tts_auth("google")

# Microsoft Cognitive Services Text to Speech REST API
tts_auth("microsoft")

# Coqui TTS
tts_auth("coqui")
```

`tts_bind_wav` *Bind WAVs together*

Description

As the data are split due to limits of the API, `tts_bind_wav()` allows the text and the results to be harmonized

Usage

```
tts_bind_wav(result, same_sample_rate = TRUE)
```

Arguments

`result`	A `data.frame` from `tts()`.
`same_sample_rate`	A logical value indicating whether to force the same sample rate.

Value

A `data.frame` with the same structure as that of `tts`

Examples

```
## Not run:
# Same sample rate
tts_bind_wav(res, same_sample_rate = TRUE)

# Different sample rate
tts_bind_wav(res, same_sample_rate = FALSE)


## End(Not run)
```

tts_default_voice *Default voice for text-to-speech engine*

Description

Default voice for text-to-speech engine

Usage

```
tts_default_voice(service = c("amazon", "google", "microsoft", "coqui"))
```

Arguments

service	Text-to-speech engine

tts_speak_engine *Speak Engine for* knitr

Description

Speak Engine for knitr

Usage

```
tts_speak_engine(options)
```

Arguments

options	A list of chunk options. Usually this is just the object options passed to the engine function; see knit_engines

Value

A character string generated from the source code and output using the appropriate output hooks.

Examples

```
## Not run:
knitr::knit_engines$set(speak = tts_speak_engine)
options = list(
code = "hey let's go to the park",
eval = FALSE,
label = "random",
fig.path = tempdir(),
echo = TRUE, results = "asis",
engine = "speak")
tts_speak_engine(options)
```

```
    if (tts_auth("google")) {
       options$eval = TRUE
       tts_speak_engine(options)
    }

## End(Not run)
```

`tts_voices` *Text-to-Speech (Speech Synthesis) Voices*

Description

Various services offer a range of voice options:

- Amazon Polly : https://docs.aws.amazon.com/polly/latest/dg/voicelist.html
- Microsoft Cognitive Services Text to Speech REST API : https://learn.microsoft.com/en-us/azure/cognitive-services/speech-service/language-support?tabs=tts#voice-styles-and-roles
- Google Cloud Text-to-Speech API : https://cloud.google.com/text-to-speech/docs/voices
- Coqui TTS : https://huggingface.co/spaces/coqui/CoquiTTS

Usage

```
tts_voices(service = c("amazon", "google", "microsoft", "coqui"), ...)

tts_amazon_voices(...)

tts_microsoft_voices(region = "westus")

tts_google_voices(...)

tts_coqui_voices()
```

Arguments

`service`	Service to use (Amazon, Google, Microsoft, or Coqui)
`...`	Additional arguments to service voice listings.
`region`	(Microsoft only) Region of your Microsoft Speech Service API Key

Value

(Amazon, Microsoft, and Google) A standardized `data.frame` featuring the following columns:

- `voice` : Name of the voice
- `language` : Spoken language
- `language_code` : Abbreviation for the language of the speaker

- gender : Male or female
- service : The text-to-speech engine used

(Coqui TTS) A tibble featuring the following columns:

- language : Spoken language
- dataset : Dataset the deep learning model was trained on
- model_name : Name of deep learning model
- service : The text-to-speech engine used

Examples

```
# Amazon Polly
if (tts_auth(service = "amazon")) {
tts_voices(service = "amazon")
}


# Microsoft Cognitive Services Text to Speech REST API
if (tts_auth(service = "microsoft")) {
tts_voices(service = "microsoft")
}

# Google Cloud Text-to-Speech API
if (tts_auth(service = "google")) {
tts_voices(service = "google")
}

# Coqui TTS
if (tts_auth(service = "coqui")) {
tts_voices(service = "coqui")
}
```

Index