Package 'stt.api'

June 19, 2026

Title 'OpenAI' Compatible Speech-to-Text API Client

Version 0.3.0

Description A minimal-dependency R client for 'OpenAI'-compatible speech-to-text APIs (see <https://platform.openai.com/docs/api-reference/audio>) with optional local fallbacks. Supports 'OpenAI', local servers, and the 'whisper' package for local transcription.

License MIT + file LICENSE

Encoding UTF-8

URL https://github.com/cornball-ai/stt.api

BugReports https://github.com/cornball-ai/stt.api/issues

Imports curl, jsonlite

Suggests tinytest, whisper

NeedsCompilation no

Author Troy Hernandez [aut, cre] (ORCID: <https://orcid.org/0009-0005-4248-604X>), Cornball AI [cph]

Maintainer Troy Hernandez <troy@cornball.ai>

Repository CRAN

Date/Publication 2026-06-19 19:40:06 UTC

Contents

clear_native_whisper_cache *Clear native whisper model cache*

Description

Removes cached native whisper models from memory. Call this to free GPU/RAM after batch processing is complete.

Usage

```
clear_native_whisper_cache()
```

Value

No return value, called for side effects (frees memory by removing cached models and triggers garbage collection).

Examples

```
clear_native_whisper_cache()
```

set_stt_base *Set the API Base URL*

Description

Sets the base URL for OpenAI-compatible STT endpoints.

Usage

```
set_stt_base(url)
```

Arguments

url	Character string. The base URL (e.g., "http://localhost:4123" or "https://api.openai.com").

Value

Invisibly returns the previous value.

Examples

```
set_stt_base("http://localhost:4123")
getOption("stt.api_base")
```

set_stt_key *Set the API Key*

Description

Sets the API key for hosted STT services (e.g., OpenAI). Local servers typically ignore this.

Usage

```
set_stt_key(key)
```

Arguments

key	Character string. The API key.

Value

Invisibly returns the previous value.

Examples

```
set_stt_key("test-key-123")
getOption("stt.api_key")
```

stt *Speech to Text*

Description

Convert an audio file to text using a local whisper backend or an OpenAI-compatible API.

Usage

```
stt(file, model = NULL, language = NULL,
    response_format = c("json", "text", "verbose_json"),
    backend = c("auto", "whisper", "openai"),
    source = c("auto", "api", "package"), prompt = NULL)
```

Arguments

file	Path to the audio file to convert.
model	Model name to use for transcription. For API backends, this is passed directly (e.g., "whisper-1"). For whisper, this is the model size (e.g., "tiny", "base", "small", "medium", "large"). If NULL, uses the backend's default.
language	Language code (e.g., "en", "es", "fr"). Optional hint to improve transcription accuracy.
response_format	Response format for API backend. One of "text", "json", or "verbose_json". Ignored for whisper backend.
backend	Which engine to use: "auto" (default), "whisper", or "openai". Auto mode tries whisper first, then the openai API (if configured). See `source` for *where* the engine runs.
source	Where the engine runs: "auto" (default), "api" for an HTTP service (OpenAI, or a self-hosted whisper server; see `set_stt_base`), or "package" for the in-process whisper R package. "auto" runs whisper in-process and openai via the API, matching the previous behavior. Use `backend = "whisper", source = "api"` to reach a whisper `serve()` endpoint.
prompt	Optional text to guide the transcription. For API backend, this is passed as initial_prompt to help with spelling of names, acronyms, or domain-specific terms. Ignored for whisper backend.

Value

A list with components:

text The transcribed text as a single string.

segments A data.frame of segments with timing info, or NULL.

words A data.frame of word-level timestamps (word, start, end), present only when the API returns word granularity (verbose_json); otherwise absent.

language The detected or specified language code.

backend The legacy execution route ("api" or "whisper"). This reports *where* the engine ran, not the engine itself; the resolved `backend/source` pair lives in the `"call_record"` attribute.

raw The raw response from the backend.

The result also carries a `"call_record"` attribute (cornball_sidecar v1, as in xtx.api/tts.api): the resolved request, elapsed seconds, and a timestamp – provenance that rides with the transcription when callers serialize it.

Examples

```
## Not run:
# Using OpenAI API
set_stt_base("https://api.openai.com")
set_stt_key(Sys.getenv("OPENAI_API_KEY"))
result <- stt("speech.wav", model = "whisper-1")
```

```
result$text

# Using a self-hosted whisper serve() endpoint
set_stt_base("http://troy-g5:7809")
result <- stt("speech.wav", backend = "whisper", source = "api")

# In-process whisper package
result <- stt("speech.wav", backend = "whisper", source = "package")

## End(Not run)
```

`stt_health` *Check STT Backend Health*

Description

Checks whether a transcription backend is available and working.

Usage

```
stt_health()
```

Value

A list with components:

ok Logical. TRUE if a backend is available.

backend Character. The available backend ("api" or "whisper"), or NULL if none available.

message Character. Status message with details.

Examples

```
## Not run:
h <- stt_health()
if (h$ok) {
  message("STT ready via ", h$backend)
}

## End(Not run)
```

Index