

Package ‘n3d’

October 14, 2026

Title Speaker Diarization with 'Nemotron 3 Diarization' in R 'torch'

Version 0.1.0

Date 2026-10-01

Description Determines ``who spoke when" in recorded or streaming audio with a native R 'torch' implementation of the 'NVIDIA' 'Nemotron 3 Diarization' model <<https://huggingface.co/nvidia/Nemotron-3-Diarization>>, a Streaming 'Sortformer' that tracks up to eight speakers in order of first arrival. Log-mel feature extraction, the transformer encoder and the arrival-order speaker cache are implemented in R; weights are downloaded from 'Hugging Face' with the 'hfhub' package on request.

License Apache License (>= 2)

Copyright see inst/COPYRIGHTS

Depends R (>= 4.4.0)

Imports torch (>= 0.17.0), safetensors, hfhub, av, utils

Suggests stats, tinytest

URL <https://github.com/cornball-ai/n3d>

BugReports <https://github.com/cornball-ai/n3d/issues>

Encoding UTF-8

NeedsCompilation no

Author Troy Hernandez [aut, cre] (ORCID:
<<https://orcid.org/0009-0005-4248-604X>>),
cornball.ai [cph],
The HuggingFace Team [cph] (portions ported from the transformers
library (Apache-2.0); see inst/COPYRIGHTS)

Maintainer Troy Hernandez <troy@cornball.ai>

Repository CRAN

Date/Publication 2026-10-14 12:22:07 UTC

Contents

diarize	2
download_n3d	3
load_audio	4
load_n3d	4
n3d_exists	5
n3d_revision	6
n3d_stream	6
n3d_stream_finish	7
n3d_stream_push	8
probs_to_segments	9
Index	10

diarize	<i>Speaker Diarization</i>
---------	----------------------------

Description

Finds who spoke when in a recording. Speakers are numbered from 1 in order of first arrival; up to eight are tracked.

Usage

```
diarize(  
  audio,  
  model = NULL,  
  threshold = 0.5,  
  probs = FALSE,  
  sample_rate = 16000L  
)
```

Arguments

audio	Path to an audio or video file, or a numeric vector of 16 kHz mono samples.
model	A model from load_n3d . Loaded on first use when NULL.
threshold	Speaker probability above which a frame counts as speech.
probs	Whether to also return the per-frame probabilities.
sample_rate	Sampling rate of numeric audio; must be 16000.

Details

The recording is processed offline in 27.2 s chunks with 3.2 s of look-ahead, carrying the arrival-order speaker cache between chunks, so its length is not limited.

Value

A data frame of segments with columns `start`, `end` (seconds) and `speaker`. With `probs = TRUE`, a list of segments and `probs`, a frames x 8 matrix with one frame per 10 ms.

Examples

```
if (n3d_exists()) {
  model <- load_n3d()
  mp3 <- system.file("samples", "Synapsis-Wonderland.mp3", package = "av")
  diarize(mp3, model)
}
```

download_n3d

Download the Model Weights

Description

Downloads the Nemotron 3 Diarization weights (about 400 MB) into the Hugging Face cache with **hfhub**. In interactive sessions it asks for consent first. The weights are published by NVIDIA under the OpenMDW License 1.1.

Usage

```
download_n3d(revision = n3d_revision(), force = FALSE)
```

Arguments

<code>revision</code>	Hugging Face commit to download.
<code>force</code>	Download again even when the weights are cached.

Value

The path of the cached `model.safetensors`, invisibly.

Examples

```
if (interactive()) {
  download_n3d()
}
```

load_audio

Load Audio

Description

Decodes an audio or video file with **av**, downmixed to mono and resampled to 16 kHz.

Usage

```
load_audio(file)
```

Arguments

file Path to an audio or video file in any format FFmpeg reads.

Value

Numeric vector of samples in $[-1, 1]$.

Examples

```
mp3 <- system.file("samples", "Synopsis-Wonderland.mp3", package = "av")
samples <- load_audio(mp3)
length(samples) / 16000 # seconds
```

load_n3d

Load the Diarization Model

Description

Builds the network and loads the Nemotron 3 Diarization weights from the Hugging Face cache.

Usage

```
load_n3d(
  device = default_device(),
  download = FALSE,
  revision = n3d_revision()
)
```

Arguments

device Torch device, "cuda" when available, otherwise "cpu".

download Whether to download the weights when they are not cached. Defaults to FALSE; see [download_n3d](#).

revision Hugging Face commit to load. Defaults to the commit this package was validated against.

Details

On CUDA each encoder layer runs as a traced TorchScript graph, which frees intermediate tensors as it goes (peak device memory about 0.45 GiB instead of about 2 GiB) and removes most per-operation R overhead. Tracing and warming it up take about a second, once per model. Set `options(n3d.jit = FALSE)` to run the plain R functions instead. The CPU always runs them.

Value

An `n3d_model` object: a list holding the network and its configuration.

Examples

```
if (n3d_exists()) {  
  model <- load_n3d(device = "cpu")  
}
```

n3d_exists

Check Whether the Weights Are Cached

Description

Check Whether the Weights Are Cached

Usage

```
n3d_exists(revision = n3d_revision())
```

Arguments

revision Hugging Face commit to look for.

Value

TRUE if the model weights are in the Hugging Face cache.

Examples

```
n3d_exists()
```

n3d_revision	<i>Model Revision</i>
--------------	-----------------------

Description

The Hugging Face commit of nvidia/Nemotron-3-Diarization that [load_n3d](#) and [download_n3d](#) use by default.

Usage

```
n3d_revision()
```

Value

A 40-character commit hash.

Examples

```
n3d_revision()
```

n3d_stream	<i>Streaming Diarization</i>
------------	------------------------------

Description

Opens a streaming session: push audio as it arrives and get speaker probabilities for each completed chunk. The speaker cache keeps speaker numbering consistent across the whole stream.

Usage

```
n3d_stream(model, mode = "low_latency")
```

Arguments

- | | |
|-------|---|
| model | A model from load_n3d . |
| mode | Streaming mode: "low_latency", "very_low_latency" or "ultra_low_latency". |

Details

Latency is the audio a chunk waits for, its frames plus look-ahead: 1.04 s for "low_latency", 0.64 s for "very_low_latency" and 0.32 s for "ultra_low_latency", excluding compute time.

Every step re-encodes the speaker cache and FIFO queue (about 540 encoder frames) to score one chunk, so keeping up with live audio needs a GPU; on a CPU a step takes seconds.

Value

An n3d_stream session.

See Also

[n3d_stream_push](#), [n3d_stream_finish](#)

Examples

```
if (n3d_exists()) {
  model <- load_n3d()
  mp3 <- system.file("samples", "Synopsis-Wonderland.mp3", package = "av")
  samples <- load_audio(mp3)
  s <- n3d_stream(model)
  probs <- list()
  for (start in seq(1, length(samples), by = 8000)) {
    piece <- samples[start:min(start + 7999, length(samples))]
    probs[[length(probs) + 1]] <- n3d_stream_push(s, piece)
  }
  probs[[length(probs) + 1]] <- n3d_stream_finish(s)
  probs_to_segments(do.call(rbind, probs))
}
```

n3d_stream_finish	<i>Finish a Stream</i>
-------------------	------------------------

Description

Scores the remaining buffered audio, including the look-ahead frames of the previous chunk, and closes the session.

Usage

```
n3d_stream_finish(stream)
```

Arguments

stream A session from [n3d_stream](#).

Value

A matrix of speaker probabilities for the remaining frames.

See Also

[n3d_stream](#) for a complete example.

Examples

```
if (n3d_exists()) {
  s <- n3d_stream(load_n3d())
  n3d_stream_push(s, numeric(32000))
  dim(n3d_stream_finish(s))
}
```

n3d_stream_push	<i>Push Audio to a Stream</i>
-----------------	-------------------------------

Description

Push Audio to a Stream

Usage

```
n3d_stream_push(stream, samples)
```

Arguments

stream	A session from n3d_stream .
samples	Numeric vector of 16 kHz mono samples.

Value

A matrix of speaker probabilities, frames x 8, for the frames completed by this push (possibly zero rows), one frame per 10 ms.

See Also

[n3d_stream](#) for a complete example.

Examples

```
if (n3d_exists()) {
  s <- n3d_stream(load_n3d())
  p <- n3d_stream_push(s, numeric(32000)) # 2 s of silence
  dim(p)
}
```

probs_to_segments	<i>Speaker Probabilities to Segments</i>
-------------------	--

Description

Thresholds per-frame speaker probabilities and returns one row per contiguous run of activity. Overlapping speech gives overlapping segments.

Usage

```
probs_to_segments(probs, threshold = 0.5, frame_duration = 0.01)
```

Arguments

probs	Numeric matrix, frames x speakers, of speaker activity probabilities (one frame per 10 ms), as returned by diarize with probs = TRUE or by n3d_stream_push .
threshold	Probability above which a frame counts as speech.
frame_duration	Seconds per frame.

Value

A data frame with columns start and end (seconds) and speaker (integer, numbered from 1 in order of first arrival), sorted by start time then speaker.

Examples

```
p <- matrix(0, 300, 8)
p[1:120, 1] <- 0.9
p[100:300, 2] <- 0.8
probs_to_segments(p)
```

Index

diarize, [2](#), [9](#)

download_n3d, [3](#), [4](#), [6](#)

load_audio, [4](#)

load_n3d, [2](#), [4](#), [6](#)

n3d_exists, [5](#)

n3d_revision, [6](#)

n3d_stream, [6](#), [7](#), [8](#)

n3d_stream_finish, [7](#), [7](#)

n3d_stream_push, [7](#), [8](#), [9](#)

probs_to_segments, [9](#)