Experiments in generating audiobook files with epub2tts-edge

John C Nash

2024-10-19

Initial version 2024-07-29, revised 2024-10-19

Background

Audiobooks have become a popular way to hear rather than read the content of books, in addition to being an important resource for the visually impaired. Unfortunately, creating good-quality audiobook files is not easy. Written content has many features of typography, layout, illustration and idiosynchratic expression that do not carry over to the spoken word. Moreover, most readers, even professional actors, have difficulty maintaining their enunciation and providing expressiveness over the generally extended duration of a book. Furthermore, the services of professional actors and readers are expensive relative to the resources of the visually impaired and amateur authors.

The development of large language models for converting text to speech (i.e. TTS) has reached the stage where it could be considered a possibility for carrying out the task of generating audiobooks from textual form, for example, epub files. This article is a set of notes on some experiments with freely available software to do precisely this.

Popularly such software is called "artificial intelligence", but "large language models" more correctly describes the programs. In particular, the software used in developing this report comes from the Github project **epub2tts-edge** of Christopher Aedo. See https://github.com/aedocw/epub2tts-edge. Aedo has done a great service in this and the earlier **epub2tts** project which is based on the Coqui AI TTS tools (https://github.com/coqui-ai/TTS). Moreover, he has responded quickly and helpfully to queries.

There are undoubtedly other tools that I could have used, but experimentation is very time consuming, and preliminary experiments showed promise, first with **epub2tts**, then somewhat better results with **epub2tts-edge**. This led to more, and more ambitious, experiments, as well as some investigations into ways to improve outputs or make them more usable to potential listeners.

Source works

I have generally put my creative writing on the UK writers' site *https://obooko.com.* This currently is twelve novels and two collections of memoir/short story material. A feature of several of my works is that they involve more than one language. My first novel, **Thursday Afternoon**, for example, has a scene where a young Canadian airforceman is working with some British soldiers and a Flemish woman to find and clear a landmine in 1944 Flanders. The Flemish woman communicates with the Canadian in French, but falls back into Flemish on occasions. There are thus three languages mixed in a couple of pages of text, which is a challenge to the software. Moreover, I have yet to discover how to inform the software of the language to be read, though I feel certain there must be a way to do this.

Installation of the software

Setup followed instructions in the github repositories to install the software for the two projects. Note that this may involve a considerable download of code and binaries depending on the existing software installation. I worked with Linux Mint. One limitation for me is a lack of CUDA capable graphical processing units

(GPUs), so performance of the tools on my computers was somewhat lacklustre. Typically rendering of an epub to an m4b or mp3 took over an hour.

Note that the `epub2tts-edge` software currently calls for Python support using at least version 3.11 or later. The audio rendering also needs an active internet connection.

The software also was developed for a Linux environment. While there is the possibility in Windows of running the Windows Subsystem for Linux, I have chosen to use Oracle VirtualBox to install Linux Mint 22 (Wilma). Indeed, because my working platform is an earlier Linux Mint and I do not wish to disturb some stable facilities, I have installed LM22 under my existing version. My writing colleague and coordinator of the Stittsville Creative Writers Group, R J Partington, installed LM22 under Windows. This has the advantage of isolating the text to speech work from other ongoing activities.

Standard usage of the software

We assume we have followed instructions in the Github repository to install the software.

epub2tts-edge is intended to be applied to an **epub** file. (https://www.w3.org/publishing/epub3/, https://en.wikipedia.org/wiki/EPUB). Such files are packaged collections of HTML text and metadata. Suppose that we have a file **mybook.epub**. Then we first convert this file to a properly structured plain text file **mybook.txt** by opening a terminal (command line screen) and issuing the command

```
        epub2tts-edge mybook.epub
```

This takes only a few seconds to execute. This creates a file mybook.txt and extracts a cover image mybook.png from the epub file. The text file begins with two lines of the form

```
Title: Dodging the Potholes
Author: John C. Nash
```

It also has some lines that commence with a hash symbol (#) that give "chapters" or "sections" and the text behind the hash is read out in the output audio as a chapter heading. However, the algorithm by which these lines are created is not clear to me, but they can be artificially inserted into the file, or if generated ones like

```
# Part 24
```

they can be edited to a more desirable phrase. Text following the hash is read as the title of the section. Below we discuss how to break up the work and have separate audio files, which also serves to allow for correction of individual parts.

Failure of the epub to text process

We have observed some cases where `epub2tts-edge` fails to carry out the conversion to text and extraction of the cover image. There are two other approaches we have used.

1) **calibre** program `ebook-convert`

This tool is part of the **calibre** ebook management software. We simply use the command

```
ebook-convert myfile.epub myfile.txt
```

Generally `myfile.txt` will be more or less correctly formatted, though we have noticed that the initial "Title:" and "Author:" may be combined in one line. This can, of course, be corrected with a text editor.

2) **pandoc**

```
pandoc -f epub -t plain -o myfile.txt myfile.epub
```

will create a text version of the epub. Unfortunately, it often breaks paragraphs into multiple lines which cause undesirable pauses in the eventual audio file(s). Moreover, the section delimiters ("# (some text)") are often missing. This is the difficult workaround when all else fails.

Conversion to audio file(s)

The much longer process is converting the text file to an audio file of type m4b (https://www.lifewire.com/m4b-file-2621958). For a novel, this may take of the order of an hour if the work is in a single computer text file. Breaking up the text file into parts is therefore helpful for repairing glitches in individual sections, and my experience is that these glitches are inevitable.

The second stage run of **epub2tts-edge** generally requires us to specify a "voice" or "speaker". If no speaker is given, a neutral English-only speaker with a quite moderate American accent is used, which can be specified by the identifier **en-US-AndrewNeural**. Moreover, for multilingual texts we need to use one of a few special multilingual voices. For example, the specifier **en-US-AndrewMultilingualNeural** will use the same voice as the default **for English**, but process multiple languages using **different** speaker voices for the other languages, which is disconcerting for the listener. Note that it seems that different "speaker" specifications will handle the same text differently and some words pronounced correctly by one voice will be mangled by another.

The second stage command can also specify a cover image which can be in jpg or png format. The command is

```
epub2tts-edge --speaker speakerID --cover mybook.png  mybook.txt
```

where speakerID is a valid speaker identifier and mybook.png (or some other png or jpg image file) is a cover image available in the same directory as mybook.txt. The output will be a file mybook-speakerID.m4b. This can be a very large file of several hundred megabytes for a full novel, and take 8-16 hours to play. The m4b format allows bookmarks so the user can pause and restart. On the other hand, older devices may not be able to play these files. They are, however, convertible to other audio file formats, in particular mp3.

Note that line breaks in the text file mybook.txt cause quite long pauses in the voice reading. Therefore line breaks that are non-pausing should be removed, giving full paragraphs on a single "line".

Some important general problems

Omitted text

WARNING: I have discovered that text in an epub file that is in a monospaced (i.e., typewriter-like) font is somehow eliminated completely from the file mybook.txt and must be re-inserted manually. This is a nuisance and can cause grief. For example, in my novels, I use such fonts for "letters" and other correspondence or quotations such as the text on a tombstone. I have had some email correspondence with Christopher Aedo about this, but so far we have not found the source of the glitch. (?? Note 2024-11-01: This may have been fixed and needs a test.)

It is feasible to replace the first stage run of epub2tts-edge with a different translator from epub to text as we have noted above. Some other choices are noted here.

- I could not get the software **epub2txt2** (https://github.com/kevinboone/epub2txt2) to compile (and hence was unable to install it).
- **epub2txt** is a Python program (https://github.com/ffreemt/epub2txt) that can be easily installed with the command `pip install epub2txt` if Python is properly set up. This seems to get all the material, but gives no spaces between what would be "paragraphs". Moreover, verbatim material has line endings, so some attention to that detail may be relevant.
- **pandoc**, as noted above, does not keep paragraphs together as a single "line". (See https://unix.stackexchange.com/questions/647686/convert-epub-to-txt-and-preserve-original-formatting). `pandoc` leaves two line endings between "paragraphs" but has line endings within paragraphs to make the text easy to display in a text editor that does not wrap lines. It should be noted that epub2tts-edge then pauses for each line ending, so these single line endings need to be removed. However, there are some parts of the epub such as the title and cover pages where we would like to retain the single line endings (or double them to ensure separation of important fields that should be read separately). Note that

omitting the "-t plain" option leaves some tags within the resulting text file that could cause unwanted spoken material.

- The `ebook-convert` program from the **calibre** software, as noted above, works reasonably well. It should be the first alternative to `epub2tts-edge` when that program fails to convert the epub successfully.

Lost features

I have not been able to get strikeout, bold or italic to be expressed in the audio output.

Emphasis, by way of capitalization, is also lost, as can be seen by processing the file **cap4emphasis.txt** through epub2tts-edge. Note line endings have been removed.

```
# Capitals for emphasis

Putting words in capitals to emphasize them will only get them pronounced letter by letter in some case

"I said," often has awkward pause or mis-emphasis
```

Difficulties with language change

When the language of the text changes, particularly with dialogue, there can be unwelcome results:

- Sometimes the language model, which appears to be linked into the speaker voice chosen, either fails to recognize the language change, so continues in the current language setting inappropriately, or else will continue after the "new" language in that language when it is not correct.
- The voice of the speaker may change drastically in the new language, which is disconcerting for the reader.

I have not (at 2024-8-5) yet found work-arounds for these issues.

Workarounds

Some glitches in pronunciation can be overcome by use of phonetic text. Here are a number of examples, all of which are small files that can be tested using

```
  epub2tts-edge [file].txt
```

where `[file]` is replaced by the appropriate identifier. Note that in some cases the long lines needed by epub2tts-edge give display problems for the Rmarkdown editor in R-studio (https://www.rstudio.com/categories/rstudio-ide/) and the resulting pdf output.

0pauses.txt

```
0pauses.txt

Various things cause pauses.

Line endings.

Like
these
ones.

A period will give a break. Just like that. But shorter than line ending.

But dashes -- seem to be ignored.
```

```
What about a semicolon; does that work? More or less. Perhaps dashes are same.
```

titles.txt

```
# titles.txt

Lt. Cmdr. Try full words "Lieutenant Commander" or phonetically Leftenant Commander

Putting M I 9 together gets MI9.  Better separated.

Ms. (not sounding like Miz). Try Miz.

Mrs. could be better MISSUS or Missus. Note that epub2tts-edge seems to get this to work OK.

We can also try versus and vs.

Monsieur Lefebvre not M. Lefebvre

POWs  ("s" not pronounced sometimes). Try P O W's.
```

titles.txt

```
# names.txt

Vaudreuil try  vohdroy
A / C or Ae slash C for air conditioning or alternating current. Spell it out.
Arnprior Arnpryor Arnprye-ore
Belisle  Belleel
Jean Godin use zjonn gohdann
Lebreton Lebretton
Levis try Levee
Long Sault  (pronunciation)  Long Soo
saltpetre  saltpeter
Paule try Pohl

Almonte use Almont
arboretum
Aylmer
Boucherville probably OK
Dieppe
Domenico Nacci or Nachi
Erminia
escapees
escheat
Frieda
Greenock -- seems OK
I-81
Jocelyn
kinesiology and phys. ed. or phonetic fiz ed
Leica
Maniac
Model A Ford
Myrna pronounced strangely by female voice default of epub2tts
Ontario
Rideau pronounced Rydoh in epub2tts
```

```
Rimouski
Tremblay mispronounced  try T R O M B L A Y  Tromblay
V I M Y or Vimy pronounced Veemy or Vy me? By using V I M M Y we get Vimmy.
```

homonymns.txt

```
# homonyms.txt

"read" sometimes reed, not red

bough of a tree, bow of a ship, a ribbon in a bow.

bowed down

learned vs learnt

row -- pronounced as rou not roe

lead pronounced as "leed" (led for weight)
```

numberpunctuation.txt

```
#numberpunctuation.txt

Numbers, units and punctuation give difficulties.

2/3 versus two thirds

.2 or point 2 is just 2 with pause. Not Point 2

Brackets (), [], and {} are not handled well

web addresses are often mangled

Lower case M M may be millimeters. M M M is mmm.
35mm vs 35 mm
No. 2  not number 2
cwt. versus hundredweight
```

other.txt

```
# other.txt

en-US-AndrewNeural speaker handles these correctly
escapees
steno for stenographer
`unmaking` not pronounced correctly in epub2tts
```

Dialect

Sometimes we want to write dialogue with the speaker using local pronunciation. Particular cases may be the dropping of the letter H by some working-class British speakers. This can give problems, and need phonetic replacement.

```
# dialect.txt
```

```
I'll 'ave the recording of Ave Maria.

I'll avv the recording of Ave Maria.
```

Vocalizations

Vocalizations like "oh", "er", "hmmph", "hmmm" sometimes get mangled, though epub2tts-edge with the en-US-AndrewNeural voice seems to work reasonably well.

```
# vocalizations.txt

epub2tts-edge does not seem to know how to properly enunciate some vocalizations common in human speech

The surprise expressive "oh" is often mis-said. "Oh, I really must fix that."

Disdain, as with "hmmph". Hesitation as with -- er -- and -- hmmm -- or realization with "ah" are poten
```

Audio post-processing

It is sometimes necessary to post-process the audio files.

Conversion to mp3

Since many older devices cannot play m4b files, but are intended for mp3 files, we may need to convert. Unfortunately, most such devices do NOT allow the user to save the position of a paused play. Therefore it is useful to have an audiobook in mp3 form in multiple smaller files. One way to do this is to build the audiobook in pieces, and this is the recommended approach. Nevertheless, **epub2tts-edge** produces files as m4b files, and these need conversion. I do this with **ffmpeg** and batch process them with a small (Linux / bash) command-line script.

```
#!/bin/bash
#
# convm4b
mkdir newfiles
for f in *.m4b; do ffmpeg -i "$f" -codec:v copy -codec:a libmp3lame -q:a 2 newfiles/"${f%.m4b}.mp3"; do
```

Open question What is Windows / Mac equivalent?

Breaking a large mp3 file into pieces

I have also had to break up a large file (converted from the m4b of an entire novel). For this I used the **audacity** software which is available on most common platforms. This is a graphical user interface program allowing for sections of an audio file to be cut and pasted and exported in various formats, including mp3.

Concatenating audio files

Sometimes the mp3 files are really too small, so the user needs to keep loading them. In that case we want to concatenate several files. **ffmpeg** can do this using a text file containing a list of the files to be processed in the order they appear. An example command is

```
ffmpeg -f concat -i mylist.txt -c copy output.wav
```

which will produce a file of type wav.

File mylist.txt must have lines of the form

```
file 'myfile.type'
```

where type is one of acceptable types to ffmpeg, which does not include m4b unfortunately, even though we used ffmpeg above to do conversions.

Another tool is **mp3wrap**, which uses a command

```
mp3wrap [options]  OUTPUTFILE  f1.mp3  f2.mp3  [f3.mp3]
```

Unfortunately, there is a limit to the number of files that can be included, but the number does not appear to be documented.

I was unable to get the program **mp3cat** from https://github.com/dmulholl/mp3cat to work at all.

Changing speed or tempo of audio

audacity allows the speed or tempo (pitch may change with one of these) to be altered. For some texts and some "speakers" (i.e., voices) a slowdown of approximately 8% made the material easier to comprehend.

Special Tools

To process the main source text for the second-stage of epub2tts-edge, I have written several small scripts to perform particular tasks. I have written these in **R** or in the Linux **bash** script, but those choices were purely for convenience. The tasks are relatively straightforward, but tedious to carry out without the automation.

Section numbering

Most of my books do not have regular chapters, but do have sections. These are typically, but not always, delimited by "* * *". I use a text editor and replace these (and any other suitable breakpoints) with

```
# Part ??
```

An R script numpart.R (see Appendices) is then used (after editing filename specifications within the script) to auto-replace the "??" with suitable numbers. Note that we want these numbers to all be the same length for processing to audio files, so leading zeros are strongly recommended in the numbering, and the script prepares such numbers.

The section dividers provided by epub2tts-edge can also be edited to provide useful output. I have used lines like

```
# The title of this book
```

or

```
# Capsule comment
```

with the latter for the "teaser" comment for the book.

Section splitting

It is useful to have individual files of text for each section of a book to convert to audiobook form. Such files are smaller and more quickly processed, so errors or glitches can be repaired and the files re-processed. File **splitnovel.R** in the Appendices does this.

Notes:

- we may wish to consolidate several sections after splitting so the audio playback files are of a reasonable size. This can be done with script **combpart.R** in the Appendices. Titles for the consolidated sections could be titled withing the text source for conversion by adding title lines in the form

```
# Module A of My Excellent Book
```

- there are issues with sections that are named versus 'Part 043' style, in that mixing these can be awkward. A suggestion is to use a style

```
# Part 043 -- The plot thickens
```

A recipe

Extracting the cover and the text

epub2tts-edge aims to do this in the first pass using a command like

```
epub2tts-edge mybook.epub
```

which should give the cover image in file mybook.png and "text" in file mybook.txt. The first lines of mybook.txt, which can be copied to a file mybookheader.txt, are something like

```
Title: My fabulous book
Author: Me A. Writer
```

Moreover, the software will try to create "parts" denoted with the hash (#) symbol followed by a title, but this will depend to a large extent on the epub content structure.

Unfortunately, as we have noted, **epub2tts-edge**, using the Python library **Beautiful Soup** (according to Christopher Aedo), fails to extract any material in a monospaced font.

As an alternative process, we can use **pandoc** as follows:

- A command like the following will extract text

```
pandoc -f epub -t plain -o mybookp.txt mybook.epub
```

This file will NOT have headers for parts, nor will it have the header of the type mybookheader.txt mentioned above, but the header can be added by using a text editor. Similarly, whatever divisions are present can be edited to the form `# Part ??`, or such divisions can be inserted. Blank lines above and below the division are recommended.

- The program **epub-cover** can be used to extract the cover image. This is part of the Ruby package epub-parser from https://gitlab.com/KitaitiMakoto/epub-parser.

```
epub-cover mybook.epub
```

will output whatever cover file is embedded in the epub, which may have a name somewhat different from the epub title, and may be a JPEG file. Note that one may choose to use epub2tts-edge to get mybook.png and then ignore the text file produced in case it is missing material that was in monospaced font.

Cleanup of pandoc text output

The text file produced by pandoc does not have the mybookheader.txt. Moreover, its paragraphs are NOT separated suitably for the second (main) stage of epub2tts-edge.

- Add newlines after unseparated lines as appropriate, particularly in title page(s), where we do not want paragraph run-on when we process the mybookp.txt to consolidate paragraphs.

Section dividers

- For my own books, with `* * *` section separator, convert "* * *" to "# Part ??" throughout. This can be done with most text editors.
- Check that there is a line ending for the final line of the mybook.txt file.
- Add section dividers for the title page(s), the capsule comment and any preamble. Optionally expand or diminish the section divisions.

Avoiding some awkward pauses or mispronunciations

In the text file mybook.txt.

- Remove periods after initials in author name(s) to avoid unnecessary pause.
- Remove the copyright symbol (©), if necessary expanding to the word "copyright".
- Put spaces in email addresses e.g. `j.smith@isp.org` becomes `j . s m i t h @ i s p . o r g`. Similarly expand abbreviations and postal codes so they are not read as if they are words.
- Expand province or state abbreviations.

Consolidate paragraphs

- Run the program **novelparas.R** to ensure text paragraphs are consolidated for the audio output. Even if the text has been extracted with epub2tts-edge, there may be manually inserted monospace text that requires paragraph consolidation to remove unwanted line endings.

Phonetic corrections

Proper names, dialect, homonyms and similar issues for text to speech should be corrected.

- We need a set of source / replacement choices. We shall suppose this is in the form of lines

```
old (tab)  new
```

This assumes 1-1 choices. We will manually deal with situations where there is more than one choice using a text editor. A prototype script to process the substitutions, offering the user a choice to accept the suggested change (there are situations where we do NOT wish to take the suggestion), is **wordsub.R**. This script uses a named file **phoneticsub.txt** which has entries described above. To save computing time, it is worthwhile having a "master" file, call is **phoneticsub.full.txt** with all possible replacements. This file can be copied to phoneticsub.txt and unnecessary entry lines simply removed.

Section numbering

Run **numpart.R** to get sections numbered.

Section splitting

Run **splitnovel.R** to divide the source text into individual sections. This is helpful later when or if we discover that the conversion of text to speech has made an error or glitch. We can then attempt to correct the issue for just the single section. Once the text is adjusted appropriately, we can then re-work the section, if appropriate first inserting it into the consolidated module where it belongs.

Section consolidation

To provide suitably sized audio files, we consolidate the sections into modules using **combpart.R**.

Conversion to speech

The script **runedge.R** is intended to process the text modules (or sections) and produce files of type m4b. Note that this script is currently set up to be first edited to provide the right filename root for the modules and the correct cover image.

This stage of the process takes quite a long time – typically of the order of an hour.

(Optional) Adjust cover image

The program **AtomicParsley** (https://github.com/wez/atomicparsley) can be used to insert a different cover image into m4b (and other audio) files. One syntax is

```
cp myfile.m4b tmp.m4b
AtomicParsley tmp.m4b --artwork REMOVE_ALL
AtomicParsley tmp.m4b --artwork cover.jpg -o tmpx.m4b
mv tmpx.m4b myfile.m4b
rm tmp.m4b
```

Convert audio files to desired format

While m4b files are the current recommendation for audiobooks, many player devices do not handle them. Therefore, we may wish to convert the files. The Linux **bash** script **convm4b.sh** will handle this nicely, but there are many tools available in the common operating systems. (I welcome suggestions.)

(Optionally) Adjust audio file characteristics

The program **audacity** is useful to change speed, tempo or pitch of audio files. Some text-to-speech models create output that is very quick. Slower and lower-pitched output may be helpful to listeners.

Open needs

- A fix for the first stage of epub2tts-edge so that monospaced font material is included. An alternative would be tools to work with other epub to text translation tools such as **pandoc** or **epub2txt** mentioned above that also establish the paragraph structure without line endings. At the moment a text editor is used.
- Until and unless the large language models can cope with various proper names, homonymns and dialect, tools to streamline the replacement of selected words with phonetic substitutes will be welcome. It would be helpful to have a GUI editor that highlights target words and offers replacements. Such tools already exist for spelling checkers, so this is not an idle possibility.
- epub2tts-edge needs an active internet connection to function. There are a number of situations where it would be important to be able to prepare audio output from text WITHOUT the need for a connection to the Microsoft service. Moreover, it is by no means certain that this service will be continued, or continued on the same basis as at present.

Special cases

An attempt to convert the epub of novel “Moonbeam Shadows” (file: Moonbeam.epub) to text with

epub2tts-edge Moonbeam.epub

failed with the messages as follows.

```
Namespace(sourcefile='Moonbeam.epub', speaker='en-US-AndrewNeural', cover=None)
Cover image saved to Moonbeam.png
Exporting Moonbeam.epub to Moonbeam.txt
Traceback (most recent call last):
  File "/home/john/.local/bin/epub2tts-edge", line 8, in <module>
    sys.exit(main())
  File "/home/john/.local/lib/python3.10/site-packages/epub2tts_edge/epub2tts_edge.py", line 388, in mai
    export(book, args.sourcefile)
  File "/home/john/.local/lib/python3.10/site-packages/epub2tts_edge/epub2tts_edge.py", line 113, in exp
    author = book.get_metadata("DC", "creator")[0][0]
IndexError: list index out of range
```

```
john@M21:~/Videos/moonbeam$
```

Similarly, **Dodging Potholes** gave the same error. There are likely others.

Future goals and needs

1) need a script to run pandoc to text

pandoc -f epub -t plain -o filename.txt filename.epub

2) Need a script to run all the parts
    - copy epub to shortened name
    - build profile to use for inputs
    - modify scripts to work together
    - timers and log files
    - testing and reporting.

Conclusions

This article underlines the reality that we are still in the early stages of large-scale text to speech, at least in the open-source world.

- We see a large number of small but important "glitches" in the text-to-speech engines. While not mentioned much in the article, the Coqui AI TTS was tried, but found wanting.
- Workarounds have been found for almost all the glitches discovered. These have, however, requires quite a lot of work.
- Pronunciation and emphasis issues still are found.
- The lack of a way to tell the TTS engine what language a segment of text is using is a serious fault and causes a number of issues. It would also be useful to be able to specify emphasized or softer speech, or to specify the "speaker" voice for novels with more than one narrator, such as https://www.obooko.com/free-historical-fiction-books/dodging-the-potholes .
- Some books require multiple voices. This is relatively easy to arrange if the text blocks are broken up, but tags to allow the "speaker" to be changed would be helpful.
- The **edge** TTS engine require the computer processing text to be internet- connected. This is a serious obstacle for some applications. It also suggests that users may be vulnerable to being held hostage by Microsoft should they suddenly impose a charging model on the parts of edge-TTS now open.

Appendix 1: epub2tts-edge voices

?? How to get a list

How to get audio samples

The program **get-edge-tts-samples.py** shows how to get examples of audio for different edge-tts speakers.

```
#!/usr/bin/env python3

"""
Basic example of edge_tts usage.
"""

import asyncio
import edge_tts
```

```
NAMES = ['en-US-AnaNeural', 'en-US-AndrewNeural', 'en-US-AriaNeural', 'en-US-AvaNeural', \
 'en-US-BrianNeural', 'en-US-ChristopherNeural', 'en-US-EmmaNeural', 'en-US-EricNeural', \
 'en-US-GuyNeural', 'en-US-JennyNeural', 'en-US-MichelleNeural', 'en-US-RogerNeural', \
 'en-US-SteffanNeural', 'de-DE-FlorianMultilingualNeural', 'de-DE-SeraphinaMultilingualNeural',\
 'en-US-AndrewMultilingualNeural', 'en-US-AvaMultilingualNeural', 'en-US-BrianMultilingualNeural',\
 'en-US-EmmaMultilingualNeural', 'fr-FR-RemyMultilingualNeural', 'fr-FR-VivienneMultilingualNeural']

TEXT = "This is a longer string that I am sending to text to speech, and using the python module directl

async def amain() -> None:
    """Main function"""
    for name in NAMES:
        output = f"sample-{name}.wav"
        print(f"Saving {output}")
        communicate = edge_tts.Communicate(TEXT, name)
        await communicate.save(output)


if __name__ == "__main__":
    loop = asyncio.get_event_loop_policy().get_event_loop()
    try:
        loop.run_until_complete(amain())
    finally:
        loop.close()
```

There are a number of lists of edge-tts voices. One is at https://github.com/rany2/edge-tts/blob/master/RE
ADME.md

Appendix A: Possible troubles

The following text MAY still give problems and need to be checked

```
# Oddbods.txt

Nene -- This is an early Rolls Royce jet engine using a centrifugal complressor of the
         Whittle style. It was used in the De Havilland Vampire and a few other early
         jet aircraft.  Better to use Neen to get the right pronunciation.
Sterling currency expression 17/6 is better written 17 and 6.
ha'penny  as hah penny, not haypenny
"wan" --> W A N
Marthe -- more like Martha than Mart
Feltham  "ham"
French mis-pronounced. "687 25th Avenue"
Paule, Lise (more like Liss)
souper
"Sacrément!"
dommage
compris
cintres
"C’est un très grand réfrigérateur."
tarte au sucre
erable
Houle
Decarie
Ste Catherine
```

```
Outrement
Ultramontes
Universit'e
maudit
Anges
Sinterklaas
Bolduc
Pieds nus dans l'aube
Sault Ste Marie,   Regina
Metis
Mon Dieu
V-1 may be better as V 1
```

Appendix B: Software tools associated with this article

convm4b.sh – convert m4b files to mp3

```
#!/bin/bash
#
# convm4b
mkdir newfiles
for f in *.m4b; do ffmpeg -i "$f" -codec:v copy -codec:a libmp3lame -q:a 2 newfiles/"${f%.m4b}.mp3"; do
```

combpart.R – combine individual book sections

The purpose of this script is to create texts that will have a reasonable audio file duration, particularly when in mp3 form, which is more difficult in most cases to position or resume playing. The number of parts to combine is set at `np <- 10` in the present version.

```
Warning in readLines("combpart.R"): incomplete final line found on 'combpart.R'
```

```
# combpart.R
# Combine parts of text source for audiobooks
ifnroot <- readline("input file root:")
np <- 10 # can change to number of parts to combine

pat<-paste("^",ifnroot,"...\\.txt",sep='')
ll <- list.files(pattern=pat)
ll <- sort(ll)
print(ll)
tmp<-readline("cont.")
nl <- length(ll)

f0<-ll[1]
con<-file(f0, open="r")
tt<-readLines(con)
close(con)
hd<-tt[1:3]

sufx<-c("A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N",
          "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z")

fcount<-1
ofn<- paste(ifnroot,"cc",sufx[fcount],".txt",sep='')
cat("Initial ofn:", ofn,"\n")
ocon<-file(ofn, open="wt")
```

```
writeLines(hd, ocon) # puts in header


for (i in 1:nl) {
  cat("Input file ", ll[i],"\n")
  con<-file(ll[i], open="r")
  tt<-readLines(con)
  nt<-length(tt)
  otxt<-tt[4:nt]
  close(con)
  writeLines(otxt, ocon)
  if (i >= nl) { # finished
     flush(ocon)
     close(ocon)
     break
  } else { # check if need new output file
    if ( (i %% np) == 0 ) { # New output file?
      flush(ocon)
      close(ocon)
      fcount <- fcount + 1 # next output file
      ofn<- paste(ifnroot,"cc",sufx[fcount],".txt",sep='')
      cat("New ofn:", ofn,"\n")
      ocon<-file(ofn, open="wt")
      writeLines(hd, ocon) # puts in header
    }
  }
} # end loop over files
cat("Done!\n")
```

novelparas.R – remove single line endings to get paragraphs

epub2tts-edge will pause on line endings. Paragraphs are therefore assumed to be terminated by a line ending.

```
# novelparas.R
library(stringr)
# Consolidate paragraph lines in novel audiobook source texts
ifn<-readline("Input file name(text):")
con<-file(ifn, open="r")
ofn<-readline("Output file nam (txt):")
ocon<-file(ofn, open="w")
y<-readLines(con)
close(con)
ny<-length(y)
bb<-which(y=="")
nbb<-length(bb)
nbb<-nbb+1
ny<-length(y)+1
bb[nbb] <- ny # beyond end of file
inpos<-0
lb <- 1 # initial first line in para
for (ib in 1:nbb){ # loop over paras
  tpara <- ""
  ub <- bb[ib]-1 # last line in para
  for (ll in lb:ub){
      inpos<-ll
      lyn<-y[inpos]
      tpara <- paste(tpara,lyn," ",sep='')
      # May want to remove extra white space afterwards
   }
   # should get rid of terminating and extra whitespace in tpara
   tpara<-str_squish(tpara)
#   allp[ib]<-tpara
   writeLines(tpara, ocon)
   writeLines("", ocon) # Just in case
   lb <- ub + 1 # to reset for next para
} # end loop over paras
flush(ocon)
close(ocon)
```

numpart.R – number sections that have generic divider naming

This script is intended to convert dividers `# Part ??` into numbered dividers. It requires generic dividers of the form `* * *` to be converted to `# Part ??` with a text editor.

```
# numpart.R
# converts book that has '*  *  *' sections changed to '# Part ??'
# ?? needs testing

ifn <- readline("Input filename:")
pn <- 0
pat <- "# Part ??"

con=file(ifn, open="r")
ofn <- readline("Output file (text):")
ofcon <- file(ofn, open="w")

while ( TRUE ) {
  line = readLines(con, n = 1)
  if (length(line)==0) {
    # close(con)
    break
  }
  if ( substr(line, 1, 9) == pat) {
     pn <- pn + 1
     pnc <- as.character(pn)
     while(nchar(pnc) < 3) {pnc<-paste("0",pnc,sep='')}
# convert number to text with leading zeros so section names sort.
# Suggest 3 character form, as books with more than 999 sections too long.
     oline <- paste("# Part ",pnc,sep='')
     writeLines(oline, ofcon)
  } else { writeLines(line, ofcon) }
}
close(con)
flush(ofcon)
close(ofcon)
```

splitnovel.R – divide book text file into individual sections with headers

```
# splitnovel.R
# Split novel into sections
## ?? possibly not finishedifn <- readline("input file:")
ifn <- readline("Input file name:")

ofnroot <- readline("Output file root:")

pat <- "# Part "
npat<-nchar(pat)
con=file(ifn,open="r")

tt <- readLines(con)
ntt<- length(tt)
close(con)
hd<-tt[c(1,3,4)]

plines <- which(substr(tt, 1, npat)==pat)
npl <- length(plines)
plines<-c(plines, ntt+1) # add an upper bound at end

lb <- 5 # initial start line
sn <- 0
snc <- as.character(sn)
while(nchar(snc) < 3) {snc<-paste("0",snc,sep='')}
if (snc != "000") stop("Bad number")
ofn <- paste(ofnroot,snc,".txt",sep='')
ofcon <- file(ofn, open="w")
ub <- plines[1] - 1
tfile<-tt[lb:ub]
writeLines(hd, ofcon)
writeLines(tfile, ofcon)
writeLines("", ofcon)
flush(ofcon); close(ofcon)

lb<-ub+1
for (ii in 1:npl){
   ub<-plines[ii+1] - 1
   cat(lb," ", ub,"\n")
   sn<-sn+1
   snc <- as.character(sn)
   while(nchar(snc) < 3) {snc<-paste("0",snc,sep='')}
   tfile<-tt[lb:ub]
   ofn <- paste(ofnroot,snc,".txt",sep='')
   ofcon <- file(ofn, open="w")
   writeLines(hd, ofcon)
   writeLines(tfile, ofcon)
   writeLines("", ofcon)
   flush(ofcon); close(ofcon)
   lb<-ub+1 # update
}
```

Appendix C: List of voices usable by epub2tts-edge

Multilingual voices

Name: de-DE-FlorianMultilingualNeural Gender: Male

Name: de-DE-SeraphinaMultilingualNeural Gender: Female

Name: en-US-AndrewMultilingualNeural Gender: Male

Name: en-US-AvaMultilingualNeural Gender: Female

Name: en-US-BrianMultilingualNeural Gender: Male

Name: en-US-EmmaMultilingualNeural Gender: Female

Name: fr-FR-RemyMultilingualNeural Gender: Male

Name: fr-FR-VivienneMultilingualNeural Gender: Female

Single language voices

Name: af-ZA-AdriNeural Gender: Female

Name: af-ZA-WillemNeural Gender: Male

Name: am-ET-AmehaNeural Gender: Male

Name: am-ET-MekdesNeural Gender: Female

Name: ar-AE-FatimaNeural Gender: Female

Name: ar-AE-HamdanNeural Gender: Male

Name: ar-BH-AliNeural Gender: Male

Name: ar-BH-LailaNeural Gender: Female

Name: ar-DZ-AminaNeural Gender: Female

Name: ar-DZ-IsmaelNeural Gender: Male

Name: ar-EG-SalmaNeural Gender: Female

Name: ar-EG-ShakirNeural Gender: Male

Name: ar-IQ-BasselNeural Gender: Male

Name: ar-IQ-RanaNeural Gender: Female

Name: ar-JO-SanaNeural Gender: Female

Name: ar-JO-TaimNeural Gender: Male

Name: ar-KW-FahedNeural Gender: Male

Name: ar-KW-NouraNeural Gender: Female

Name: ar-LB-LaylaNeural Gender: Female

Name: ar-LB-RamiNeural Gender: Male

Name: ar-LY-ImanNeural Gender: Female

Name: ar-LY-OmarNeural Gender: Male

Name: ar-MA-JamalNeural Gender: Male

Name: ar-MA-MounaNeural Gender: Female

Name: ar-OM-AbdullahNeural Gender: Male

Name: ar-OM-AyshaNeural Gender: Female

Name: ar-QA-AmalNeural Gender: Female

Name: ar-QA-MoazNeural Gender: Male

Name: ar-SA-HamedNeural Gender: Male

Name: ar-SA-ZariyahNeural Gender: Female

Name: ar-SY-AmanyNeural Gender: Female

Name: ar-SY-LaithNeural Gender: Male

Name: ar-TN-HediNeural Gender: Male

Name: ar-TN-ReemNeural Gender: Female

Name: ar-YE-MaryamNeural Gender: Female

Name: ar-YE-SalehNeural Gender: Male

Name: az-AZ-BabekNeural Gender: Male

Name: az-AZ-BanuNeural Gender: Female

Name: bg-BG-BorislavNeural Gender: Male

Name: bg-BG-KalinaNeural Gender: Female

Name: bn-BD-NabanitaNeural Gender: Female

Name: bn-BD-PradeepNeural Gender: Male

Name: bn-IN-BashkarNeural Gender: Male

Name: bn-IN-TanishaaNeural Gender: Female

Name: bs-BA-GoranNeural Gender: Male

Name: bs-BA-VesnaNeural Gender: Female

Name: ca-ES-EnricNeural Gender: Male

Name: ca-ES-JoanaNeural Gender: Female

Name: cs-CZ-AntoninNeural Gender: Male

Name: cs-CZ-VlastaNeural Gender: Female

Name: cy-GB-AledNeural Gender: Male

Name: cy-GB-NiaNeural Gender: Female

Name: da-DK-ChristelNeural Gender: Female

Name: da-DK-JeppeNeural Gender: Male

Name: de-AT-IngridNeural Gender: Female

Name: de-AT-JonasNeural Gender: Male

Name: de-CH-JanNeural Gender: Male

Name: de-CH-LeniNeural Gender: Female

Name: de-DE-AmalaNeural Gender: Female

Name: de-DE-ConradNeural Gender: Male

Name: de-DE-FlorianMultilingualNeural Gender: Male

Name: de-DE-KatjaNeural Gender: Female

Name: de-DE-KillianNeural Gender: Male

Name: de-DE-SeraphinaMultilingualNeural Gender: Female

Name: el-GR-AthinaNeural Gender: Female

Name: el-GR-NestorasNeural Gender: Male

Name: en-AU-NatashaNeural Gender: Female

Name: en-AU-WilliamNeural Gender: Male

Name: en-CA-ClaraNeural Gender: Female

Name: en-CA-LiamNeural Gender: Male

Name: en-GB-LibbyNeural Gender: Female

Name: en-GB-MaisieNeural Gender: Female

Name: en-GB-RyanNeural Gender: Male

Name: en-GB-SoniaNeural Gender: Female

Name: en-GB-ThomasNeural Gender: Male

Name: en-HK-SamNeural Gender: Male

Name: en-HK-YanNeural Gender: Female

Name: en-IE-ConnorNeural Gender: Male

Name: en-IE-EmilyNeural Gender: Female

Name: en-IN-NeerjaExpressiveNeural Gender: Female

Name: en-IN-NeerjaNeural Gender: Female

Name: en-IN-PrabhatNeural Gender: Male

Name: en-KE-AsiliaNeural Gender: Female

Name: en-KE-ChilembaNeural Gender: Male

Name: en-NG-AbeoNeural Gender: Male

Name: en-NG-EzinneNeural Gender: Female

Name: en-NZ-MitchellNeural Gender: Male

Name: en-NZ-MollyNeural Gender: Female

Name: en-PH-JamesNeural Gender: Male

Name: en-PH-RosaNeural Gender: Female

Name: en-SG-LunaNeural Gender: Female

Name: en-SG-WayneNeural Gender: Male

Name: en-TZ-ElimuNeural Gender: Male

Name: en-TZ-ImaniNeural Gender: Female

Name: en-US-AnaNeural Gender: Female

Name: en-US-AndrewMultilingualNeural Gender: Male

Name: en-US-AndrewNeural Gender: Male

Name: en-US-AriaNeural Gender: Female

Name: en-US-AvaMultilingualNeural Gender: Female

Name: en-US-AvaNeural Gender: Female

Name: en-US-BrianMultilingualNeural Gender: Male

Name: en-US-BrianNeural Gender: Male

Name: en-US-ChristopherNeural Gender: Male

Name: en-US-EmmaMultilingualNeural Gender: Female

Name: en-US-EmmaNeural Gender: Female

Name: en-US-EricNeural Gender: Male

Name: en-US-GuyNeural Gender: Male

Name: en-US-JennyNeural Gender: Female

Name: en-US-MichelleNeural Gender: Female

Name: en-US-RogerNeural Gender: Male

Name: en-US-SteffanNeural Gender: Male

Name: en-ZA-LeahNeural Gender: Female

Name: en-ZA-LukeNeural Gender: Male

Name: es-AR-ElenaNeural Gender: Female

Name: es-AR-TomasNeural Gender: Male

Name: es-BO-MarceloNeural Gender: Male

Name: es-BO-SofiaNeural Gender: Female

Name: es-CL-CatalinaNeural Gender: Female

Name: es-CL-LorenzoNeural Gender: Male

Name: es-CO-GonzaloNeural Gender: Male

Name: es-CO-SalomeNeural Gender: Female

Name: es-CR-JuanNeural Gender: Male

Name: es-CR-MariaNeural Gender: Female

Name: es-CU-BelkysNeural Gender: Female

Name: es-CU-ManuelNeural Gender: Male

Name: es-DO-EmilioNeural Gender: Male

Name: es-DO-RamonaNeural Gender: Female

Name: es-EC-AndreaNeural Gender: Female

Name: es-EC-LuisNeural Gender: Male

Name: es-ES-AlvaroNeural Gender: Male

Name: es-ES-ElviraNeural Gender: Female

Name: es-ES-XimenaNeural Gender: Female

Name: es-GQ-JavierNeural Gender: Male

Name: es-GQ-TeresaNeural Gender: Female

Name: es-GT-AndresNeural Gender: Male

Name: es-GT-MartaNeural Gender: Female

Name: es-HN-CarlosNeural Gender: Male

Name: es-HN-KarlaNeural Gender: Female

Name: es-MX-DaliaNeural Gender: Female

Name: es-MX-JorgeNeural Gender: Male

Name: es-NI-FedericoNeural Gender: Male

Name: es-NI-YolandaNeural Gender: Female

Name: es-PA-MargaritaNeural Gender: Female

Name: es-PA-RobertoNeural Gender: Male

Name: es-PE-AlexNeural Gender: Male

Name: es-PE-CamilaNeural Gender: Female

Name: es-PR-KarinaNeural Gender: Female

Name: es-PR-VictorNeural Gender: Male

Name: es-PY-MarioNeural Gender: Male

Name: es-PY-TaniaNeural Gender: Female

Name: es-SV-LorenaNeural Gender: Female

Name: es-SV-RodrigoNeural Gender: Male

Name: es-US-AlonsoNeural Gender: Male

Name: es-US-PalomaNeural Gender: Female

Name: es-UY-MateoNeural Gender: Male

Name: es-UY-ValentinaNeural Gender: Female

Name: es-VE-PaolaNeural Gender: Female

Name: es-VE-SebastianNeural Gender: Male

Name: et-EE-AnuNeural Gender: Female

Name: et-EE-KertNeural Gender: Male

Name: fa-IR-DilaraNeural Gender: Female

Name: fa-IR-FaridNeural Gender: Male

Name: fi-FI-HarriNeural Gender: Male

Name: fi-FI-NooraNeural Gender: Female

Name: fil-PH-AngeloNeural Gender: Male

Name: fil-PH-BlessicaNeural Gender: Female

Name: fr-BE-CharlineNeural Gender: Female

Name: fr-BE-GerardNeural Gender: Male

Name: fr-CA-AntoineNeural Gender: Male

Name: fr-CA-JeanNeural Gender: Male

Name: fr-CA-SylvieNeural Gender: Female

Name: fr-CA-ThierryNeural Gender: Male

Name: fr-CH-ArianeNeural Gender: Female

Name: fr-CH-FabriceNeural Gender: Male

Name: fr-FR-DeniseNeural Gender: Female

Name: fr-FR-EloiseNeural Gender: Female

Name: fr-FR-HenriNeural Gender: Male

Name: fr-FR-RemyMultilingualNeural Gender: Male

Name: fr-FR-VivienneMultilingualNeural Gender: Female

Name: ga-IE-ColmNeural Gender: Male

Name: ga-IE-OrlaNeural Gender: Female

Name: gl-ES-RoiNeural Gender: Male

Name: gl-ES-SabelaNeural Gender: Female

Name: gu-IN-DhwaniNeural Gender: Female

Name: gu-IN-NiranjanNeural Gender: Male

Name: he-IL-AvriNeural Gender: Male

Name: he-IL-HilaNeural Gender: Female

Name: hi-IN-MadhurNeural Gender: Male

Name: hi-IN-SwaraNeural Gender: Female

Name: hr-HR-GabrijelaNeural Gender: Female

Name: hr-HR-SreckoNeural Gender: Male

Name: hu-HU-NoemiNeural Gender: Female

Name: hu-HU-TamasNeural Gender: Male

Name: id-ID-ArdiNeural Gender: Male

Name: id-ID-GadisNeural Gender: Female

Name: is-IS-GudrunNeural Gender: Female

Name: is-IS-GunnarNeural Gender: Male

Name: it-IT-DiegoNeural Gender: Male

Name: it-IT-ElsaNeural Gender: Female

Name: it-IT-GiuseppeNeural Gender: Male

Name: it-IT-IsabellaNeural Gender: Female

Name: ja-JP-KeitaNeural Gender: Male

Name: ja-JP-NanamiNeural Gender: Female

Name: jv-ID-DimasNeural Gender: Male

Name: jv-ID-SitiNeural Gender: Female

Name: ka-GE-EkaNeural Gender: Female

Name: ka-GE-GiorgiNeural Gender: Male

Name: kk-KZ-AigulNeural Gender: Female

Name: kk-KZ-DauletNeural Gender: Male

Name: km-KH-PisethNeural Gender: Male

Name: km-KH-SreymomNeural Gender: Female

Name: kn-IN-GaganNeural Gender: Male

Name: kn-IN-SapnaNeural Gender: Female

Name: ko-KR-HyunsuNeural Gender: Male

Name: ko-KR-InJoonNeural Gender: Male

Name: ko-KR-SunHiNeural Gender: Female

Name: lo-LA-ChanthavongNeural Gender: Male

Name: lo-LA-KeomanyNeural Gender: Female

Name: lt-LT-LeonasNeural Gender: Male

Name: lt-LT-OnaNeural Gender: Female

Name: lv-LV-EveritaNeural Gender: Female

Name: lv-LV-NilsNeural Gender: Male

Name: mk-MK-AleksandarNeural Gender: Male

Name: mk-MK-MarijaNeural Gender: Female

Name: ml-IN-MidhunNeural Gender: Male

Name: ml-IN-SobhanaNeural Gender: Female

Name: mn-MN-BataaNeural Gender: Male

Name: mn-MN-YesuiNeural Gender: Female

Name: mr-IN-AarohiNeural Gender: Female

Name: mr-IN-ManoharNeural Gender: Male

Name: ms-MY-OsmanNeural Gender: Male

Name: ms-MY-YasminNeural Gender: Female

Name: mt-MT-GraceNeural Gender: Female

Name: mt-MT-JosephNeural Gender: Male

Name: my-MM-NilarNeural Gender: Female

Name: my-MM-ThihaNeural Gender: Male

Name: nb-NO-FinnNeural Gender: Male

Name: nb-NO-PernilleNeural Gender: Female

Name: ne-NP-HemkalaNeural Gender: Female

Name: ne-NP-SagarNeural Gender: Male

Name: nl-BE-ArnaudNeural Gender: Male

Name: nl-BE-DenaNeural Gender: Female

Name: nl-NL-ColetteNeural Gender: Female

Name: nl-NL-FennaNeural Gender: Female

Name: nl-NL-MaartenNeural Gender: Male

Name: pl-PL-MarekNeural Gender: Male

Name: pl-PL-ZofiaNeural Gender: Female

Name: ps-AF-GulNawazNeural Gender: Male

Name: ps-AF-LatifaNeural Gender: Female

Name: pt-BR-AntonioNeural Gender: Male

Name: pt-BR-FranciscaNeural Gender: Female

Name: pt-BR-ThalitaNeural Gender: Female

Name: pt-PT-DuarteNeural Gender: Male

Name: pt-PT-RaquelNeural Gender: Female

Name: ro-RO-AlinaNeural Gender: Female

Name: ro-RO-EmilNeural Gender: Male

Name: ru-RU-DmitryNeural Gender: Male

Name: ru-RU-SvetlanaNeural Gender: Female

Name: si-LK-SameeraNeural Gender: Male

Name: si-LK-ThiliniNeural Gender: Female

Name: sk-SK-LukasNeural Gender: Male

Name: sk-SK-ViktoriaNeural Gender: Female

Name: sl-SI-PetraNeural Gender: Female

Name: sl-SI-RokNeural Gender: Male

Name: so-SO-MuuseNeural Gender: Male

Name: so-SO-UbaxNeural Gender: Female

Name: sq-AL-AnilaNeural Gender: Female

Name: sq-AL-IlirNeural Gender: Male

Name: sr-RS-NicholasNeural Gender: Male

Name: sr-RS-SophieNeural Gender: Female

Name: su-ID-JajangNeural Gender: Male

Name: su-ID-TutiNeural Gender: Female

Name: sv-SE-MattiasNeural Gender: Male

Name: sv-SE-SofieNeural Gender: Female

Name: sw-KE-RafikiNeural Gender: Male

Name: sw-KE-ZuriNeural Gender: Female

Name: sw-TZ-DaudiNeural Gender: Male

Name: sw-TZ-RehemaNeural Gender: Female

Name: ta-IN-PallaviNeural Gender: Female

Name: ta-IN-ValluvarNeural Gender: Male

Name: ta-LK-KumarNeural Gender: Male

Name: ta-LK-SaranyaNeural Gender: Female

Name: ta-MY-KaniNeural Gender: Female

Name: ta-MY-SuryaNeural Gender: Male

Name: ta-SG-AnbuNeural Gender: Male

Name: ta-SG-VenbaNeural Gender: Female

Name: te-IN-MohanNeural Gender: Male

Name: te-IN-ShrutiNeural Gender: Female

Name: th-TH-NiwatNeural Gender: Male

Name: th-TH-PremwadeeNeural Gender: Female

Name: tr-TR-AhmetNeural Gender: Male

Name: tr-TR-EmelNeural Gender: Female

Name: uk-UA-OstapNeural Gender: Male

Name: uk-UA-PolinaNeural Gender: Female

Name: ur-IN-GulNeural Gender: Female

Name: ur-IN-SalmanNeural Gender: Male

Name: ur-PK-AsadNeural Gender: Male

Name: ur-PK-UzmaNeural Gender: Female

Name: uz-UZ-MadinaNeural Gender: Female

Name: uz-UZ-SardorNeural Gender: Male

Name: vi-VN-HoaiMyNeural Gender: Female

Name: vi-VN-NamMinhNeural Gender: Male

Name: zh-CN-XiaoxiaoNeural Gender: Female

Name: zh-CN-XiaoyiNeural Gender: Female

Name: zh-CN-YunjianNeural Gender: Male

Name: zh-CN-YunxiNeural Gender: Male

Name: zh-CN-YunxiaNeural Gender: Male

Name: zh-CN-YunyangNeural Gender: Male

Name: zh-CN-liaoning-XiaobeiNeural Gender: Female

Name: zh-CN-shaanxi-XiaoniNeural Gender: Female

Name: zh-HK-HiuGaaiNeural Gender: Female

Name: zh-HK-HiuMaanNeural Gender: Female

Name: zh-HK-WanLungNeural Gender: Male

Name: zh-TW-HsiaoChenNeural Gender: Female

Name: zh-TW-HsiaoYuNeural Gender: Female

Name: zh-TW-YunJheNeural Gender: Male

Name: zu-ZA-ThandoNeural Gender: Female

Name: zu-ZA-ThembaNeural Gender: Male