SRA の技術情報誌
GS LetterNeo
Vol.186　2024 年 1 月

Whisper を用いた文字起こし

今井 紘太

アドバンストクラウドエンジニアリング事業部

はじめに

「仕事をする上で手間がかかるけど決してなくならないものは何？」と問われた際に、いくつか回答候補が思い浮かぶかと思います。その中で今回は会議の議事録に焦点を当ててみます。一見記録するだけであれば音声データを取得しておいて、聞き直しが可能な状態にしておけば問題なさそうです。それだけであればただ会議を録音するだけでクリアすることが可能です。ただし、文字データで会議内容を記録して議事録としてまとめている現場がほとんどではないでしょうか。

文字データとして記録するメリットは、会議内で決定された方針を手早く再確認できることや、保存する際のデータ量を少なくできることです。対してデメリットは、議事録作成のためにかかる時間的コストや人的コストの高さが挙げられます。これらのデメリットの解決を目指して、今回は Whisper を用いて音声データから文字起こしの自動化に取り組んでみます。

Whisper とは

OpenAI から 2022 年 9 月から提供されている音声認識モデルです。68 万時間もの大規模で多様なデータセットで学習が行われており、英語だけでなく多様な言語に対応しています。バックグラウンドノイズにも強く、テレワーク環境で多くなってしまう環境音を排斥した状態での出力も期待されます。Whisper の詳細については Whisper 公式ページを参照してください。(参考[1])

Whisper を用意する

Whisper には、API 版と OSS 版があります。API 版を利用すると従量課金が必要となるため、今回は OSS 版の Whisper を利用します。

Whisper のパッケージをインストールします。

```
> pip install git+https://github.com/openai/whisper.git
```

API 版とは利用手順が異なるため、注意が必要です。

Whisper 公式で互換性があるとされている python バージョンは 3.8~3.11 です。今回の記事では python3.9.2 を利用します。

音声データを扱うには FFmpeg のインストールを別途行う必要があります。インストール手順については参考[2]のサイトを参考に行いました。

ファイルの指定を行います

Whisper で処理できるファイル形式には、m3a、mp3、mp4、mpeg、mpga、wav、webm などがあります。

```
import whisper

#Audio File
audio = "audio.mp3"
```

今回は参考[3]のサイトからフリーの音声をお借りして、文字起こしを試します。音声データの形式は mp3 で音声内容は以下となります。録音時間は 26 秒です。

```
本日はご来場いただきまして誠にありがとうございます。開演に先立ちましてお客様にお願い申し上げます。携帯電話など音の出るものの電源はお切りください。また許可のない録音、撮影はご遠慮ください。皆様のご協力をよろしくお願いいたします。
```

モデルの指定を行います

利用できるモデルには以下の種類があります。Parameters が増えるにつれて認識精度が向上し、実行にかかる時間が増えます。トレードオフなため、用途に合わせて使い分けする必要があります。

表 *1* モデルの比較

Size	Parameters	English-only model	Multilingual model	Required VRAM	Relative speed
tiny	39 M	tiny.en	tiny	~1 GB	~32x
base	74 M	base.en	base	~1 GB	~16x
small	244 M	small.en	small	~2 GB	~6x
medium	769 M	medium.en	medium	~5 GB	~2x
large	1550 M	N/A	large	~10 GB	1x

認識精度を優先してモデル large をまず利用してみます。

```
#Whisper Model
Model = whisper.load_model("large")
```

large の場合およそ 2.87G のダウンロードが発生します。Windows10 の場合、ダウンロードされたモデルはデフォルトでは以下のパスに配置されます。

```
C:¥Users¥<username>¥.cache¥whisper¥<model>
```

2 回目以降の実行の際はモデルのダウンロードは不要です。ダウンロード先のパス変更も可能です。

文字起こしの実行

音声ファイルを指定することで、そのファイルの文字起こしが実行されます。冒頭の 30 秒で言語の判定が自動で行われます。オプションで明示的に言語を指定することも可能です。また、verbose=Trueのオプションを設定することで発言時間を記録することもできます。

```
#Transcribe
result = model.transcribe(audio,verbose=True,language="ja")
print(result["text"])
```

実行した出力は以下のようになりました。

```
[00:00.000 --> 00:04.060] 本日はご来場いただきまして誠にありがとうございます。
[00:06.120 --> 00:10.160] 開演に先立ちましてお客様にお願い申し上げます。
[00:11.900 --> 00:15.960] 携帯電話など音の出るものの電源はお切りください。
[00:17.480 --> 00:21.540] また許可のない録音、撮影はご遠慮ください。
[00:22.700 --> 00:26.440] 皆様のご協力をよろしくお願いいたします。
```

出力されたデータに誤差はありませんでした。

出力までにかかる時間を測定したところ、およそ 72 秒でした。26 秒の音声ファイルとなるため、およそ 3 倍程度の時間がかかったようです。今回の実行環境は CPU が Intel Corei5 4 コア 2.21 GHz、GPU は未搭載です。

同じ環境で 20 分の会議音声の文字起こしを実行したところ、およそ 4600 秒かかりました。さきほどよりも相対的に時間がかかり、元ファイルの 4 倍近くの時間がかかりました。実際の会議の時間が 30~60 分のものが多いことを考えると、処理速度がもう少し速いと使い勝手が良さそうです。処理速度改善のためにより良い環境を用意するか、追加の対応を検討する必要がありそうです。

処理速度改善について

簡単な処理速度改善案は、利用するモデルを Parameters が少ないものに変更することです。medium と small のモデルを利用した際の処理速度と出力を確認してみましょう。

medium

```
[00:00.000 --> 00:04.060] 本日はご来場いただきまして誠にありがとうございます。
[00:06.120 --> 00:10.160] 開演に先立ちましてお客様にお願い申し上げます。
[00:11.900 --> 00:15.960] 携帯電話など音の出るものの電源はお切りください。
[00:17.480 --> 00:21.540] また許可のない録音、撮影はご遠慮ください。
[00:22.700 --> 00:26.440] 皆様のご協力をよろしくお願い致します。
```

処理時間は約 40 秒ほどでした。large と同様に出力誤差はありませんでした。今回利用した音声データと同等の音質のものかつ専門的な用語が出ないのであれば、モデルは medium で十分なようです。

small

```
[00:00.000 --> 00:04.060] 本日はご来場いただきまして誠にありがとうございます。
[00:06.120 --> 00:10.160] 会員に先立ちましてお客様にお願い申し上げます。
[00:11.900 --> 00:15.960] 携帯電話など音の出るものの電源はお切りください。
[00:17.480 --> 00:21.540] また許可のない録音、撮影はご遠慮ください。
[00:22.700 --> 00:26.440] 皆様のご協力をよろしくお願いいたします。
```

処理時間は約 15 秒でした。誤差があったのは「開演」が「会員」になった部分のみでした。4 分の 1 以下の時間でこの出力となるのであれば、small でも実用に耐えそうです。

Whisper では prompt に事前情報を与えることができます。専門用語や人名などを与えることで認識精度を向上させることが可能です。

今回 small で認識誤差が発生した「開演」を prompt に渡して、認識誤差がなくなるか確認してみましょう。

```
   #Transcribe
   result = model.transcribe(audio, verbose=True,language="ja",
initial_prompt="開演")
```

```
[00:00.000 --> 00:04.060] 本日はご来場いただきまして誠にありがとうございます。
[00:06.120 --> 00:10.160] 開演に先立ちましてお客様にお願い申し上げます。
[00:11.900 --> 00:15.960] 携帯電話など音の出るものの電源はお切りください。
[00:17.480 --> 00:21.540] また許可のない録音、撮影はご遠慮ください。
[00:22.700 --> 00:26.440] 皆様のご協力をよろしくお願いいたします。
```

prompt に与えた「開演」のおかげで、認識誤差がなくなりました。prompt を利用することで処理時間を優先して小さいモデルを利用した場合でもある程度は質を確保することができそうです。

おわりに

今回 Whisper によって文字起こしが自動化できることが確認できました。ただし、音声データの文字起こしの完全自動化にはまだ課題があります。そのうちの 1 つが、話者の自動判定です。話者分離と呼ばれ、音声データ内で誰がどの発言をしたかを判定する必要があります。Whisper では prompt の機能を応用することで話者分離を実現することが可能です。また、別のオープンソースフレームワークである pyannote(参考[4])を利用することでも話者分離が可能です。今後は、pyannote を用いた話者分離にも挑戦してみたいと思います。

引用文献

[1] OpenAI, 2023, “Introducing Whisper – OpenAI”, （2023 年 12 月 25 日取得, https://openai.com/research/whisper）.

[2] HXYoUGZ(Deri Soru), Qiita, 2023 年 12 月 27 日, “ffmpeg のインストールと使い方”, https://qiita.com/HXYoUGZ/items/960d9bb4e61c55ab9bbc, (2024 年 1 月 10 日取得) .

[3] 効果音ラボ, “声素材 - アプリ・音声案内(落ち着いた女性)”, (2024 年 1 月 10 日取得, https://soundeffect-lab.info/sound/voice/info-lady1.html) .

[4] Github, 2024 年 1 月 9 日, “pyannote/pyannote-audio”, (2024 年 1 月 11 日取得, https://github.com/pyannote/pyannote-audio) .

GSLetterNeo Vol.186
2024 年 1 月 20 日発行
発行者　株式会社 SRA 技術本部 先端技術研究室

編集者　熊澤努　方 学芬
バックナンバー https://www.sra.co.jp/public/sra/gsletter/
お問い合わせ　gsneo@sra.co.jp

株式会社SRA
〒171-8513 東京都豊島区南池袋 2-32-8

夢を。Yawaraka Innovation
やわらかいのべーしょん