mimi®

mimi® ASR API Service 利用ガイド

2018 beta 4

Fairy Devices

mimi® ASR HTTP API 仕様

この API は、サーバーに音声データを送信すると、音声データ中の人間の声（日本語）を、単語ごとに認識する音声認識機能を提供するバッチ API です。認識途中の結果を得ることはできません。音声認識時間処理に要する時間は、入力音声長の約半分となります。つまり、10 秒の音声を入力した場合、約 5 秒の認識時間が掛かることになります。音声ファイル等の認識処理に好適です。

マイク入力等から逐次的に認識したい場合は mimi® ASR WebSocket API（リアルタイム API）をご利用ください。

エンドポイント

mimi-trial.fd.ai

ポート

443（SSL 接続が必要です）

アクセストークン（ハッカソン専用）

f7950a3a-00c6-11e8-ba89-0ed5f89f718b

標準入力音声形式

・16kHz, 16bit, 1ch, PCM（audio/x-pcm;bit=16;rate=16000;channels=1）
・16kHz, 16bit, 1ch, FLAC（audio/x-flac;bit=16;rate=16000;channels=1）
※ただし、ひとつの音声は 60 秒以下であること

必須 HTTP リクエストヘッダー

HTTP リクエストヘッダー	値
Authorization	Bearer アクセストークン
Content-Type	入力音声形式を指定する上記の文字列

出力 JSON 形式

```
{
  "response" : [
  {"pronunciation":"ヨミガナ 1","result":"認識単語 1”,"time":[開始時刻 1,終了時刻 1]},
  {"pronunciation":"ヨミガナ 2","result":"認識単語 2”,"time":[開始時刻 2,終了時刻 2]},
    … 以下文章の終わりまで単語毎の認識結果が続く…
  ],
  "session_id":"結果のユニーク ID",
  "status":"recog-finished",
  "type":"asr#mimilvcsr"
}
```

開始時刻、終了時刻は、音声ファイルの先頭を時刻ゼロとする相対時刻で単位はミリ秒です。認識されたその単語が、音声ファイル中のどこからどこまでの区間だったかを示します。具体例は実行結果例を御覧ください。

実行例

```
$ curl https://mimi-trial.fd.ai:443
-H "Authorization: Bearer f7950a3a-00c6-11e8-ba89-0ed5f89f718b"
-H "content-type: audio/x-pcm;bit=16;rate=16000;channels=1" --data-binary @audio.raw
```

Linux / Mac OS X 環境では curl コマンドが利用できます。この例では、curl コマンドでカレントディレクトリの audio.raw という音声ファイルをサーバーに送信しています。

実行結果例

```
{
  "response" : [
  { "pronunciation" : "ローニャク", "result" : "老若", "time" : [ 300, 810 ] },
  { "pronunciation" : "ナンニョ", "result" : "男女", "time" : [ 810, 1180 ] },
  { "pronunciation" : "ガ", "result" : "が", "time" : [ 1180, 1390 ] },
  { "pronunciation" : "ヒ", "result" : "火", "time" : [ 1650, 1890 ] },
  { "pronunciation" : "ヲ", "result" : "を", "time" : [ 1890, 1960 ] },
```

```
{ "pronunciation" : "カコン", "result" : "囲ん", "time" : [ 1960, 2350 ] },
{ "pronunciation" : "デ", "result" : "で", "time" : [ 2350, 2460 ] },
{ "pronunciation" : "ノミ", "result" : "のみ", "time" : [ 2460, 2840 ] },
{ "pronunciation" : "テ", "result" : "手", "time" : [ 3100, 3250 ] },
{ "pronunciation" : "ヲ", "result" : "を", "time" : [ 3250, 3350 ] },
{ "pronunciation" : "ツナイ", "result" : "つない", "time" : [ 3350, 3700 ] },
{ "pronunciation" : "デ", "result" : "で", "time" : [ 3700, 3850 ] },
{ "pronunciation" : "ウタウ", "result" : "歌う", "time" : [ 3850, 4310 ] } ],
"session_id" : "8c0f9fdc-00da-11e8-b96e-42010af00006",
"status" : "recog-finished",
 "type" : "asr#mimilvcsr" }
```

説明のために pretty print していますが、実際の出力は pretty print されません。

より進んだ利用方法（知識のある方向け）

サーバーサイドリサンプラー有効化

標準ではサンプリングレート 16 kHz の音声信号を受け付けますが、サーバー側にリサンプラーを備えており、それを有効化することで任意のサンプリングレートの音声を受け付けることができるようになります。以下のリクエストヘッダーを用います。デフォルトでは off になっています。

HTTP リクエストヘッダー	値
x-mimi-resample	on/off

サーバーサイド VAD（発話区間抽出器）有効化

サーバー側で VAD（発話区間抽出器）を動作させ、入力音声中の人間の声の部分のみを抽出し、認識を行います。ただし、雑音が大きい音声の場合、発話区間を誤推定する場合があり、そのような場合、パラメータ調整が必要ですが、より複雑になるため、本文書では説明を省略します。

HTTP リクエストヘッダー	値
x-mimi-vad	on/off

mimi® ASR WebSocket API 仕様

この API は、サーバーに音声データを送信すると、音声データ中の人間の声（日本語）を、単語ごとに認識する音声認識機能を提供するリアルタイム API です。音声データは逐次的に送信することができ、認識の途中結果が逐次的に返されます。音声認識処理に要する時間は、音声の送信時間が実時間だった場合、最後の音声データチャンク送信後、約 0.5 秒です。つまり、マイク入力をそのまま送信した場合、その発話時間に加えて、発話終了コマンドをサーバーが受領してから約 0.5 秒で最終結果が返されます。

音声ファイルからバッチ的に入力したい場合は、より簡単に利用できる mimi® ASR HTTP API（バッチ API）をご利用ください。

エンドポイント

mimi-trial.fd.ai

ポート

443（SSL 接続が必要です）

アクセストークン（ハッカソン専用）

f7950a3a-00c6-11e8-ba89-0ed5f89f718b

標準入力音声形式

・16kHz, 16bit, 1ch, PCM（audio/x-pcm;bit=16;rate=16000;channels=1）
・16kHz, 16bit, 1ch, FLAC（audio/x-flac;bit=16;rate=16000;channels=1）

必須 HTTP リクエストヘッダー

HTTP リクエストヘッダー	値
Authorization	Bearer アクセストークン
Content-Type	入力音声形式を指定する上記の文字列

出力 JSON 形式

```
{
  "response" : [
  {"pronunciation":"ヨミガナ 1","result":"認識単語 1”,"time":[開始時刻 1,終了時刻 1]},
  {"pronunciation":"ヨミガナ 2","result":"認識単語 2”,"time":[開始時刻 2,終了時刻 2]},
    … 以下文章の終わりまで単語毎の認識結果が続く…
  ],
  "session_id":"結果のユニーク ID",
  "status":"認識の経過情報",
  "type":"asr#mimilvcsr"
}
```

開始時刻、終了時刻は、音声ファイルの先頭を時刻ゼロとする相対時刻で単位はミリ秒です。認識されたその単語が、音声ファイル中のどこからどこまでの区間だったかを示します。具体例は実行結果例を御覧ください。

認識の経過情報は、recog-in-progress もしくは recog-finished のどちらかが返されます。前者は、サーバーが発話終了信号を受信しておらず、認識が途中であるとされている状態、後者は、発話終了信号を受信し、最終結果であるとされている状態を示します。一般に、複数回の recog-in-progress の結果が返され、1 回のみの recog-finished の結果が返されます。それらの結果全てにおいて、session_id は同一となります。

通信手順

（1） WebSocket 仕様（RFC6455）に従って、エンドポイントに WebSocket 接続を開きます。このとき、必須 HTTP リクエストヘッダーの情報を合わせて与えなければならないことに留意してください。また、HTTP API で示した、より進んだオプションについても利用することが可能です。

（2） 音声データチャンクをバイナリフレームとして、当該接続に送信します。音声データチャンクサイズは任意ですが、音声データとして意味を持つこと、つまり必ず偶数バイトで送信しなければならないことに留意してください。

（3） サーバーから適時、認識の途中結果が得られます。出力結果は、HTTP API の場合と認識の経過情報を除き同一です。

（4） 音声データの送信を終える際には、テキストフレームで、音声データ送信終了命令（後述）

を送信します。

（５） サーバーから最終の認識結果が得られます。出力結果は、HTTP API の場合と同一です。コネクションは RFC6455 に従ってサーバーから切断されます。

WebSocket 接続の簡単な実行例はありません。利用するプログラム言語に応じて、適切な WebSocket ライブラリをご利用ください。出力例は省略します。

音声データ送信終了命令について

```
{
  "command": "recog-break"
}
```

テキストフレームで、上記の JSON データをサーバーに送信することで、サーバーは本命令を受信する前までに受け取った音声データで確定認識結果を返す処理に入ります。本命令を送信した後に、さらに音声データを送信することはできません。

以上