# 回答根拠確認チュートリアル

## はじめに

本書は、本サービスが提供する回答根拠確認APIによって、LLMの生成結果の回答根拠を確認するためのチュートリアルです。

本書の対象読者は以下を想定しています。

- 本サービスと連携したシステムや製品開発を行う開発者

なお、本書ではPythonのサンプルコードを記載しておりますが、本サービスのAPIはREST形式のため、他の言語からもご利用いただけます。
サポートしているAPIとリクエスト・レスポンスの詳細については、APIリファレンスを参照してください。

- 本ガイドに記載する「https://<サーバのドメイン名>/」はGenerative AI FWがインストールされているサーバのドメイン名に置き換えてAPIを実行してください。
- 本サービスではHTTPSに既定では自己証明書を使用しています。そのためAPI利用時に考慮が必要です。詳細は「スタートアップマニュアル（導入準備編）」をご確認ください。

## 回答根拠確認APIとは

 LLM が生成したテキストと、テキスト生成時に情報源としたテキストを入力とすることで、生成されたテキストが情報源のどの部分に対応するかを確認するAPIです。

回答根拠確認を使用することでLLMの要約結果の回答根拠の確認、検索対話の回答根拠の確認を行うことができます。

## チュートリアルの流れ

本チュートリアルの流れは以下です。

1. 要約結果の回答根拠確認
2. 検索対話の回答根拠確認
3. （応用）requestsライブラリを利用したpythonプログラムで回答根拠確認を利用する

## 要約結果の回答根拠確認

要約結果と要約前のテキストを入力とすることで、要約結果の回答根拠を確認することができます。

## APIの実行方法

以下のcurlコマンドを用いて、APIを呼び出します。APIの詳細は「Generative AI FW APIリファレンス」を参照してください。

```
curl -X POST https://<サーバのドメイン名>/genai-api/v1/alignment ¥
-H "Content-Type: application/json" ¥
-d "<リクエストパラメータ>" ¥
-H "Authorization: <API Key>"
```

以下はリクエストパラメータの例です。answerには要約後のテキスト、sourcesには要約前のテキストを指定します。

```
{
  "answer": "NECは、高い日本語性能を有する軽量なLLM(Large Language Model：大規模言語モデル)「cotomi」を...",
  "sources": [
    {
      "text": "NECは、高い日本語性能を有する軽量なNEC開発のLLM(Large Language Model：大規模言語モデル)..."
    }
  ]
}
```

応答がJson形式で返却されます。explainContentsには入力したテキストとそのテキストに紐づけされた回答根拠となるテキストが返却されます。

```
{"explainContents":{"answer":"NECは、高い日本語性能...", ...}}
```

## 回答根拠の見方

```
{
  "explainContents": [
    {
      "answer":"NECは、高い日本語性能を有する軽量なLLM(Large Language Model：大規模言語モデル)「cotomi」を..
      "explains": [
        {
          "text":"NECは、高い日本語性能を有する軽量なNEC開発のLLM(Large Language Model：大規模言語モデル)...
          "score":0.982276201248169
        },
        {
          "text": "具体的には、2024年春からNECが持つ業種・業務ノウハウをもとにした特化モデルを中核にお客様の
        }
      ]
    }
  ]
}
```

各パラメータの詳細は以下の通りです。

| パラメータ | | 説明 |
|---|---|---|
| answer | | LLMの回答内容。リクエストパラメータのanswerを文単位に分けた内容。 |
| explains | text | 説明文。 |

| | | |
|---|---|---|
| | score | 回答根拠の数値(0-1)。1に近いほど根拠の可能性が高い。回答根拠の前後の文の場合はなし。 |
| metadata | | リクエストパラメータのmetadataで指定した回答根拠となったソースドキュメントのメタデータ。<br><br>回答根拠となったソースドキュメントにメタデータがない場合や回答根拠がない場合はなし。 |

## 検索対話の回答根拠確認

検索対話の回答のテキストと、検索対話に使われた情報源のテキストを入力とすることで検索対話の回答根拠を確認することができます。

### APIの実行方法

以下のcurlコマンドを用いて、APIを呼び出します。APIの詳細は「Generative AI FW APIリファレンス」を参照してください。

```
curl -X POST https://<サーバのドメイン名>/genai-api/v1/alignment ¥
-H "Content-Type: application/json" ¥
-d "<リクエストパラメータ>" -H "Authorization: <API Key>"
```

以下はリクエストパラメータの例です。answerには検索対話の回答、sourcesのtextに検索対話の応答のsourceDocumentsのdocContent、metadataにsourceDocumentsのmetadataを指定します。

```
{
  "answer": "NECは、高い日本語性能を有する軽量なLLM(Large Language Model：大規模言語モデル)「cotomi」を...",
  "sources": [
    {
      "text": "「NEC Digital Platform」に「cotomi」は含まれます。",
      "metadata": {
        "url": "https:¥/¥/jpn.nec.com¥/..."
      }
    },
    {
       "text": "NECは、高い日本語性能を有する軽量なNEC開発のLLM(Large Language Model：大規模言語モデル)...",
      "metadata": {
        "url": "https:¥/¥/jpn.nec.com¥/..."
      }
    }
  ]
}
```

応答がJson形式で返却されます。

```
{explainContents:{"answer":"", ...}}
```

# requestsライブラリを利用したpythonプログラムで回答根拠確認を利用する

## requestsライブラリのインストール

インストールコマンドの例

```
pip install requests
```

## Pythonコードを実装する

Pythonコードの例

Pythonのエディタを開き、下記のコードを記述します。ファイル名を「alignment_main.py」として保存します。

```
import requests

# KEYはAPIキー
KEY = 'abcdefg1234567890'
# BASEはサービスのURL
BASE = 'https://<サーバのドメイン名>/genai-api/v1'

def func():
    url = BASE + '/alignment'
    key = 'Bearer ' + KEY
    payload = { "answer": "NECは、高い日本語性能を有する軽量なLLM(Large Language Model：大規模言語モデル)...
                "sources": [
                  { "text": "NECは、高い日本語性能を有する軽量なNEC開発のLLM(Large Language Model：..."}
                ]}
    headers = { 'content-type': 'application/json',
                'Authorization': key }
    # リクエスト
    response = requests.post(url, json=payload, headers=headers)
    print(response.status_code)
    print(response.text)
    #print(response.headers)

if __name__ == '__main__':
    func()

```

## Pythonの実行

実行コマンド

```
python alignment_main.py
```

実行結果の例

```
200
{explainContents:[{"answer":"", ...}]}
```