Neural Network Console クラウド版
スターターガイド
-API機能説明編-

ソニーネットワークコミュニケーションズ株式会社

目次

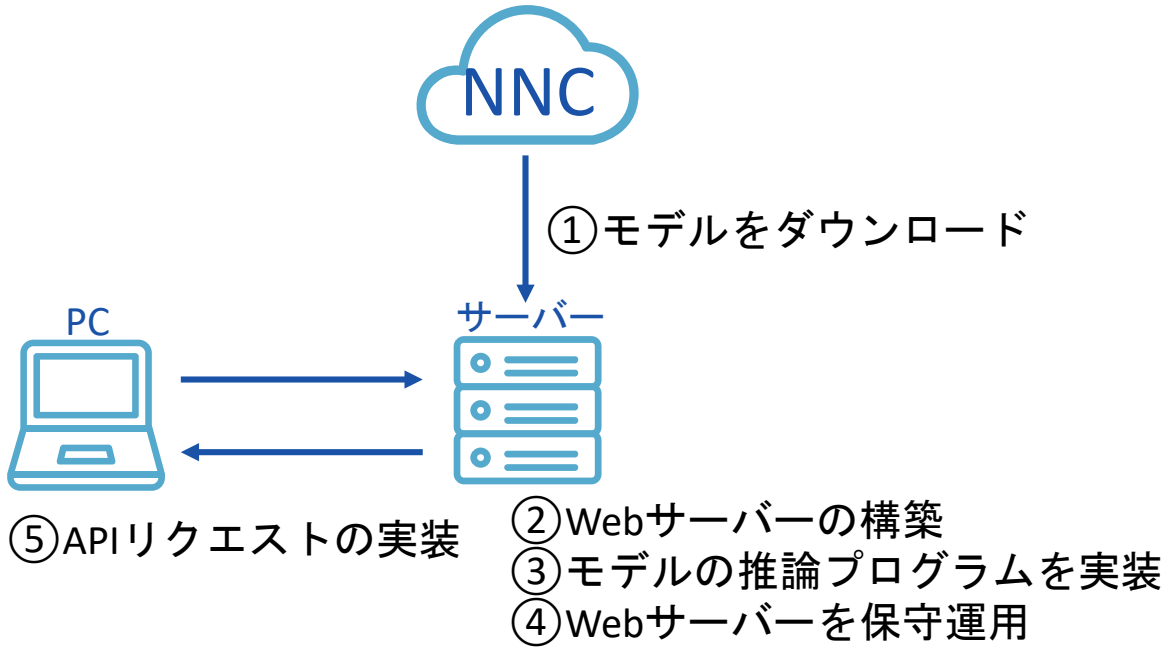
API機能とは

API機能とは作成したモデルをNNC上で運用する機能です。インターネット経由で推論データをNNCのサーバーに送信すると、NNC上で自動でモデルが実行され、推論結果が返却されます。

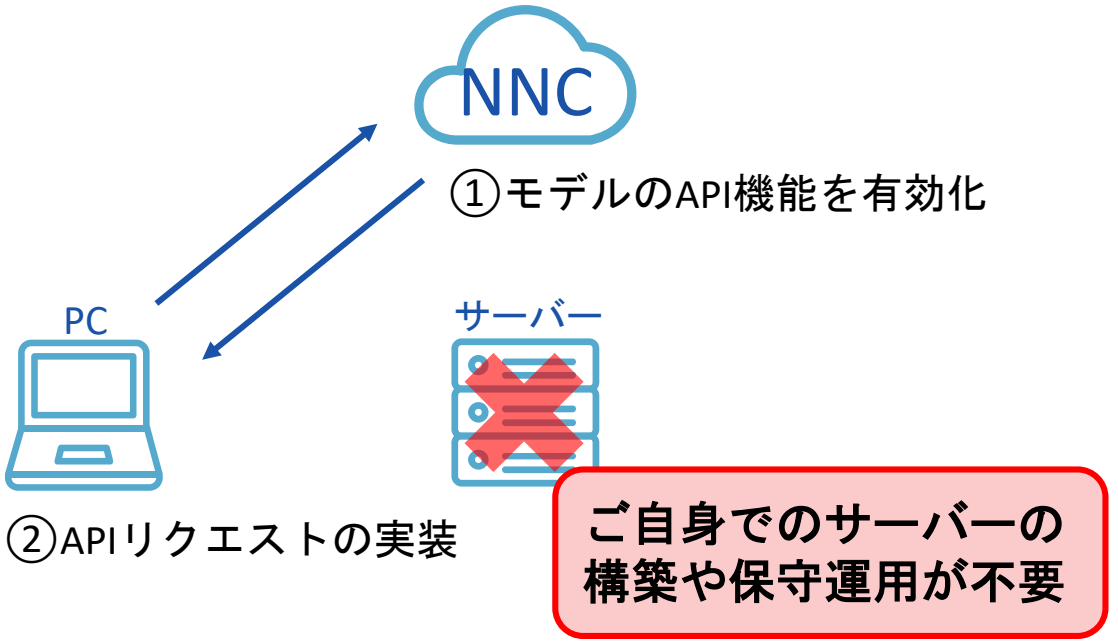
作成したモデルをダウンロードして利用することもできますが、API機能を利用することで、サーバーの構築や保守運用などが不要になり、手軽にモデル運用ができます。

モデルの運用方法

モデルをダウンロードして運用する場合



API機能を利用する場合



API機能の実行タイプ

API機能の実行タイプとして、リクエストごとに実行処理するリクエスト数課金タイプと、常にインスタンスを起動させておくインスタンス占有タイプの２種類があります。
データやモデルのサイズ、期待する処理速度などに応じて、適切なタイプを選択ください。

		リクエスト数課金タイプ	インスタンス占有タイプ
概要		• リクエストごとにCPUで実行 • Cost parameter※1が64MB未満で、30秒以内にモデル実行が終了することが条件	• CPUもしくはGPUのインスタンスを常時起動 • 事前設定により起動・停止の予約が可能
課金体系		• APIのリクエスト数課金※2 (ただし、500回までは無料利用可能)	• インスタンスの起動時間課金※2
対応データ	画像データ	対応(DICOM形式を除く)	対応
	数値データ (表や時系列など)	対応	対応
	音声データ	非対応	対応
利用シーン		• 軽量のモデルを安価に運用	• 大きなデータやモデルの運用 • 処理速度を高速に運用

※1: Cost ParameterはTrainingタブの右下に表示されるNetwork Statisticsの中で確認できます。
※2: 詳細な料金体系はウェブページをご参照ください。
有料プランの利用にはクレジットカード登録もしくは法人契約が必要になります。クレジットカード登録はユーザ設定を、法人契約はウェブサイトをご参照ください。

目次

API機能を利用するためのNNC上での操作

API機能を利用するために、NNC上でモデルのAPI機能を有効化し、APIのURLとキーを取得する必要があります。APIのURLとキーはAPIを利用する際に必要な情報になります。

	操作	概要
1	API機能の有効化	APIとして利用するモデルを選択し、そのAPI機能を有効化します
2	APIのURLとキーの取得	外部からAPIを利用するためのURLとキーを取得します

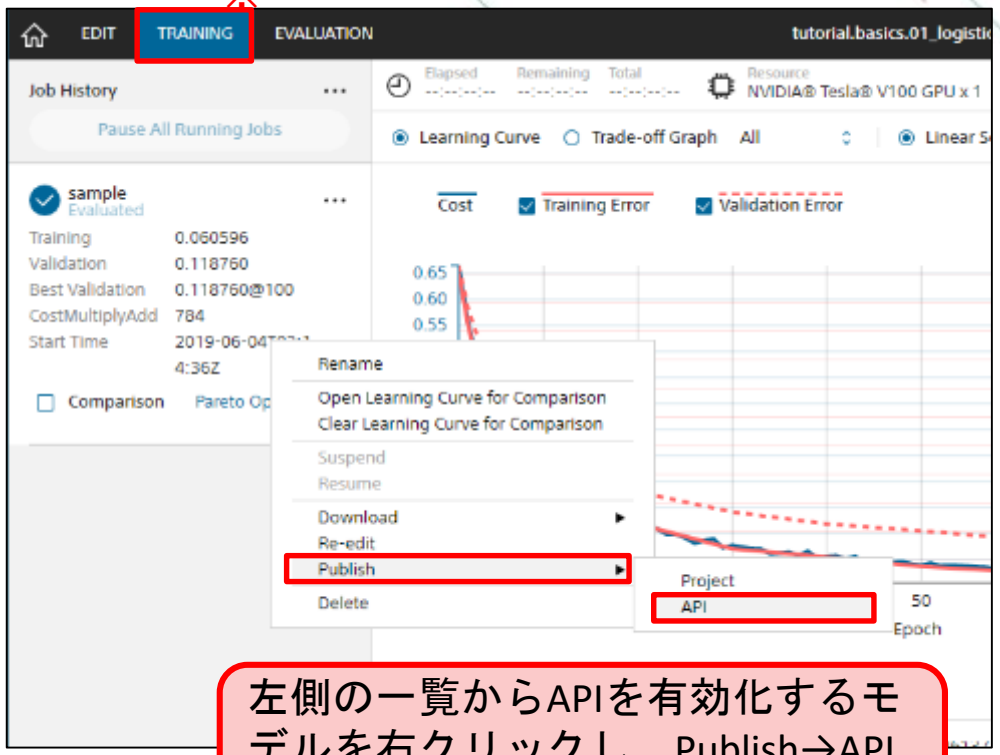
API機能の有効化

API 有効化 → URL,キーの取得

Projectで作成したモデルの一覧からAPIとして利用するモデルをTRAININGタブないしはEVALUATIONタブから選択します。選択後に表示されるポップアップの内容は次頁で解説します。

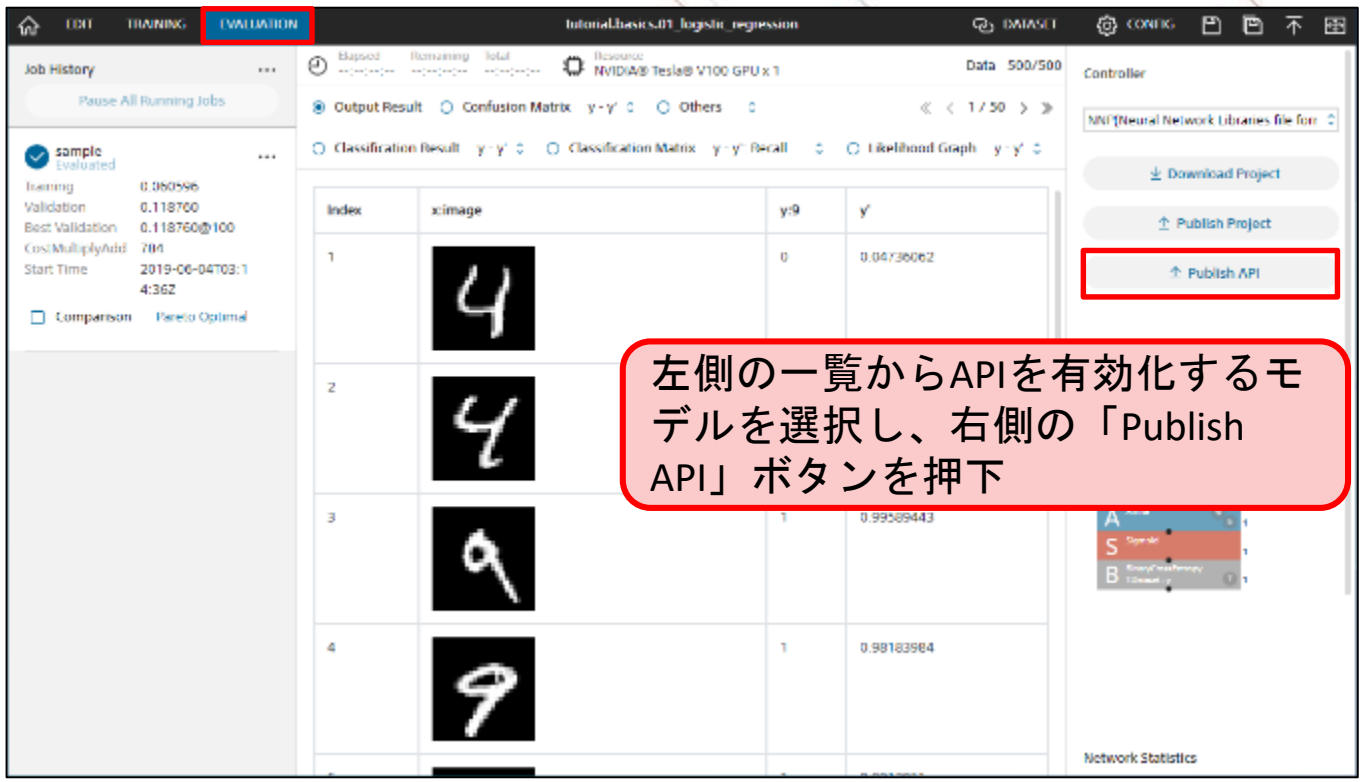
モデル一覧から有効化

※



左側の一覧からAPIを有効化するモデルを右クリックし、Publish→APIと選択

EVALUATIONタブのボタンから有効化



左側の一覧からAPIを有効化するモデルを選択し、右側の「Publish API」ボタンを押下

※モデル一覧からの有効化はTRAININGタブ、EVALUATIONタブの両方から実行可能です。

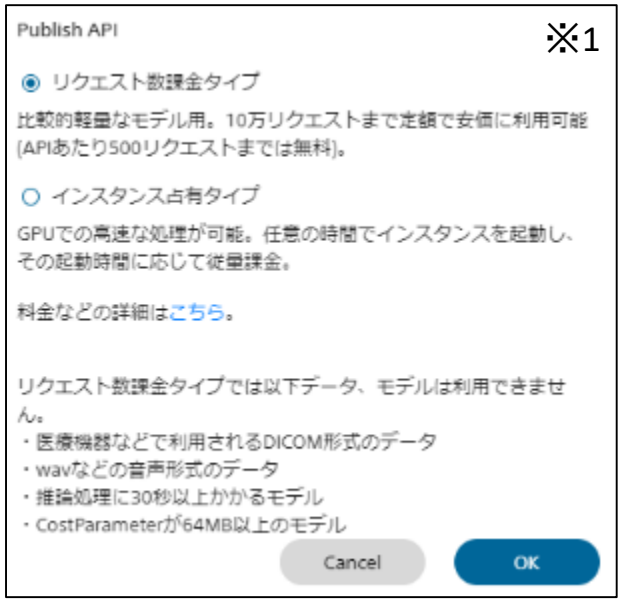
API機能の実行タイプの選択

API 有効化 → URL,キーの取得

ポップアップにて、APIの実行タイプやインスタンス占有タイプを選択した場合の詳細設定を実施します。
APIの設定は後ほど変更することも可能です。

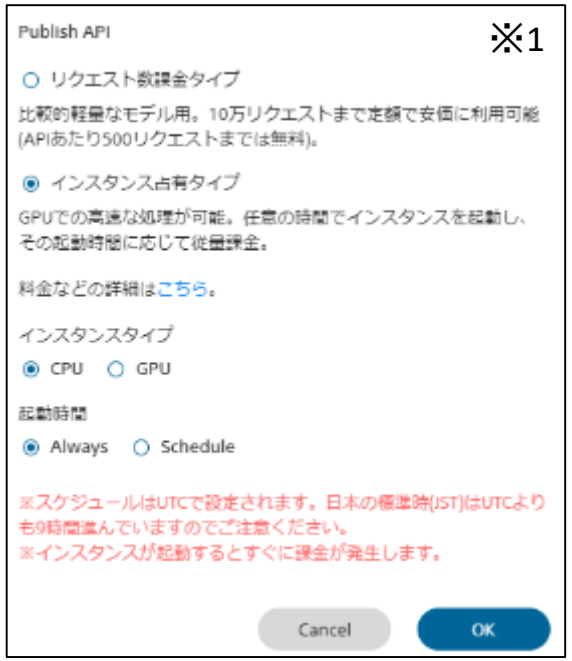
1．APIの実行タイプの選択

APIの実行タイプを選択します。
リクエスト数課金タイプを選択の場合、操作はここで終了です。



2．インスタンスなどの設定

インスタンス占有タイプのCPUと起動時間を選択します。起動時でAlways(常時起動)を選択された場合、操作はここで終了です。



3．起動時間の設定

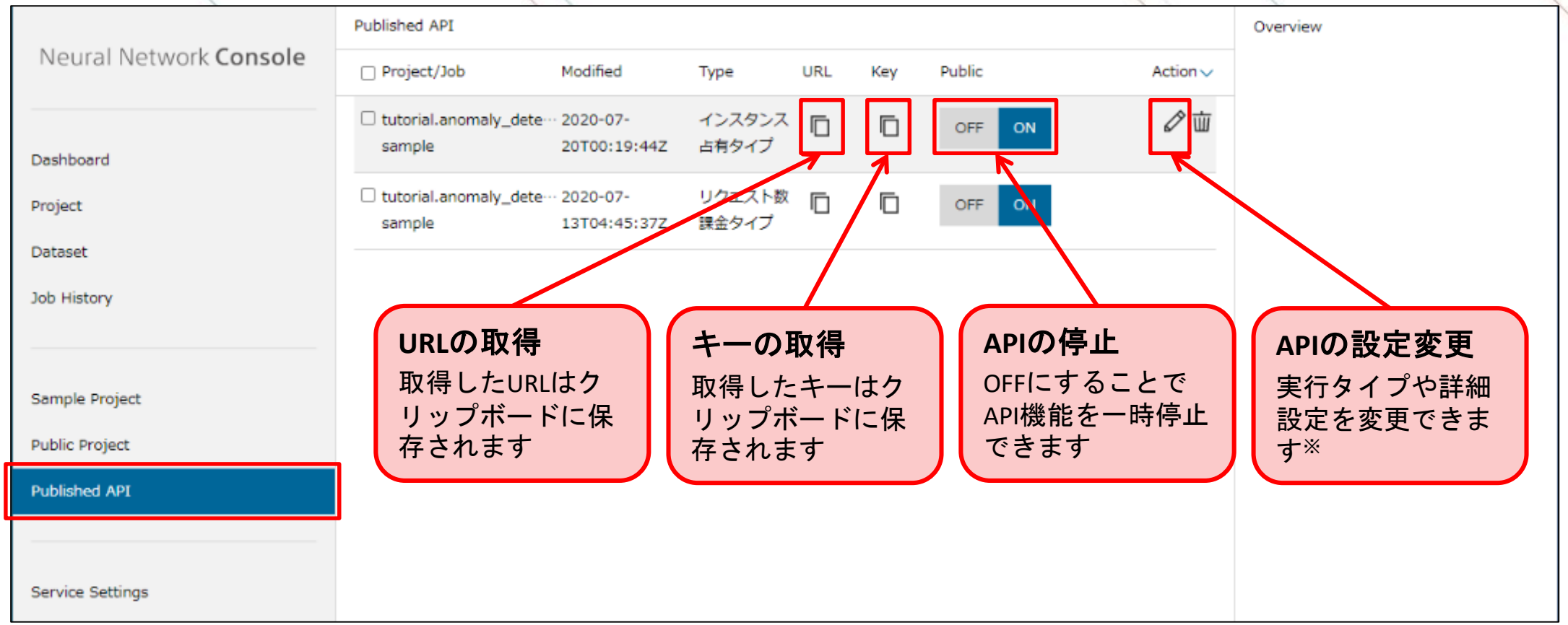
起動時間を設定します。
起動時間は起動の時間帯と曜日で設定することができます。
時間帯はUTCで設定ください※2。

※2 日本標準時から9時間引いた時刻がUTCです。
例えば、日本標準時で9:00～18:00の起動設定をする場合は、0:00～9:00と設定ください。

※1 設定によってポップアップが英語で表示される場合があります。言語表示の変更はユーザ設定をご参照ください。

APIのURLとキーの取得

DashboardのPublished APIに有効化されたAPIの一覧が表示されます。
ここから、API利用時に必要となる各APIのURLやキーの取得が可能です。
また、APIの実行タイプ変更やインスタンス占有タイプのインスタンスや起動時間の変更も可能です。



※実行タイプを変更した際には、URLが変更されますのでご注意ください。

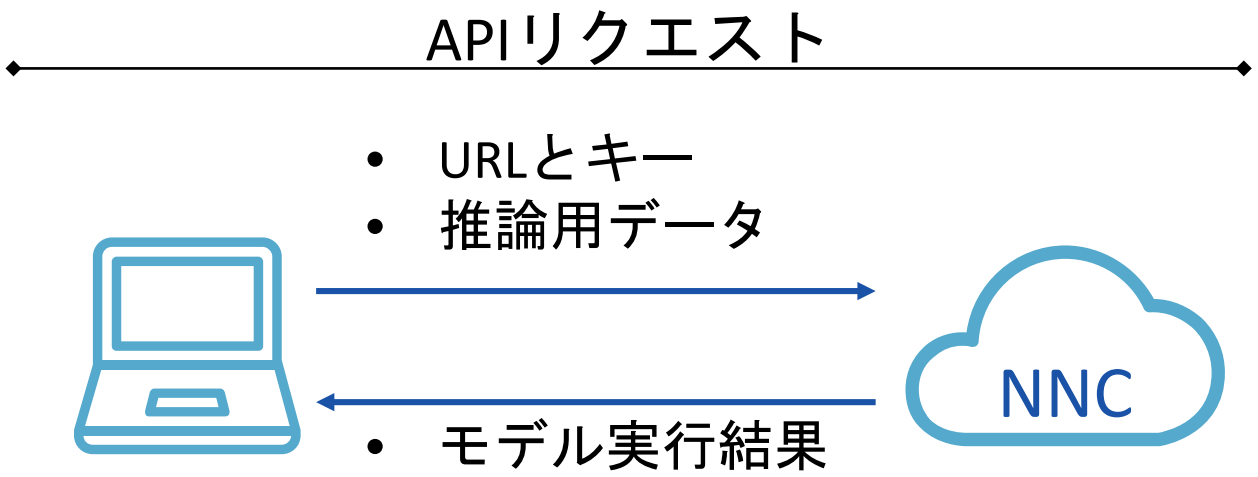
目次

API機能の実行

ユーザ側からHTTP通信でWeb APIリクエストを行うことで、モデルの実行結果を受け取ることができます。

リクエスト実行時にはウェブ画面上から取得したURLとキー、推論用データを合わせて送信します。

リクエストの実装サンプルとして、次頁以降でデータ形式ごとのPythonでの実行方法とcurlを用いたコマンドライン上での実行方法を紹介します。



画像データのAPI実行方法(例)

画像データはモデル作成時のデータセットと同様の規格(画像サイズ、カラー/モノクロ)で準備をし※1、base64値に変換した後に、API実行時に入力データとして送信します。

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import requests
import json
URL = 'https://xxxxx.dl.sony.com/v1/serverless/classifiers/dasdfadfd/inference'
KEY = '6626b3e4-1316-4b1a-b10b-6dsaf3dfsdab995d8'
PIC = './picture.png'
headers = {
    'Content-Type' : 'application/json',
    'x-api-key' : KEY
}
b64 = base64.encodebytes(open(PIC, 'rb').read()).decode('utf8')
data = {
    "executor": "Executor",
    "inputs": [{
        "name": "x",
        "type": "png",
        "data": b64
    }]
}
r = requests.post(URL, data=json.dumps(data), headers=headers)
print(r.text)
```

URL: NNC上で取得したURLを入力
KEY: NNC上で取得したキーを入力
PIC: 画像のファイルパスを入力

name: モデルの入力データ名を入力
type: 画像データの拡張子※2を入力

※1 モデル作成時にImage Normalizationを利用された場合には、API実行時にも同様の処理が実施されます。
※2 対応している拡張子は‘png’, ‘jpg’, ‘jpeg’, ‘gif’, ‘bmp’, ‘tif’, ‘tiff’, ‘dcm’になります。

数値データのAPI実行方法(例)

表データや時系列データなどの数値データはモデル作成時の入力サイズと同様のものをリストで準備し、API実行時に入力データとして送信します。

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import requests
import json
URL = 'https://xxxxx.dl.sony.com/v1/serverless/classifiers/dasdfadfd/inference'
KEY = '6626b3e4-1316-4b1a-b10b-6dsaf3dfsdab995d8'
headers = {
    'Content-Type' : 'application/json',
    'x-api-key' : KEY
}
data = {
    "executor": "Executor",
    "inputs": [{
        "name": "x",
        "type": "vector",
        "data": [[5], [3.5], [1.3], [0.3]]
    }]
}
r = requests.post(URL, data=json.dumps(data), headers=headers)
print(r.text)
```

URL: NNC上で取得したURLを入力
KEY: NNC上で取得したキーを入力

name: モデルの入力データ名を入力

data: データをリストで準備
リストサイズは右図を参照

《データのリストサイズ》
モデル作成時のInputレイヤーに表示されるレイヤーサイズでデータを準備します。

音声データのAPI実行方法(例)

音声データはモデル作成時のデータセットと同じ長さで準備をし※、base64値に変換した後に、API実行時に入力データとして送信します。

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import requests
import json
URL = 'https://xxxxx.dl.sony.com/v1/serverless/classifiers/dasdfadfd/inference'
KEY = '6626b3e4-1316-4b1a-b10b-6dsaf3dfsdab995d8’
WAV = './data.wav'
headers = {
    'Content-Type' : 'application/json',
    'x-api-key' : KEY
}
b64 = base64.encodebytes(open(WAV, 'rb').read()).decode('utf8')
data = {
    "executor": "Executor",
    "inputs": [{
        "name": "x",
        "type": “wav”,
        "data": b64
    }]
}
r = requests.post(URL, data=json.dumps(data), headers=headers)
print(r.text)
```

URL: NNC上で取得したURLを入力
KEY: NNC上で取得したキーを入力
WAV: 音声のファイルパスを入力

name: モデルの入力データ名を入力

※ 対応している音声データのフォーマットは‘wav’のみになります。

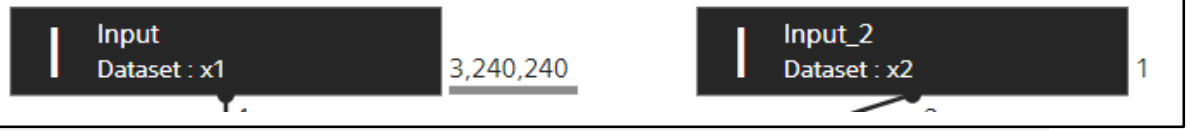
複数データのAPI実行方法(例)

モデルの入力が複数の場合には、各データの形式に応じてp.12～14で説明した形で準備をし、API実行時に入力データとして並列にして送信します。

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
import requests
import json
URL = 'https://xxxxx.dl.sony.com/v1/serverless/classifiers/dasdfadfd/inference'
KEY = '6626b3e4-1316-4b1a-b10b-6dsaf3dfsdab995d8'
PIC = './picture.png'
headers = {
    'Content-Type' : 'application/json',
    'x-api-key' : KEY
}
b64 = base64.encodebytes(open(PIC, 'rb').read()).decode('utf8')
data = {
    "executor": "Executor",
    "inputs": [{
        "name": "x1",
        "type": "png",
        "data": b64
    },{
        "name": "x2",
        "type": "vector",
        "data": [0.25]
    }]
}
r = requests.post(URL, data=json.dumps(data), headers=headers)
print(r.text)
```

入力データが複数の場合には、inputsの中に複数を並列で記載

《ネットワークの入力》



curlを用いた実行方法(例)

curlを用いたコマンドラインからの実行例は以下の通りです。
jsonファイルに実行するデータを記載します。フォーマットについては前頁までのPythonによる実行方法のdata部分と同様です。

《curlによる実行例》

```
curl -H "x-api-key:6626b3e4-1316-4b1a-b10b-6dasab995d8" ¥
     -H "Content-Type:application/json" ¥
     -d @data.json ¥
     "https://xxxxx.dl.sony.com/v1/serverless/classifiers/dasddfd/inference"
```

NNC上で取得したキー

データを記載したjsonファイル

NNC上で取得したURL

《データを記載したjsonファイルの例》

・画像データ

```
{
  "executor": "Executor",
  "inputs": [{
    "name": "x",
    "type": "png",
    "data": "ajf;aejf;oafaf・・・¥n"
  }]
}
```

拡張子を入力※1

base64値を入力※2

・数値データ

```
{
  "executor": "Executor",
  "inputs": [{
    "name": "x",
    "type": "vector",
    "data": [5, 3.5, 1.3, 0.3]
  }]
}
```

・音声データ

```
{
  "executor": "Executor",
  "inputs": [{
    "name": "x",
    "type": "wav",
    "data": "ajf;aejoafasdf・・・¥n"
  }]
}
```

base64値を入力※2

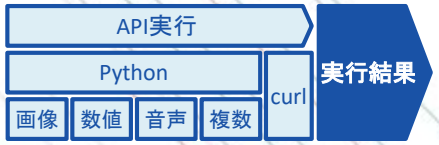
・複数データ

入力データが複数の場合には、Inputsの中に並列で複数を記載ください。各データは左の形式でご準備ください。

※1 対応している拡張子は‘png’, ‘jpg’, ‘jpeg’, ‘gif’, ‘bmp’, ‘tif’, ‘tiff’, ‘dcm’になります。
※2 画像データと音声データは事前にbase64値に変換いただく必要があります。
Pythonによる変換例は以下の通りです。

```
base64.encodebytes(open(filepath, 'rb').read()).decode('utf8')
```

実行結果



実行が正常に完了すると、出力レイヤーの名称とデータなどがjson形式で返却されます※1。
異常時には”error”を含むデータが返却されますので、”message”を確認したうえで、キーやデータなどを再確認ください。

《正常時の実行結果》

{"version":"20200707_051647035778","outputs"[{"name":"y","data":"0.9994778037071228, 0.0005221762694418 43, 9.620963747902778e-12"}]}

outputsに出力レイヤーの名称とデータが記載

《異常時の実行結果》

{"code":"4031020301""error":"ACCESS_DENIED","message""Invalid API key""data":{"request_id":"e5aefec8-cbab-4a45-af65-7905e4e26fa4"}}

実行結果に”error”が含まれると実行に失敗

“message”の内容で失敗の原因を確認ください。

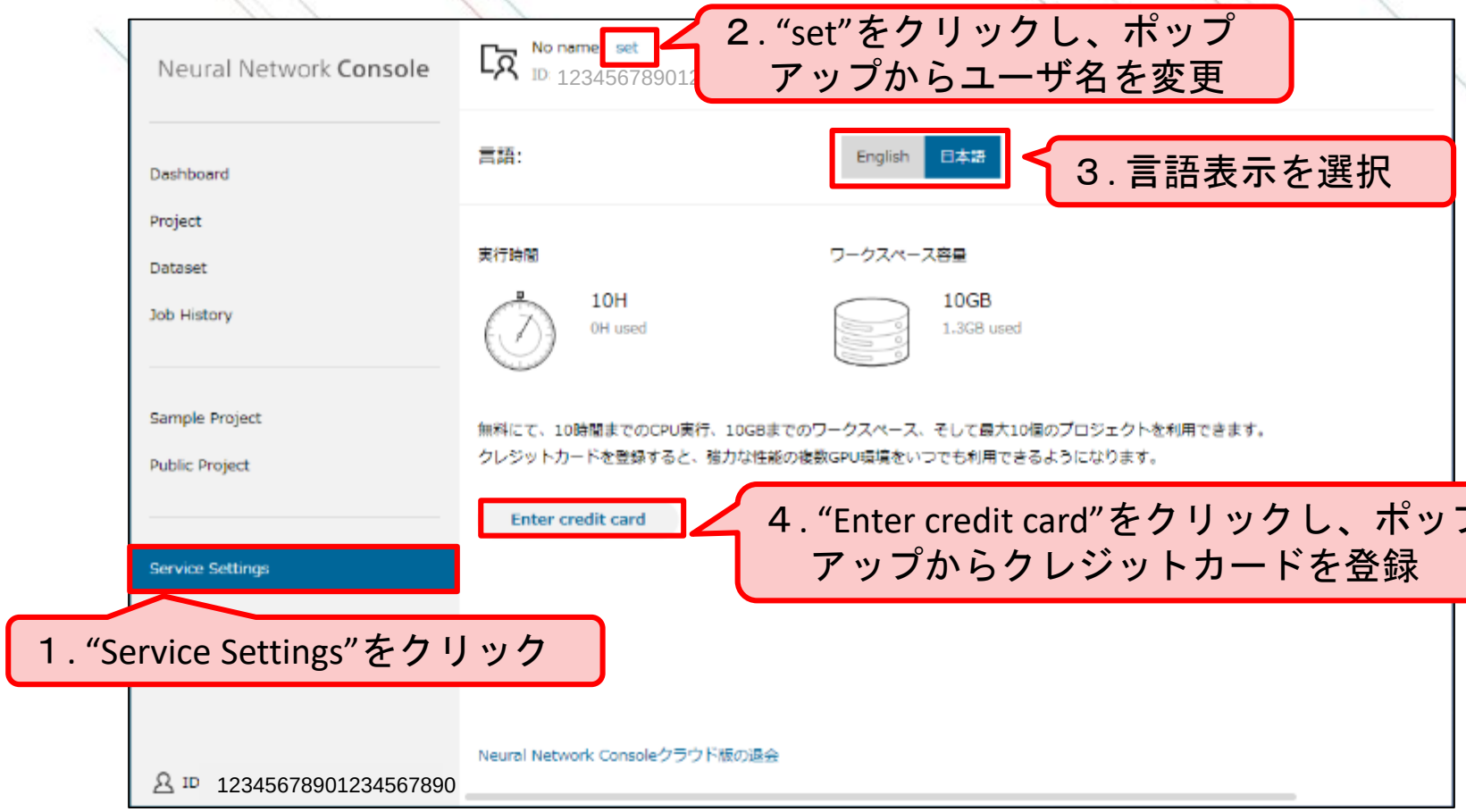
- “Invalid API key”の場合
  APIのキーを再確認ください。
- “Invalid request”の場合
  送付したデータを再確認ください。
- “Can not publish api”の場合
  無料利用枠を使い切ったため、有料プランをご利用ください※2。
- その他
  APIが有効になっているかをご確認の上で、少し時間をおいて再度お試しください。改善しない場合には、問合せフォームかコミュニティでご連絡ください。

※1 実行時にjson形式の返却がなくエラーが発生する場合には、URLが誤りの可能性がありますので、ご確認ください。
※2 有料プランの利用にはクレジットカード登録もしくは法人契約が必要になります。クレジットカード登録はユーザ設定を、法人契約はウェブサイトをご参照ください。

Appendix

ユーザ設定

Service Settingsからユーザ名や言語表示の変更、クレジットカード登録※1などが可能です。
グループ機能を利用される場合には、ユーザ名を設定することで作業者が明確になり便利です※2。



※１GPUなど有料のサービスを利用される場合には、クレジットカード登録ないしは法人契約が必要になります。
※２ すでにグループ登録をされている方は、画面上に表示されるタブを、Personalに変更ください。(本紙には記載がありません)