2024.7/1 作成

---**本資料目次**

---

① 本稿での仮想環境概要

PCは2台使用しています．1台は外付けGPU搭載パソコンで．データセットを作成～YOLOモデルをトレーニングします．もう1台は内蔵GPU搭載パソコン（CPU：Intel N100）で作成済みモデルを使って物体検出実験をしています．

特集第 1 部では，Windows パソコンで Stable Diffusion と YOLOv8 を実行できるよう環境を整備し．Python プログラムを作成します．その基盤となる実行環境は仮想環境を使って，図 1 のように，用途に応じて 3 つの仮想環境（env-da，env-ga，env-ai）を構築します．仮想環境 env-da と env-ga でデータ拡張を含むデータセット作成を行い，env-ai で物体検出モデルをトレーニングしたり，モデルを利用したりします．

仮想環境を使えば，Windows 環境に影響を与えにくく，何かしらの操作で Windows が動かなくなってしまう可能性を減らせます．Python 標準機能に仮想環境を作るコマンドがあるので，これを使います．

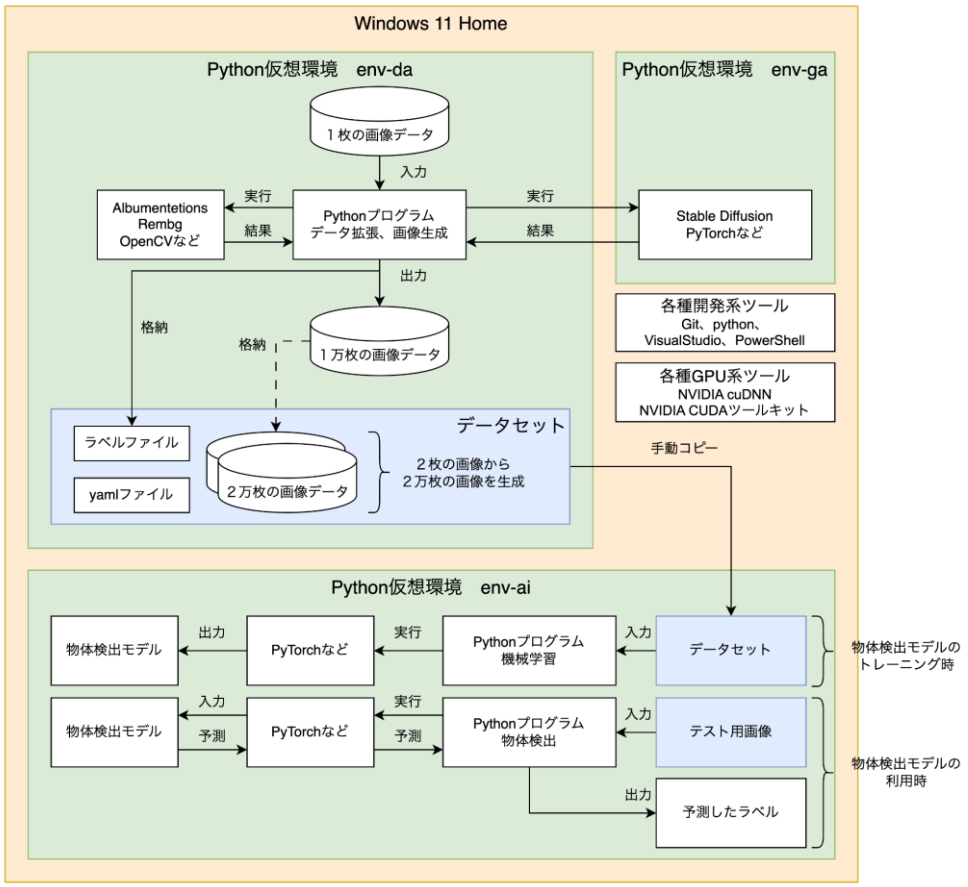

図１　作成する仮想環境とその用途

●用途別に仮想環境を作る理由

用途に応じて３つの仮想環境を作成します．用途によって使う機械学習フレームワーク．ツールやパッケージなどが異なります．オープンソースのツールでは．バージョン・アップが頻繁に行われることがあり．バージョン・アップのタイミングによっては．依存関係にあるライブラリやパッケージの管理が重要になります．

同一実行環境上で．すべてのツールが動作するように維持するのは難しいです．オープンソースでは．開発されたツールによってテストされている実行環境が異なるためです．同じ実行環境上で．メンテナンスしていこうとすると．依存関係にあるライブラリやパッケージが異なり．ツール間で異なるバージョンが要求されることもあります．

そのため．実行環境の整備を簡単にするため．以下の表１のように実行環境を仮想的に分離します．Python には．仮想環境を作る venv ツールが用意されています．このツールで実行環境を仮想的に分けることで．それぞれの仮想環境で異なるバージョンのライブラリやパッケージを同じパソコン上で保持し．それぞれの独立した実行環境で処理することができます．

表 1　Python 仮想環境一覧

仮想環境の用途	主な利用ツール	パソコン上の仮想環境	Python プログラムなどの格納
学習用データのデータ拡張	Albumentetions Rembg OpenCV	env-da	./env-da/user
学習用データの画像生成	Stable Diffusion PyTorch	env-ga	
物体検出モデルの機械学習	PyTorch	env-ai	./env-ai/user

上記の表 1 から．Albumentetations．Rembg．Stable Diffusion の実行に必要な Python ライブラリのバージョンが異なるので．Python 実行環境を仮想環境で分けます．ライブラリの依存関係も複雑にならず．ツールや実行環境のメンテナンスも簡単になります．また．Python が使用する RAM も分けることができますから．大きな RAM 領域を活かすことができます．

Stable Diffusion や PyTorch はフォルダ構成の階層を浅くして．パスが短くなるように環境を作ると良いとされています．Windows の環境設定でパスを通すなど工夫をすると良いでしょう．なお．筆者のパソコンでは C:¥Users¥if の配下に仮想環境を作成しました．できるだけ．コマンドや Python プログラムの実行時にパスが長くなり．エラーにならないようにしています．

●すぐに切り替え可能

仮想環境は次のコマンドで，使うときだけ有効化して，使用しないときには無効化して切り替えます．この切り替えには Windows OS の再起動なしに，有効化・無効化が可能ですので，使用する仮想環境だけを有効化して，Python プログラムやツールを実行します．このように Windows 環境と仮想環境を分けておくことで，それぞれの環境が干渉しにくくなります．

・有効化コマンド…active
・無効化コマンド…deactive

●パソコン上に仮想環境を構築

Windows に図 2 のような仮想環境を作成します．

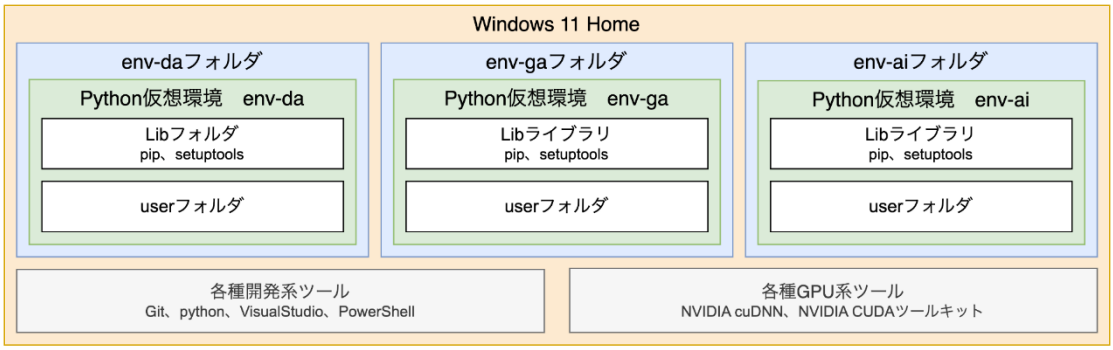

図 2　Windows OS 内の仮想環境の位置づけ

リスト 1 のコマンド実行例では，カレントパスが C:¥Users¥if>ですので，次の
.¥[仮想環境名]¥Scripts¥activate
の形式でコマンドを実行します．ここでは，データ拡張に使用する仮想環境 env-da の例をあげていますが，他にも学習用データの画像生成の仮想環境 env-ga，物体検出モデルの機械学習の仮想環境 env-ai も同じ要領で作成します．

仮想環境が有効化されると，プロンプト文の先頭に(env-da) が表示されます．無効時には，(env-da) が表示が消えますので見分けやすいと思います．上記の例ですと「(env-da)」が表示されていると有効化されているとわかります．

リスト 1 仮想環境の作成．有効化．無効化のコマンド

```
C:¥Users¥if>python -m venv env-da                    # 仮想環境の作成
C:¥Users¥if>.¥env-da¥Scripts¥activate                # 仮想観葉の有効化
(env-da) C:¥Users¥if>

(env-da) C:¥Users¥if>deactivate                      # 仮想環境の無効化
```

●設定

▲共通のもの

仮想環境内の pip コマンドと setuptools コマンドを最新化します．コマンド・プロンプトで実行するコマンドはリスト 2 の通りですが，仮想環境に応じて「env-da」の部分を，各仮想環境名（env-ga）に読み替えてください．

リスト 2　共通の環境設定

```
# カレントフォルダ変更
(env-da) C:¥Users¥if>cd env-da
```

```
(env-da) C:¥Users¥if¥env-da>

# Python 関連ツールのインストール
(env-da) C:¥Users¥if¥env-da>python -m pip install --upgrade pip
(env-da) C:¥Users¥if¥env-da>pip install --upgrade setuptools

# user フォルダ作成
(env-da) C:¥Users¥if¥env-da>md user
```

注意点としては, 有効化せずに pip コマンドで Python パッケージをインストールすると, 仮想環境内ではなく Windows OS に直接インストールされてしまうことです. また, 仮想環境の使用後は無効化します. 無効化を忘れると正常に動作しなくなるかもしれません. パソコンをシャットダウンするときにも, 無効化を忘れないようにしましょう.

機械学習フレームワークで使用する Python ライブラリやパッケージを Lib フォルダにインストールする前に, 最新バージョンでないと, ツールやライブラリのインストール作業時にエラーが発生するかもしれません. インストールすると, C:¥Users¥if¥env-da の下に格納されます. また, ユーザが作成する Python プログラムやデータ拡張される画像は, user ディレクトリを作成して格納します. user フォルダにまとめているので分かりやすいと思います.

▲仮想環境 env-da

仮想環境によって異なる機能を持たせるために, 異なるツールやパッケージなどをインストールします. リスト 3 は仮想環境 env-da の設定で実行するコマンドです.

リスト 3　仮想環境 env-da の環境構築で実行するコマンド

```
#  データ拡張関連ツールのインストール
(env-da) C:¥Users¥if¥env-da>pip install optimum[onnxruntime-gpu]
(env-da) C:¥Users¥if¥env-da>pip install -U rembg[gpu,cli]
(env-da) C:¥Users¥if¥env-da>pip install -U albumentations webuiapi

#  前処理用フォルダ作成
(env-da) C:¥Users¥if¥env-da>cd user
(env-da) C:¥Users¥if¥env-da¥user>md sample_img
```

仮想環境 env-da は, user フォルダ配下に中間ファイルや出力ファイルを格納するフォル

ダを作りますが，これらは 2 種類のボルトの「OK」と「NG」のクラスに対し，ラベル番号として 0 と 1 を割り当てていますので，それぞれの処理でフォルダを作成します．

最終的には，個別にコマンドや Python プログラムを実行しなくて済むようにします．データ拡張の処理と合わせて，クラスごとに処理する内容とフォルダ作成をまとめて，バッチファイル data_exp.bat で処理します．

▲仮想環境 env-ga

環境構築時に実行するコマンドをリスト 4 に示します．仮想環境を分けるのは手間ではあるのですが，各環境で使用するパッケージ，コンポーネント，ライブラリなどは，相互に依存するバージョンが一致していない部分があります．インストール時にこうした調整をしたり，将来的にバージョンアップしたりするときにバージョンや依存関係を管理することが難しくなります．そのため，仮想環境を機能ごとに分離することで，環境構築やパッケージ管理を簡単にしています．また，リスト 4 最後のコマンドで v1.7.0 にする必要があります．

リスト 4　仮想環境 env-ga の環境構築で実行するコマンド

```
# Stable Diffusion 関連ルールのインストール
(env-ga) C:¥Users¥if¥env-ga>git clone https://github.com/AUTOMATIC1111/stable-diffusion-webui.git
(env-ga) C:¥Users¥if¥env-ga>cd stable-diffusion-webui
(env-ga) C:¥Users¥if¥env-ga¥stable-diffusion-webui>
(env-ga) C:¥Users¥if¥env-ga¥stable-diffusion-webui>git checkout cf2772f
```

---

②仮想環境 env-ga への Stable Diffusion 導入のコツ

●高速処理を実現する xformers

xformers は，facebook 社が作ったコンポーネントで，NVIDIA 製 GPU の処理能力を使ってコンピュータビジョンや NLP の処理を高速に実行するのに用いられます．Stable Diffusion は，xformers のコンポーネントを利用するとより高速に処理できます（図 3）．特に，反復的な処理を可能な限り高速で行い，高いメモリ効率を実現します．また，xFormers には，独自の CUDA カーネルが含まれる最先端のコンポーネントです．必要に応じて他のライブラリに切り替えられます．

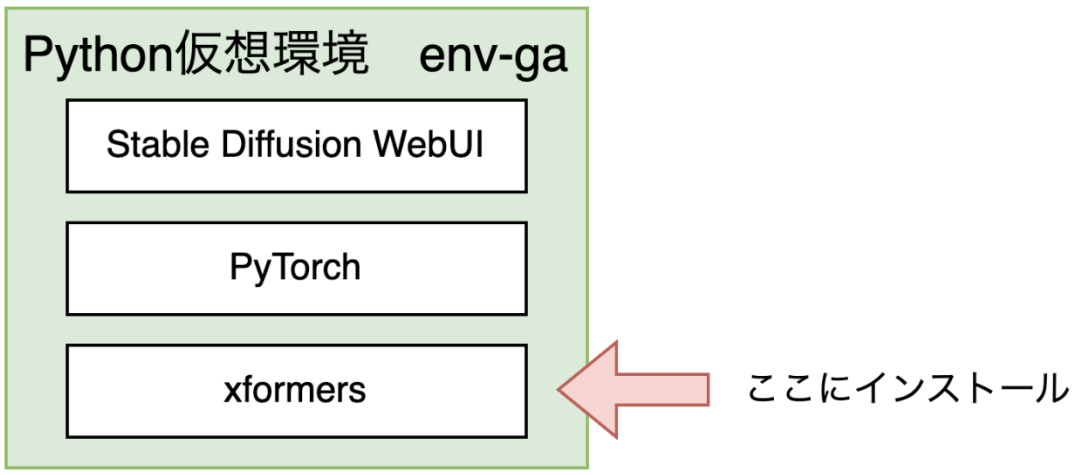

図 3　仮想環境 env-ga の主なツール

便利で高機能な xformers ですが，PyTorch などの主要なツールやライブラリで，標準的なインストールパッケージには，このコンポーネントが含まれていません．そのため，Stable Diffusion や PyTorch とは別に，ここで改めてインストールする必要があります．A1111 の標準インストールでも，このコンポーネントが含まれておらず，初期状態のまま起動しても自動インストールされません．

通常は，pip コマンドでインストールしますが，この方法では WebUI の開発者がテスト済みのバージョンに合わせるのに手間がかかります．そこで，stable-diffusion-webui フォルダ内の webui-user.bat（Stable Diffusion WebUI Automatic1111 に標準インストールでダウンロードされているもの）の次の箇所を修正し，WebUI を起動すると，起動時に xformers のコンポーネントや関連パッケージが最適なバージョンでインストールされます．

次のように Stable Diffusion 標準のバッチファイルを修正して実行すると，開発者が用意した関連パッケージ等のインストール情報を使ってインストールできるため失敗がありません．

＜１回目の変更＞

変更前：set COMMANDLINE_ARGS=

変更後：set COMMANDLINE_ARGS=--xformers --reinstall-xformers

変更を保存してバッチファイルを実行して，WebUI を起動します．Stable Diffusion WebUI の起動途中で，初回実行時に xformers が使えるように関連パッケージなどが自動でインストールされます．

なお，通常通り pip コマンドでインストールしたところ，パスを通しても WebUI から正常にアクセスできませんでした．仮想環境にインストールしたことでパスを見つけられないことが原因のようです．Windows の環境設定でパスを通してもよいのですが，その他の

パッケージとの依存関係がありますので，仮想環境に分ける意味がなくなってしまいます．ここでは，このような方法を選択しました．

WebUI を起動するためには，コマンド・プロンプトから次のコマンドを実行します．正常にインストールが終わると，Stable Diffusion WebUI が実行され，自動で Web ブラウザ上にページが表示されます．

＜WebUI（Automatic1111）の起動＞

(env-ga) C:¥Users¥if¥env-ga¥stable-diffusion-webui>webui-user.bat

※参考：facebookresearch/xformers　https://github.com/facebookresearch/xformers

●API 有効化と WebUI の無効化

次に，WebUI の API を有効にします．Python プログラムで API 通信できるようになると，ウェブ・ブラウザでの操作が不要になるため WebUI を無効化します．この設定も，先程の webui-user.bat を次のように修正すると設定できます．

変更前：set COMMANDLINE_ARGS=--xformers --reinstall-xformers

変更後：set COMMANDLINE_ARGS=--xformers --api –nowebui

「--reinstall-xformers」は不要ですので削除します．この修正を忘れても WebUI を使うことができますが，毎回，起動時に xformers を再インストールしますので，起動に余計な時間がかかります．

修正後，webui-user.bat を実行したコマンド・プロンプトで，Ctrl+c を入力し，WebUI のプロセスを停止するため「Y」を入力します．プロセスが停止した後，修正済みの webui-user.bat を実行すると，上記の「--api」の設定によって，API と通信できるようになります．

次に，引数に「--nowebui」を設定しましたので，WebUI のプロセス起動時に，ウェブ・ブラウザを自動起動しなくなりますので，RAM 使用量が節約できます．もし，RAM の空き容量に余裕があるのであれば，この引数を設定しないのも 1 つの手です．そのようにすると WebUI と API のどちらでも操作できるようになります．

---

③Stable Diffusion WebUI に拡張機能を導入

Stable Diffusion WebUI を閉じる前に，拡張機能をインストールします．作業は，WebUI の Extensions タブから GUI を操作します．追加する拡張機能は Dynamic Prompts です（図 4）．さまざまなパターンの画像を生成するときに使います．

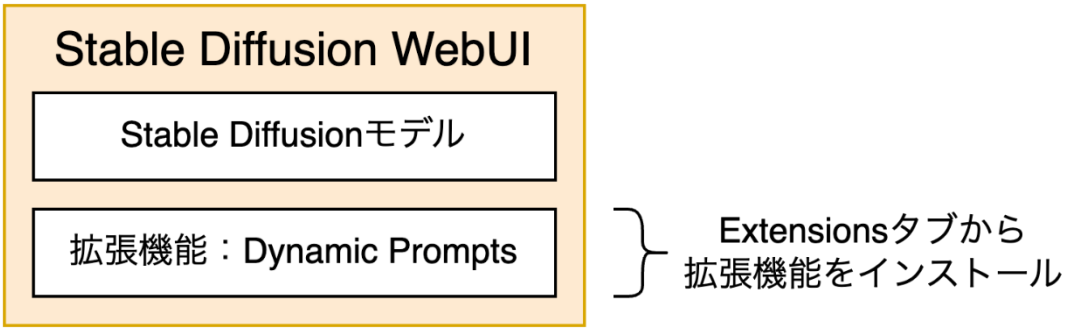

図 4　拡張機能の追加イメージ

この機能を追加すると，プロンプト文に{ red | blue | green }のように記載すると，|で単語を区切っていると解釈されます．プロンプトに記述すると，いずれかの単語がランダムに選ばれて入力されます．

この機能は，Stable Diffusion WebUI の標準でインストールされないため，拡張機能として追加インストールします．インストールと設定は次のように行います．

1.Settings タブで「Dynamic Prompts enabled」にチェックを入れて適用する
2.インストールが開始される
3.インストール後，WebUI のプロセスの再起動する
4.ウェブブラウザを閉じて，env-ga を有効化しているコマンド・プロンプトで Ctlr+c を入力する
5.Stable Diffusion のプロセスを停止する
6.webui-user.bat を実行する
7.追加した拡張機能が WebUI 上に表示されて，機能が使用できるようになる

---
④物体検知のための環境設定

物体検知は，データ拡張やモデル・トレーニングに用いてきた PC とは別の，あえてスペックが高くない PC を使います．この PC に仮想環境 env-yl を構築します．

リスト 5　仮想環境 env-yl の構築

```
python -m venv env-yi
# Python 関連ツールのインストール
python -m pip install --upgrade pip
pip install --upgrade setuptools
```

```
# YOLOv8 のインストール
pip install ultralytics
```