

Infini-Brain A101/B, A101/BH SDK 使い方ガイド補足情報

(Model Management の使い方)

目次

本書をお読みになる前に	3
このマニュアルについて	3
安全にお使いいただくために	3
本書の表記	3
Windows の操作	4
用語	4
商標および著作権	4
第 1 章 お使いになる前に	
1 Model Management の概要	6
2 インストール	7
Model Management のインストール／アンインストール	7
Model Encryption Tool のインストール／アンインストール	8
3 環境変数	9
第 2 章 暗号化モデル導入の流れ	
1 Model Management データフロー	11
2 暗号化モデルの導入と推論までのフロー	12
第 3 章 AI 拡張ボードの再セットアップ	
1 再セットアップ手順	14
メインボード交換	14
AI 拡張ボードの構成変更	14
第 4 章 Model Management のご使用の流れ	
1 AI モデル暗号化	16
2 スクリプトの準備	18
3 Dockerfile 作成	19
4 Docker コンテナ作成	20
5 暗号化秘密情報の登録	21
6 Docker コンテナ立ち上げ	22
7 推論結果の確認	23

本書をお読みにする前に

このマニュアルについて

本マニュアルは、「Model Management」の使い方について説明しています。

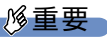

安全にお使いいただくために

本製品を安全に正しくお使いいただくための重要な情報が『取扱説明書』に記載されています。特に、「安全上のご注意」をよくお読みになり、理解されたうえで本製品をお使いください。

本書の表記

■ 本書の記号

本書に記載されている記号には、次のような意味があります。

 重要	お使いになるときの注意点や、してはいけないことを記述しています。必ずお読みください。
 POINT	操作に関連することを記述しています。必要に応じてお読みください。
→	参照ページを示しています。

■ キーの表記と操作方法

本書中のキーの表記は、キーボードに書かれているマークを記述するのではなく、説明に必要な文字を使い、次のように記述しています。

例：【Ctrl】キー、【Enter】キー、【→】キーなど
また、複数のキーを同時に押す場合には、次のように「+」でつないで表記しています。
例：【Ctrl】+【F3】キー、【Shift】+【↑】キーなど

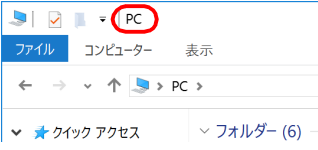
■ 連続する操作の表記方法

本書中の操作手順において、連続する操作手順を、「→」でつなげて記述しています。
例：コントロールパネルの「システムとセキュリティ」をクリックし、「システム」をクリックし、「デバイスマネージャー」をクリックする操作
↓
「システムとセキュリティ」→「システム」→「デバイスマネージャー」の順にクリックします。

■ ウィンドウ名の表記

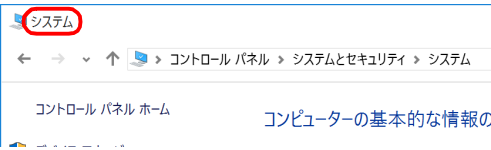
本文中のウィンドウ名は、アドレスバーの最後に表示されている名称を表記しています。

例：



↓

「PC」ウィンドウ



↓

「システム」ウィンドウ

■ 画面例およびイラストについて

本文中の画面およびイラストは一例です。お使いの機種やモデルによって、実際に表示される画面やイラスト、およびファイル名などが異なることがあります。また、イラストは説明の都合上、本来接続されているケーブル類を省略したり形状を簡略化したりしていることがあります。

■ 製品の呼び方


本書では、製品名称を次のように略して表記します。

製品名称	本書の表記		
Windows 10 IoT Enterprise 2019 LTSC	Windows 10 IoT Enterprise	Windows 10	Windows
Infini-Brain A101/B	Infini-Brain	本製品	
Infini-Brain A101/BH			
NVIDIA® Jetson™	NVIDIA Jetson		
NVIDIA® Jetson™ TX2	NVIDIA Jetson TX2	NVIDIA Jetson	
NVIDIA® Jetson AGX Xavier™	NVIDIA Jetson AGX Xavier		

Windows の操作

■ アクションセンター

アプリからの通知を表示するほか、クリックすることで画面の明るさ設定や通信機能の状態などを設定できるアイコンが表示されます。

- 1 画面右下の通知領域にある  をクリックします。
画面右側に「アクションセンター」が表示されます。

■ 「コントロールパネル」ウィンドウ

次の手順で「コントロールパネル」ウィンドウを表示させてください。

- 1 「スタート」ボタン→「Windows システム ツール」→「コントロールパネル」の順にクリックします。

■ 「コマンドプロンプト」ウィンドウ


次の手順で「コマンドプロンプト」ウィンドウを表示させてください。

- 1 「スタート」ボタン→「Windows システム ツール」の順にクリックします。
- 2 「コマンドプロンプト」を右クリックし、「その他」→「管理者として実行」をクリックします。

■ ユーザーアカウント制御

本書で説明している Windows の操作の途中で、「ユーザーアカウント制御」ウィンドウが表示される場合があります。これは、重要な操作や管理者の権限が必要な操作の前に Windows が表示しているものです。表示されるメッセージに従って操作してください。

■ 通知領域のアイコン

デスクトップ画面右下の通知領域にすべてのアイコンが表示されていない場合があります。
表示されていないアイコンを一時的に表示するには、通知領域の  をクリックします。

用語

次の用語について、本マニュアルでは次のように表記しています。

用語	本文中の表記
AI モデル	学習済みモデル
暗号化 AI モデル	Encrypted Model file
Docker イメージ	イメージ名 (リポジトリ名: [タグ名])
Docker イメージの tar	tar
推論結果の依頼スクリプト	推論依頼アプリ (メインボード側)
推論結果の送信スクリプト	推論結果を返すスクリプト (AI 拡張ボード側)

商標および著作権

NVIDIA、CUDA、Jetson、Jetpack、NVIDIA Jetpack、TensorRT は、アメリカ合衆国および / またはその他の国における NVIDIA Corporation の商標または登録商標です。
本製品には、Apache License V2.0 に基づきライセンスされるソフトウェアに当社が必要な改変を施して使用しております。
本製品には、BSD、GNU General Public License (GPL)、MIT、その他のライセンスに基づくオープンソースソフトウェアが含まれています。
オープンソースソフトウェアのライセンスに関する詳細およびソフトウェアのソースコードについては、本製品のマニュアルをご覧ください。
FUJITSU Hardware Monitor、Model Management、SDK Distributed Manager、SDK Support Tool、バーチャル LAN ドライバー、ブリッジコントローラードライバーは、富士通クライアントコンピューティング株式会社の製品です。著作権は富士通クライアントコンピューティング株式会社にあります。
その他の各製品名は、各社の商標、または登録商標です。
その他の各製品は、各社の著作物です。
その他のすべての商標は、それぞれの所有者に帰属します。

Copyright FUJITSU LIMITED 2020

1

第 1 章 お使いになる前に

Model Management の機能概要と提供ツールについて説明します。

1. Model Management の概要	6
2. インストール	7
3. 環境変数	9

1. Model Management の概要

本アプリは、SDK Distributed Manager を利用しているユーザーの AI モデルを暗号化し、セキュアに管理する機能を提供します。
暗号・復号化ツールと、暗号化された AI モデルを AI 拡張ボードにデプロイするための API を提供します。

- ユーザーの開発環境（NVIDIA Jetson 開発者キット、AI 拡張ボード部）で実行できる暗号化ツール（Model Encryption Tool）を提供します。
 - ・本ツールで AI モデルの暗号化を行います。
 - ・本ツールで AI モデルの暗号化と暗号化秘密情報が生成されます。
暗号化秘密情報を本製品に登録することで、復号化するとき秘密情報を取得し自動で復号化します。詳しくは、第 2 章「暗号化モデル導入の流れ」（→ P.10）をご覧ください。
（暗号化秘密情報は、第三者に渡ってしまうと号化した AI モデルが解読される可能性があるため、ユーザー自身で厳重に管理してください。）
- 暗号化秘密情報は、本製品の TPM キーで暗号化され、厳重に管理されます。
- AI 拡張ボード部の Docker を TLS 証明書でセキュアに接続し、Windows 側で操作できる API を提供します。
- API は Windows 上で Python 言語により活用することができます。

■ Model Management で提供するツールと機能

ツール	ツール実行環境	概要	ファイルパス（インストール後に展開）
Model Management	メインボード部	AI 拡張ボード上の Docker を TLS 証明書でセキュアに接続し、Windows 側で Docker コマンドを操作できる API です。	C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\Basic\Model Management\bin\ModelMgmt.exe
Model Encryption Tool	NVIDIA Jetson 開発者キット	ユーザーの開発環境（NVIDIA Jetson 開発者キット）で AI モデルを暗号化するためのツールです。deb パッケージを開発環境にインストールすることで使用できます。	C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\Basic\Model Management\tools\modelencryption-1.0-1.deb
menctool	NVIDIA Jetson 開発者キット	Model Encryption Tool のヘルパースクリプトです。本スクリプトを使用することで、会話形式で暗号化に必要なパラメータを入力できます。本ツールで簡単に AI モデルを暗号化することが可能です。Model Encryption Toolを開発環境にインストールすることで使用できます。	/usr/bin/menctool
Model Decryption Tool	AI 拡張ボード部（Docker コンテナ）	暗号化モデルの復号化ツールとして、Docker コンテナの中に本ツールをバンドルしてください。コンテナへの実装手順は、第 2 章「暗号化モデル導入の流れ」（→ P.10）をご覧ください。	C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\Basic\Model Management\tools\ModelDecryptionTool
ModelAgentd	AI 拡張ボード部	Model Management API のコマンドパーサーです。常駐サービスで AI 拡張ボード部で起動します。	/opt/fccl/sdk/basic/modelagent/ModelAgentd

機能	概要
Model Management API	メインボードから AI 拡張ボードに Docker イメージをデプロイし、docker コンテナを操作する機能をコマンド API で提供しています。
Docker 通信機能	メインボードと AI 拡張ボード間の Docker 通信において、TLS 証明書でセキュアに通信します。
AI モデルの暗号／復号化機能	ユーザーの AI モデルを AES で暗号化します。また、暗号化時にユーザーが登録したパスワードを、さらに TPM キーを使って暗号化し、よりセキュアに管理します。 ユーザーが、Docker コンテナを API で起動時に自動で暗号化モデルが復号化します。
AI モデルの改竄／複製防止機能	暗号化された AI モデルは、暗号化秘密情報と対で管理されており、本製品に暗号化秘密情報をユーザー自身が適用していない環境では、AI モデルの復号化はできません。 AI 拡張ボードの盗難、複製された場合においても本機能により AI モデルの解読はできません。

2. インストール

Model Management のインストール／アンインストール

Model Management のインストール、アンインストール手順です。なお、アプリのインストール／アンインストールは、管理者権限のアカウントで行ってください。

■ インストール

□ メインボード部のセットアップ

- 1 管理者権限で「コマンドプロンプト」を起動します（→ P.4）。
- 2 次のコマンドを入力し、[Enter] キーを押します。
`> cd "C:\Fujitsu\Bundle\Model Management"`
「C:\Fujitsu\Bundle\Model Management」フォルダーへ移動します。
- 3 次のコマンドを実行し、サービスの登録を行います。
`> SetupUI.bat`
- 4 画面の指示に従って実行します。
インストール完了後、「Close」をクリックします。

□ AI 拡張ボード部のセットアップ

- 1 管理者権限で「コマンドプロンプト」を起動します（→ P.4）。
- 2 次のコマンドを入力し、[Enter] キーを押します。
`> cd "C:\Fujitsu\Bundle\Model Management\SubsystemInstaller"`
「C:\Fujitsu\Bundle\Model Management\SubsystemInstaller」フォルダーへ移動します。
- 3 「.¥SubsystemInstaller」フォルダーに移動し、インストールを実行します。
`> SubsystemInstaller.exe --install`


以上で、インストールは終了です。

重要

▶ 本アプリをアップデートする場合は、アンインストール（→ P.8）をした後に本手順でアップデート版をインストールしてください。

■ インストール確認手順

□ メインボード部のインストール確認

- 1 「スタート」ボタン→（設定）→「アプリ」の順にクリックします。
- 2 画面左側のメニューで「アプリと機能」をクリックします。
- 3 画面右側の「アプリと機能」に「Model Management」が表示されていることを確認します。
- 4 Model Management の EXE が展開されていることを確認します。
`C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED¥SDK¥Basic¥Model Management¥Bin¥ModelMgmt.exe`

□ AI 拡張ボード部のインストール確認

搭載されている AI 拡張ボード部（#2 ～ #7）のインストールを確認します。

- 1 ssh コマンドで AI 拡張ボード部にログインします。
対象の AI 拡張ボード部のバーチャル LAN の IP アドレスを確認し、SSH で対象の AI 拡張ボード部にログインします。
例）ユーザー名「fujitsu」でログインして拡張ボード部（#2）にログインする場合、「コマンドプロンプト」で次のコマンドを入力します。
`> ssh fujitsu@192.168.1.102`
※ ログイン情報の詳細は、『管理者ガイド』で確認してください。
- 2 dpkg コマンドで、次のパッケージが AI 拡張ボード部にインストールされていること確認します。
対象の AI 拡張ボード部にログインした状態で確認してください。

[確認するパッケージ]

ansible
modelagent
auditd

例）ansible パッケージを確認する場合、確認するパッケージを grep コマンドで検索し確認してください。

```
> sudo dpkg -l | grep ansible
> ii  ansible 2.5.1+dfsg-1ubuntu0.1 all Configuration management, deployment, and task execution system
```

- 3 systemctl で、次のパッケージのサービスが、active (running) になっていることを確認します。
対象の AI 拡張ボード部にログインした状態で確認してください。

```
> sudo systemctl status modelagent
● modelagent.service - Model Management Agent
Loaded: loaded (/etc/systemd/system/modelagent.service; enabled; vendor preset: enabled)
Active: active (running) since Fri 2020-01-24 20:50:50 JST; 6min ago
```

※ status が、active (running) でない場合は、本製品の電源を再度入れた後、確認してください。

■ アンインストール

□ AI 拡張ボード部のアンインストール

- 1 「.¥SubsystemInstaller」フォルダーに移動し、アンインストールを実行します。
> SubsystemInstaller.exe --remove

□ メインボード部のアンインストール

- 1 管理者アカウントでサインインします。
- 2 管理者権限で「コマンドプロンプト」を起動します (→ P.4)。
- 3 本アプリが格納されているドライブ、フォルダーへ移動します。
- 4 次のコマンドを実行してサービスの登録を行います。
> Uninstall.bat
- 5 「はい」をクリックしアンインストールを実行します。

👉 重要

▶ Model Management アンインストールでの削除対象は次のディレクトリとレジストリです。
C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED¥SDK¥Basic¥Model Management 配下
¥HKEY_LOCAL_MACHINE¥SOFTWARE¥FUJITSU CLIENT COMPUTING LIMITED¥SDK¥Basic¥Model Management

Model Encryption Tool のインストール／アンインストール

Model Encryption Tool のインストール、アンインストール手順です。
本ツールは、AI モデルを暗号化するためのツールです。

■ ハードウェア／ソフトウェア条件

ハードウェア：NVIDIA Jetson 開発者キット、AI 拡張ボード部
ソフトウェア：UbuntuOS:18.04, NVIDIA® Jetpack™ 4.2.2

■ インストール

👉 重要

▶ 本手順は、AI モデルの暗号化を AI 拡張ボード部 (#2) で実行する場合の例です。
ユーザー開発環境で実行する場合は、modelencryption-1.0-1.deb をコピーし、手順5のコマンドでインストールしてください。
▶ アップデートする場合も、本手順を行ってください。

- 1 管理者アカウントでサインインします。
- 2 管理者権限で「コマンドプロンプト」を起動します (→ P.4)。
- 3 本アプリが格納されているドライブ、フォルダーへ移動します。
C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED¥SDK¥Basic¥Model Management¥tools¥Model Encryption Tool
- 4 次のコマンドを実行して AI 拡張ボードにインストールパッケージを転送します。
> scp modelencryption-1.0-1.deb <user>@<192.168.1.102>:~/.
- 5 AI 拡張ボードにログインしてパッケージをインストールします。
> ssh <user>@<192.168.1.102>
AI 拡張ボードにログイン後にインストーラーを実行します。
> sudo dpkg -i modelencryption-1.0-1.deb

以上でインストールは終了です。

■ インストール確認手順

- 1 AI 拡張ボードにログインし、Model Encryption フォルダーに実行モジュールが存在することを確認します。
> ls /opt/fcc/sdk/modelenc/
ls コマンドの出力内容に「ModelEncryptionTool」があることを確認してください。

■ アンインストール

- 1 AI 拡張ボードにログインし、次のコマンドを実行してアンインストールします。

```
sudo dpkg --remove modelencryption  
sudo dpkg --purge modelencryption
```

- 2 下記のコマンドを実行し、パッケージが完全になくなっていることを確認します。

```
sudo dpkg -l | grep modelencryption
```

上記コマンドの結果一覧で何も出てない場合は、アンインストールが正常に終了しました。

3. 環境変数

ModelMgmt.exe のパスを環境変数に設定しておくくと便利です。ここでは PATH の設定方法について説明します。

- 1 「コントロール パネル」ウィンドウ (→ P.4) を表示します。
- 2 「システムとセキュリティ」→「システム」の順にクリックします。
- 3 画面左側の「システムの詳細設定」をクリックします。
「システムのプロパティ」が表示されます。
- 4 「詳細設定」タブで「環境変数」をクリックします。
- 5 「システム環境変数」で「Path」変数を選択し、「編集」をクリックします。
- 6 「新規」をクリックし、Model Management の実行ファイルのパスを追加します。
追加するパス：C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\Basic\Model Management\Bin
- 7 「OK」をクリックします。

これで、コンピューターの任意のディレクトリから ModelMgmt.exe の実行にアクセスできます。
更新された変数を取得するには、「コマンドプロンプト」の新しいインスタンスを起動してください。

2

第 2 章 暗号化モデル導入の流れ

暗号化モデルの導入方法について説明します。

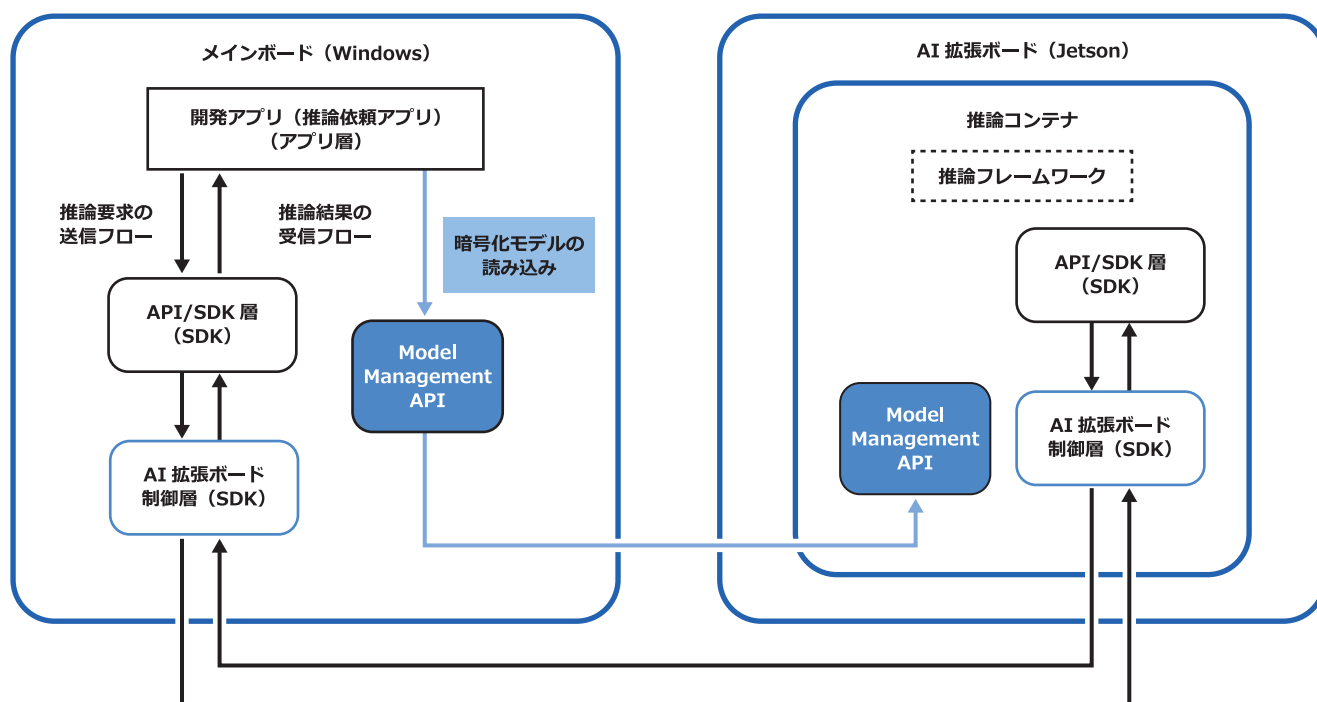
1. Model Management データフロー	11
2. 暗号化モデルの導入と推論までのフロー	12

本導入手順は、SDK Distributed Manager の開発者を対象に説明しています。

- Model Management データフロー
- 暗号化モデルの導入と推論までのフロー
- Model Encryption Tool を使ったユーザーの AI モデルの暗号化方法(ユーザー開発環境)
- 暗号化した AI モデルの Docker コンテナ作成方法 (ユーザー開発環境)
- Model Management の準備と Docker コンテナの登録 (本製品上で操作)
- サンプルプログラムの動作 (本製品上で操作)

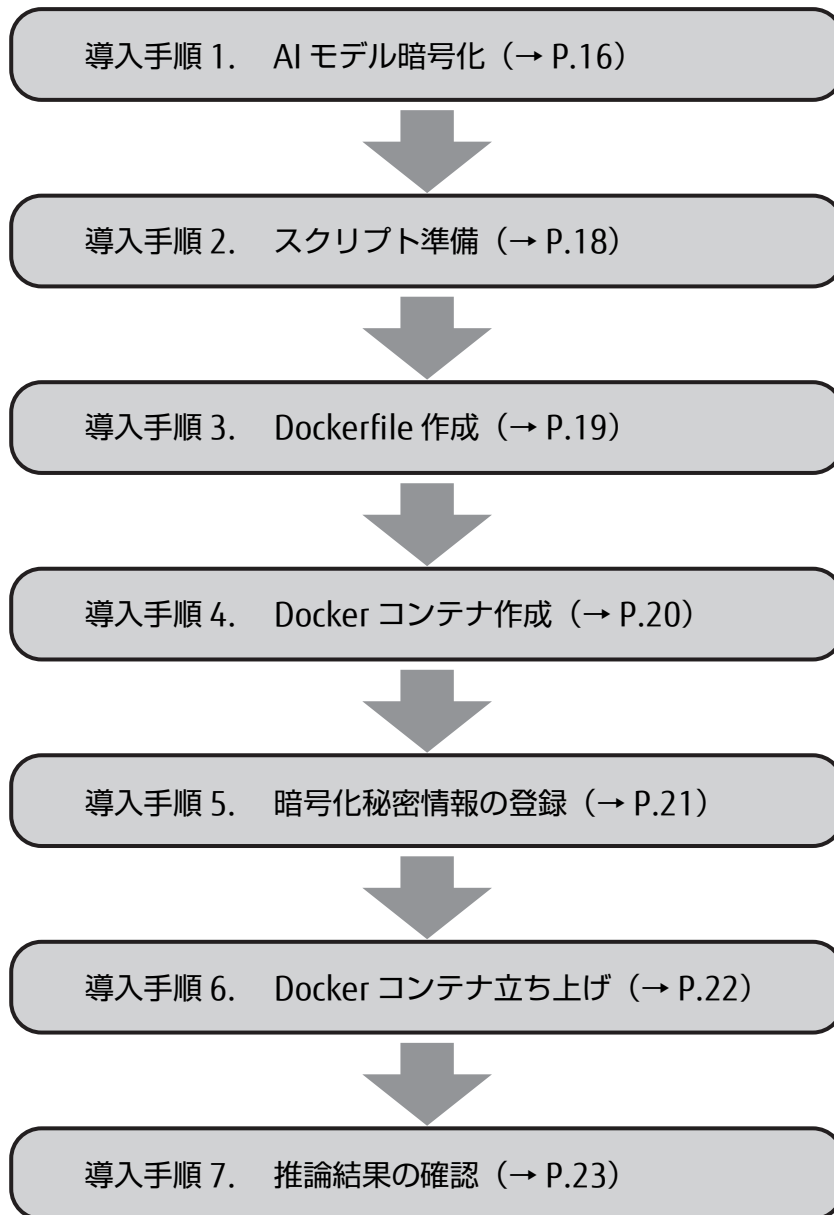
1. Model Management データフロー

推論コンテナ上で動作する推論プログラムを開発するには、SDK Distributed Manager のパッケージ式に含まれる Connector クラスを使用します。Connector クラスとは、メインボードと推論コンテナが通信するための API を備えたクラスです。推論実行時のメインボードと推論コンテナのデータフローは下図のようになります。Model Management API の loaddockerimage で、推論コンテナをデプロイします。また、startdockercontainer で、暗号化モデルが復号化され利用可能になります。



2. 暗号化モデルの導入と推論までのフロー

導入手順については、第4章「Model Management のご使用の流れ」(→ P.15) の各項目をご覧ください。



3

第 3 章

AI 拡張ボードの再セットアップ

メインボードの交換、AI 拡張ボードの交換などの構成変更やリカバリに伴う、Model Management の再セットアップ手順を説明します。

Model Management を利用しているユーザーは、暗号化秘密情報を再登録するため本章の再設定手順を行ってください。

1. 再セットアップ手順.....	14
-------------------	----

1. 再セットアップ手順

メインボード交換

修理でメインボード交換した場合、次の手順を行う必要があります。

■ メインボード側のシステムイメージ復元後の Model Management 再セットアップ手順

- 1 メインボード交換後にシステムイメージを復元します。
システムイメージの復元については、『管理者ガイド』の第4章「バックアップと復元」－「メインボード部のバックアップと復元」をご覧ください。
- 2 Model Management AI 拡張ボード部をアンインストールします。
- 3 Model Management AI 拡張ボード部をインストールします。
- 4 管理者権限で「コマンドプロンプト」を起動します（→ P.4）。
- 5 「1.1.22 ModelInfo - create」コマンドで、暗号化秘密情報を登録します。
ユーザーが使用する暗号化モデルの暗号化秘密情報を本コマンドで再度登録してください。
[コマンドプロンプト（管理者権限）]
`ModelMgmt.exe -c create -i {"table":"","modelinfo":"","filepath":"","C:¥¥model¥¥mymodel_v1_model.txt¥¥"}`

■ メインボード側でご購入時の状態の場合の Model Management 再セットアップ手順

- 1 Model Management メインボード部をインストールします。
- 2 Model Management AI 拡張ボード部をインストールします。
- 3 管理者権限で「コマンドプロンプト」を起動します（→ P.4）。
- 4 「1.1.22 ModelInfo - create」コマンドで、暗号化秘密情報を登録します。
ユーザーが使用する暗号化モデルの暗号化秘密情報を本コマンドで再度登録してください。
[コマンドプロンプト（管理者権限）]
`ModelMgmt.exe -c create -i {"table":"","modelinfo":"","filepath":"","C:¥¥model¥¥mymodel_v1_model.txt¥¥"}`

AI 拡張ボードの構成変更

AI 拡張ボードの交換や取り付け／取り外しなどの構成変更、リカバリなどによる変更後、次の手順を行う必要があります。

■ AI 拡張ボードの構成変更後の再セットアップ手順

- 1 Model Management AI 拡張ボード部をアンインストールします。
- 2 再セットアップします。
AI 拡張ボードの再セットアップについては、『管理者ガイド』の第3章「再セットアップ（AI 拡張ボード）」をご覧ください。
- 3 Model Management AI 拡張ボード部をインストールします。
- 4 管理者権限で「コマンドプロンプト」を起動します（→ P.4）。
- 5 「1.1.22 ModelInfo - create」コマンドで、暗号化秘密情報を登録します。
ユーザーが使用する暗号化モデルの暗号化秘密情報を本コマンドで再度登録してください。
[コマンドプロンプト（管理者権限）]
`ModelMgmt.exe -c create -i {"table":"","modelinfo":"","filepath":"","C:¥¥model¥¥mymodel_v1_model.txt¥¥"}`

4

第 4 章

Model Management のご使用の流れ

AI モデルの暗号化から、実際にユーザー環境で利用するまでの導入手順を説明します。

1. AI モデル暗号化	16
2. スクリプトの準備	18
3. Dockerfile 作成	19
4. Docker コンテナ作成	20
5. 暗号化秘密情報の登録	21
6. Docker コンテナ立ち上げ	22
7. 推論結果の確認	23

1. AI モデル暗号化

Model EncryptionTool を使って AI モデルの暗号化手順を記載しています。

ここでは、Model EncryptionTool のヘルパースクリプトの menctool を使った AI モデルの暗号化手順について説明します。

```
192.168.1.102 - fujitsu@copro: ~ VT
File Edit Setup Control Window Help
fujitsu@copro:~$ menctool
Default Values for ModelEncryptionTool
Input Model Path: /home/fujitsu/model
Output Model Path: /home/fujitsu/modelenc.tar.enc
Model Name: sample
Model Version: v1.0
Do You Wish To Use Default Values For Encrypting Model Data?: n
Enter Absolute Input Model Folder Path: /home/fujitsu/project/model
Enter Absolute Output Model Path with filename in .tar.enc format: /home/fujitsu/
ssd_inception_2017_without_relu6.tar.enc
Enter Model Name: ssdmodel
Enter Model Version: v4.2.2
Enter Input Password (Eg. SaMplE123):
Enter Password Again (Eg. SaMplE123):
fujitsu@copro:~$ tail -5 ModelEnc.log
2020-05-06 17:28:52 INFO encryption.c:381: Checksum: 0883108856685f5d

2020-05-06 17:28:52 INFO encryption.c:382: Output Filename:ssd_inception_2017_without_relu6
.tar.enc
2020-05-06 17:28:52 TRACE encryption.c:385: MODEL ENCRYPTION TOOL DONE WITH ENCRYPTION
ENABLED
fujitsu@copro:~$
```

■ 環境

NVIDIA Jetson 開発者キット、AI 拡張ボード部

■ 操作手順

- 1 Model EncryptionTool がインストールされた環境で、「menctool」コマンドを実行します。

```
192.168.1.102 - fujitsu@copro: ~ VT
File Edit Setup Control Window Help
fujitsu@copro:~$ menctool
```

- 2 コマンド実行後、次のパラメータが要求されます。ユーザーの環境に合わせて入力してください。

```
Default Values for ModelEncryptionTool
Input Model Path: /home/fujitsu/model
Output Model Path: /home/fujitsu/modelenc.tar.enc
Model Name: sample
Model Version: v1.0
```

◆ 入力パラメータ

Input Model Path : 暗号化させるモデルファイルまたは、フォルダーのフルパスを指定します。
Output Model Path : 暗号化されたモデルの出力先とファイル名を指定します。また、このパスはコンテナの中でのモデルパスになります。
Model Name : モデル名を指定します。ユニークな名称を任意で指定します。
Model Version : モデルバージョンを指定します。ユニークなバージョンを任意で指定します。

※ 「Model Name」、「Model Version」は、Docker コンテナ作成時にここで登録したパラメータと同じ名称にする必要があります。

- 3 「y」を入力します。

```
Input Model Path: /home/fujitsu/model
Output Model Path: /home/fujitsu/modelenc.tar.enc
Model Name: sample
Model Version: v1.0
Do You Wish To Use Default Values For Encrypting Model Data?: y
```

- 4 暗号化ファイルと暗号化秘密情報が生成されます。

Output Model Path で指定した出力パスに以下が生成されます。

1. 暗号化 AI モデル (例 /home/ibadmin/ssd_inception_2017_without_relu6.tar.enc)
2. 暗号化秘密情報 (例 .mymodel_v1_model.txt)

※/usr/bin/menctool/ModelEnc.log に実行結果がロギングされます。

■ 実行例

```

192.168.1.102 - fujitsu@copro: ~ VT
File Edit Setup Control Window Help
fujitsu@copro:~$ menctool
Default Values for ModelEncryptionTool
Input Model Path: /home/fujitsu/model
Output Model Path: /home/fujitsu/modelenc.tar.enc
Model Name: sample
Model Version: v1.0
Do You Wish To Use Default Values For Encrypting Model Data? : y
Enter Input Password (Eg. SaMplE123):
Enter Password Again (Eg. SaMplE123):
fujitsu@copro:~$ tail -5 ModelEnc.log
2020-05-06 17:33:03 INFO encryption.c:381: Checksum: 36c64307fb26920b

2020-05-06 17:33:03 INFO encryption.c:382: Output Filename: modelenc.tar.enc

2020-05-06 17:33:03 TRACE encryption.c:385: MODEL ENCRYPTION TOOL DONE WITH ENCRYPTION
ENABLED
fujitsu@copro:~$

```

■ エラー情報

ModelEnc.log にロギングされるエラー内容と対処方法について説明します。

エラーメッセージ	エラー詳細	エラー対処方法
Failed To Create Logger File	作業フォルダーにログファイルの生成に失敗しました。	作業フォルダーにファイル作成の権限があるか確認してください。
System Call Error [Syscall name will be written in logs with errno]	TAR コマンドで実行エラーが発生しました。	詳しくは ModelEnc.log で確認してください。TAR コマンド実行時のエラー情報から必要な対処を行ってください。
Failed to Tar File	TAR コマンドでモデルファイルの暗号化に失敗しました。	対象ファイルまたは、作業フォルダーの書き込み権限を確認してください。

■ 参考

ModelEncryptionTool のコマンド仕様です。menctool を利用せず、本コマンドを実行することで同様のことができます。ユーザー自身でシェルスクリプトを作成する場合にご利用頂けます。

形式 : ./ModelEncryptionTool [OPTIONS]

Note :For Encryption : -p <numeric value> -i -o -n -v が必須
:-x が任意

For Non-Encryption: -p <numeric value> -i are required

[引数一覧] :

- p <1 : 暗号化する, 0 : 暗号化しない>
- x <ユーザーが登録する AI モデルのパスワード (英数字 大文字 / 小文字) >
- n <モデルの名前>
- v <モデルのバージョン>
- i <対象モデルの絶対パス>
- o <暗号化済みの TAR を置く場所のパス (絶対パス)>
- h <ヘルプ>

例) ./ModelEncryptionTool -p 1 -z userpwd -n ssd -i /home/ibadmin/ssd_model -o /home/ibadmin/ssd_model_output/ssd_model.tar.enc -v v2

```

192.168.1.102 - fujitsu@copro: ~ VT
File Edit Setup Control Window Help
fujitsu@copro:~/project$ cd /opt/fccl/sdk/modelenc/
fujitsu@copro:/opt/fccl/sdk/modelenc$ ./ModelEncryptionTool -p 1 -z userpwd -n ssd -i /home/
fujitsu/project/model -o /home/fujitsu/project/output/ssd_model.tar.
enc -v v4.2.2
fujitsu@copro:/opt/fccl/sdk/modelenc$ cd /home/fujitsu/project/output
fujitsu@copro:~/project/output$ ll
total 16
drwxrwxr-x 2 fujitsu fujitsu 4096 May  6 17:41 ./
drwxrwxr-x 4 fujitsu fujitsu 4096 May  6 17:41 ../
-rw-rw-r-- 1 fujitsu fujitsu 128 May  6 17:41 ssd_model.tar.enc
-rw-rw-r-- 1 fujitsu fujitsu 218 May  6 17:41 ssd_v4.2.2_model.txt

```

2. スクリプトの準備

メインボード部から推論依頼するスクリプトと推論結果を送信するスクリプトの準備をします。

本導入手順は、SDK Distributed Manager の開発者を対象に説明しています。

本導入手順は、SDK Distributed Manager を利用中のユーザーが Model Management 機能を導入するためのポイントをサンプルコードで変更差分について説明します。

本手順を参考にユーザーのスクリプトを変更してください。

■ 環境

NVIDIA Jetson 開発者キット、AI 拡張ボード部

■ 操作手順

ここでは、Output Model Path で「/tmp/sdkdata/.4152/」を指定した場合を例に挙げて説明します。

1 メインボード部の開発アプリ（推論依頼アプリ）の Docker 制御処理を Model Management の実装に置き換えます。

1. ModelMgmt でコンテナをロードする
2. ModelMgmt でコンテナを開始する

The image shows two code editor windows. The left window, titled 'intel_aiclustertest.py', contains the following code snippet:


```
# create container config
log.debug("#####load_inference#####")
config = ContainerConfig(name = inference_name, implementation_type = 'docker',
    image_path = container_path, image_filename = file_name, image_tag = tag)
```

 The word 'docker' is highlighted with a red box. The right window, titled 'intel-emc.py', contains a similar snippet:


```
# create container config
log.debug("#####load_inference#####")
config = ContainerConfig(name = inference_name, implementation_type = 'other',
    image_path = container_path, image_filename = file_name, image_tag = tag)
```

 The word 'other' is highlighted with a red box.

2 AI 拡張ボード部の推論結果の送信スクリプトに Model Management の実装を追加します。

1. モデルのパス（復号化済みのモデルは /tmp/sdkdata/.4152/ の配下に格納される）
2. モデル復号化までの待ちするコード

The image shows two code editor windows. The left window, titled 'jetson_code.txt', contains the following code snippet:


```
aiclustest = AIClustest()

# create container config
log.debug("#####load_inference#####")
config = ContainerConfig(name = inference_name, implementation_type = 'docker',
    image_path = container_path, image_filename = file_name, image_tag = tag)

# load container to coprocessor
containers = aiclustest.load_inference(container_config=config, coprocessor_name = copro)

# logging
for i, container in enumerate(containers):
    log.debug("container {} -> name: {}, allocated: {}".format(i, container.name, container.allocated_coprocessor))
```

 The right window, titled 'jetson_code_emc.txt', contains a similar snippet but with additional code for Model Management:


```
aiclustest = AIClustest()
# create container config
log.debug("#####load_inference#####")
config = ContainerConfig(name = inference_name, implementation_type = 'other',
    image_path = container_path, image_filename = file_name, image_tag = tag)
# load container to coprocessor #LUNENcrypted
containers = aiclustest.load_inference(container_config=config, coprocessor_name = copro)
if hostaddress == "192.168.1.102":
    iContainerName = 'enc_sample'
    ModelMgmtExe = 'ModelMgmt.exe'
    ImageName = 'ssdmodel:4.2.2'
    # logging
    for i, container in enumerate(containers):
        log.debug("container {} -> name: {}, allocated: {}".format(i, container.name, container.allocated_coprocessor))
    print("Starting.. load container")
    code, out, err = run(["ModelMgmt.exe", '-c', 'loaddockerimage', '-i', {"hostaddress": "192.168.1.102", "imagepath": "C:\\\\i"}])
    if code < 0:
        print("err: {}".format(err))
    print("Starting.. create container")
    code, out, err = run(["ModelMgmt.exe", '-c', 'createdockercontainer', '-i', {"hostaddress": "192.168.1.102", "optionalpar"}])
    if code < 0:
        print("err: {}".format(err))
    print("Starting.. start container")
    code, out, err = run(["ModelMgmt.exe", '-c', 'startdockercontainer', '-i', {"hostaddress": "192.168.1.102", "modelName": ""}])
    if code < 0:
        print("err: {}".format(err))
```

 The code between the 'load container' and 'start container' sections is highlighted with a red box.

3. Dockerfile 作成

ユーザーのアプリ環境に合わせた Docker 推論コンテナ環境を準備してください。
ここでは、Model Management で暗号化モデルを利用する場合に必要なファイルを Dockerfile に追加します。

■ 環境

NVIDIA Jetson 開発者キット、AI 拡張ボード部

■ 操作手順

1 ユーザーのアプリ環境に合わせた Docker 推論コンテナ環境の Dockerfile を準備します。

2 Dockerfile に暗号モデル関連のファイルを追加します。

◆ 追加ファイル

復号化関連のライブラリ : ModelDecryptionTool の復号化用の依存ライブラリを追加します。

apt-get install openssl libssl-dev

(OpenSSL のライブラリのインストール時に自動で libcrypto.so.1.0.0 のモジュールもインストールされます。)

暗号化モデル名 : 「1. AI モデル暗号化」(→ P.16) で指定した、「Output Model Path」を指定してください。

復号化ツール : ModelDecryptionTool の追加と、実行権限の付与
ModelDecryptionTool は起動時にコンテナのデフォルトユーザーで実行権限を与えてください。

3 手順 2 の変更内容を以下に記載します。参考にユーザー環境の Dockerfile に追加してください。

◆ 【変更前】

```
FROM aarch64:ubuntu16.04

ENV DEBIAN_FRONTEND=noninteractive

RUN apt-get update
RUN apt-get install -y software-properties-common && apt-get install -y apt-utils

RUN useradd -ms /bin/bash fujitsu
RUN apt-get install -y vim python3 python3-pip

COPY init.py /home/fujitsu/init.py
COPY modelfolder /home/fujitsu/

RUN chown -R fujitsu /home/fujitsu

USER fujitsu

ENTRYPOINT python3 /home/fujitsu/init.py
```

◆ 【変更後】

```
FROM aarch64:ubuntu16.04

ENV DEBIAN_FRONTEND=noninteractive

RUN apt-get update
RUN apt-get install -y software-properties-common && apt-get install -y apt-utils

RUN useradd -ms /bin/bash fujitsu
RUN apt-get install -y vim python3 python3-pip openssl libssl-dev

COPY ModelDecryptionTool /home/ibadmin/.model_tool/ModelDecryptionTool

COPY init.py /home/fujitsu/init.py
COPY ssd_inception_2017_without_relu6.pb.tar.enc /home/fujitsu/ssd_inception_2017_without_relu6.pb.tar.enc

RUN chmod a+x fujitsu /home/ibadmin/.model_tool/ModelDecryptionTool
RUN chown -R fujitsu /home

USER fujitsu

ENTRYPOINT python3 /home/fujitsu/init.py
```

■ 実行例

1 対象の暗号化したモデルのパスを次のコマンドで確認できます。

※ 事前に、暗号化秘密情報を登録 (→ P.21) しておく必要があります。

Dockerfile に追加する暗号化したモデル名は、コマンドプロンプトで次のコマンドを実行することで確認できます。

ModelMgmt.exe -c list -i {"table":"modelinfo"}

◆ 出力例

modelpath:/home/ibadmin/ssd_inception_2017_without_relu6.pb.tar.enc

```
C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\BasicSDK Distributed Manager\IBSDK\Sample\SDK_Operation_Verification>ModelMgmt.exe -c list -i {"table":"modelinfo"}

{"data":{"checksum":"","iscontainer":"true","iv":"","modelName":"twistlock/authbroker","modelpath":"C:\\Program Files\\FUJITSU CLIENT COMPUTING LIMITED\\SDK\\Basic\\Model Management\\data\\plugin-v1.0.2.tar.gz","modelversion":"v1.0.2","x":"","y":"","z":"","table":"modelinfo"}}

{"data":{"checksum":"dd79bf32c45cc207","iscontainer":"false","iv":"","mHstBstOcUpZytwQ","modelName":"ssdmodel","modelpath":"/home/ibadmin/ssd_inception_2017_without_relu6.pb.tar.enc","modelversion":"v4.2.2","x":"406CE54B0867ADDEBF2E4F5D5C098421C93932F6D0406BCA0E8BA0592B5A8093198D10AC956B54D5E339BFDCEB0CABAF3331B916C21B93F63618A548A95604B7343A0C5A6C32AE45A1C7FDCF105100135DF76B6E507A9FF5C879221703D0A1EA86A9B0AB3CA4D93F25CE5ACCF86075F1AB1815A16076B9E18CBA8BA4C2606E9D73FB891FC1A5933FA84FAE83EB72BB731D02096F17DD5C170DD97757EA8E923A4D01AF184820FFC45FB0094A7FF8AAF622C6AEDF0499F9079FD02C6D28305301859C18FFCE0E34CAEA0BB934E8FFF1787C1631D31FC594B62F3F385BA20B142530260482AA21FF2E0A3378A2C77C6BD13B440DB49BE2A09BEA00FCB88703F8","y":"A3B4C7ECFBAB329FEF3DE07A4D7A9677CAA4D989C9EAA8BF66E2B4E968EAC881787911184322074CFA3C7B0E6FA447130B8841C47C3D99C9D266A6753D287DB68AB89092539EB58F494230C24E53C14AF26A8CFA1DB5760CEDC2F83E40D28E59CF4D1749510984D0CB0621932C352325DECCA87D9C94332742273FC580970517670CF2D52D0CC1488517803960F750FFCCD4FCE2A693C959B29223736F1C382B9D2D1887E9FEC3E424E1D62368F4DCA773B09E18CBF4B75F206F5F877A01430E2F9738B619291B4D605CF76F943A053F87CF7513AD2597D2045E4BA094D0E8B65C7BC467392353101C8B3378A20D49A26DD68A3D735F95329A6DD185C41BE47","z":"40596B73CD181F4CD854AFB0D39DC2F01880DE06778C26AE3F9D0FD8C42B5D3D9815D0B6C99147A64726597A3272F431E2C36C2D6FCF1F819F485FB97A71612B2262FF1FC9E087DE455769A61C5067A7CFE1C1CB231786B832BA13C2326D91466B6F9B6470F33FD89A2EAD5770CD9E4575AFDEEA223634615FDED666F2586F3CE06E7F321A7B58AF8406B1C49166AA7029D8EF433075AB6C57FC7196B8E8F37605EAE4516749D0781713C04123E359B5BF8C93180635EAAA0BE353D3BDDF4BA5D017C6155A05208493A8088E145A48383351F3D840441B6BCB0BD7A4F13B5EC915A8C20EB9767C04C04B5ABA2A0A19C853C47F4A18D50A78CF364E33872E"},"table":"modelinfo"}
```

4. Docker コンテナ作成

ユーザーのアプリ環境に合わせた Docker 推論コンテナ環境を準備してください。

ここでは、docker image build コマンド を利用して、「3. Dockerfile 作成」(→P.19)で準備した Dockerfile から Docker コンテナを tar ファイルで保存します。

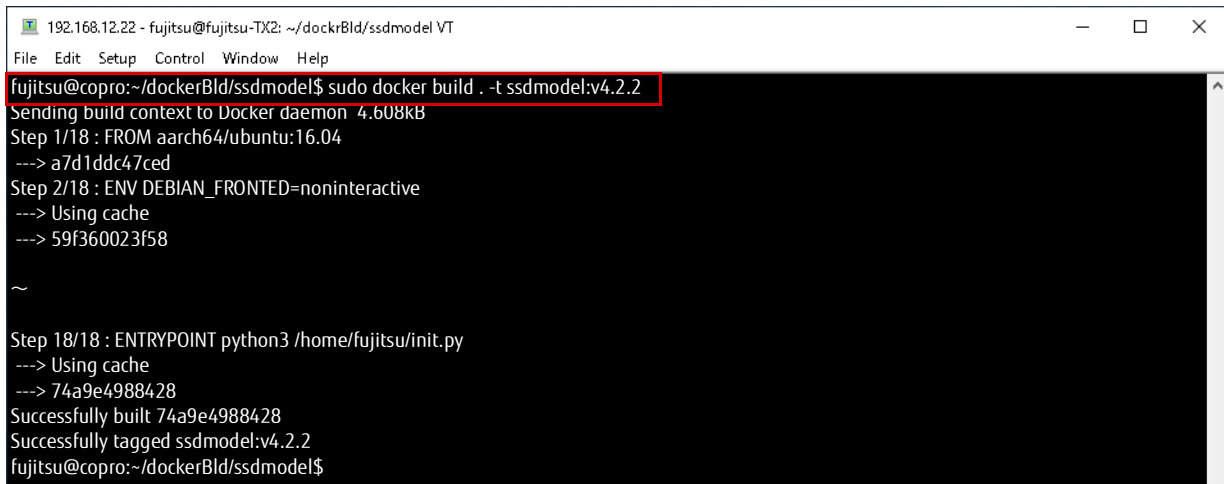
■ 環境

NVIDIA Jetson 開発者キット、AI 拡張ボード部

■ 操作手順

- 1 「3. Dockerfile 作成」(→P.19) で準備した、Dockerfile から docker image build コマンドでビルドします。

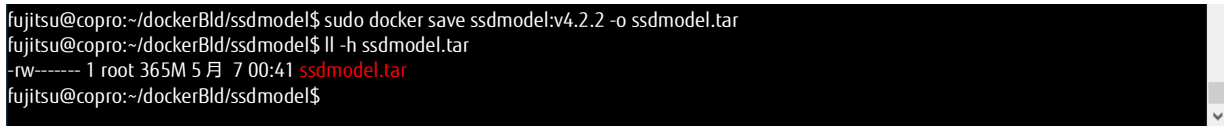
sudo docker build -t <イメージ名>:<バージョンタグ> .



```
192.168.12.22 - fujitsu@fujitsu-TX2: ~/dockerBld/ssdmodel VT
File Edit Setup Control Window Help
fujitsu@copro:~/dockerBld/ssdmodel$ sudo docker build . -t ssdmodel:v4.2.2
Sending build context to Docker daemon 4.608kB
Step 1/18 : FROM aarch64/ubuntu:16.04
--> a7d1ddc47ced
Step 2/18 : ENV DEBIAN_FRONTEND=noninteractive
--> Using cache
--> 59f360023f58
~
Step 18/18 : ENTRYPOINT python3 /home/fujitsu/init.py
--> Using cache
--> 74a9e4988428
Successfully built 74a9e4988428
Successfully tagged ssdmodel:v4.2.2
fujitsu@copro:~/dockerBld/ssdmodel$
```

- 2 手順 1 でビルドされた docker イメージをビルド docker save コマンドで tar ファイルに保存します。

sudo docker save <イメージ名>:<バージョンタグ> -o <tar ファイル名>



```
fujitsu@copro:~/dockerBld/ssdmodel$ sudo docker save ssdmodel:v4.2.2 -o ssdmodel.tar
fujitsu@copro:~/dockerBld/ssdmodel$ ll -h ssdmodel.tar
-rw-r--r-- 1 root 365M 5月 7 00:41 ssdmodel.tar
fujitsu@copro:~/dockerBld/ssdmodel$
```

5. 暗号化秘密情報の登録

「1. AI モデル暗号化」(→ P.16) で生成した暗号化秘密情報と、「4. Docker コンテナ作成」(→ P.20) で生成した Docker コンテナを Model management に登録します。

これにより、登録済みの暗号化された Docker コンテナは、Model Management の「startdockercontainer」コマンドで、自動で AI モデルが復号化され利用できます。

■ 環境

Infini-Brain 本体

■ 操作手順

- 1 「1. AI モデル暗号化」(→ P.16) で生成した暗号化秘密情報と、「4. Docker コンテナ作成」(→ P.20) で生成した Docker コンテナをメインボード部にコピーします。

1. 暗号化 AI モデル (Ex. ssd_model.tar.enc)
2. 暗号化秘密情報 (Ex. mymodel_v1_model.txt) ※ 暗号化秘密情報は難読化されています。

ユーザーの開発環境からメインボード部に USB またはネットワーク経緯で任意フォルダーにコピーしてください。

- 2 暗号化秘密情報を ModelManagement コマンドで、登録します。

```

C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\Basic\SDK Distributed Manager\IBSDK\Sample\SDK_Operation_Verification>ModelMgmt.exe -c list -i [{"table":"modelinfo"}]

C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\Basic\SDK Distributed Manager\IBSDK\Sample\SDK_Operation_Verification>ModelMgmt.exe -c create -i [{"table":"modelinfo","filepath":"C:\\img\\ssdmodel_v422_model.txt"}]
Data Inserted Successful

C:\Program Files\FUJITSU CLIENT COMPUTING LIMITED\SDK\Basic\SDK Distributed Manager\IBSDK\Sample\SDK_Operation_Verification>ModelMgmt.exe -c list -i [{"table":"modelinfo"}]
{"data":{"checksum":"","iscontainer":true,"iv":"","modelname":"twistlock/authzbroker","modelpath":"C:\\Program Files\\FUJITSU CLIENT COMPUTING LIMITED\\SDK\\Basic\\Model Management\\data\\plugin-v1.0.2.tar.gz","modelversion":"v1.0.2","x":"","y":"","z":""},"table":"modelinfo"}

{"data":{"checksum":"dd79bf32c45cc207","iscontainer":"false","iv":"mHstBst0cUpZYtwQ","modelname":"ssdmodel","modelpath":"/home/ibadmin/ssd_inception_2017_without_relu6.pb.tar.enc","modelversion":"v4.2.2","x":"406CE54B0867ADDEBF2E4F5D5C098421C93932F6D0406BCA0EBBA0592B5A8093T98D10AC956B54D5E339BFDCEB0CABAF3331B916C21B93F63618A548A95604B7343A0C5A6C32AE45A1C7FDCF105100135DF76B8E507A9FF5C879221703D0A1EA86A9B0AB3CA4D93F25CE5ACCF86075F1AB1815A16076B9E18CBABBA4C2606E9D73FBB91FC1A5933FA84FAE83EB72BB731D02096F17DD5C170DD97757EA8E923A4D01AF184820FFC45FB0094A7FF8A4AF622C6AEDF0499F9079FD02C6D28305301859C13FFCE0E34CAEAOBB934E8FFF1787C1631D31FC594B62F3F385B420B142530260482AA21FF2E0A3378A2C77C6BD13B440DB49BE2A09BEA00FC8AB3703F8","y":"A3B4C7ECFBAB329FEF3DE07A4D7A9677CA4D989C9EAABF66E2B4E968EAC881787911184322074CFEA3C7B0E6FA447130B8841C47C3D99C9D266A753D287DB68AB89092539EB58F494230C24E53C14AF26A8CFA1DB5780CEDC2F83E40D28E59C4FD1749510984D0CB0621932C352325DECCA87D9C94332742273FC580970517670CF2D52D0CC1488517B03960F750FFCCD4FCE2A693C959B29223F36F1C382B92D21887E9FEC3E424E1D62368F4DCA773B09E18CBF4B75F206F5F877A01430E2F9738B619291B4D605CF76F943A053F87CF7513AD2597D2045E4BA094D0E8B65C7C8C467392353101C8B3378A20D49A26DD68A3D735F95329A6DD185C41BE47","z":"40596B73CD181F4CD854AFB0D39DC2F01880DE06778C26AE3F9D0FFD8C42B5D3D9815D0B6C99147A64726597A3272F431E2C36C2D6FCF1F819F485FB97A71612B2262FF1FC9E087DE455769A61C5067A7CFE1C1CB2317868B32BA13C2326D91466B6F9B6470F33FD89A2EAD5770CD9E4575AFDEA223634615FDEDD666F2586F3CC0E6E7F321A7B58AF8406B1C49166AA7029D8EF4330754B6C57FC7196B8E8F37605EAAE4516749D0781713C04123E359B5BF8CC93180635EAA0BE353D3BBD4BA5D017C6155A05208493A8088E145A48383351F3D840441B6BCB0BD7A4F13B5EC915A8C20EB9767C04C04B5ABA2A0A19C853C47F4A18D50A78CF364E33872E"},"table":"modelinfo"}

```

6. Docker コンテナ立ち上げ

「1. AI モデル暗号化」(→ P.16) で生成した暗号化秘密情報と、「4. Docker コンテナ作成」(→ P.20) で生成した Docker コンテナを Model management に登録します。

これにより、登録済みの暗号化された Docker コンテナは、Model Management の「startdockercontainer」コマンドで、自動で AI モデルが復号化され利用できます。

- Docker コンテナ load
- SDK Distributed Manager の AI Cluster に登録
- Docker コンテナ create
- Docker コンテナ start

■ 環境

Infini-Brain 本体

■ 操作手順

ここでは、Output Model Path で「/tmp/sdkdata/.4152/」を指定した場合を例に挙げて説明します。

1 ModelManagement の API を使って Load します。

◆ 実装サンプル

```
load_command='ModelMgmt.exe -c loaddockerimage -i "%{hostaddress}%":%192.168.1.102%", "%imagepath%":%C:%img%ssdimage3.tar %%"'
os.system(load_command)
```

2 AI Cluster に登録

下記の Python コードを使って ai_model の inference_name を登録しないと推論受信待ちのコンテナが利用できません。
下記のように inference_name を登録します。

◆ 実装サンプル

```
copro = 'subsys2'
inference_name = 'object_detection'
config = ContainerConfig(name = inference_name, implementation_type = 'other')
containers = aicuster.load_inference(container_config=config, coprocessor_name = co
```

3 ModelManagementTool の API を使ってコンテナを AI 拡張ボードに生成します。

◆ 実装サンプル

```
os.system('"' + self.modelMgmtExe_path + ' -c createdockercontainer -i "%{hostaddress}%":%192.168.1.102%", "%optionalparams%":%192.168.1.102%", "%device%":%dev/nvhost-ctrl-gpu --device=/dev/nvhost-ctrl-gpu --device=/dev/nvhost-prof-gpu --device=/dev/nvmap --device=/dev/nvhost-gpu --device=/dev/nvhost-as-gpu -v /usr/local/cuda:/usr/local/cuda -v /usr/local/cuda-10.0:/usr/local/cuda-10.0 --env PYTHONPATH=$PYTHONPATH:/opt/ -env LD_LIBRARY_PATH=/usr/local/cuda-10.0/lib64:/usr/lib/aarch64-linux-gnu:/usr/lib/aarch64-linux-gnu/tegra --log-driver=syslog --env ai_name=%self.containerName% --env ai_hostip=%self.aiHostip% --env ai_pubport=%self.aiPubport% --env ai_subport=%self.aiSubport% --env ai_rcvhw=%self.aiRcvhw% --env ai_sndhw=%self.aiSndhw% --name=mini2 --rm %self.container_Tag% python3 /ssd_before/sdk_jetsonSide.py %%"')"
```

4 ModelManagementTool の startdockercontainer を使って停止中のコンテナを起動します。

ModelManagementTool の startdockercontainer は自動的に秘密情報を使ってモデルの復号化します。
暗号化モデルを使用する前に「5. 暗号化秘密情報の登録」(→ P.21) で、暗号化秘密情報を登録する必要があります。
復号化された AI モデルは、Docker コンテナの「/tmp/sdkdata/.4152」配下に展開されます。

◆ 実装サンプル

```
ModelMgmt.exe -c startdockercontainer -i "%{hostaddress}%":%192.168.1.102%", "%optionalparams%":%myssd%"
```

7. 推論結果の確認

■ 推論結果確認方法

- 1 AICluster のサービスを開始します。
- 2 推論依頼スクリプトをメインボードに起動します。
スクリプトの中では
 - ・ Docker イメージの tar をロードします。
 - ・ コンテナの開始と自動復号化します。
 - ・ AI 拡張ボードのスクリプトは推論待ち状況になって依頼が来たら結果を zmq で返します。
 - ・ メインボードのスクリプトは結果を必要な形式に格納します。

Infini-Brain A101/B, A101/BH
SDK 使い方ガイド補足情報（Model Management の使い方）

B6FY-5131-01 Z0-00

発行日 2020 年 6 月
発行責任 富士通株式会社

〒105-7123 東京都港区東新橋 1-5-2 汐留シティセンター

- このマニュアルの内容は、改善のため事前連絡なしに変更することがあります。
- このマニュアルに記載されたデータの使用に起因する第三者の特許権およびその他の権利の侵害については、当社はその責を負いません。
- 無断転載を禁じます。