



# Getting started with Stellar Studio AI plugin for Artificial Intelligence (AI)

## Introduction

This user manual delineates the procedural framework for constructing a comprehensive AI project tailored for Stellar microcontrollers within the Stellar Studio IDE environment. It encompasses a systematic approach to the automatic transposition of pre-trained neural networks (NN) into an efficiently optimized library and delineates the integration process within the project. The StellarStudio.AI module, a constituent fully compatible with the Stellar Studio suite, is the focal point of this documentation.

The core of this manual is a practical tutorial that guides users through the configuration of the StellarStudio.Al module. It provides detailed instructions on expediting the creation of a Stellar Al centric project. Users will learn to navigate the StellarStudio.Al's features and leverage its capabilities to streamline project development.

StellarStudio.AI is built upon the widely recognized ST Edge AI Core command-line interface (CLI) technology, known as ST Edge AI. This technology is specifically designed for deployment across various STMicroelectronics devices including STM32, Stellar MCUs and MEMS. Comprehensive documentation on the ST Edge AI Core technology is included within the final installation package for users seeking in-depth understanding and advanced operational guidance.

By following this manual, users will acquire the necessary knowledge to efficiently develop AI projects on Stellar microcontrollers, from the inception to the completion, utilizing the StellarStudio.AI component as an integral tool within the Stellar Studio ecosystem.

#### Figure 1. ST Edge AI Core CLI technology

Al editor			
<ul> <li>✓ Datform:/resource/CIFAR/CIFAR.ai</li> <li>✓ A I Component CIFAR</li> <li>♦ Mcu</li> <li>♦ Mode</li> <li>♦ Network network (enabled)</li> <li>♦ Actions</li> <li>♦ Validate (enabled)</li> </ul>	Artificial Intelligence component	STEdge Al Core CLI technology. Copyright (C) STMicroelectr	onics. All rights reserved.





# 1 Requirements

The StellarStudio.AI component requires the latest available Stellar Studio version (with the latest versions of the SDKs and TOOLS packages installed) for Windows operating system and it is compatible with the Stellar automotive Arm<sup>®</sup>-based device family.

In particular, the supported Stellar microcontrollers can be selected in the MCU target selection panel of the StellarStudio.AI component.

# 2 Overview

The StellarStudio.AI component extends the Stellar Studio by providing an automatic NN library generator optimized in computation and memory (RAM and flash memory) that converts pre-trained neural networks from the most used DL frameworks (such as Keras, TensorFlow<sup>™</sup> lite, and ONNX) into a library that is automatically integrated in the final Stellar Studio software development kit (Stellar SDK). The project is automatically set up, ready for compilation and execution on the Stellar microcontrollers.

The StellarStudio.AI component can be easily installed in few steps directly from the Stellar Studio tool:

- 1. From the menu, select [Help]>[Install new software...]
  - a. For external users, type https://contrib.srxstudio.org and press Next.

Figure 2. StellarStudio.Al external installation

🖅 Install	🖻 install – 🗆 X								
Available S	Software								
Check the	items that you wish to install.					() <u>_</u>			
Work with:	https://contrib.srxstudio.org				<ul> <li>✓ <u>A</u>dd</li> </ul>	<u>M</u> anage			
type filter to	ext					Select All			
Name		Version				^ Deselect All			
∨ ⊠ u s ∨ ⊠ u	tellar/Contrib Il Artificial Intelligence								
	R STEdgeAl component	1.2.0.202403251511				~			
3 items selec	ted								
Details									
						<u>ئ</u> ا			
Show onl	y the latest versions of available software		Hide items that are already installed						
Group ite	ms by category		What is <u>already installed</u> ?						
Show onl	y software applicable to target environment								
Contact a	Il update sites during install to find required software								
G				< Back Next >	Einish	Cancel			

b. For internal users, type the latest available installation link (for example, http:// Imecxd0438.Ime.st.com/ stellarstudio/journey) and press *Next*.

#### Figure 3. StellarStudio.Al internal installation

💯 install			_	ΟX
Available Software				
Check the items that you wish to install.				
				0
Work with: <sup>®</sup> AI - http://Imecxd0438.Ime.st.com/stellarstudio/heart/		~	<u>A</u> dd	<u>M</u> anage
type filter text				Select All
Name	Version		^	Deselect All
✓ ■ 100 Stellar/Contrib				
✓ Multi Anticia menigerice ✓ I W STEdgeAl component	1.2.0.202403251511		~	
1 item selected				
Details				
Show only the latest versions of available software	Hide items that are already inst	talleri		
Group items by category	What is already installed?	anco		
Show only software applicable to target environment				
Contact all update sites during install to find required software				
?		< <u>B</u> ack <u>N</u> ext >	Einish	Cancel

- 2. Press Finish to start the installation.
- 3. At the end of the installation a Stellar Studio restart is required.

#### Figure 4. StellarStudio.Al installation restart

Software Updates	×
Restart StellarStudio All In One to apply the software u	pdate?
	<u>R</u> estart Now <u>N</u> o

The StellarStudio.Al component can generate two kinds of project:

- Validation project that validates incrementally the results returned by the NN, stimulated by either random or user test data, for the Stellar device selected in the MCU target selection panel.
- Application template project allowing the building of AI-based applications.

It is based on a graphical user interface (GUI) that allows to define a neural network on which the SDK project is based and the type of processing (analyze, generate, or validate) to execute. The project, after generating the neural networks files, can be compiled and flashed on the board to validate the neural network. When the validation on target is successful, the neural network files can be used in any user application to run inferences. The StellarStudio.AI configuration is based on the following panels (MCU, mode, network, actions, and validate)

as shown in Figure 5. ST Edge AI Core - StellarStudio.AI GUI.

57/



Figure 5.	ST Edge	Al Core	- StellarStudio.Al	GUI
-----------	---------	---------	--------------------	-----

Al editor	
Bylatform/resource/SRSE1AL,Dex/Clou     Al Component Pri     Al Component Pri     Mode LEGAC/-MODE     Network network (enabled)     Actions     Validate (enabled)	Stellar family Target Mcu device STELLAR-STSET v
Al editor	
Implatform/resource/SRSE1_AL_DevClou     Al Component Pri     Amount Pri     Mode LEGACY-MODE     Network network (enabled)	Public client APIs Public client APIs Embedded inference APIs (regacy/ or is-a)   LEGACYMODE v
<ul> <li>Actions</li> <li>Validate (enabled)</li> </ul>	LEGACY-MODE STAI-MODE
< >	
Al editor	
Implatform/resource/SRSE1_AL_DevCleu     Al_Omponent Prj     Alson STELLAR-SRSE1     Alson STELLAR-SRSE1     Alson Stellar Stellar     Alson Stellar Stellar     Alson Stellar Stellar     Alson Stellar Stellar	Neural network
<ul> <li>Validate (enabled)</li> </ul>	Settings 2
	Name network
	rype Nerias Compression Nome Optimization Delianced O
	Advanced settings *
	Use activation buffer for input buffer (allocate-inputs)
	Use activation buffer for the output buffer (allocate-outputs)
	Split weights during code generation (split-weights) Allocate the activation space in the generated code (allocate-activations)
	Allocate the states buffers in the generated code (allocate-states)
	Force classifier validation output (classifier)
	Extra command line options Extra command line options Curtom Issuer Support
	Enable
	Custom Layer Json File Browse
E Al editor	
All Component Prj     All Component Prj     Acu STELLAR-SR5E1     Mode LEGACY-MODE     Network network (enabled)	Tools version A
<ul> <li>Validate (enabled)</li> </ul>	AG3001
	Available networks * Enabled networks
	Name: network, Type: keras, Compression: none, Optimization: balanced, Model path: source/model, Complexity: 95105 MACC
	Network enabled for validation on target: - network
	Analyze & Generate actions \$
	Analyze Analyze
	Generate Generate
< >	
Al editor	
Instruction of the second	Target validation
Validate (enabled)	Validate settings
	retwork to validate network
	Validation Type Custom Input Data V Com Post COMR
	Input Path source/validation/validate_set.npz Browse
	Output Path Browse
	Validate action *
	Validate Validate
	In order to run the Validate on Target:
	Enable the waldkillow view calcelog the Enabled flag     Solved from the "Audidans settings" tab:     21 the Network To Validate     Solved from the "Audidans settings" tab:     21 the Network Service     Consense the Validate Project Code using the Generate button from the "Actions" panel     4. Comparise the Validate Project Code using the Generate button from the "Actions" panel     4. Comparise the Validate Project Code using the Generate button from the "Actions" based     5. Select from the "Validate parameter" tab:     6. Select from the "Validate parameter" tab:     6. The Validate 'project Parameter' tab:
< >>	6.2 the com Port 7. Press the vialidate button For the selected network a report will be displayed in the console and will be stored in a dedicated file inside the project folder: Proj/dg.

The next figure shows a typical workflow for developing a user-defined application based on neural networks using the StellarStudio.AI component.







# 3 MCU parameters

The MCU panel allows to specify which target device of the Stellar family is used. Stellar SR5E1 line and SR6P line devices can be selected and are fully supported.

## Figure 7. MCU parameters

Al editor			
Platform:/resource/SR5E1_AI_DevClou     Al Component Prj     Mcu STELLAR-SR5E1     Mode LEGACY-MODE	Stellar family	Stellar	
Actions     Validate (enabled)	Target Mcu device	STELLAR-SRSE1	~



# 4 Mode parameters

The Mode panel allows to specify which embedded public C-API is generated at runtime: standard legacy vs st-ai. By default, legacy mode is used.

Depending on this value, the files generated are different.

For a full description of this CLI option (--c-api), you can refer to the ST Edge AI Core technology documentation.

## Figure 8. Mode parameters

Al editor		
Platform:/resource/SRSE1_Al_DevClou     Al Component Prj     Arcus STELLAR-SRSE1     Mode LEGACV-MODE     Network network (enabled)     Actions     Validate (enabled)	Public client APIs Embedded Inference APIs ('legacy' or 'st-ai')	LEGACY-MODE



# 5 Network parameters

The network panel allows specifying one or more networks on which the StellarStudio.Al component is based on each network defined in the network panel and it can be enabled or disabled. When a network is disabled, it is not processed. In this way the user can temporarily remove one or more networks from the StellarStudio.Al process. But at least one network in the network panel must be enabled anytime.

For each new entry in the network panel, it is possible to specify the following options:

- Enabled: the flag to enable or disable the neural network.
- **Name:** the name of the neural network. Two or more networks with the same name can be defined in the network list, but only one can be enabled. If more networks with the same name are enabled at the same time, an error is shown. Please note that two names that differ only for the lower case/upper case of one or more characters are considered identical.
  - This name is used to generate the C neural networks file names and functions. For a full description of this CLI option (-n/--name), you can refer to the ST Edge AI Core technology documentation.
- **Type:** the type of the deep learning (DL) framework. The following types are supported in the StellarStudio.Al component:
  - Keras
  - TensorFlow lite
  - ONNX

For a list of the layers supported, you can refer to the ST Edge AI Core technology documentation.

- **Compression:** the expected global factor of compression to reduce the size of the deployed c-model. Only the weights for the floating-point dense or fully connected layers are considered. Supported values are:
  - None: no compression, default value.
  - Lossless: applied algorithms ensuring the accuracy (structural compression).
  - Low: applied algorithms trying to reduce the size of the parameters with a minimum of accuracy loss.
  - Medium: more aggressive algorithms, the final accuracy loss can be more important.
  - High: extreme aggressive algorithms (not used).
     For a full description of this CLI option (-c/--compression), you can refer to the STMicroelectronics Edge AI Core technology documentation.
- Optimization: it is used to indicate the objective of the optimization passes, which are applied to deploy the c-model. Note that the accuracy/precision of the generated model is not impacted. By default, a tradeoff (balanced value) is considered. Supported values are:
  - Time: applied the optimization passes to reduce the inference time (or latency). In this case, the size
    of the used RAM (activation buffer) can be impacted.
  - Ram: applied the optimization passes to reduce the RAM used for the activations. In this case, the inference time can be impacted.
  - Balanced: trade-off between the 'Time' and the 'Ram' objectives.
     For a full description of this CLI option (-O/--optimization), you can refer to the ST Edge AI Core technology documentation.
- **File path:** the path in which the model files must be stored. If the model file path is not valid or the folder does not contain any valid model file, the StellarStudio.Al processing is stopped, and an error is returned. For a full description of this CLI option (-m/--model), you can refer to the ST Edge AI Core technology documentation.
- Allocate inputs: it's enabled by default in the ST Edge AI Core CLI command line and it indicates that the activation buffer is also used to handle the input buffers, else, they should be allocated separately in the user memory space. Depending on the size of the input data, the activation buffer may be bigger but overall less than the sum of the activation buffer plus the input buffer.
   For a full description of this CLI option, you can refer to the ST Edge AI Core technology documentation.
- Allocate outputs: it's enabled by default in the ST Edge AI Core CLI command line and it indicates that
  the activation buffer is also used to handle the output buffers, else, they should be allocated separately in
  the user memory space.

For a full description of this CLI option, you can refer to the ST Edge AI Core technology documentation.



- **Split weights:** if enabled, this flag indicates that one c-array is generated by the weights/bias data tensor instead of having a unique C-array (weights buffer) for the whole. For a full description of this CLI option (--split-weights), you can refer to the ST Edge AI Core technology documentation.
- Allocate activations: if enabled, this flag allocates the activation buffers (to store the intermediate results) in the generated code (used only when the st-ai mode is selected). Moreover, in case of default behavior, they should be allocated separately in the user memory space.
   For a full description of this CLI option (--allocate-activations), you can refer to the ST Edge AI Core technology documentation.
- **Classifier:** if enabled, this flag indicates that the provided model should be considered as a classifier vs regressor. This implies that the computation of the "CM" and "ACC" metrics are evaluated, and an autodetection mechanism is used to evaluate if the model is a classifier or not. For a full description of this CLI option (--classifier), you can refer to the ST Edge AI Core technology documentation.
- **Extra options:** This field must be used when running the ST Edge AI Core CLI commands. For a full list of all CLI options, you can refer to the ST Edge AI Core technology documentation.
- Custom Layer
  - Enabled: to enable a custom layer (json file format).
  - Custom layer Json file: the path of the configuration file (.json file) to support the custom layers. The new .c file is automatically generated during the generation phase and built during the compilation phase.

For a full description of this CLI option (--custom), you can refer to the ST Edge AI Core technology documentation.

The figure below shows the network panel parameters.

Al editor							
In platform:/resource/SR5E1_Al_DevClou     Al Component Pri     Au STELLAR-SR5E1     Mode LEGACY-MODE     Network network (enabled)     Artiges	Neural network Enabled 🗹 Settings						*
<ul> <li>Validate (enabled)</li> </ul>	Name network						
	Type Keras	✓ Compression	None		<ul> <li>Optimization</li> </ul>	Balanced	~
	File Path source/model			Browse			
	Advanced settings 2						
	Use activation buffer for input buffer (allocate-inputs)						
	Use activation buffer for the output buffer (allocate-outputs)						
	Split weights during code generation (split-weights)						
	Allocate the activation space in the generated code (allocate-activ	ations)					
	Allocate the states buffers in the generated code (allocate-states)						
	Force classifier validation output ( classifier)						
	Extra command line options	Extra command line options					
	Custom Layer Support						
	Enable 🗹	Enable 🗹					
	Custom Layer Json File Custom Layer Json File				Browse		
< >							

### Figure 9. Neural networks parameters

The table below summarizes the model file extensions for the different network types.

#### Table 1. Neural network model file extensions

DL framework	Туре	File extension
Keras	Keras	.h5 or .hdf5 and .json
TensorFlow lite	TFLite	.tflite
ONNX	Onnx	.onnx

# 6 Al commands

The StellarStudio.AI component supports the following commands (refer to the ST Edge AI Core CLI technology documentation):

- Version
  - Get the version of all AI tools
- Analyze
  - Import the model.
  - Map, render, and optimize internally the model.
  - Log and display a report.
- Generate
  - Import the model.
  - Map, render, and optimize internally the model.
  - Export the specialized C-files.
  - Log and display a report.
- Validate
  - Import the model.
  - Map, render, and optimize internally the model.
  - Execute the generated C-model (on a target).
  - Execute the original model using the original deep learning runtime framework for x86.
  - Evaluate the metrics.
  - Log and display a report.

Pushing the version command, the ST Edge AI Core tools versions are returned.

For the other commands, the same preliminary steps are applied. A report (.txt file) is systematically created and fully or partially displayed. Additional JSON files (dictionary based) are generated in the workspace directory to be parsed by the StellarStudio.Al component to retrieve the results. Note that they can also be used in a non-regression environment. The format of these files is out of the scope of this document.

<workspace-directory-path>\<network\_name>\_report.json, <network\_name>\_c\_graph.json

<output-directory-path>\<network\_name>\_<cmd>\_report.txt

Version, Analyze and Generate commands are inside the actions panel, see Figure 10. Version, analyze, and generate commands.

The Validate command is inside the validate panel, see Figure 11. Validate command.

The Analyze command is the primary command to import, parse, check, and render an uploaded pre-trained model. A detailed report provides the main system metrics to know if the generated code can be deployed on the target Stellar device. It also includes rendering information by layer or/and operator. After completion, the user can be fully confident of the imported model in terms of supported layer/operators.

The Generate command is used to generate specialized networks and data C-files. They are generated in the specified output directory. The files generated depend on the embedded public C-API selected in the mode parameter, see Figure 8. Mode parameters. The name of the files generated depends on the neural network name parameter chosen in the network panel.

The Validate command allows to import, render, and validate the generated C-files. For the validation on target, the target board must be flashed with a valid validation firmware.

When the validation on target is performed, the DL model is compared with the C model that runs on the targeted device. It requires a special StellarStudio.AI test application that embeds the generated neural networks libraries and the COM agent to communicate with the host system.

Be aware, that the main purpose of the underlying validation process is to test the generated C files with the associated network runtime library by comparison with the imported DL model.

Subsequently, only a representative and limited part of a whole validation or test dataset can be used. It has not been designed to validate the pre-trained model as during a training/test phase.

# Figure 10. Version, analyze, and generate commands

🖾 Al editor		
<ul> <li>✓</li></ul>	ACTOM	
<ul> <li>Network network (enabled)</li> <li>Actions</li> </ul>	Tools version	*
Validate (enabled)	Version Version	
	Available networks	*
	Enabled networks: - Name: network, Type: keras, Compression: none, Optimization: balanced, Model path: source/model, Complexity: 95105 MACC	
	Network enabled for validation on target: - network	
	Analyze & Generate actions	*
	Analyze Analyze	
	Generate Generate	
< >		



Al editor							
V D platform:/resource/CIFAR/CIFAR.ai							
Al Component CIFAR     Mcu     Mcu	Target validation						
<ul> <li>Mode</li> <li>Network network (enabled)</li> <li>Actions</li> </ul>	Enabled 🖸						
Validate (enabled)	Validate settings	*					
	Network To Validate network	~					
	Serial SRSEL_UARTI	~					
	Validate parameters	*					
	Velidation Type Custom Input Data V Com Port	~					
	Input Path source/validation_nn/int8_100_cifar10_1.npz Browse						
	Output Path Browse						
	Validate action	*					
	Volidate Validate						
	In order to run the Validate on Target:						
	1. Enable the validation view selecting the Enabled flag						
	2. Select from the "Validate settings" tab: 2.1 the Network To Validate						
	2.2 the Validate Serial 3. Generate the Validate Project code using the Generate button from the "Actions" namel						
	4. Complet the validate Project cost and the Central Cost of the rection parts						
	5. Upload and run it on the target 6. Select from the "Validate parameters" tab:						
	6.1 the Validate Type (Random or Custom Input/Output Data)						
	7. Press the Validate button						
For the selected network a report will be displayed in the console and will be stored in a dedicated file inside the project folder: \CIFAR(dg)							

57



6.1

## Version

The Version command returns the information of the ST Edge AI Core tools version and is displayed on the AI console, as the figure below.

For a full description of this CLI option (--tools-version), you can refer to the ST Edge AI Core technology documentation.

Al editor	
✓	Tools version 2 Version 2
	Available networks
	Enabled networks: - Name: network, Type: keras, Compression: none, Optimization: balanced, Model path: source/model_nn, Complexity: 160339094 MACC Network enabled for validation on target: - network Analyze & Generate actions & & & & & & & & & & & & & & & & & & &
	Generate
🗐 Console 🗙 🚺 Debug Shell 🖹 Problem	ns 🕡 Executables 🗎 🖗 🖻 🙂 😋 🕶 🗖
Al Console	
stedgeal - STEdgeAI Core v9.1.0-A1 - Python version : 3.9.13 - Numpy version : 1.23.0 - TF version : 2.12.0 - TF Keras version : 2.12.0 - ONEX version : 1.10.2	
- ONNX RT version : 1.15.0	

## Figure 12. Version process

## 6.2 Analyze

The Analyze command is used to check a DL model. For each of the enabled networks in the network panel it generates a report that is shown in the Stellar Studio console during the command execution and is also stored in a .txt file within the project folder cpi</c>/<ai\_component\_name</c>/Cfg/. The name of the report is <network</pre> name analyze report.txt.

For a full description of the CLI Analyze command, you can refer to the ST Edge AI Core technology documentation.

The report allows us to check the imported models in terms of supported layers/operators. To run the analyze command, select the actions panel and click on the analyze button (see the figure below).

Figure	13.	Analy	ze proces	S
--------	-----	-------	-----------	---

🏠 Project Explorer 🗙	🖻 🗣 🎖 📍 🗖	🗖 "Prj.ai 🗙		- 0
> 😂 HAR		Al editor		
S LES HPE ATT LAY				
> ES IDPS		✓  ☐ platform:/resource/SR5E1_AI_DevClou		
> 🐸 KerasiD001		Al Component Prj	Action	
> 😂 KerasiD002		Mode LEGACY-MODE		
> ES KerasiD003		Network network (enabled)	Technologi	*
> 😂 KerasiD005		Actions	IOOS VEINOI	~
> 🎒 KerasiD006		<ul> <li>Validate (enabled)</li> </ul>	Version Version	
> 🖆 KerasiD007				•
> > MobileNet			Available networks	^
> S OnnxiD002			Enabled networks:	
> 10 SR5E1_AI_Demo_7000P			<ul> <li>Name: network, Type: keras, Compression: none, Optimization: balanced, Model path: source/model, Compression: MACC.</li> </ul>	
SRSE1_AI_DevCloud			Network enabled for validation on target:	
> 🍪 Binaries			- network	
> B CommonBuild			Analyze & Generate school	*
> 🕼 Modules				
∽ 🈥 Prj			Analyze Analyze	
v ≽ cfg			Generate Generate	
> metwork_analyze_report.txt				
> 😂 source				
> c main.c				
🖹 c_objs.lst				
Naketile				
a stellar_ai.mk				
				_



## Figure 14. Example of the analyze report

STEdge	AI Core v1.0.0											
Parame	eters : analyzetarget stellar-e -m C:/Stell	arStudio-5.0/workspace/	CIFAR\source/model_nn\d	cifar10_qkeras_g	ray_6795_1.h5name netw	worktype kerascomp	pression noneworks	pace C:/StellarStudio-5.0/wo	nkspace/CIFAR\CIFAR\ws\	-output C:/StellarStud	io-5.0/workspace/CIFA	AR\CIFAR\cfg
Exec/n	report summary (analyze)											
model type c_name compre optimi target worksp output model_ model_	file         C. \StellarStudio-5.0Wworkspace\CLFAM.co           keras         heras           station         nome           tation         balanced           \station         balanced           \station         some           \station         some	urce\model_nn\cifar10_q FAR\ws FAR\cfg	keras_gray_6795_1.h5									
params input output macc weight activa ram (t	<pre>i</pre>	QLinear(1.000000000,0, 7,979,240(-96.8%) vs fl 40	int8), user oat model									
model	name - citario_qkeras_gray_0/95_1					1						
	Tayer (type,or ignaz)	Charles and and a	parato size	MACC	connecceu co	1 Care	Caace.	c_type				
	input_1 (input, inputtayer)	[0:1,0:52,0:52,0:1]										
1	act_0 (Conversion, QActivation)	[b:1,h:32,w:32,c:1]		2,848	input_1		-2,848(-100.0%)					
2	conv2d_A (Conv2D, QConv2D)	[b:1,h:32,w:32,c:64]	576/2,304	589,824	act_0	-1,472(-63.9%)		Conv2D_[0]				
3	batch_normalization (ScaleBias, BatchNormalization)	[b:1,h:32,w:32,c:64]	128/512	131,072	conv2d_A	-512(-100.0%)	-131,072(-100.0%)					
4	act_1 (Conversion, QActivation)	[b:1,h:32,w:32,c:64]		131,072	batch_normalization	<u> </u>	-131,072(-100.0%)					
5	conv2d_8 (Conv2D, QConv2D)	[b:1,h:32,w:32,c:64]	36,864/147,456	37,748,736	act_1	-142,592(-96.7%)		Conv2D_[1]				
6	<pre>batch_normalization_1 (ScaleBias, BatchNormalization)</pre>	[b:1,h:32,w:32,c:64]	128/512	131,072	conv2d_B	-512(-100.0%)	-131,072(-100.0%)					
7	act_2 (Conversion, QActivation)	[b:1,h:32,w:32,c:64]		131,072	batch_normalization_1	1	-131,072(-100.0%)					
8	<pre>max_pooling2d (Pool, MaxPooling2D)</pre>	[b:1,h:16,w:16,c:64]		65,536	act_2	1		Pool_[2]				
9	conv2d_C (Conv2D, QConv2D)	[b:1,h:16,w:16,c:128]	73,728/294,912	18,874,368	max_pooling2d	-285,184(-96.7%)		Conv2D_[3]				
10	batch_normalization_2 (ScaleBias, BatchNormalization)	[b:1,h:16,w:16,c:128]	256/1,024	65,536	conv2d_C	-1,024(-100.0%)	-65,536(-100.0%)					
11	act_3 (Conversion, QActivation)	[b:1,h:16,w:16,c:128]		65,536	batch_normalization_2	1	-65,536(-100.0%)					
12	conv2d_D (Conv2D, QConv2D)	[b:1,h:16,w:16,c:128]	147,456/589,824	37,748,736	act_3	-578,888(-96.8%)		Conv2D_[4]				
13	batch_normalization_3 (ScaleBias, BatchNormalization)	[b:1,h:16,w:16,c:128]	256/1,024	65,536	conv2d_D	-1,024(-100.0%)	-65,536(-100.0%)					
14	act_4 (Conversion, QActivation)	[b:1,h:16,w:16,c:128]		65,536	batch_normalization_3	1	-65,536(-100.0%)					
15	max pooling2d 1 (Pool, NaxPooling2D)	[b:1,h:8,w:8,c:128]	• •••••	32,768	act 4			Pool [5]				
16	conv2d E (Conv2D, QConv2D)	[b:1,h:8,w:8,c:256]	294,912/1,179,648	18,874,368	max pooling2d 1	-1,141,768(-96.8%)		Conv2D [6]				
17	batch normalization 4 (ScaleBias, BatchNormalization)	[b:1,h:8,w:8,c:256]	512/2,048	32,768	conv2d E	-2,848(-108.0%)	-32,768(-100.0%)					
18	act 5 (Conversion, QActivation)	[b:1,h:8,w:8,c:256]		32,768	batch normalization 4	··· ·····	-32,768(-100.0%)					
19	conv2d F (Conv2D, QConv2D)	[b:1,h:8,w:8,c:256]	589,824/2,359,296	37,748,736	act 5	-2,284,544(-96.8%)		Conv2D [7]				
20	batch normalization 5 (ScaleBias, BatchNormalization)	[b:1,h:8,w:8,c:256]	512/2,048	32,768	conv2d F	-2,848(-108.8%)	-32,768(-100.0%)					
		(hat had und hatte)			habels annualdantidan f		33 700 ADD OK)					

# 57

# 6.3 Generate

The Generate command is used to generate the C-library files for all the enabled networks within the network panel. Then users can design and develop specific applications based on the APIs of these C-libraries. For each of the enabled networks, the generate command creates the specific neural network files starting with the name of the neural network chosen within the project folder <prj><ai\_component\_name>/cfg/ and generates a report that is shown in the AI console during the command execution and it is also stored in a .txt file within the same folder. The name of the report will be <network\_name>\_generate\_report.txt.

For a full description of the CLI Generate command, you can refer to the ST Edge AI Core technology documentation.

If an error occurs during the generation process of one of the enabled networks, the generate command continues to process the other enabled networks.

To run the generate command, select the actions panel and click on the generate button (see the figure below). A list of all available networks enabled and ready for the generation is also shown.



### Figure 15. Generate process

The Figure 16. Example of the generate report is an example of the generated generate report.



#### Figure 16. Example of the generate report

STEdgeat Core v9 8	a_10281
Created date	: 2024-03-19 09:28:13
Parameters	: generatetarget stellar-ec-api legacy -m C:/StellarStudio-5.0/workspace/SR5E1_AI_DevCloud\source/mode\CONU_LSTN.h5name networktype keras compression noneworkspace C:/StellarStudio-5.0/workspace/SR5E1_AI_DevCloud\Prj\ws\output C:/StellarStudio-5.0/workspace/SR5E1_AI_DevCloud\Prj\cfg
Exec/report summar	y (generate)
model file	: C:\StellarE\Projects\AITests\TestD01\source\model lstm\CNN LSTM.h5
type	: keras
c_name	: lstm
compression	: none
options	: allocate-inputs, allocate-outputs
optimization	: balanced
target/series	: generic
workspace dir	: C:\StellarE\Projects\AITests\TestID01\TestID01\ws
output dir	: C:\StellarE\Projects\AITests\TestID01\TestID01\cfg
model_fmt	: float
model_name	: CNN_LSTM
model_hash	: b84fd65fa67e2b886f5631c67837406e
params #	: 8,903 ltems (35.01 K18)
loout 1/1	· · · · · · · · · · · · · · · · · · ·
Tubac 1/1	: Input_o (uumainiattivations/ "uerauit")
output 1/1	· ou scenes (dense) (dense) (askojsta) · 'dense 1 dense' (dense) (askojsta)
output 1/1	1 liters, 4 B, ai float, float, (1.1.1)
macc	95,105
weights (ro)	35,980 B (35.14 KiB) (1 segment) / +128(+0.4%) vs float model
activations (rw)	: 2,304 B (2.25 KiB) (1 segment) *
ram (total)	: 2,304 B (2.25 KiB) = 2,304 + 0 + 0
(*) 'input'/'outp	ut' buffers can be used from the activations buffer
Pass: 79% ########	9   84/105 [00-00/00.02.81+/+]
PASS: 97% ########	##7 103/100 [00:00<00:00, 27.55it/s]
Generated files (	7)
Cal Shalloof) Deede	and (ATT and TOAL) Tout TOAL ( Sold ) (Sold ) (Sold ) (Sold )
C:\StellarE\Proje	
C:\StellarE\Proje	the state is the state of the s
C:\StellarE\Proje	es (allest (Test Dellog (Test Del)) fellom data narans h
C:\StellarE\Proje	ts/Alfests/TestD01/TestD01/fe/1stm_dta_parame.c
C:\StellarE\Proje	cts\Allests\TestID01\TestID01\cfs\lstm data.h
C:\StellarE\Proje	cts/AITests/TestID01/TestID01/cfg/lstm_data.c
PASS: 97% #######	mm7  103/105 [00:00<00; 27.55it/s]
Creating txt repor	t file C:\StellarE\Projects\AITests\TestID01\TestID01\Cfg\lstm_generate_report.txt
DACC. O'NI PROPERTY	##71 303/306 F00-00-00 37 55it/-1
PA33: 97% #########	##/1 #3/#00 [W010940408 27 554/6]
PASS: 100% gauges	and 146/146 [2010/04:00 14 [30]]
Contraction and an and an and an and an and an and an	Taoi sao fasisi asiasi shaasi sh

## 6.4 Validate

The Validate command allows to import, render, and validate the C-libraries related to the enabled networks in the network panel. To execute the validate command, the validate procedure must be enabled by selecting the validate panel and setting the enabled flag (see Figure 17. Validate enable flag).

For a full description of the CLI Validate command, you can refer to the ST Edge AI Core technology documentation.

<ul> <li>platform:/resource/SR5E1_AI_DevClou</li> <li>Al Component Prj</li> </ul>	Target validation						
<ul> <li>Mcu STELLAR-SR5E1</li> <li>Mode LEGACY-MODE</li> </ul>							
<ul> <li>Mode LEAC-Y-MOPE</li> <li>Network network (enabled)</li> <li>Actions</li> <li>Validate (enabled)</li> </ul>	Enabled 🗹						
	Validate settings				*		
	Network To Validate network				~		
	Serial SR5E1_UART1				~		
	Validate parameters				*		
	Validation Type Custom Input Data	¥	Com Port	COM8	~		
	Input Path source/validation/Validate_set.npz		Browse.	-			
	Output Path		Browse.				
	Validate action				*		
	Validate Validate						
	In order to run the Validate on Target:						
	1 Enable the validation view calecting the Enabled flag						
	2. Select from the "Validate settings" tab:						
	2.1 the Network To Validate						
	2.2 the Validate Serial 3. Generate the Validate Project code using the Generate button from the "Actions" panel						
	4. Compile the Validate Project						
	5. Upload and run it on the target						
	<ol> <li>Select from the "Validate parameters" tab:</li> <li>A the Validate Type (Department of Cuttors legit Data as Cuttors)</li> </ol>	international and the	(				
	6.2 the Com Port	ustom input/Output b	/ata)				
	7. Press the Validate button						
	For the selected network a report will be displayed in the co	insole and will be stor	red in a dedica	ated file inside the project folder:			
	\Prj\cfg\						

#### Figure 17. Validate enable flag

A simple and quick validation mechanism is provided to compare the accuracy of a generated model and the uploaded DL model from a numerical standpoint. Both models are fed with the same input tensors (fixed random inputs or custom dataset). To be more accurate, additional metrics are reported to evaluate the generated C model.

#### Figure 18. Validate flow overview



Only the validation on target is provided to compare the DL model with the C model that runs on the targeted device. It requires a special AI test application that embeds the generated NN libraries and the COM agent to communicate with the host system.





The Validate procedure is based on the communication via serial port between the host (that sends the validation data to the target) and the target (that processes the received data).

For this reason, within the validate panel of the StellarStudio.AI component a validate serial port must be selected. It is possible to select as validate serial one of the UART available for the Stellar device selected in the MCU Target selection panel. If no validate serial port is selected an error is returned during the compilation phase.

The network to validate must also be selected from a list that contains the names of all enabled networks. Only one network at a time can be validated. It must be chosen before running the generating process. If no network is selected (for example, NONE) as the network to validate, an error is returned during the compilation of the validate project.

The Figure 20. Validate settings shows the section validate settings of the validate panel of the StellarStudio.Al component in which it is possible to select the network to validate and the validate serial.

#### Figure 20. Validate settings

Al editor						
∀ ≧ platform://resource/SRSE_LAL_DevClou	Target validation	network		*		
		Juger govern				
	Validate parameters Validation Type C Input Path source	ustom Input Data e/validation/Validate_set.npz	Com Port COM8 Browse	*		
	Uurput Path Browse					
	Validate Validate					
	In order to run the Va 1. Enable the validatio 2. Select from the 'Va 2.1 the Network To 2.2 the Validate Ser 3. Generate the Validat 4. Compile the Validate Typ 6.2 the Com Port 6.1 the Validate Typ 6.2 the Com Port 7. Press the Validate I For the selected netw 	lidate on Target: in view selecting the Enabled flag idiate settings' tab: volladte al the Project code using the Generate buttor a Project idiate parameters tab: e (Random or Custom Input Data or Custo utton ork: a report will be displayed in the consc	from the "Actions" panel m Input/Output Data) e and will be stored in a dedicated file inside the project folder:			

To validate the C-libraries, the following steps are required:

- The C-libraries of all enabled networks must be generated and included in a validate project.
  - Some configuration files, depending on the selected network to validate, are generated too. Because
    of that, it is needed to select the right network to validate (if more than one is present and enabled) in
    the validate settings before the generation command execution.
- The validate project must be compiled, downloaded on the target, and executed.
  - A makefile is also automatically generated to build the necessary files.
  - The validate procedure must be run from the StellarStudio.AI component pushing the validate button.

The first step is the same as the generate command with the validation enabled flag set in the validate panel and the StellarStudio.AI component generation is run (button generate in Figure 15. Generate process).

Note: For each new network to validate (if more than one has been loaded and enabled in the network panel), a new generation phase is needed, after having selected the right network to validate in the validate settings. If NONE is selected, a compilation error is generated when building the validate project.

When the generation of the neural network C-libraries is completed, the next step is to create and build (using one of the SDK C-compilers supported) a Stellar SDK validate project including the neural network C-libraries generated. Inside the main application, after the platform setup, it is enough to invoke the API AIValidateStart(). The code below shows a typical main source code of a Stellar SR5E1 line SDK validate project to start the validation of a neural network C-model.

```
#include <test_env.h>
#include <uart.h>
#include <io.h>
#include <io.h>
#include <irq.h>
#include <stdio.h>
#include "stellar_ai_cfg.h"
#if (STELLAR_AI_VALIDATE == TRUE)
#include "stellar_ai.h"
#endif /* #if (STELLAR_AI_VALIDATE == TRUE) */
/*
 * Example of system initialization function for SR5E1-EVB3000D board
 */
Void SystemInit(void)
{
   /* Enable interrupts.*/
   osal_sys_unlock();
   test env init((TestInit t)
```



```
(TEST INIT CLOCK
                 TEST INIT GPIO
                 TEST_INIT_BOARD
                 TEST INIT IRQ
                 TEST INIT OSAL));
  gpio_set_pin_mode(UART1_RX, UART1_RX_CFG);
  gpio set pin mode(UART1 TX, UART1 TX CFG);
  /* Initialize UART driver instance used for IO redirection.*/
  uart init(&DRV UART1);
  /* Configure UART driver instance used for IO redirection.*/
  (void)uart_set_prio(&DRV_UART1, IRQ_PRIORITY_5);
  (void)uart_set_rx_drv_mode(&DRV_UART1, UART_RX_DRV_MODE_INT_SYNC);
  (void)uart_set_tx_drv_mode(&DRV_UART1, UART_TX_DRV_MODE_INT_SYNC);
  (void)uart_set_baud(&DRV_UART1, UART_BAUDRATE_115200);
  (void)uart set presc(&DRV UART1, UART PRESCALER DIV1);
  (void)uart set parity(&DRV UART1, UART PARITY NONE);
  (void)uart_set_over(&DRV_UART1, UART_OVERSAMPLING_16);
  (void)uart_set_sbit(&DRV_UART1, UART_STOPBIT_1);
  /* Initialize Runtime IO module.*/
  io init(&DRV UART1);
  /* Start Runtime IO module.*/
 io_start();
  /* Enabling the Data Cache when validation is disabled.*/
 SCB EnableDCache();
#if (STELLAR AI VALIDATE == TRUE)
* Application entry point for validation process
*/
\ main(void) {
  /* System initialization.*/
 SystemInit();
  /* Run the validate procedure.*/
 aiValidateStart();
  /* never here...*/
 while (true) {
  }
#else
* Application entry point for inference run
*/
int main(void) {
  /* System initialization.*/
 SystemInit();
 printf("#### AI application to run inference.\r\n");
  /* Run user application based on the AI neural network. */
 ai application();
  /* never here...*/
 while (true) {
  }
#endif /* #if (STELLAR AI VALIDATE == TRUE) */
```

57

After compiling, the validate project must be flashed on a target Stellar board using one of the available tools (via USB Stellar link or JTAG interface). The procedure of flashing depends on the target board used. For example, using a discovery SR5E1-EVB3000D revision B board it can be done simply by using the OpenOCD software under Stellar studio environment and the StellarLINK hardware interface of the board.



#### Figure 21. SR5E1-EVB3000D revision B board

After power-on the SR5E1-EVB3000D revision B board with a USB cable (attached to the PC), the command to flash and download the validation binary code with the OpenOCD software is:

*C:\StellarStudio-5.0\openocd\bin\openocd.exe -d0 -s C:\StellarStudio-5.0\openocd\scripts -f board\sr5e1\_evb.cfg -c "program "C:/StellarStudio-5.0/workspace/HAR/build/sr5e1/evbe3000d/core1/Release/HAR.elf" reset exit" and can be simply executed under a command prompt:* 





The messages \*\* Programming started \*\* and \*\* Programming finished \*\* show the result of the command.

For more details about how to install StellarLINK drivers and how to use OpenOCD software you can refer to the Stellar Studio documentation.

After the flash programming is completed successfully, the code is automatically executed on the target board, and it waits in a loop for the incoming data to validate the model under evaluation.

The next step is to select the validate parameters in the section validate parameters of the validate panel (see Figure 23. Validate parameters ):



Al editor		
♥	Target validation	*
	Network To Validate network	~
	Serial SR5E1_UART1	~
	Validate parameters	*
	Validation Type Custom Input Data  Com Port COM8	~
	Input Path source/validation/Validate_set.npz Browse	
	Output Path Browse	
	Validate action	*
	Validate Validate	
	In order to run the Validate on Target:	
	1. Enable the validation view selecting the Enabled flag 2. Select from the "Validate settings" tab: 2. The Network To Validate 2. The Validate settings" tab: 2. The Validate Serial 3. Generate the Validate Project 5. Upload and run it on the target 6. Select from the "Validate project and the target 6. Select from the "Validate project 7. The Selected retwork a report will be displayed in the console and will be stored in a dedicated file inside the project folder:	

The figure below shows the flowchart of the validate procedure.





The validate parameters to set are:

Com port: it is the HOST COM on which the target is connected on. It can be selected manually between
one of the COM ports available in the list or if "Auto" (default choice) is selected an automatic detection will
start looking for the available COM ports in the system.



- **Validate type**: it is the type of custom test data used by the validate procedure. The user can select among:
  - Random: an internal self-generated random dataset is used (default value).
  - Custom input data: a custom dataset is used. In this case the user has to provide a single file containing the dataset. The supported file extensions are:
    - .npz: in this case the file can contain both inputs and expected outputs or the only inputs. If the only inputs are provided, the expected outputs are automatically obtained by the model files using the l2r metric.
    - **.npy** or **.csv:** in this case the file contains only the inputs. The expected outputs are automatically obtained by the model files using the I2r metric.
  - Custom input/Output data: a custom dataset is used. In this case the user has to provide both the custom input data and expected custom output data. The supported file extensions are.npz, .npy or .cvs for both custom input and expected output data. Note that if an .npz file containing both input and expected output data, the custom output data file is ignored.
- Input path: it is the custom input data file (.npz, .npy or .csv).
- Output path: it is the custom output data file (.npz, .npy or .csv).

For a full description of these CLI options (-vi/valinput and -vo/valoutput), you can refer to the ST Edge AI Core technology documentation.

When all the validate parameters are correctly selected, the validate procedure can be started by clicking the validate button in the validate panel. See the figure below.

	urce/HAK/HAK.a						
Al Compone	ent HAR	Target validation					
Mode Network i	network (enabled)	Enabled	9				
+ Actions		Validate settings					
♦ Validate (e	(enabled)	Network To Validat	network				
		Colu					
		Senai	SKSET_DAARTT				
		Validate paramete					
		Validation Type	Custom Input Data		Y Com Port	COM6	
		Input Path so	rce/validation_nn/valinput.npz		Browse		
		Output Path			Browse		
		Validate action					
				Validate Validate			
		In order to run the	alidate on Target:				
		1. Enable the valid	tion view selecting the Enabled flag				
		2. Select from the 2.1 the Network	/alidate settings" tab: o Validate				
		2.2 the Validate	1.1				
		3 Consuming the life	eriai data Basia di anda union tha Consenta	better from the Stational second			
		3. Generate the Va 4. Compile the Val	atal date Project code using the Generate late Project	button from the "Actions" panel			
		3. Generate the Va 4. Compile the Val 5. Upload and run 6. Select from the	nnai date Project code using the Generate late Project t on the target (vidate parameters" tabi	button from the "Actions" panel			
		3. Generate the Va 4. Compile the Val 5. Upload and run 6. Select from the 6.1 the Validate	rnai date Project code using the Generate ate Project : on the target /alidate parameters" tab: :pe (Random or Custom Input Data o	button from the "Actions" panel r Custom Input/Output Data)			
		<ol> <li>Generate the Val</li> <li>Compile the Val</li> <li>Upload and run</li> <li>Select from the 6.1 the Validate 6.2 the Com Por 7. Press the Valida</li> </ol>	rnal date Project code using the Generate date Project : on the target (addate parameters" tab: :pe (Random or Custom Input Data o : button	button from the "Actions" panel r Custom Input/Output Data)			
		3. Generate the Va 4. Compile the Val 5. Upload and run 6. Select from the 6.1 the Validate 6.2 the Com Por 7. Press the Valida For the selected n 	rnai date Project code using the Generate date Project : on the target 'addate parameters'' tab: 'addate parameters'' tab: pe (Random or Custom Input Data o : button work a report will be displayed in the	button from the "Actions" panel r Custom input/Output Data) console and will be stored in a d	ledicated file inside t	he project folder:	
Console X (i) De	Vebua Shell 🁏 Proble	3. Generate the Va 4. Compile the Val 5. Upload and run 6. Select from the 6.1 the Validate 6.2 the Com Por 7. Press the Valida For the selected n \HAR\(cfg)	anai date Project code using the Generate ate Project con the target addate parameters" tab: addate parameters" tab: addate parameters" tab: button work a report will be displayed in the Progress Information	button from the "Actions" panel r Custom input/Output Data) console and will be stored in a d	Sedicated file inside t	he project folder:	
Console × ① De Console	Nebug Shell 👔 Probl	3. Generate the V4 4. Compile the Val 5. Upload and run 6. Select from the 6.1 the Validate 6.2 the Com Por 7. Press the Validate For the selected n \HAR\cfg\ ems ② Executables	real bide Project code using the Generate ate Project con the target raidfate parameters' tab: per Random or Custom input Data o button work a report will be displayed in the get Progress Information Widea the generation	button from the "Actions" panel r Custom Input/Output Data) console and will be stored in a d	ledicated file inside t	he project folder:	ik a ⊮ ≓ ¤ • ∩ •
Console × De Console AI.IO: 1%	Pebug Shell 🔝 Probl	a. Generate the V4 b. Compile the V4 b. Upload and run b. Subload and run b. Select from the b. the Validate b. the Com Por composition compo	All and the second seco	button from the "Actions" panel r Custom Input/Output Data) c console and will be stored in a d	Jedicated file inside t	he project folder:	B, 5 9 ⊂ 0 • d •
I Console × j De Console AI.IO: 1%  AI.IO: 1%	Nebug Shell 🖹 Probl   8/913 [0   9/913 [0	B. Generate the Val     Compile the Val     Compile the Val     Compile the Val     S. Upload and run     G. Select from the Validate     G. The Validate     G. The Validate     C. The Validate     Or the selected n    \HAR\cfg\     Decutables      Decutables      Del<02:26, 6.1731      Del<02:248, 5.3531	and a project code using the Generate ale Project con the target stickte parameters "tab: pe (Random or Custom Input Data o work a report will be displayed in the work a report will be displayed in the stickton work a report will be displayed in the work a report will be displayed in the work a second will be displayed in the work as the second second second second work as the second second second second work as the second second second second second second work as the second second second second second second second work as the second second second second second second second work as the second	button from the "Actions" panel r Custom input/Odput Data) s console and will be stored in a d ted C-files for the 'network' network.	Jedicated file inside t	he project folder:	% <b>छ ७</b>   <b>२</b> 0 • ∩ •
Console × ] De Console AI.IO: 1%  AI.IO: 1%  AI.TO: 1% 1	Nebug Shell 🖹 Probl   8/913 [0   9/913 [0   10/913 [	3. Generate the V     4. Compile the Val     4. Compile the Val     5. Upload and run     6. Site 4. Set of the Validate     6. The Validate     6. The Validate     7. Press the Validat     For the selected n\HAR\(clg)      00:01<02:26, 6.17i(1 00:01<02:26, 5.35i( 00:01<02:48, 5.35i( 00:01<02:48	man program of the second seco	button from the "Actions" panel r Custom Input/Output Data) console and will be stored in a d ted C-files for the 'network' network'	ledicated file inside t	he project folder:	8월 19일 년 89년 - 13 -
) Console × ) De Console AI.IO: 1%   AI.IO: 1%   AI.IO: 1%   AI.IO: 1%	Nebug Shell 🖹 Probl   8/913 [6   9/913 [   18/913 ]	3. Generate the V     4. Complet the Val     4. Complet the Val     5. Select from the     6. Select from the     6. Select from the     6. The Validate     6.2 The Com Por     7. Press the Validate     for the selected n    HARR(cfg)     weilles26.6 6.1711     weilles26.6 6.1711     weilles21.63.7 4.433     weilles21.63.7 4.433	An arroyat arroyat da wing the Generate at Project code using the Generate ion the target solidate parameters 'tab': per Glavidour or Costom Impd Data or button work a report will be displayed in the project solidate parameters and the general project of the ge	button from the "Actions" panel r Custom Input/Output Data) console and will be stored in a d ted C files for the 'network' network' s 3 (002 < 63-48, 3-546-)2(A	fedicated file inside t	he project folder:	B, 3 영 년 9 · 년 •
Console × () De Console AI.IO: 1% AI.IO: 1% AI.IO: 1% I AI.IO: 1% I AI.IO: 1% I XI.IO: 1%	Proble Shell ▲ Proble   6/913 [0   10/913 [1   12/923     12/923	3. Generate the V 4. Compile the Val 5. Upload and run 6. Select from the 6.1 the Validate 6.2 the Com Por 7. Press the Validate For the selected n VARAC(fg) 00:01           00:01         0.2 the Compile 00:01         0.2 the Compile 00:01         0.2 the Compile 00:01           00:01         0.0 the 00:01         0.0 the 00:01         0.0 the 00:01         0.0 the 00:01           00:01         0.0 the 00:02         0.0 th	ana argon da	button from the "Actions" panel r Castom Input/Output Data) console and will be stored in a d ted C files for the "network" network. s s 2020-20348, 3546*;3[2](A 1002-20348, 346*;3[2](A	Jedicated file inside t	he project folder:	등 13 명( 전 9 · C •
Console × () De Console AI.IO: 1%   AI.IO: 1%   AI.IO: 1%   AI.IO: 1%   AI.IO: 1%   AI.IO: 1%   AI.IO: 1%	Proble   8/913 [€   9/913 [€   18/913 [   11/913     12/913     12/913	3. Generate the V     4. Compile the Val     4. Compile the Val     5. Upload and rum     6. Select from the     6.1 the Validate     6.2 the Com Por     7. Press the Validate     for the selected n    \HARX.cfg()     80:01       Beilot     Beilot       Beilot     Beil	An arroyat and arroyat of a visit of the Generate at Project code using the Generate and Project Code using the State parameters that and the State parameters that and the State and the State an	button from the "Actions" panel r Custom Input/Output Data) console and will be stored in a d ted C files for the 'network' network' s J0002-0040, J.946r/JE(A J0002-0054, J.866r/JE(A	edicated file inside t	he project folder:	张句 (에 에 이 · 더 ·
Console × ) De Console AT.IO: 1%   AT.IO: 1%	Nebug Shell 🖹 Probl   8/913 [6   9/913 [1   12/913 [   12/913 [   12/913 [   13/913 ]	3. Generate the V     4. Complete Val     4. Complete Val     5. Upload and run     6. Salect from the     6.1 the Validate     6	An arroyat da using the Generate at Project code using the Generate at Project. I on the target Solidate parameters' tab: per Glavidow or Costom Impd Data o button work a report will be displayed in the Progress Information Projects Information Projects Information Projects Information Projects Information Statuo, 25(1) [1491 Statuo, 25(1) [1491 Sta	button from the "Actions" panel r Custom input/Output Data) console and will be stored in a co ted C-files for the 'network' network. s 10002-030-40, 3-940-73C/A 10002-030-40, 3-940-73C/A 10002-030-30, 3-960-72(A	ledicated file inside t	he project folder:	% <b>छ ७</b> ट • त •
Console × De amade X.I.O: 1% X.I.O: 1% X.I.O: 1% X.I.O: 1% X.I.O: 1% X.I.O: 1% X.I.O: 1% X.I.O: 2% X.I.O: 2%	Nebug Shell 🖹 Probl   8/913 [6   9/913 [1   10/913   12/913   12/913   12/913   14/913 [   14/913 [	3. Generate the V     4. Compile the Val     4. Compile the Val     5. Splect from the     6. The Validate     6. The Va	An array of the second	button from the "Actions" panel r Custom input/Output Data) a console and will be stored in a d ted C files for the "network" network. s 10002-0304, 3.94er/JICA 10002-0305, 3.04er/JICA 10003-0305, 3.74er/JICA	ledicated file inside t	he project folder:	86일 영( 4 0 · C ·
Console X December Int.10. 1% I.T.10. 1% I.T.10. 1% I.T.10. 1% I.T.10. 1% I.T.10. 1% I.T.10. 1% I.T.10. 2% I.T.10. 2%	tebug Shell € Proble   8/93.2 (t   9/93.2 (t   18/953.3 (t   12/953.3 (t   12/953.3 (t)   13/953.3 (t)   14/953.3 (t)   15/953.3 (t)	S. Generate Hu V     Completite Val     Completite	An arroyat code using the Generate ale Project code using the Generate ale Project code using the Generate and the State parameters tab: Using the Generate state state work a report will be displayed in the Using the Generation of Content (State State	button from the "Actions" panel r Custom Input/Odiput Data) console and will be stored in a d ted C files for the 'network' network' s 10002-00548, 3040-10204 10002-00548, 3040-10204 10002-00558, 3040-10204 10002-00558, 3740-10204 10002-0058, 3740-10204	fedicated file inside t	he project folder:	%, 월 91 년 0 · 년 •
I Console X D De Console AT. TO: 1% AT. TO: 2% AT. TO: 2% AT. TO: 2%	Abbug Shell 20 Proble   8/933 [0   18/933 [0   18/933 [1   12/933 [1   12/933 [1   13/935 [1   14/933 [1   15/953 [1   16/933 [1	3. Generate the V     4. Compile the Val     4. Compile the Val     5. Upload and run     6. The Validate     6. The Validate     6. The Validate     6. The Validate     6. The SoleCold     for the soleCold     100-102-126, 6.17/1     100-101-02-126, 6.17/1     100-101-02-126, 6.17/1     100-101-02-126, 6.17/1     100-102-126, 7.17/1     100-102-126, 7.17/1	AP opport         color           are Project         color           in the target         solid           solid         parameters' tab:           performers' tab:         performers' tab:           bittom         month are point           vold         Progress information           sol         Progress information           sol         Progress information           SUL0         23(1)           SUL0         23(2)           SUL0         23(2)           SUL0         23(2)           SUL0         23(2)           SUL0         23(2)	button from the "Actions" panel r Custom input/Output Data) console and will be stored in a d ted C-files for the 'network' network' f 10002+0040, 3:940+3)2(A 10002+0040, 3:940+3)2(A 10003+0050, 3:740+3)2(A 10003+0050, 3:740+3)2(A 10003+0050, 3:740+3)2(A 10003+0050, 3:740+3)2(A		he project folder:	등 3 명 년 0 · 13 ·
I Console × 〕 De Console AT.IO: 18  AT.IO: 18  AT.IO: 18  AT.IO: 18  AT.IO: 28  AT.IO: 28  AT.IO: 28  AT.IO: 28  AT.IO: 28	Hebug Shell 🛃 Proble   6/933 [   9/933 [   12/933 ]   12/933 [   12/933 ]   12/933 ]   13/933 ]   14/953 ]   15/953 ]   15/953 ]	1. Generate the Va 4. Complet the Value 4. Complet the Value 4. Complet the Value 5. Decisit from Value 6. The Value 6. The Value 6. The Value 7. Press t	Bit Project         Column Column           as Project         Column Column         Column           and the trajet         Solidate parameters' table         Column           assessment         Solidate basessment         Solidate parameters' table           assessment         Solidate basessment         Solidate parameters' table           fill         Solidate basessment         Solidate basessment           fill         Solidate basessment         Solidate basessment           fill         Solidate basessment         Solidate basessment           fill         Solidate basessment         Solidate bases           fill         Solidate basessment         Solidate bases           fill         Solidate bases         Solidate bases           fill         Solidate b	button from the "Actions" panel c Custom input/Output Data) c console and will be stored in a d ted C files for the 'network' network. (0002-034), 1346/1024 (0002-0353, 1366/1024 (0003-0353, 1366/1024 (0003-0353, 1366/1024 (0003-0353, 1376/1024 (0003-0353, 1376/1024 (0003-0353, 1376/1024 (0003-0353, 1376/1024 (0003-0353, 1376/1024 (0003-0353, 1376/1024 (0003-0353, 1376/1024	lectoared file inside u	he project folder:	© छ छ। ≪ 0 • C •
Console X () De Console X AT.10: 18] AT.10: 18] AT.10: 18] AT.10: 18] AT.10: 18] AT.10: 18] AT.10: 28] AT.10: 28] AT.10: 28] AT.10: 28]	Notice         Stress         Notice           1         8/1913         10           1         8/1913         11           1         14/1913         11           1         12/1913         11           1         3/1913         1	Longrate the Vale     Longrate     Longr	An arroyat code using the Generate at Project code using the Generate at Project code using the Generate at Project code using the Generate at Program of Code of the Generate at Code of the G	button from the "Actions" panel r Castom Input/Odiput Data) console and will be stored in a d ted C-files for the 'network' network' a 10002-03-04, 3-984-7020A 10002-03-05, 3-984-7020A 10003-0355, 3.784-7020A 10003-0355, 3.784-7020A 10003-0355, 3.784-7020A 10003-0355, 3.784-7020A	fedcated file inside t	he project folder:	ik bi ⊯ ⊄ 0 • d •

#### Figure 25. Validate process

The validate procedure will generate for the selected network validated a report that is shown in the Stellar Studio console during the command execution and is also stored in a .txt file within the project folder <prj>/ <ai\_component\_name>/cfg/. The name of the validation report generated will be <network\_name>\_validate\_report.txt.

Please, before starting the validate procedure, verify that the COM related to the UART selected as validate serial is not busy on another task, otherwise the communication between the host and the target fails and the validation procedure returns an error. The communication is done @ 115200 bps.



When a validate procedure is completed, a new one with different parameters can be started from the validate panel. But if the user wants to add new networks to the validate project or wants to enable/disable some networks already defined in the network panel, it is recommended to restart the validate procedure doing a manual code generation file cleaning and then a new code generation.

The validate procedure can also be stopped during its execution. Note, if the validate procedure is stopped, it could be necessary to disconnect and then reconnect the target to the host before starting a new validate procedure.

When a validate procedure has been completed, a new one with different parameters can be started from the validate panel.

But if the user wants to add new networks or wants to enable/disable some networks already defined, it is recommended to restart the validate procedure by doing a manual code generation file cleaning and then a new code generation. The validate procedure can also be stopped during its execution. If the validate procedure is stopped, it could be necessary to disconnect and then reconnect the target to the host before starting a new procedure. If in the validate project, the RuntimeIO driver has been also added, the information about the networks added is printed on the serial port. To see them, just open a terminal emulator on that serial port.

Note, if the user configures the same serial in both RuntimeIO driver and in the StellarStudio.AI component, it will be mandatory to disconnect the terminal emulator before starting the validate procedure to avoid a communication failure.

The Figure 26. Neural network runtime information shows the typical information of the networks included in the Stellar Studio SDK validation project printed on the serial console when the RuntimelO driver is included in the validate project.

# Tera Term - [disconnected] VT File Edit Setup Control Windo Help Validation 7.1 AT led with GCC 10. AR-F denice conf onfiguration... vID:0x2511 (SR5E1x) RevID:0x0000 - FPU used 300 MHz 300 MHz \$I/\$D=(True,True) Validation 7.1 GCC 10 uration (2511 (SR5E1x) RevID:0x0000 (True,True) k + DWT (delay(1)=0.999 ms) latform (API 1.1.0 - RUNTIME 9.0.0) overing the network(s)... "network" etwork "network".. -0f2f740ba9d67d1fa2fba7b2357b41 Mar 15 09:43:05 2024 15 2024 09:43:45 .0 05 MACC net the (1,20,1,4)80/float32 (User Domain)/320 (1,1.1)1/float32 (User Domain)/4 READY to receive a CMD from the HOST... ! Note: At this point, default ASCII-base terminal should be closed and a serial COM interface should be used (i.e. Python al\_runner module). Protocol version = 3.1

#### Figure 26. Neural network runtime information



# 7 Supported compilers

The StellarStudio.AI component is fully working with the following Arm compilers:

- GNU Arm embedded toolchain 10.3-2021.10
- HighTec clang version 8.1.0
- IAR ANSI C/C++ compiler V9.30.1.335/W64
- ARMCLANG Arm compiler for embedded 6.21

The Stellar Studio ecosystem with both StellarStudio.AI component and SDKs installed already contain some AI demo applications for some specific boards, to be used as reference.

# **Revision history**

## Table 2. Document revision history

Date	Version	Changes
04-Sep-2023	1	Initial release.
		Document status changed from ST Restricted to public. Updated Section Introduction, Section 1: Requirements and Section 2: Overview.
		Added Section 3: MCU parameters and Section 4: Mode parameters
		Updated Section 5: Network parameters and Section 6: AI commands
		Added Section 6.1: Version.
14-Jun-2024	2	Updated Section 6.2: Analyze, Section 6.3: Generate and Section 6.4: Validate
		Removed "Embedded inference client API" and all subsection.
		Updated Section 7: Supported compilers.
		Removed "Supported deep learning toolboxes and layers" and all subsection.
		Removed "How to run a c-model locally".
		Removed "Key metrics" and all subsection.
18-Nov-2024	3	Updated Section 1: Requirements, Section 2: Overview, Section 3: MCU parameters, Section 5: Network parameters, Section 6.4: Validate and Section 7: Supported compilers.



# Contents

1	Requ	lirements	2
2	Over	view	3
3	MCU	parameters	7
4	Mode	e parameters	8
5	Netw	ork parameters	9
6	Al co	mmands	.11
	6.1	Version	. 13
	6.2	Analyze	. 13
	6.3	Generate	. 15
	6.4	Validate	. 16
7	Supp	orted compilers	.24
Revi	sion l	nistory	.25
List	of tab	les	.27
List	of fig	ures	.28

57

# List of tables

Table 1.	Neural network model file extensions	10
Table 2.	Document revision history	25



# List of figures

Figure 1.	ST Edge AI Core CLI technology	. 1
Figure 2.	StellarStudio.AI external installation.	. 3
Figure 3.	StellarStudio.Al internal installation	. 3
Figure 4.	StellarStudio.Al installation restart.	. 4
Figure 5.	ST Edge AI Core - StellarStudio.AI GUI	. 5
Figure 6.	ST Edge AI Core - StellarStudio.AI workflow.	. 6
Figure 7.	MCU parameters	. 7
Figure 8.	Mode parameters	. 8
Figure 9.	Neural networks parameters.	10
Figure 10.	Version, analyze, and generate commands	12
Figure 11.	Validate command	12
Figure 12.	Version process	13
Figure 13.	Analyze process	13
Figure 14.	Example of the analyze report	14
Figure 15.	Generate process	15
Figure 16.	Example of the generate report.	16
Figure 17.	Validate enable flag	16
Figure 18.	Validate flow overview	17
Figure 19.	Validate on target	17
Figure 20.	Validate settings	18
Figure 21.	SR5E1-EVB3000D revision B board	20
Figure 22.	How to flash the code on a target board using OpenOCD.	20
Figure 23.	Validate parameters	21
Figure 24.	Validate flowchart	21
Figure 25.	Validate process	22
Figure 26.	Neural network runtime information	23

#### IMPORTANT NOTICE - READ CAREFULLY

STMicroelectronics NV and its subsidiaries ("ST") reserve the right to make changes, corrections, enhancements, modifications, and improvements to ST products and/or to this document at any time without notice. Purchasers should obtain the latest relevant information on ST products before placing orders. ST products are sold pursuant to ST's terms and conditions of sale in place at the time of order acknowledgment.

Purchasers are solely responsible for the choice, selection, and use of ST products and ST assumes no liability for application assistance or the design of purchasers' products.

No license, express or implied, to any intellectual property right is granted by ST herein.

Resale of ST products with provisions different from the information set forth herein shall void any warranty granted by ST for such product.

ST and the ST logo are trademarks of ST. For additional information about ST trademarks, refer to www.st.com/trademarks. All other product or service names are the property of their respective owners.

Information in this document supersedes and replaces information previously supplied in any prior versions of this document.

© 2024 STMicroelectronics – All rights reserved