پروژه ۲: پیاده سازی مدل یادگیری ماشین "پیشبینی مصرف انرژی" با نرم افزار x-cube-ai

گام اول: نحوه ی عملکرد مدل و چگونگی پیش بینی داده ها

۱. خواندن مدل و پردازش اولیه آن

Return first 5 rows.

	Start time UTC	End time UTC	Start time UTC+03:00	End time UTC+03:00	Electricity consumption in Finland
0	2015-12-31 21:00:00	2015-12-31 22:00:00	2016-01-01 00:00:00	2016-01-01 01:00:00	10800.0
1	2015-12-31 22:00:00	2015-12-31 23:00:00	2016-01-01 01:00:00	2016-01-01 02:00:00	10431.0
2	2015-12-31 23:00:00	2016-01-01 00:00:00	2016-01-01 02:00:00	2016-01-01 03:00:00	10005.0
3	2016-01-01 00:00:00	2016-01-01 01:00:00	2016-01-01 03:00:00	2016-01-01 04:00:00	9722.0
4	2016-01-01 01:00:00	2016-01-01 02:00:00	2016-01-01 04:00:00	2016-01-01 05:00:00	9599.0

تصویر ۱- نمایش ۵ سطر اول مدل

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 52966 entries, 0 to 52965
Data columns (total 5 columns):
 #   Column                             Non-Null Count  Dtype
---  ------                             --------------  -----
 0   Start time UTC                     52966 non-null  object
 1   End time UTC                       52966 non-null  object
 2   Start time UTC+03:00               52966 non-null  object
 3   End time UTC+03:00                 52966 non-null  object
 4   Electricity consumption in Finland 52966 non-null  float64
dtypes: float64(1), object(4)
memory usage: 2.0+ MB
```

تصویر ۳- اطلاعات مدل

	Electricity consumption in Finland
count	52966.000000
mean	9488.750519
std	1576.241673
min	5341.000000
25%	8322.000000
50%	9277.000000
75%	10602.000000
max	15105.000000

تصویر ۲- بررسی آماری داده های ستون مصرف

در ادامه سه ستون اول حذف شده و ستون زمان به ستون های ماه، سال، تاریخ، زمان، هفته و روز تقسیم می شود.

DateTime	Consumption	Month	Year	Date	Time	Week	Day
2016-01-01 01:00:00	10800.0	1	2016	2016-01-01	01:00:00	53	Friday
2016-01-01 02:00:00	10431.0	1	2016	2016-01-01	02:00:00	53	Friday
2016-01-01 03:00:00	10005.0	1	2016	2016-01-01	03:00:00	53	Friday
2016-01-01 04:00:00	9722.0	1	2016	2016-01-01	04:00:00	53	Friday
2016-01-01 05:00:00	9599.0	1	2016	2016-01-01	05:00:00	53	Friday

چون می خواهیم بر اساس هفته بررسی کنیم، هفته های ناقص را حذف می کنیم. (4-1-2016 تا 26-12-2021)

سپس برای تحلیل اولیه این نمودار ها را ترسیم می کنیم :

- نمودار خطی از مصرف برق در فنلاند بین سال‌های ۲۰۱۶ تا ۲۰۲۱
- نمودار سری زمانی مصرف انرژی را با استفاده از مقادیر ستون Consumption از دیتاست
- ۶ نمودار مجزا از مصرف انرژی برای هر یک از سال‌های ۲۰۱۶ تا ۲۰۲۱ را در یک صفحه
- توزیع انرژی مصرفی را در مجموعه داده با استفاده از هیستوگرام و نمودار چگالی
- یک نمودار جعبه‌ای از مصرف انرژی بر اساس ماه‌های سال برای نمایش توزیع مصرف انرژی در هر ماه

در نهایت فرکانس نمونه برداری از ساعت به روز کاهش می دهیم

DateTime	Consumption	Month	Year	Week
2016-01-04	12300.625000	1	2016	1
2016-01-05	12945.375000	1	2016	1
2016-01-06	13192.750000	1	2016	1

۲. دسته بندی داده ها

یک سری زمانی (Series) از ستون "Consumption" در y می سازیم و داده های آن را با نرمالیزه کردن بین صفر و یک مقیاس می کنیم. سپس آنها را به سه دسته تقسیم می کنیم:

	توضیحات	تعداد	تعریف
train_data	داده های آموزش	0.8* total	y[0:training_size-val_size,:]
test_data	داده های تست	0.2* total (437, 1)	y[training_size:len(y),:1]
val_data	داده های اعتبار سنجی	0.8* training_size	y[len(y)-test_size-val_size:len(y)-test_size,:1]

۳. تعریف تابع سازنده ورودی مدل

این تابع یک کرنل کانولوشن یک‌بعدی به ابعاد ۱۰۱×۱ تعریف می‌کند که تمامی نمونه‌های داده های دیتاست اصلی را در بر می‌گیرد. ۱۰۰ ورودی اول به شکل یک لیست به dataX اضافه می‌شود و ورودی ۱۰۱ به عنوان مقدار هدف به لیست dataY اضافه می‌شود. در نهایت این تابع را روی train_data ، test_data و val_data پیاده می کنیم.

```
X_train shape:  (1297, 100)   y_train shape:  (1297,)
X_test shape:  (336, 100)     ytest shape:  (336,)
X_val shape:  (248, 100)      yval shape:  (248,)
```

داده ای ورودی برای آموزش مدل‌های یادگیری ماشین به فرمت سه بعدی تبدیل می شوند.

```
X_train shape:  (1297, 100, 1)
X_test shape:  (336, 100, 1)
X_val shape:  (248, 100, 1)
```

۴. تعریف مدل و پیاده سازی آن: این مدل میزان مصرف را بر اساس ۱۰۰ داده ورودی قبلی تخمین می زند.

ورودی مدل

ورودی مدل یک tuple با شکل (batch_size, 100, 1) است. به این معنی که:

- batch_size: تعداد نمونه‌های داده در هر دسته است.
- ۱۰۰ : طول هر دنباله ورودی است.
- ۱ : تعداد کانال‌ها، که در این مورد نشان می‌دهد داده‌ها یک بعدی هستند.

خروجی مدل

خروجی مدل یک tuple با شکل (batch_size, 1) است. به این معنی که برای هر نمونه ورودی، مدل یک مقدار اسکالر را به عنوان خروجی پیش‌بینی می‌کند.

کامپایل مدل

از الگوریتم adam برای بهینه‌سازی وزن‌های مدل استفاده کند.

برای ارزیابی عملکرد مدل، از میانگین مربعات خطا بین خروجی پیش‌بینی شده و مقدار واقعی استفاده کند.

توضیح لایه ها

Model: "sequential"

Layer (type)	Output Shape	Param #
conv1d (Conv1D)	(None, 98, 32)	128
max_pooling1d (MaxPooling1D)	(None, 49, 32)	0
flatten (Flatten)	(None, 1568)	0
dense (Dense)	(None, 64)	100,416
dense_1 (Dense)	(None, 1)	65

Total params: 100,609 (393.00 KB)
Trainable params: 100,609 (393.00 KB)
Non-trainable params: 0 (0.00 B)

تصویر ۴ – خلاصه مدل

I. لایه کانولوشن یک بعدی (Conv1D) : استخراج الگو های محلی

این لایه برای استخراج ویژگی‌ها از داده‌های یک بعدی استفاده می‌شود. در پردازش سیگنال، داده‌های زمانی یا مکانی اغلب به صورت دنباله‌های یک بعدی نمایش داده می‌شوند. این لایه با اعمال فیلترهای یادگیری شده بر روی ورودی، ویژگی‌های مهم را شناسایی می‌کند.

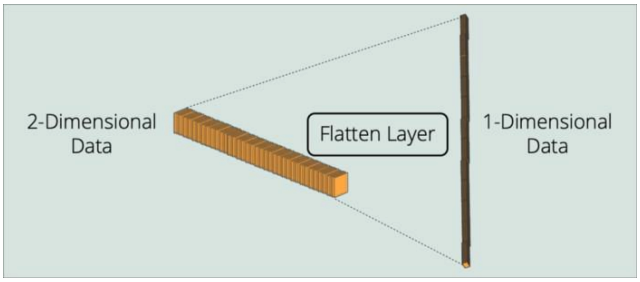
پارامتر ها	توضیحات
32	تعداد فیلترها (یا کانال‌ها) در این لایه است. هر فیلتر یک ویژگی خاص را در داده‌ها شناسایی می‌کند.
kernel_size=3	اندازه هسته فیلتر است که تعیین می‌کند چند داده متوالی در هر مرحله مورد بررسی قرار می‌گیرد.
activation='relu'	تابع فعال‌سازی ReLU است که برای معرفی غیرخطی بودن در مدل استفاده می‌شود.
input_shape=(100, 1)	شکل ورودی به این لایه است. به این معنی که ورودی یک دنباله با طول ۱۰۰ و یک کانال است.

II. لایه MaxPooling1D

لایه MaxPooling1D در شبکه‌های عصبی کانولوشنی (CNN) برای کاهش اندازه ویژگی‌های خروجی استفاده می‌شود. این لایه ابعاد خروجی را کاهش می‌دهد و مقادیر بیشتر را در هر پنجره مشخص نگه می‌دارد.

III. لایه Flatten

این لایه خروجی لایه کانولوشن را به یک بردار یک بعدی تبدیل می‌کند. این کار برای آماده‌سازی داده‌ها جهت ورود به لایه‌های متراکم بعدی ضروری است. (در این مدل $49 \times 32 = 1568$)



IV. لایه Dense

یک لایه کاملاً متصل است که داده‌های مسطح را پردازش کرده و ۶۴ ویژگی را به‌عنوان خروجی ارائه می‌دهد. این لایه برای یادگیری روابط پیچیده‌تر بین ویژگی‌ها استفاده می‌شود. پارامترها : (تعداد نورون ها = ۶۴ , تابع فعال سازی = ReLU)

V. لایه Dense_1

این لایه آخرین لایه مدل است و مقدار نهایی پیش‌بینی شده را تولید می‌کند. پارامترها : (تعداد نورون ها = ۱ , تابع فعال سازی = linear) این تابع فعال سازی برای برای پیش‌بینی مقادیر پیوسته مناسب است.

در نهایت مدل را با داده های آموزش و اعتبار سنجی، آموزش می دهیم.

۵. ارزیابی مدل

داده های تست را به مدل می دهیم و نتیجه آن را با معیار های ریشه میانگین مربعات خطا (RMSE)، میانگین قدر مطلق خطا (MAE)، میانگین درصد قدر مطلق خطا (MAPE) با مقدار واقعی آن مقایسه می کنیم. سپس برای هر سه دسته آموزش، تست و ارزیابی با مدل پیش بینی انجام داده و با این سه معیار درستی مدل را ارزیابی می کنیم. در نهایت نمودار آنها را رسم می کنیم.

مفهوم Tiny-ML

پردازش لبه به معنی انجام برخی از پردازش ها روی دستگاه کاربر (به جای سرور) است. در نتیجه پهنای باند افزایش می یابد و ترافیک شبکه، مسافت طی شده و تاخیر کاهش می یابد. Tiny-ML یا یادگیری ماشین کوچک، زیر مجموعه ای از پردازش لبه است که به معنای اجرای مدل‌های یادگیری ماشین روی دستگاه‌های کم‌مصرف و کوچک مانند میکروکنترلرها است.

ویژگی‌های اصلی

۱. مصرف انرژی پایینی دارد و برای دستگاه‌هایی طراحی شده است که در آنها میزان انرژی مصرفی اهمیت زیادی دارد.

۲. مدل‌های یادگیری ماشین در TinyML به شدت فشرده و بهینه‌سازی شده‌اند تا بتوانند روی سخت‌افزارهای کوچک اجرا شوند.

۳. داده‌ها در همان دستگاه کاربر پردازش می‌شوند و نیازی به ارسال اطلاعات به سرور یا فضای ابری نیست. این ویژگی باعث افزایش امنیت و کاهش تأخیر می‌شود.

مزایا:

- صرفه‌جویی در انرژی و هزینه: به دلیل پردازش محلی، نیازی به ارسال داده‌ها به سرور نیست، که مصرف انرژی و هزینه انتقال داده‌ها را کاهش می‌دهد.
- کاهش تأخیر: پردازش محلی سرعت پاسخ‌دهی را افزایش می‌دهد.
- افزایش امنیت و حریم خصوصی: داده‌ها در همان دستگاه پردازش می‌شوند . بنابراین نیازی به انتقال اطلاعات حساس نیست.

دلایل تبدیل مدل Keras به TF-Lite

مدل Keras برای ساخت مدل های یادگیری عمیق طراحی شده اند . این مدل ها بزرگ هستند و تمام پارامتر ها و تنظیمات مورد نیاز برای آموزش مدل را دارند. علاوه بر این، نوع داده ۳۲ بیتی (float32) را پشتیبانی می کنند. بنابراین حافظه و انرژی زیادی را مصرف می کنند. از طرف دیگر مدل TF-Lite برای اجرا روی دستگاه های کوچک (مثل میکروکنترلر ها) طراحی شده لند. این مدل ها از الگوریتم های بهینه سازی خاصی استفاده می کنند تا با فشرده کردن مدل، حجم حافظه و انرژی کمتری مصرف کنند. همچنین این مدلها از نوع داده ۱۶ بیتی (float16) یا ۸ بیتی(int8) پشتیبانی می کنند. در نتیجه این اقدامات مدل F-Lite مصرف انرژی و فضای ذخیره‌سازی کمتر و سرعت اجرای بیشتری نسبت به مدل Keras دارد و برای پیاده سازی مدل بر میکروکنترلر مناسب تر است.

تغییرات مهم در ویژگی‌های مدل:

- کاهش دقت عددی(Quantization) :

عددهای ۳۲ بیتی (float32) به ۸ بیتی (int8) یا عددهای ۱۶ بیتی کاهش می‌یابند. این کاهش دقت ممکن است تأثیر کمی روی خروجی مدل داشته باشد، اما باعث کاهش قابل توجه حجم مدل و افزایش سرعت می‌شود.

- حذف عملیات غیرضروری:

برخی عملیات‌ها یا لایه‌های پیچیده (مانند لایه‌های سفارشی) که قابل ترجمه به TF-Lite نیستند، ساده‌سازی یا حذف می‌شوند.

- بهینه‌سازی لایه‌ها:

برخی لایه‌ها مثل Dense یا Conv2D ممکن است فشرده‌تر یا با الگوریتم‌های جدید جایگزین شوند تا برای اجرا روی سخت‌افزار محدود بهینه شوند.

- کاهش سایز مدل:

به کمک روش‌هایی مثل weight pruning (حذف وزن‌های غیرضروری) یا shared weights (استفاده مجدد از وزن‌ها) حجم مدل کاهش می‌یابد.

گام دوم: بارگذاری مدل روی STM32CubeMX

I. Analysis

این گزینه یک ارزیابی از مدل انجام می دهد تا اطمینان دهد که مدل به درستی برای میکروکنترلر هدف اجرا می شود. این گزارش اطلاعات زیر را نمایش می دهد:

- خلاصه ای از ویژگی های مدل (مثل حجم مدل، وزن های آن، ویژگی های ورودی و خروجی و ...)
- بررسی ساختار مدل:

  تحلیل لایه‌های مدل، تعداد پارامترهای آموزشی و وزن‌ها

  بررسی اینکه آیا دستگاه هدف لایه های استفاده شده در این مدل را پشتیبانی می کند؟
- حافظه ای که مدل اشغال می کند.
- آیا مدل به صورت صحیح بار گذاری شده است یا خیر؟

Please wait...

Analyzing Network

100%

```
Analyzing model
C:/Users/ASUS/STM32Cube/Repository/Packs/STMicroelectronics/X-CUBE-AI/9.1.0/Utilities/windows/stedgeai.exe analyze --target stm32f4 --name network_1 -m C:
/my_project/model.tflite --compression none --verbosity 1 --allocate-inputs --allocate-outputs --workspace C:
/Users/ASUS/AppData/Local/Temp/mxAI_workspace104847531033290045191295924733666522 --output C:/Users/ASUS/.stm32cubemx/network_1_output
ST Edge AI Core v1.0.0-19894
Creating c (debug) info json file C:\Users\ASUS\AppData\Local\Temp\mxAI_workspace104847531033290045191295924733666522\network_1_c_info.json

 Exec/report summary (analyze)
 ------------------------------------------------------------------------------------------------------------
 model file         :   C:\my_project\model.tflite
 type               :   tflite
 c_name             :   network_1
 compression        :   none
 options            :   allocate-inputs, allocate-outputs
 optimization       :   balanced
 target/series      :   stm32f4
 workspace dir      :   C:\Users\ASUS\AppData\Local\Temp\mxAI_workspace104847531033290045191295924733666522
 output dir         :   C:\Users\ASUS\.stm32cubemx\network_1_output
 model_fmt          :   float
 model_name         :   model
 model_hash         :   0x3360987601741f82739e632eb2d2499b4
 params #           :   200,961 items (785.00 KiB)
 ------------------------------------------------------------------------------------------------------------
 input 1/1          :   'serving_default_conv1d_input0', f32(1x100x1), 400 Bytes, activations
 output 1/1         :   'gemm_4', f32(1x1), 4 Bytes, activations
 macc               :   213,473
 weights (ro)       :   803,844 B (785.00 KiB) (1 segment)
 activations (rw)   :   12,956 B (12.65 KiB) (1 segment) *
 ram (total)        :   12,956 B (12.65 KiB) = 12,956 + 0 + 0
 ------------------------------------------------------------------------------------------------------------
 (*) 'input'/'output' buffers can be used from the activations buffer
Computing AI RT data/code size (target=stm32f4)..
 Model name - model
```

m_id	layer (original)	oshape	param/size	macc	connected to
0	serving_default_conv1d_input0 ()	[b:1,h:100,c:1]			
	reshape_0 (EXPAND_DIMS)	[b:1,h:1,w:100,c:1]			serving_default_conv1d_input0
1	conv2d_1 (CONV_2D)	[b:1,h:1,w:98,c:32]	128/512	9,440	reshape_0
	nl_1_nl (CONV_2D)	[b:1,h:1,w:98,c:32]		3,136	conv2d_1
2	reshape_2 (RESHAPE)	[b:1,c:3136]			nl_1_nl
3	sequential_dense_MatMul ()	[h:64,c:3136]	200,704/802,816		
	sequential_dense_BiasAdd_ReadVariableOp ()	[c:64]	64/256		
	gemm_3 (FULLY_CONNECTED)	[b:1,c:64]		200,768	reshape_2 sequential_dense_MatMul sequential_dense_BiasAdd_ReadVariableOp
	nl_3_nl (FULLY_CONNECTED)	[b:1,c:64]		64	gemm_3
4	sequential_dense_1_MatMul ()	[b:1,c:64]	64/256		
	sequential_dense_1_BiasAdd_ReadVariableOp ()	[c:1]	1/4		
	gemm_4 (FULLY_CONNECTED)	[b:1,c:1]		65	nl_3_nl sequential_dense_1_MatMul sequential_dense_1_BiasAdd_ReadVariableOp

```
model: macc=213,473 weights=803,844 activations=-- io=--
Number of operations per c-layer
------- ------ ------------------------ --------- --------------
c_id    m_id   name (type)                    #op           type
------- ------ ------------------------ --------- --------------
0       1      conv2d_1 (Conv2D)            9,440   smul_f32_f32
1       1      nl_1_nl (Nonlinearity)       3,136     op_f32_f32
2       3      gemm_3 (Dense)             200,768   smul_f32_f32
3       3      nl_3_nl (Nonlinearity)          64     op_f32_f32
4       4      gemm_4 (Dense)                  65   smul_f32_f32
------- ------ ------------------------ --------- --------------
total                                    213,473
Number of operation types
---------------- --------- -----------
operation type           #           %
---------------- --------- -----------
smul_f32_f32       210,273       98.5%
op_f32_f32           3,200        1.5%
Complexity report (model)
------ ------------------------ ------------------------ ------------------------ --------
m_id   name                     c_macc                   c_rom                    c_id
------ ------------------------ ------------------------ ------------------------ --------
1      conv2d_1                 |                  5.9%   |                  0.1%   [0, 1]
3      sequential_dense_MatMul  ||||||||||||||||  94.1%   ||||||||||||||||  99.9%   [2, 3]
4      sequential_dense_1_MatMul|                  0.0%   |                  0.0%   [4]
------ ------------------------ ------------------------ ------------------------ --------
macc=213,473 weights=803,844 act=12,956 ram_io=0
Requested memory size by section - "stm32f4" target
------------------------------ ------- --------- ------- --------
module                           text    rodata    data      bss
------------------------------ ------- --------- ------- --------
NetworkRuntime910_CM4_GCC.a     8,572         0       0        0
network_1.o                       546        40   1,832      152
network_1_data.o                   48        16      88        0
lib (toolchain)*                    0         0       0        0
------------------------------ ------- --------- ------- --------
RT total**                      9,166        56   1,920      152
------------------------------ ------- --------- ------- --------
weights                             0   803,848       0        0
activations                         0         0       0   12,956
io                                  0         0       0        0
```

```
 io                                0         0       0        0
 ----------------------------- ------- --------- ------- --------
 TOTAL                          9,166   803,904   1,920   13,108
 ----------------------------- ------- --------- ------- --------
 *  toolchain objects (libm/libgcc*)
 ** RT AI runtime objects (kernels+infrastructure)
  Summary - "stm32f4" target
  ---------------------------------------------------
               FLASH (ro)      %*   RAM (rw)        %
  ---------------------------------------------------
  RT total         11,142   1.4%      2,072   13.8%
  ---------------------------------------------------
  TOTAL           814,990             15,028
  ---------------------------------------------------
  *  rt/total
Creating txt report file C:\Users\ASUS\.stm32cubemx\network_1_output\network_1_analyze_report.txt
elapsed time (analyze): 135.509s
Model file:      model.tflite
Total Flash:     814986 B (795.88 KiB)
    Weights:     803844 B (785.00 KiB)
    Library:     11142 B (10.88 KiB)
Total Ram:       15028 B (14.68 KiB)
    Activations: 12956 B (12.65 KiB)
    Library:     2072 B (2.02 KiB)
    Input:       400 B (included in Activations)
    Output:      4 B (included in Activations)
Done
Analyze complete on AI model
```

OK

II. Validate on desktop

این گزینه یک ارزیابی از مدل انجام می دهد تا اطمینان دهد که مدل به درســـتی برای میکروکنترلر هدف اجرا می شـــود. این گزارش اطلاعات زیر را نمایش می دهد:

- ارزیابی درستی اطلاعات کلیدی مدل
- اعتبار سنجی مدل با داده های تصادفی و گزارش جزئیات آن مانند زمان اجرا و ...
- ارزیابی دقت مدل با معیار های زیر

میانگین مربع خطا (Root Mean Squared Error)

میانگین قدر مطلق خطا (Mean Absolute Error)

خطای نسبی با نرم L2 (L2 relative error)

ضریب کارایی نش-ساتکلیف (Nash-Sutcliffe efficiency - NSE)

شباهت کسینوسی (Cosine similarity)

Please wait...

Validation on desktop

100%

```
Starting AI validation on desktop with random data...

C:/Users/ASUS/STM32Cube/Repository/Packs/STMicroelectronics/X-CUBE-AI/9.1.0/Utilities/windows/stedgeai.exe validate --target stm32f4 --name network_1 -m C:
/my_project/model.tflite --compression none --verbosity 1 --allocate-inputs --allocate-outputs --workspace C:
/Users/ASUS/AppData/Local/Temp/mxAI_workspace105026532961830017918445033030242097 --output C:/Users/ASUS/.stm32cubemx/network_1_output
ST Edge AI Core v1.0.0-19894
Setting validation data...
 generating random data, size=10, seed=42, range=(0, 1)
 I[1]: (10, 100, 1, 1)/float32, min/max=[0.005, 1.000], mean/std=[0.490, 0.292], serving_default_conv1d_input0
 No output/reference samples are provided
Creating c (debug) info json file C:\Users\ASUS\AppData\Local\Temp\mxAI_workspace105026532961830017918445033030242097\network_1_c_info.json
Copying the AI runtime files to the user workspace: C:\Users\ASUS\AppData\Local\Temp\mxAI_workspace105026532961830017918445033030242097\inspector_network_1\workspace

 Exec/report summary (validate)
 ------------------------------------------------------------------------------------------------------------------------
 model file         :   C:\my_project\model.tflite
 type               :   tflite
 c_name             :   network_1
 compression        :   none
 options            :   allocate-inputs, allocate-outputs
 optimization       :   balanced
 target/series      :   stm32f4
 workspace dir      :   C:\Users\ASUS\AppData\Local\Temp\mxAI_workspace105026532961830017918445033030242097
 output dir         :   C:\Users\ASUS\.stm32cubemx\network_1_output
 model_fmt          :   float
 model_name         :   model
 model_hash         :   0x336098760174f82739e632eb2d2499b4
 params #           :   200,961 items (785.00 KiB)
 ------------------------------------------------------------------------------------------------------------------------
 input 1/1          :   'serving_default_conv1d_input0', f32(1x100x1), 400 Bytes, activations
 output 1/1         :   'gemm_4', f32(1x1), 4 Bytes, activations
 macc               :   213,473
 weights (ro)       :   803,844 B (785.00 KiB) (1 segment)
 activations (rw)   :   12,956 B (12.65 KiB) (1 segment) *
 ram (total)        :   12,956 B (12.65 KiB) = 12,956 + 0 + 0
 ------------------------------------------------------------------------------------------------------------------------
 (*) 'input'/'output' buffers can be used from the activations buffer
Running the TFlite model...
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
Running the STM AI c-model (AI RUNNER)...(name=network_1, mode=HOST)
 X86 shared lib (C:\Users\ASUS\AppData\Local\Temp\mxAI_workspace105026532961830017918445033030242097\inspector_network_1\workspace\lib\libai_network_1.
dll) ['network_1']
  Summary 'network_1' - ['network_1']
  ------------------------------------------------------------------------------------------
  inputs/ouputs      :   1/1
  input_1            :   f32[1,100,1,1], 400 Bytes, in activations buffer
  output_1           :   f32[1,1,1,1], 4 Bytes, in activations buffer
  n_nodes            :   5
  compile_datetime   :   Dec 19 2024 20:13:55
  activations        :   12956
  weights            :   803844
  macc               :   213473
  ------------------------------------------------------------------------------------------
  tools              :   Legacy ST.AI 1.0.0
  capabilities       :   IO_ONLY, PER_LAYER, PER_LAYER_WITH_DATA
  device             :   AMD64, Intel64 Family 6 Model 126 Stepping 5, GenuineIntel, Windows
  ------------------------------------------------------------------------------------------
 NOTE: duration and exec time per layer is just an indication. They are dependent of the HOST-machine work-load.
  ST.AI Profiling results v1.2 - "network_1"
  ------------------------------------------------------------
  nb sample(s)   :   10
  duration       :   0.389 ms by sample (0.324/0.490/0.051)
  macc           :   213473
  ------------------------------------------------------------
   Inference time per node
   ------------------------------------------------------------------------
   c_id    m_id   type               dur (ms)       %   cumul     name
   ------------------------------------------------------------------------
   0       1      Conv2D (0x103)         0.039   10.0%   10.0%     ai_node_0
   1       1      NL (0x107)             0.006    1.4%   11.4%     ai_node_1
   2       3      Dense (0x104)          0.342   88.0%   99.5%     ai_node_2
   3       3      NL (0x107)             0.001    0.1%   99.6%     ai_node_3
   4       4      Dense (0x104)          0.001    0.2%   99.8%     ai_node_4
   ------------------------------------------------------------------------
   total                                 0.388
   ------------------------------------------------------------------------
   Statistic per tensor
   ------------------------------------------------------------------------------
   tensor   #    type[shape]:size         min     max     mean      std  name
   ------------------------------------------------------------------------------
   I.0      10   f32[1,100,1,1]:400     0.005   1.000    0.490    0.292  input_1
   O.0      10   f32[1,1,1,1]:4         0.277   0.931    0.588    0.190  output_1
   ------------------------------------------------------------------------------
Saving validation data...
 output directory: C:\Users\ASUS\.stm32cubemx\network_1_output
 creating C:\Users\ASUS\.stm32cubemx\network_1_output\network_1_val_io.npz
 m_outputs_1: (10, 1, 1, 1)/float32, min/max=[0.277, 0.931], mean/std=[0.588, 0.190], gemm_4
 c_outputs_1: (10, 1, 1, 1)/float32, min/max=[0.277, 0.931], mean/std=[0.588, 0.190], gemm_4
Computing the metrics...
 Cross accuracy report #1 (reference vs C-model)
 ----------------------------------------------------------------------------------------------------
 notes: - the output of the reference model is used as ground truth/reference value
        - 10 samples (1 items per sample)
   c=n.a., rmse=0.000776879, mae=0.000672056, l2r=0.001223652, nse=1.000, cos=1.000
   luation report (summary)
   ----------------------------------------------------------------------------------------------------------------------------------
   put       acc     rmse         mae          l2r          mean         std          nse          cos          tensor
   ----------------------------------------------------------------------------------------------------------------------------------
   ross #1   n.a.    0.0007769    0.0006721    0.0012237    0.0005859    0.0005377    0.9999909    0.9999994    gemm_4, (
   ----------------------------------------------------------------------------------------------------------------------------------
   c  : Classification accuracy (all classes)
   se : Root Mean Squared Error
   e  : Mean Absolute Error
   r  : L2 relative error
   e  : Nash-Sutcliffe efficiency criteria, bigger is better, best=1, range=(-inf, 1]
   s  : COsine Similarity, bigger is better, best=1, range=(0, 1]
   ting txt report file C:\Users\acer\.stm32cubemx\consumption_model_output\consumption_model_validate_report.txt
   sed time (validate): 3.379s
   ation ended
```