SONY

AITRIOS

Vision and Sensing Application SDK Model Quantization Functional Specifications

Version 0.2.0

2023 - 1 - 30

TOC

1. Change history

Date	What/Why
2022/11/16	Initial draft
2023/01/30	Fixed how images are stored for quantization calibration and evaluation. Added how to store the label information file used during evaluation. Removed the evaluate_ground_truth_file setting and added the evaluate_label_file setting. Fixed the default path for the dataset_image_dir, evaluate_image_dir settings. Added supported image formats. Directory structure change. Added that symbolic links must not be used. Following the deletion of the secret information, Initial was removed from the configuration file entry and added about creating a configuration file. Updated the PDF build environment.

2. Terms/Abbreviations

Terms/Abbreviations	Meaning
MCT	Open source software for quantizing neural network models
Keras	A Keras model is a type of neural network format
TFLite	TensorFlow Lite A *.tflite* model is a type of neural network format
Iteration	One occasion of neural network model training

3. Reference materials

- Reference/Related documents
  - Model Compression Toolkit (MCT)
    - https://github.com/sony/model_optimization

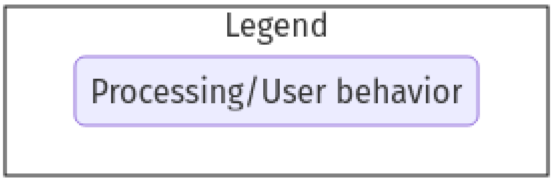
4. Expected use case

- You want to quantize the model
  You want to reduce the size of the model by quantization so that it can be deployed to the target edge AI device
- You want to run inferences and confirm accuracy using pre- and post-quantization models

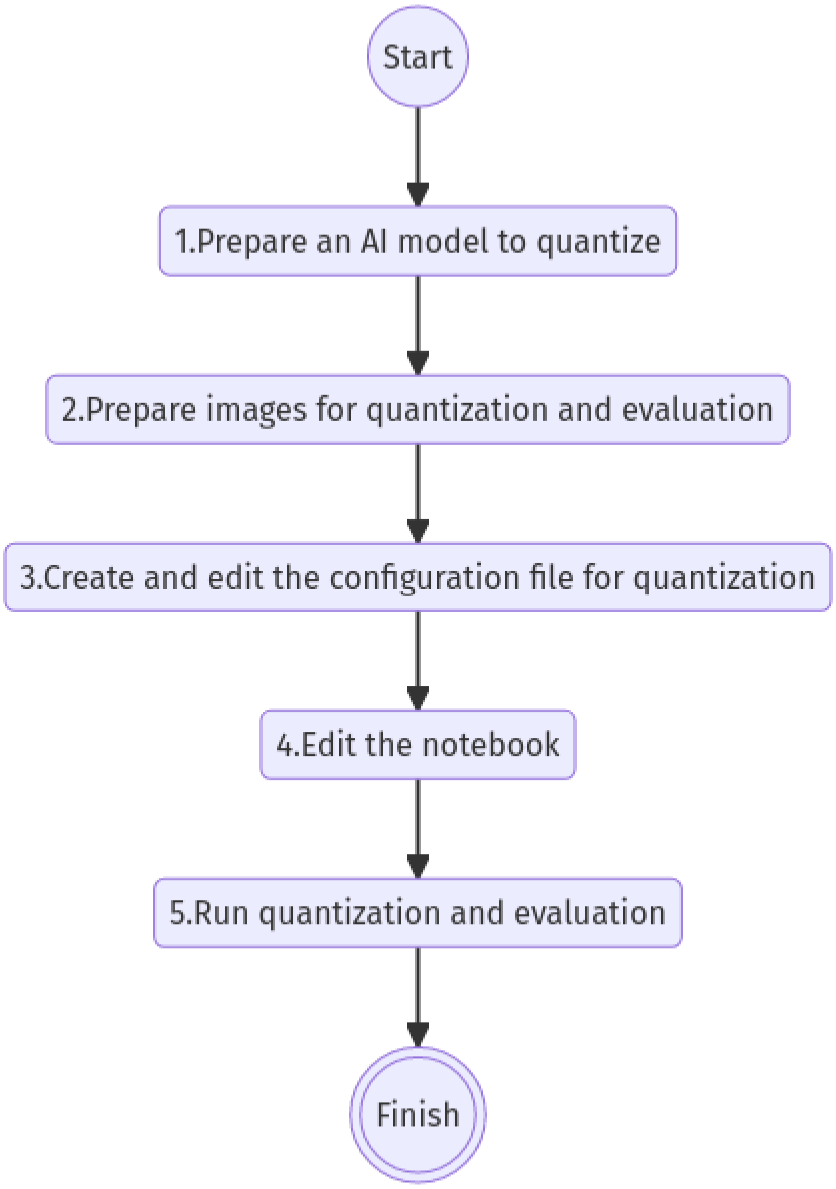
5. Functional overview/Algorithm

Functional overview

- The SDK quantizes the AI model (Keras) of image classification and converts it to an AI model (TFLite) in the following flow
- The SDK runs inferences on pre- and post-quantization AI models to get statistics (Top1 accuracy) for the inference results
- The AI models supported by the SDK conform to MCT supported features
- The image format supported by the SDK is JPEG



- Flow overview


Start
1.Prepare an AI model to quantize
2.Prepare images for quantization and evaluation
3.Create and edit the configuration file for quantization
4.Edit the notebook
5.Run quantization and evaluation
Finish

- Flow details

    1. Prepare an AI model to quantize

        - Prepare an AI model (Keras) to quantize

    2. Prepare images for quantization and evaluation

        - Prepare images used in training AI models because they will be used to calibrate the quantization

        - Prepare images and their label information for validation of AI models because they will be used as input during inference evaluation

    3. Create and edit the configuration file for quantization

        - Create and edit the configuration file *configuration.json* to configure notebook runtime settings

    4. Edit the notebook

        - Modify the implementation of the preprocessing part for calibration in the notebook according to the AI model used

    5. Run quantization and evaluation

        - Run the notebook that quantizes an AI model (Keras), converts it to an AI model (TFLite), and evaluates inferences

6. User interface specifications

How to start each function

1. Launch the SDK environment and preview the `README.md` in the top directory
2. Jump to the `README.md` in the `tutorials` directory from the hyperlink in the SDK environment top directory
3. Jump to the `README.md` in the `3_prepare_model` directory from the hyperlink in the `README.md` in the `tutorials` directory
4. Jump to the `README.md` in the `develop_on_sdk` directory from the hyperlink in the `README.md` in the `3_prepare_model` directory
5. Jump to the `README.md` in the `2_quantize_model` directory from the hyperlink in the `README.md` in the `develop_on_sdk` directory
6. Jump to the `README.md` in the `image_classification` directory from the hyperlink in the `README.md` in the `2_quantize_model` directory
7. Jump to each feature from each file in the `image_classification` directory

Prepare an AI model to quantize

1. Prepare an AI model (Keras) to quantize
    - Store the AI model (Keras) to be quantized in the SDK execution environment.

Prepare images for quantization and evaluation

1. Prepare images used in training AI models because they will be used to calibrate the quantization

    - Store the directory containing the images, about 300 files, used in training the AI model in the SDK execution environment.

        - For example, if you want to use the *tutorials/_common/dataset* directory, store it as follows:

          ```
          tutorials/
            └ _common
              └ dataset
                ├ training/  (1)
                │  ├ Image class name/
                │  │    └ Image file
                │  ├ Image class name/
                │  │    └ Image file
                │  ├ · · · ·
          ```

          (1) Dataset used during training. This directory can have any subdirectory structure.

2. Create annotation data and label information file according to the directory structure for ImageNet 1.0 format for use as input during inference evaluation.

   - Set up a directory for images to use for validation of the AI model. Store it in the SDK execution environment.

     - For example, if you want to use the *tutorials/_common/dataset* directory, store it as follows:

       ```
       tutorials/
         └ _common
           └ dataset
             ├ validation/ (1)
             |  ├ Image class name/
             |  |    └ Image file
             |  ├ Image class name/
             |  |    └ Image file
             |  ├ ・・・・
             └ labels.json (2)
       ```

       (1) Dataset used during evaluation. As described in the preceding create it according to the directory structure for ImageNet 1.0 format.

       (2) Label information file

     - The format of label information files is a json file with the label name and its id value as follows:

       ```
       {"daisy": 0, "dandelion": 1, "roses": 2, "sunflowers": 3, "tulips": 4}
       ```

> See "Convert annotation information format" in the "CVAT Image Annotation Functional Specifications" for how to convert the format of annotation information to the preceding format when quantizing a user-prepared AI model.

Create and edit the configuration file for quantization

1. Create and edit the configuration file, `configuration.json`, in the execution directory.

> If you want to run image classification, the run directory becomes `quantize_model/image_classification`.

ℹ All parameters are required, unless otherwise indicated.

ℹ All values are case sensitive, unless otherwise indicated.

ℹ Do not use symbolic links to files and directories.

Configuration	Meaning	Range	Remarks
`source_keras_model`	Path to the AI model (Keras) to convert from. Specify a directory in Keras SavedModel format, or a file in h5 format.	Absolute path or relative to notebook (*.ipynb)	
`dataset_image_dir`	Directory containing dataset images for calibration during quantization	Absolute path or relative to notebook (*.ipynb)	
`batch_size`	Number of sets of images to be calibrated during quantization to find features such as weights and biases	1 or more and less than or equal to the total number of images contained in **`dataset_image_dir`**	
`input_tensor_size`	Size of the AI model input tensor (number of pixels on one side of image)	Comply with AI model input tensor	
`iteration_count`	Number of iterations when quantizing	1 or more	
`output_dir`	Directory to store the quantized AI model	Absolute path or relative to notebook (*.ipynb)	

Configuration	Meaning	Range	Remarks
`evaluate_image_dir`	Directory containing images to use as input during inference	Absolute path or relative to notebook (*.ipynb)	
`evaluate_image_extension`	Extension of image files to use as input during inference	String	
`evaluate_label_file`	Label information for AI models	Absolute path or relative to notebook (*.ipynb)	
`evaluate_result_dir`	Directory to store statistics of inference results	Absolute path or relative to notebook (*.ipynb)	

Edit the notebook

1. Open the notebook for running quantization, **.ipynb*, in the execution directory.
2. Edit the preprocessing part of the notebook for calibration.
   - Edit the `FolderImageLoader` argument `preprocessing=[resize, normalization]` to set it equivalent to a preprocessing operation when training your AI model.

Run quantization and evaluation

1. Open the notebook for running quantization, **.ipynb*, in the execution directory, and run the python scripts in it.
    - The script does the following:
        - Checks that *configuration.json* exists in the execution directory.
            - If an error occurs, the error description is displayed and running is interrupted.
        - Checks that *configuration.json* includes values for `source_keras_model` and `dataset_image_dir`.
            - If an error occurs, the error description is displayed and running is interrupted.
        - Reads the following values from *configuration.json*, makes the necessary settings in MCT, and then quantizes and converts the AI model (Keras):
            - *configuration.json* `source_keras_model`
            - *configuration.json* `dataset_image_dir`
            - *configuration.json* `batch_size`
            - *configuration.json* `input_tensor_size`
            - *configuration.json* `iteration_count`
        - If an error occurs in external software, for example, MCT, the error output by the external software is displayed and running is interrupted.
        - Outputs the AI model quantized by MCT (TFLite) `model_quantized.tflite` , and the AI model converted to TFLite by standard TensorFlow functionality (TFLite) `model.tflite` to the directory specified in *configuration.json* `output_dir`.
            - If the directory specified by `output_dir` does not already exist, it is created at the same time.
        - During conversion, the notebook will display information as follows(when `iteration_count` is 10), for example:

          ```
            0%|          | 0/10 [00:00<?, ?it/s]
          ...
           30%|███       | 3/10 [00:15<00:35,  5.10s/it]
          ...
          100%|██████████| 10/10 [00:50<00:00,  5.07s/it]
          ```

        - Checks that *configuration.json* includes values for `output_dir`, `evaluate_image_dir`, `evaluate_label_file`.
            - If an error occurs, the error description is displayed and running is interrupted.

- Reads the following values from *configuration.json*, makes the necessary settings for the tflite interpreter:
  - *configuration.json* `output_dir`
  - *configuration.json* `evaluate_image_dir`
  - *configuration.json* `evaluate_image_extension`
  - *configuration.json* `evaluate_labe_file`
  - *configuration.json* `evaluate_result_dir`
- Runs inference and displays statistics for three types of AI model: the original AI model (Keras), the AI model converted to TFLite by standard TensorFlow functionality (TFLite), and the AI model quantized by MCT (TFLite).
- Saves statistics as the file `results.json` in the directory specified in `evaluate_result_dir`.
- If an error occurs in external software, for example, TensorFlow, the error output by the external software is displayed and running is interrupted.
- While the AI model (TFLite) is being inferred, information is displayed as follows (when the number of images is 10), for example:

```
  0%|          | 0/10 [00:00<?, ?it/s]
...
 40%|████      | 4/10 [00:03<00:05,  1.08it/s]
...
100%|██████████| 10/10 [00:09<00:00,  1.08it/s]
```

- While the AI model (Kera) is being inferred, logs from TensorFlow library are displayed.
- While processing, you can interrupt with the Stop Cell Execution of notebook cell function.

7. Target performances/Impact on performances

- When the SDK environment is built, AI models (Keras) can be quantized and converted to AI models (TFLite) without any additional installation steps
- UI response time of 1.2 seconds or less
- If processing takes more than 5 seconds, then the display during processing can be updated sequentially

8. Assumption/Restriction

- None

9. Remarks

- To check the versions of Model Compression Toolkit (MCT) and TensorFlow
    - See *requirements.txt* in the SDK environment root directory.

10. Unconfirmed items

- None