Documentation of Search Action Basesd on Reinforcement Learning

Yanzhou Jin, Chenghao Li

March 28, 2024

1 Introduction to the program

Our program implements mixed-precision search based on reinforcement learning. It offers three different operation modes, which is introduced as follows:

- **Training from sketch:** In this operation mode, the search agent is trained from sketch based on the configuration file (*.toml), the trained model will be saved in the file `mase_output`, and the auto-saved models can be find in the file named `logs`. This operation mode can be selected by setting `mode` to 'train' inside the configuration file.
- **Loading trained model:** In this operation mode, the agent will perform mixed-precision search action based on the trained model. In this mode, the model will perform a specified number of searches and then output the best result. To run this operation mode, `mode` need to be set to 'load' inside the configuration file.
- **Continue training:** In this operation mode, the agent will trained based on a trained model, that is, continue training on the saved model. In this mode, the model will train for specific steps defined in the configuration file. To run this operation mode, `mode` need to be set to 'continue-training' inside the configuration file.

In order to make parameter adjustment simpler, parameters configuration is done using a configuration file following the example of optuna search. The `vgg7_rl_search.toml` file was created to store the reinforcement learning search parameters for the VGG7 model. These parameters mainly include the parameter for quantization and other parameters used in reinforcement learning. The names and value ranges of the main parameters are shown in the table 1.

In the table, `X_width` and `X_frac_width` denote quantization parameters such as `bias_width` and `bias_frac_width`, which are represented as multi-dimensional integer arrays. `algorithm` is the policy used in reinforcement learning, `load_path`is used to specify the file path of any stored reinforcement learning model, usually used in load or continue training mode. `device` is the processor that runs the reinforcement learning algorithm, `save_name` is the name where the trained model is stored, `mode` is the mode in which the program runs, train represents train from sketch, and load represents reading. `load_path` model, continue-training means reading the `load_path` model and continuing training.`total_timesteps` is the total time steps in the training process.

Name	Value	Name	Value
X_width	`Integer_array`	**X_frac_width**	`Integer_array`
algorithm	`'a2c', 'ppo'`	**load_path**	`String`
device	`'cpu', 'cuda'`	**save_name**	`String`
env	'mixed_precision'	**mode**	'load', 'train',
total_timesteps	`Integer`		'continue-training'

Table 1: Configuration Parameters

Program Execution By running the following code, the program can be executed in the terminal. The execution mode can be switched by modifying the configuration file located at `'mase/machop/configs/examples /vgg7_rl_search.toml '`.

./ch search –config /path/to/toml-file –load /path/to/check-point

2 Main Modification to the Code

The modification gathered mainly in three files, which are `core_algorithm.py`, `env.py` and `quantize.py`. The first two are responsible for executing the core algorithm of reinforcement learning and defining the environment respectively. The reason for modifying `quantize.py` is due to a specific problem: every time a quantization operation is performed, the program does not follow the settings specified by the configuration file, causing the average bitwidth to remain unchanged. To solve this problem, the `graph_iterator_quantize_by_type2` function was copied and inserted into `quantize.py`, and one line of code was modified, as shown in the figure 1.

```
# Original code
node_config = get_config(config, get_mase_op(node))

# Modified code
node_config = get_config(config, node.name)
```

Figure 1: Modification in the quantize.py

Through the above modifications, the integration of reinforcement learning search action in the MASE system is achieved. Users can now execute commands via the terminal and correctly configure the *.toml configuration file to perform parameter adjustment for training and reading of the reinforcement learning model.