Introduction

The submitted artifact is a novel **sequential meta-learning model** for software configuration performance prediction, as proposed in the paper:

> Jingzhi Gong and Tao Chen, **Predicting Configuration Performance in Multiple Environments with Sequential Meta-Learning**, FSE'24

Availability

Souce codes and datasets of the artifact are available at:

> https://github.com/ideas-labo/SeMPL
>
> https://zenodo.org/doi/10.5281/zenodo.11072487

Installation

1. Download all the files into the same folder / clone the repository.
2. Install the specified version of Python: the codes have been tested with **Python 3.6 - 3.9**, **tensorflow 2.12 - 2.16**, and **keras < 3.0**, other versions might cause errors.
3. Using the command line: cd to the folder with the codes, and install all the required packages by running:

```
pip install -r requirements.txt
```

Run *SeMPL*

- **Command line**: cd to the folder with the codes, input the command below, and the rest of the processes will be fully automated.

```
python SeMPL_main.py
```

- **Python IDE (e.g. Pycharm)**: Open the *SeMPL_main.py* file on the IDE, and simply click 'Run'.

Demo Experiment

The main program *SeMPL_main.py* runs a demo experiment that evaluates *SeMPL* with 5 sample sizes of *ImageMagick*, each repeated 30 times, without hyperparameter tuning (to save demonstration time).

A **successful run** would produce similar messages as below:

```
Dataset: imagemagick-4environments
Number of expriments: 30
Total sample size: 100
Number of features: 5
Training sizes: [11, 24, 45, 66, 70]
Total number of environments: 4
--- Subject system: imagemagick, Size: S_1 ---
Training size: 11, testing size: 89, Meta-training size (100% samples): 100
> Sequence selection...
    Target_environment: [best sequence] --- {0: [[1, 3, 2]], 1: [[0, 2, 3]], 2: [[1, 3, 0]], 3: [[1, 0, 2]]}
    >> Sequence selection time (min): 0.03

> Meta-training in order [1, 3, 2] for target environment E_0...
    >> Learning environment 1...
    >> Learning environment 3...
    >> Learning environment 2...
    >> Meta training time (min): 0.07

> Fine-tuning...
    >> Run1 imagemagick-4environments S_1 E_0 MRE: 7.80, Training time (min): 0.02
    >> Run2 imagemagick-4environments S_1 E_0 MRE: 8.99, Training time (min): 0.01
    >> Run3 imagemagick-4environments S_1 E_0 MRE: 8.32, Training time (min): 0.01
         ...
```

The results will be saved in a file at the *results* directory with name in the format *'System_Mainenvironment_MetaModel_FineTuningSamples-MetaSamples_Date'*, for example *'imagemagick-4environments_T0_M[3, 1, 2]_11-100_03-28.txt'*.

Experiment Results Replication

To replicate the experiments in the paper, simple copy the codes to replace the lines 18-33 in *SeMPL_main.py*.

```
########### experiment parameters ###########
selected_sys = range(9)  # set the subject systems to evaluate
selected_sizes = [0,1,2,3,4]  # set the training sample sizes to evaluate
save_MRE = True # save the evaluation results
test_mode = True # to tune the DNN hyperparameters
save_best_sequence = False # to save the selected best sequences
save_meta_model = False # to save the pre-trained meta models
read_meta_model = True # Load the pre-trained meta model if exists
seed = 2
N_experiments = 30
start = 0
meta_sample_percentage = 100 # from 0 to 100
max_epoch = 1000
learned_environments = [] # to exclude the specified target environments
learned_meta_models = [] # to exclude the specified meta models
########### experiment parameters ###########
```

To run other experiment settings, alter the codes following the instructions below and comments in *SeMPL_main.py*.

To switch between subject systems

```
Change the line 19 in SeMPL_main.py

E.g., to run SeMPL with DeepArch and SaC, simply write 'selected_sys = [0, 1]'.
```

To tune the hyperparameters (takes longer time)

```
Set line 22 with 'test_mode = False'.
```

To change the number of experiments for specified sample size(s)

```
Change 'N_experiments' at line 27.
```

State-of-the-art Performance Prediction Models

Below are the repositories of the SOTA performance prediction models, which are evaluated and compared with *SeMPL* in the paper.

Single Environment Performance Models

- DeepPerf

  A deep neural network performance model with L1 regularization and efficient hyperparameter tuning.

- RF

  A commonly used ensemble of trees that tackle the feature sparsity issue.

- DECART

  An improved regression tree with a data-efficient sampling method.

- SPLConqueror

  Linear regression with optimal binary and numerical sampling method and stepwise feature selection.

- XGBoost

  A gradient-boosting algorithm that leverages the combination of multiple weak trees to create a robust ensemble.

Joint Learning for Performance Models

- BEETLE

  A model that selects the bellwether environment for transfer learning.

- tEAMS

  A recent approach that reuses and transfers the performance model during software evolution.

- MORF

A multi-environment learning version of RF where there is one dedicated output for each environment of performance prediction.

Meta-Learning Models

- MAML

  A state-of-the-art meta-learning framework that has been widely applied in different domains, including software engineering.

- MetaSGD

  Extends the MAML by additionally adapting the learning rate along the meta-training process, achieving learning speedup over MAML

To compare *SeMPL* with other SOTA models, please refer to their original pages (you might have to modify or reproduce their codes to ensure the compared models share the same set of training and testing samples).