PROGRESS REPORT MAY 27th 2024 – CUI SHUAIWEN

**I – Smart Adaptive Triggering**

**Fixed Issue:** the particle swamp optimization process does not converge or converge appropriately, making the results less convincing.

**Results:**

The convergence issue was fixed.

Current optimization results: (beta = 1.75 for f-beta score)

A screenshot of a computer program

Description automatically generated

Iteration details:

Iteration: 1 / 15

current max fitness: 0.7089297886843899

Global Best: 0.708930 | threshold: 0.020095 | duration: 3.000000 | precision: 0.991383 | recall: 0.648591 [UPDATED!!!]

Iteration: 2 / 15

current max fitness: 0.9292932027271075

Global Best: 0.929293 | threshold: 0.002179 | duration: 4.000000 | precision: 0.777661 | recall: 0.992483 [UPDATED!!!]

Iteration: 3 / 15

current max fitness: 0.9252403358384668

Iteration: 4 / 15

current max fitness: 0.7170419598996817

Iteration: 5 / 15

current max fitness: 0.7122661291413575

Iteration: 6 / 15

**current max fitness: 0.9348037112182279**

**Global Best: 0.934804 | threshold: 0.003876 | duration: 3.971550 | precision: 0.960490 | recall: 0.926711 [UPDATED!!!]**

Iteration: 7 / 15

current max fitness: 0.709362940897705

Iteration: 8 / 15

current max fitness: 0.9222317888022508

Iteration: 9 / 15

current max fitness: 0.9215273888311025

Iteration: 10 / 15

current max fitness: 0.7164566195242534

Iteration: 11 / 15

current max fitness: 0.857590801067918

Iteration: 12 / 15

current max fitness: 0.925119320761199

Iteration: 13 / 15

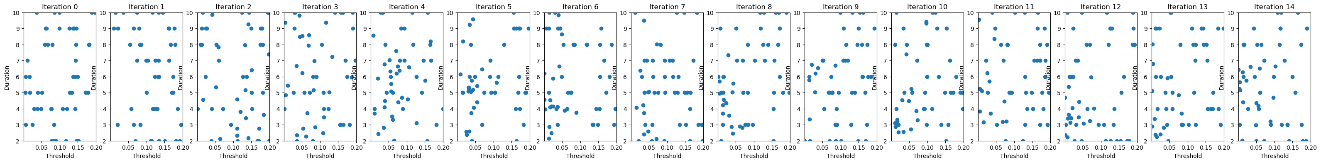
current max fitness: 0.9301252091627717

Iteration: 14 / 15

current max fitness: 0.9332263014202633

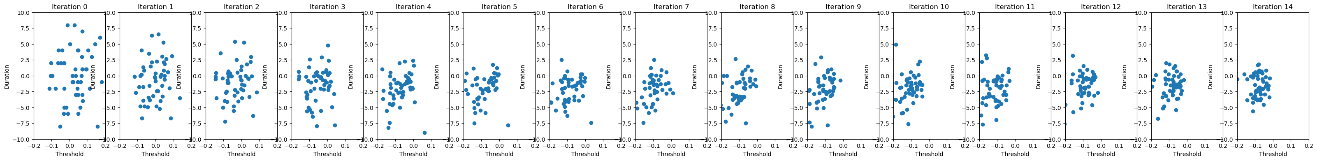
**Visualization:**

Position iteration (parameter iteration)

****

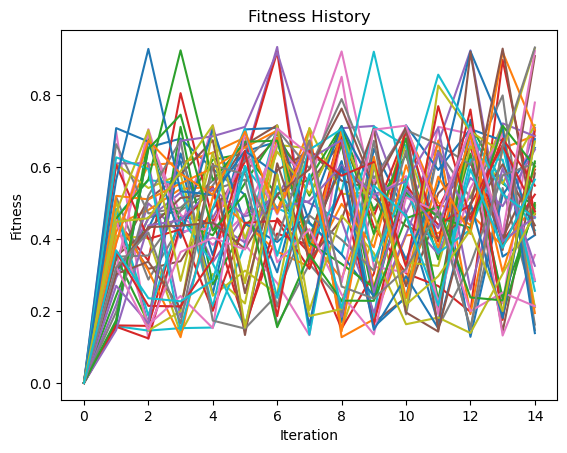
note: some particles aggregated to same positions (local / global maxima)

Velocity iteration (parameter updates in each iteration)

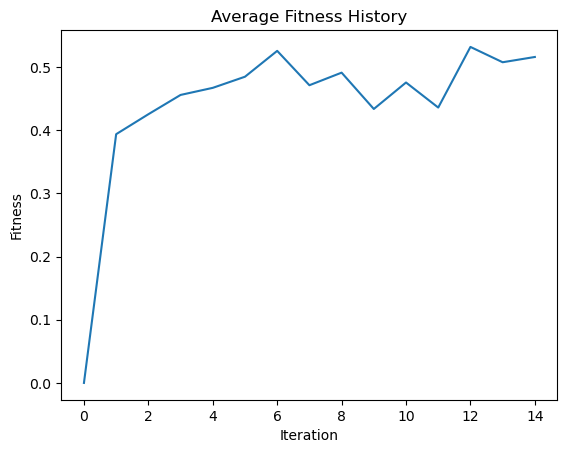


note: the update strides are getting smaller as the number of iterations moves forward, suggesting convergence. But the updates strides did not converge to (0, 0) as the constrains for parameters were applied, i.e., the duration must be integers, and the particles lying outside the scope were reset to appear in random position within the scope.

Fitness History for all particles



Averaged Fitness History



Note: as shown, the convergence trend of the fitness value can be observed.

**Discussion：**

The reason why that the iteration did not converge in previous setup is the parameter constraints were too strict, i.e., the constraints were applied before each iteration. So, I changed the strategy, for each iteration step, conducting iteration without considering constraints first and applying constraints after getting the results.

More specifically, in each iteration of PSO, we do position (triggering parameter) updating first, then calculating the fitness (value) , and finally the velocity updating (direction and stride for parameter updating). The strategy adopted for now is to treat it as a continuous problem first, then do iteration, and then discretize the results to get ideal values. In our case, the triggering threshold is continuous, but the triggering duration is discrete. So, for position updating, the position is updated first, then the duration position is discretized. (take the ceiling integer). For particles outside the scope after updating, their positions will be reset by generating random position within the searching space.

**Next step:**

* Reinforcement Learning as a comparison
* Use estimated event distribution to solve observability issues.

**II – PROTECT 2024**

**Sudden Damage Detection Workflow:**

WT-> ICA->CNN

**Adi’s work:**

Onboard Wavelet Transformation was successfully implemented on STM32 boards, and Adi is now working on onboard ICA.

**My work on onboard CNN classification:**

This part aims to facilitate (1) smart adaptive triggering & (2) transformer onboard implementation & (3) sudden damage detection for PROTECT2024

Together with Yuxiao and Yuqing, we have explored X-Cube-AI for onboard NN implementation on STM32 series MCU, obtaining preliminary results. With the model, data and label properly prepared, the X-Cube-AI plugin can help to analyze the performance of the model, validate and compare the results on board and PC.

As there are model compatibility issues, I rewrote CNN model for event classification with pure Keras and saved it as .h5 file for validation. And the model was validated on a small scale dataset (40 events for each type, 160 in total) and reached 100% accuracy on both computer and dev boards. I am currently trying larger dataset further test its performance.

X-Cube-AI also provides API for users to integrate the model in their own codes, but for now, more time for exploration is quired. This part should be able to help the sudden damage detection project and smart adaptive triggering mechanism in future.

However, should note that X-Cube-AI is exclusively designed for STM32 series MCU, meaning this method can not be extended for Xnode, we need to find other ways for Xnode NN implementation.