GPM: Genetic Programming with Memory for Symbolic Regression

Author: Bc. Tadeáš Jůza
Supervisor: prof. Ing. Lukáš Sekanina, Ph.D.

BRNO UNIVERSITY OF TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2024

The symbolic regression task is solved by Genetic Programming with Memory (GPM) which uses small memory to store various data points to better approximate the original data set.

$$y'(x) = \begin{cases} AM(x) & if\ x\ is\ in\ AM \\ G(x) & otherwise \end{cases}$$

Equation 1: If x is in Associative Memory (AM) then return AM(x) else G(x), where G is an expression evolved by Cartesian Genetic Programming (CGP). CGP minimizes MAE(y, y').

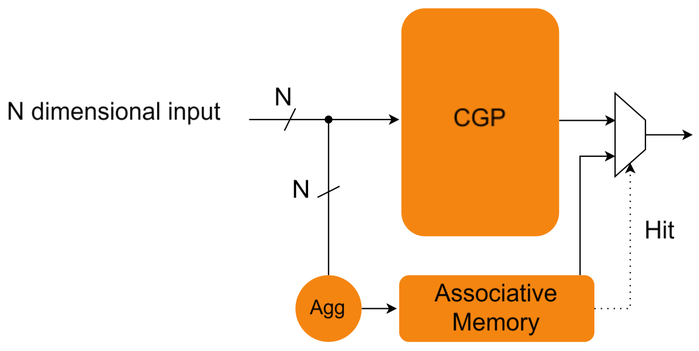
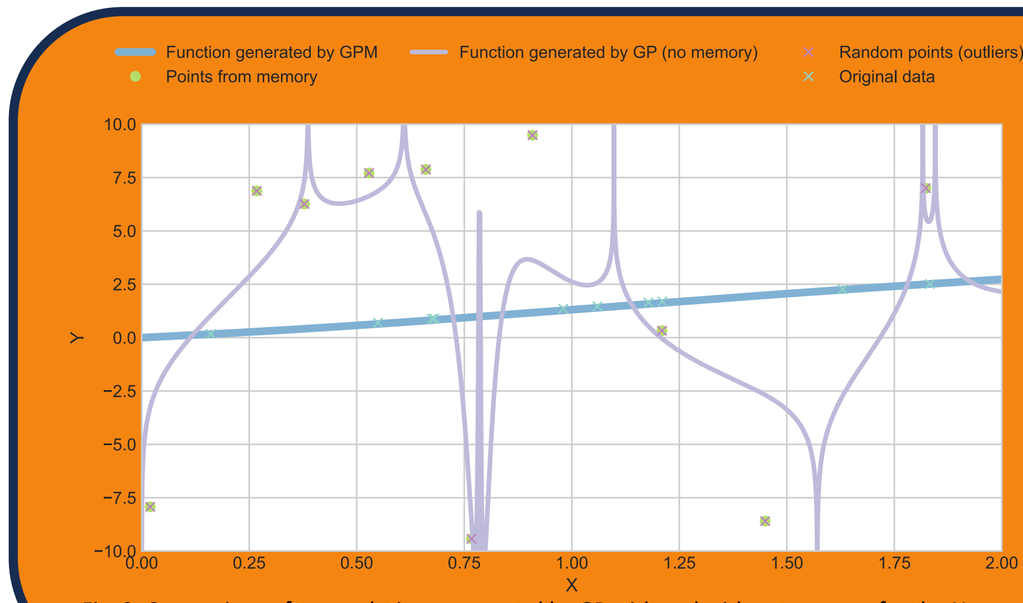


Fig. 1: Overview of GPM.

Table 1: Benchmark datasets for testing the GPM.

Name	Functions	Number of points	Interval	Step
Koza-1	$f(x) = x^4 + x^3 + x^2 + x$	40	[-1, 1]	0.05
Nguyen-7	$f(x) = \ln(x + 1) + \ln(x^2 + 1)$	20	[0, 2]	random
Nguyen-10	$f(x_0, x_1) = 2 * \sin(x_0) * \cos(x_1)$	100	[-1, 1]	random
Korns-4	$f(x_0, x_1, x_2, x_3, x_4) = -2{,}3 + 0{,}13 * \sin(x_2)$	10 000	[-50, 50]	random
Keijzer-1	$f(x) = 0{,}3 * x * \sin(2 * \pi * x)$	20	[-1, 1]	0.1
Keijzer-8	$f(x) = \sqrt{x}$	100	[0, 100]	1
Vladislavleva-5	$f(x_0, x_1, x_2) = 30\frac{(x_0 - 1) * (x_2 - 1)}{x_1^2 * (x_0 - 10)}$	300	x_0, x_2:[0.05, 2] x_1:[1, 2]	random
Vladislavleva-1	$f(x_0, x_1) = \frac{e^{-(x_0-1)^2}}{1{,}2 + (x_1 - 2{,}5)^2}$	100	[0.3, 4]	random
Vladislavleva-2	$f(x) = e^{-x} * x^3 * (\cos(x) * \sin(x)) * (\cos(x) * \sin^2(x) - 1)$	100	[0.05, 10]	0.1



The graph shows that by adding memory to GP, a complex function (purple) becomes a much simpler function (blue):
$G(x) = \ln(|e^x + x^3|)$

The random points contained in the dataset are stored in the memory.

Fig. 2: Comparison of two solutions generated by GP with and without memory for the Nguyen-7 function augmented with random points.

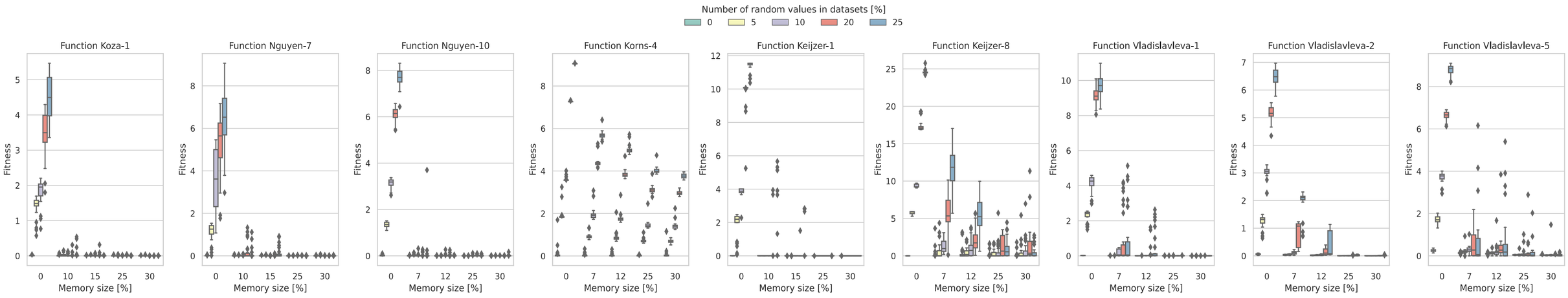


Fig. 3: Results on benchmark functions (contaminated with some randomly generated values) obtained by GPM utilizing various memory capacity. The goal is to minimize fitness.

GPM-based On-chip Weight Generator for CNNs

GPM serves as an on-chip weight generator for CNN to reduce expensive reads from external weight memory. GPM's associative memory is replaced by cheaper standard memory in this application.

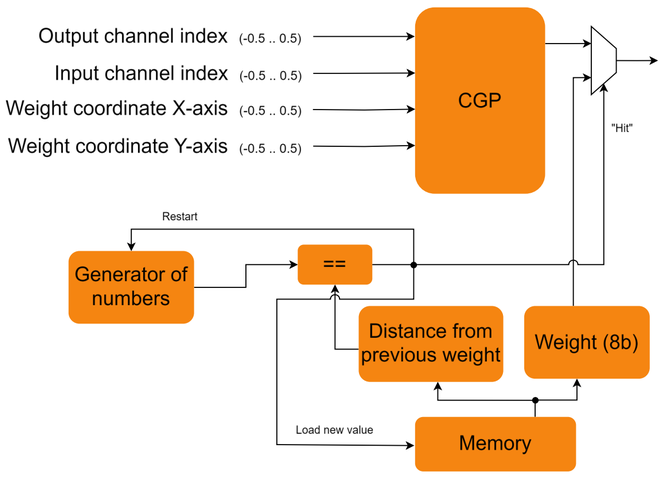


Fig. 4: Generating weights for CNN with an already found function and memory content.

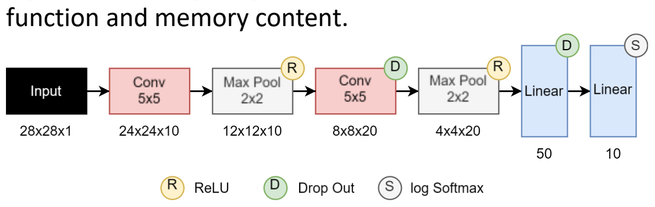


Fig. 5: CNN for MNIST classification used to evaluate the proposed method.

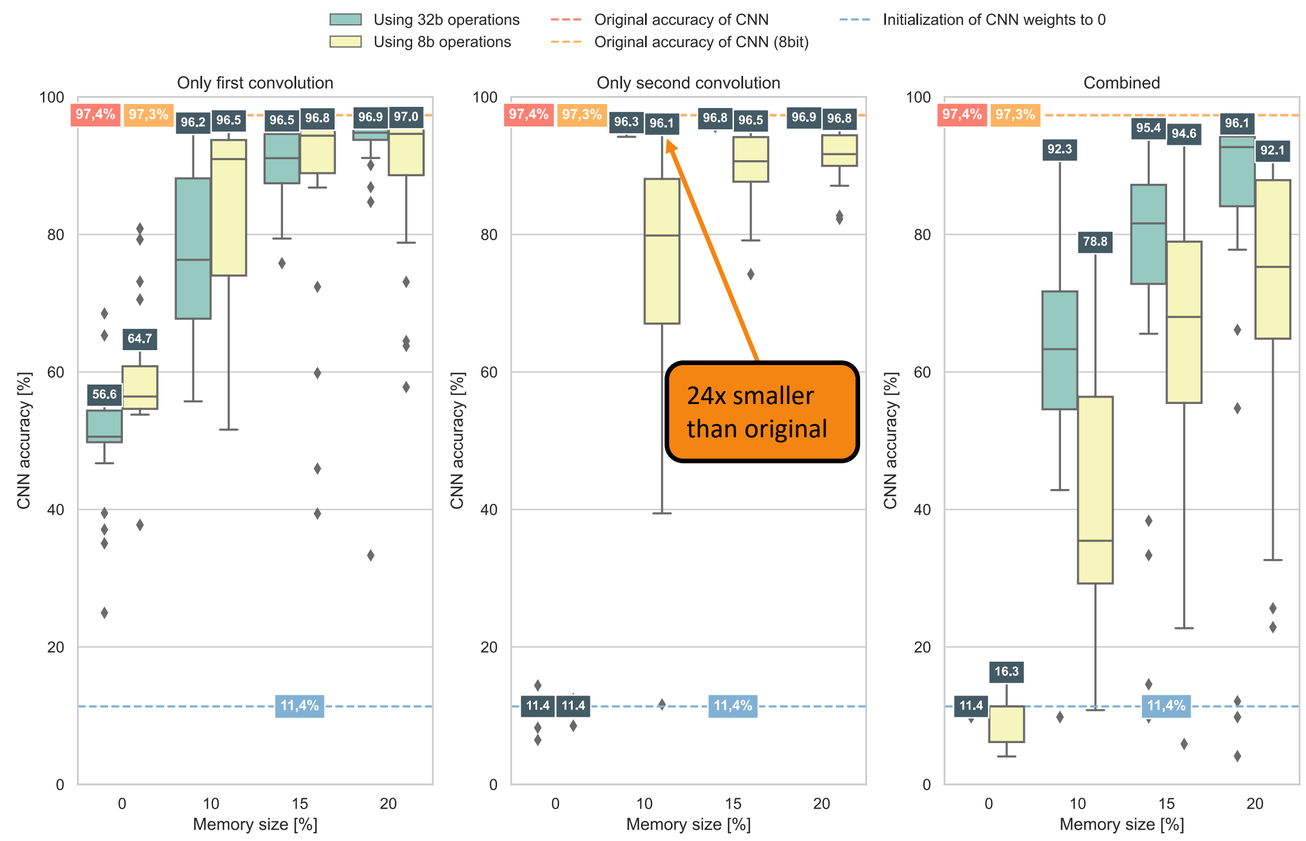


Fig. 6: Classification accuracy when the original weights are replaced by the weights generated by GPM utilizing a fraction [%] of the original weights.