GPL, MLP

Are symbolic regressor:

Takes in (R^(n+1) → string)

Eg it takes in {(1, 2), (2, 3), (3, 4)} → “x + 1”

To get a baseline for MLP:

Give three things (and withhold one thing)

X, Y, and TX (withhold TY)

MLP will take X and Y and return eqn.

We take the eqn and TX and we get PY

Then we can compare PY with TY for the error score.

Focus on 1 variable

30 points?

Try to get it to work

On [0, 3]

y=x

y=3-x

|  |  |  |
| --- | --- | --- |
| **Hyperparameter** | **Simple** | **Complicated** |
| # variables | 1 | 2+ |
| Datapoints spacing | Mesh | random |
| # datapoints | Many [30] | few |
| Set of operators | Few [+, x, sqrt, sin] | many |
| Equation complexity | Simple (few levels) 2 | Complex (many levels) |
| Interval | Smaller [0, 3] | Bigger |
| Input to model | (x,y) | (x, y) |
| # decimals | 2 | 4+ |
| # number of training equations | Many (5 million?) | Many many more |
| Constant values: how many | Not too many (50%) | Many (80%) |
| Constant values: range | Small [-1, 1] | Large (-5, 5) |
|  |  |  |
| Use pointnet? | Yeah - deep!! |  |
| Sort datapoints? | irrelevant |  |
| Output constants in eqn? | No, learn structure only |  |

x:

(2.8, 2.8)->f10

(0.1, 0.1)->f1

(0.2, 0.2)->f2

…

(2.9, 2.9)->f11 ---> f2

3-x:

(0.1, 2.9)->g1

(0.2, 2.8)->g2

(2.8, 0.2) ->g10

(2.9, 0.1)->g11