###EvoloPy-NN: An open source nature-inspired optimization Framework for Training Multilayer Perceptron Neural Network in Python

The EvoloPy-NN framework provides classical and recent nature-inspired metaheuristic for training a single layer Multilayer Perceptron Neural Network. The list of optimizers that have been implemented includes Particle Swarm Optimization (PSO), Multi-Verse Optimizer (MVO), Grey Wolf Optimizer (GWO), and Moth Flame Optimization (MFO). The full list of implemented optimizers is available here https://github.com/7ossam81/EvoloPy/wiki/List-of-optimizers

<div style="text-align:center"><img src="https://cloud.githubusercontent.com/assets/17023748/21052168/d61dd09e-be23-11e6-9c59-58f000bff11e.jpg" height="40%" width="40%"></div>

##Features

- Six nature-inspired metaheuristic optimizers are implemented.

- The implimentation uses the fast array manipulation using [`NumPy`] (http://www.numpy.org/).

- Matrix support using [`SciPy`'s] (https://www.scipy.org/) package.

- More optimizers are comming soon.

- Only binary classification problems are supported so far.

- The implimentation uses the powerful Neural Network Library [`neurolab`] (https://pythonhosted.org/neurolab/).

##Installation

- Python 3.xx is required.

Run

pip3 install -r requirements.txt

(possibly with `sudo`)

That command above will install `sklearn`, `NumPy` and `SciPy` for

you.

- If you are installing EvoloPy-NN onto Windows, please Install Anaconda from here https://www.continuum.io/downloads, which is the leading open data science platform powered by Python.

- If you are installing onto Ubuntu or Debian and using Python 3 then

this will pull in all the dependencies from the repositories:

sudo apt-get install python3-numpy python3-scipy liblapack-dev libatlas-base-dev libgsl0-dev fftw-dev libglpk-dev libdsdp-dev

##Get the source

Clone the Git repository from GitHub

git clone https://github.com/7ossam81/EvoloPy-NN.git

##Quick User Guide

EvoloPy-NN Framework contains six datasets (All of them are obtainied from UCI repository).

The main file is the main.py, which considered the interface of the framewok. In the main.py you can setup your experiment by selecting the optmizers, the datasets, number of runs, number of iterations, number of neurons

and population size. The following is a sample example to use the EvoloPy-NN framework.

To choose PSO optimizer for your experiment, change the PSO flag to true and others to false.

Select optimizers:

PSO= True

MVO= False

GWO = False

MFO= False

CS= False

After that, Select datasets:

datasets=["BreastCancer", "Diabetes", "Liver", "Parkinsons", "Vertebral"]

The folder datasets in the repositoriy contains 6 binary datasets (All of them are obtained from UCI repository).

To add new dataset:

- Put your dataset in a csv format (No header is required)

- Normalize/Scale you dataset ([0,1] scaling is prefered).

- Split the dataset into 66% training, and 34% testing.

- Rename the training and testing sets based on the following pattern such as:

[Dataset Name]Train.csv

[Dataset Name]Test.csv

% replace the [Dataset Name] part by the actual name of the dataset.

Example: If the datset name is Seed, the two files will be like the following:

SeedTrain.csv

SeedTest.csv

- Place the new datset files in the datasets folder.

- Add the dataset to the datasets list in the main.py (Line 18).

For example, if the dastaset name is Seed, the new line will be like this:

datasets=["BreastCancer", "Diabetes", "Liver", "Parkinsons", "Vertebral", "Seed"]

Change NumOfRuns, PopulationSize, and Iterations variables as you want:

For Example:

NumOfRuns=10

PopulationSize = 50

Iterations= 1000

Now your experiment is ready to go. Enjoy!

The results will be automaticly generated in excel file called Experiment which is concatnated with the date and time of the experiment.

The results file contains the following measures:

Optimizer: The name of the used optimizer

Dataset: The name of the dataset.

objfname: The objective function/ Fitness function

Experiment: Experiment ID/ Run ID.

startTime: Experiment's starting time

EndTime: Experiment's ending time

ExecutionTime : Experiment's executionTime (in seconds)

trainAcc: Trainig Accuracy

trainTP: Training True Positive

trainFN: Training False Negative

trainFP: Training False Positive

trainTN: Training True Negative

testAcc: Trainig Accuracy

testTP: Training True Positive

testFN: Training False Negative

testFP: Training False Positive

testTN: Training True Negative

Iter1 Iter2 Iter3... : Convergence values (The bjective function values after every iteration).

##Contribute

- Issue Tracker: https://github.com/7ossam81/EvoloPy-NN/issues

- Source Code: https://github.com/7ossam81/EvoloPy-NN

##Support

Use the [issue tracker](https://github.com/7ossam81/EvoloPy-NN/issues).

##Citation Request:

Please include these citations if you plan to use this Framework:

- Hossam Faris, Ibrahim Aljarah, Sayedali Mirjalili, Pedro Castillo, and J.J Merelo. "EvoloPy: An Open-source Nature-inspired Optimization Framework in Python". In Proceedings of the 8th International Joint Conference on Computational Intelligence - Volume 3: ECTA,ISBN 978-989-758-201-1, pages 171-177.

- Hossam Faris, Ibrahim Aljarah, Nailah Al-Madi, and Seyedali Mirjalili. "Optimizing the Learning Process of Feedforward Neural Networks Using Lightning Search Algorithm." International Journal on Artificial Intelligence Tools 25, no. 06 (2016).

- Ibrahim Aljarah, Hossam Faris, Seyedali Mirjalili, Nailah Al-Madi, "Training radial basis function networks using biogeography-based optimizer", Neural Computing and Applications, Springer, August 2016.

- Hossam Faris, Ibrahim Aljarah and Seyedali Mirjalili, "Training feedforward neural networks using multi-verse optimizer for binary classification problems", Applied Intelligence, Springer, March 2016.

- Ibrahim Aljarah, Hossam Faris and Seyedali Mirjalili, “Optimizing connection weights in neural networks using the whale optimization algorithm", Soft Computing, Springer, November 2016