Assessing neural network regularization as a multi-objective optimization problem

A multi-objective optimization algorithm called the Vector Evaluated Particle Swarm Optimization (VEPSO) is benchmarked against a Particle Swarm Optimization (PSO) algorithm which uses regularization tactics. The objective of both approaches is to simultaneously optimize a neural network’s architecture and the neural networks error function on two sets of function approximation and classification problems.

Stuart Gordon Reid
Student number 10026942
Email: stuartgordonreid@gmail.com
Department of Computer Science
University of Pretoria

** A special thanks is paid to Simon van Dyk for an immense amount of assistance in this assignment, especially in setting up and using CiLib. **
Introduction and problem statement

One challenge with using Artificial Neural Networks is the choice of the neural network’s architecture or topology. This choice has an immediate impact on the ability of the neural network to model the problem it is presented as well as the computational complexity of the neural network itself. This challenge has given rise to a new paradigm whereby neural networks are allowed to optimize their own architecture whilst concurrently solving the problem at hand. In essence this creates a multi-objective optimization problem:

1. Optimize the neural network architecture and
2. Optimize the performance of the neural network

One technique, called regularization, penalizes neural networks which are more complex (and not any better) than others. Other techniques attempt to optimize these two problems concurrently without altering the original problem statement. One algorithm for realizing this technique is the Vector Evaluated Particle Swarm Optimization (VEPSO) algorithm. VEPSO has been used previously with varied results to optimize complex multi-objective optimization problems. This research study will attempt to compare these two techniques to optimizing a neural networks architecture while optimizing the neural network’s performance.
Table of contents

Introduction and problem statement
Table of contents
1. Background information
   1.1. Optimization problems
   1.2. Artificial Neural Networks (ANN)
      1.2.1. Feedforward neural networks (FFNN’s)
      1.2.2. Why optimize a neural network in the first place?
   1.3. Particle Swarm Optimization (PSO) algorithms
   1.4. Vector Evaluated Particle Swarm Optimization (VEPSO) algorithms
   1.5. Related literature
2. Design and implementation
   2.1. Computational Intelligence Library (CiLib)
   2.2. Problem sets
      2.2.1. Classification problem set
      2.2.2. Function approximation problem set
   2.3. Algorithm parameters
      2.3.1. Particle Swarm Optimization
      2.3.2. Vector Evaluated Particle Swarm Optimization
3. Results
   3.1. Classification results
      3.1.1. Anderson's Iris data set
      3.1.2. Breast cancer data set
      3.1.3. Breast tissue data set
      3.1.4. Fertility data set
      3.1.5. Parkinsons data set
   3.2. Function approximation results
      3.2.1. Michalewicz function
      3.2.2. Rosenbrock function
      3.2.3. Schubert function
      3.2.4. Schwefel's function
      3.2.5. Sphere function
3. Results continued
   3.3. Pareto fronts
      3.3.1. Anderson's Iris data set
      3.3.2. Breast cancer data set
      3.3.3. Breast tissue data set
      3.3.4. Fertility data set
      3.3.5. Parkinsons data set
   3.4. Key observations in results
4. Conclusions
   4.1. Diversity control in VEPSO
   4.2. Knowledge transfer strategies in VEPSO
   4.3. Fitness of solutions found using VEPSO
   4.4. Performance of regularization
   4.5. Optimization problems
5. Research recommendations
6. Sources and acknowledgements
1. Background information

This section introduces relevant background information. This includes the types of problems being solved as well as the computational models and algorithms being used to solve them.

1.1. Optimization problems

Optimization problems involve finding optimal solutions to a problem. These optimal solutions must lie within the feasible space of the problem. Optimization problems consist of [1]:

1. Objective function(s) - these are functions that quantify the problem being optimized. These can either be minimization or a maximization objective functions.
2. Set of unknown variables - this is the set of variables that have an effect on the objective function. A set of values for the unknown values is referred to as a candidate solution.
3. Set of constraints - constraints are restrictions on candidate solutions. If a candidate solution satisfies the constraints it is said to be a feasible solution to the problem

Optimization problems are categorized according to these elements and depending on how they are categorized the approach to solving them may change. The category dealt with in this research study related to the cardinality of objective functions.

1. Single objective optimization problems (SOO) - these are optimization problems to which there is only one objective function e.g. maximize the return of this trading strategy
2. Multi objective optimization problems (MOO) - these are optimization problems to which there are multiple objective functions e.g. maximize the return of this strategy and minimize the return of this trading strategy

Multi objective optimization problems can be difficult to solve, especially when the objective functions conflict with one another as is this case with risk versus return. As such, a number of techniques have been formulated to help researchers find solutions to these problems:
1. Aggregation methods - the two objective functions are combined with one another to form one objective function. An example is to optimize the so-called ‘risk adjusted return’ using Jensen’s Alpha of the trading strategy.

2. Regularization - the primary objective function is penalized when the secondary optimization problem performs poorly. An example is to penalize trading strategies which take excess risks without improving returns.

3. MOO algorithms - specialized algorithms are created which are able to optimize both objective functions concurrently without simplifying the problem being solve. Maximize each strategies expected returns while reducing risk concurrently.

As stated in the problem statement, this research study will compare the performance of regularization against the VEPSO algorithm which can solve multi objective optimization problems concurrently. The two objectives are to minimize an artificial neural network’s error function and maximize the neural network architecture’s performance.

1.2. Artificial Neural Networks (ANN)

The abilities of the human brain have prompted computer scientists to understand its functions and mechanisms in an attempt to replicate them. Our ability to model the human brain completely is a long way off, but successes have been achieved through simple constructions that closer resemble the brain of a rodent or bat. Such constructions are called artificial neural networks (ANN’s), or neural networks for short. Neural networks are essential computational models capable of approximating non-linear mathematical relationships between input and output data. Neural Networks have been applied to the following classes of problems:

- Classification - predicting the class of an input vector
- Pattern matching - produce a pattern best matching the input vector
- Pattern completion - complete missing parts of an input vector
- Optimization - find the optimal value for parameters in an optimization problem
- Control - given an input vector appropriate actions are suggested
- Time series modelling - functions approximated to learn the functional relationships between inputs and desired output vectors
- Data mining - aim is to discover hidden patterns from data

---

1 Partial list taken from ‘Computational Intelligence - An introduction’ by Andries Engelbrecht
The models used to perform the above mentioned activities are constructed by connecting multiple perceptrons (artificial neurons) that work using activation functions and weights. The specific layout of these neurons with respect to one another is referred to as the architecture of the neural network. The objective function in neural networks is to minimize the sum squared error (SSE) of the network thereby increasing the accuracy of it's mapping. The objective functions in this assignment will be to optimize the SSE of the network as well as the network's architecture concurrently.

1.2.1. Feedforward neural networks (FFNN's)
Feedforward neural networks are designed using the feedforward control pattern. In this pattern there exist multiple layers of neurons (perceptrons). The first layer is the input layer and the last layer is the output layer. Any and all layers in between are called hidden layers. Connections are created between each one of the neurons between each layer. Each neuron is equipped with an activation function that will fire signals forward through the network if some threshold is met. This concept is illustrated below for clarity.

![Feedforward neural network diagram](image-url)
1.2.2. Why optimize a neural network in the first place?

Neural networks can be optimized with respect to the number of neurons present in each layer as well as the number of connections found between layers. Optimization of neural networks is done to improve the performance and the quality of the approximated model in terms of accuracy and reliability. The approach taken to optimizing the neural networks in this research study is to try drive down irrelevant weights on connections between neurons to 0.

1.3. Particle Swarm Optimization (PSO) algorithms

PSO's are population-based iterative search algorithms that manipulate a pool of potential solutions, called particles, using mathematics borrowed from the field of dynamics to find optima. In essence particles are 'flown' over hyper dimensional search spaces toward a global optima. Such algorithms were originally derived from the behaviour of flocks of birds. In a PSO a swarm, is initialized wherein each particle represents a candidate solution to an optimization problem. For an n-dimensional optimization problem, the candidate solution is represented as an n-dimensional vector. During each iteration of the algorithm the global or neighbourhood best position is found from the swarm. The best particle, called gbest, is then used in conjunction with the historical personal best position(s) of each particle in the swarm to calculate a new position in the search space for that particle. These two components are respectively called the social, and cognitive components. This process is shown diagrammatically below:
1.4. Vector Evaluated Particle Swarm Optimization (VEPSO) algorithms

A VEPSO algorithm works similarly to a PSO algorithm except that there are multiple swarms, one for each objective function being optimized. In the case of their being two objective functions (e.g. risk and return) one swarm will be created with particles optimizing the risk and another swarm will be created with particles optimizing the return. The major difference between the PSO and the VEPSO is that the social component (the global best solution, gbest) for each particle is the gbest particle from the other swarm. This is shown diagrammatically below:
1.5. Using PSO’s to train Neural Networks

Particle Swarm Optimization algorithms have been used to train traditional feedforward neural networks. This is done by creating a population of neural networks and encoding each neural network’s internal weights into a vector. Those vectors are then manipulated directly in the search space using the PSO algorithm. This approach is illustrated below:

![Diagram of PSO's application to Neural Networks]

1.6. Related literature

The following papers and textbooks are recommended for readers interested in using optimization algorithms to optimize the architecture of neural networks:


4. Using Particle Swarm Optimization to Train Feedforward Neural Networks in Dynamic Environments, Anna Rakitianskaia, 2011.
2. Design and implementation

This section details the problems used in this research study as well as the configurations used for the simulations which produced the results which supports the conclusion. As such, the simulations are repeatable.

2.1. Computational Intelligence Library (CiLib)

CiLib, the Computational Intelligence Library, consists of two primary libraries: the CiLib source library and the CiLib simulation library. The source library contains coded implementations of a number of computational intelligence constructs, algorithms and other relevant features. These include neural networks and various optimization algorithms including a variety of PSO’s. The simulation library is an XML framework that provides a front-end-interface to the CiLib source library. New algorithms and simulations can be created and tested using the CiLib simulation library. Simulations used in producing this report were set up and run through CiLib.

2.2. Problem sets

In this research study two problem sets consisting of five problems each were used in the simulations. The first set contained five classification problems and the second set contained five function approximation problems.

2.2.1. Classification problem set

Classification problems are ones where a neural network is trained to identify to what category a particular vector of inputs belongs. An example would be to classify people buying credit cards as being at high, medium, or low risk of defaulting. The input vectors may include information such as the person’s spending habits, their credit rating, their bank balances, their previous education, where they live, what they do for a living, etc. The specific problems analyzed in this report come from the health industry and are:

1. Andersons Iris data set\(^2\) - a multivariate data set containing data about three different species of the Iris flower. This data set contains information about each flower’s sepal length and width and each flowers petal width and length.

\(^2\) For more information about this data set please visit [http://en.wikipedia.org/wiki/Iris_flower_data_set](http://en.wikipedia.org/wiki/Iris_flower_data_set)
2. Breast cancer data set - Fine-needle aspiration biopsy is a diagnostic procedure used to investigate lumps or masses just under the skin. Digital images produced during this procedure were analyzed to produce this dataset where the target values are whether the patient was either benign or malignant.

3. Breast tissue - electrical impedance spectroscopy is a minimally invasive technique for living tissue characterisation. This technique is used on breast tissue for breast cancer detection. The outputs of this dataset is the tissue type.

4. Fertility data set - contains semen concentration values for one hundred males and socio-demographic data, environmental factors, health status, and life habits recorded as inputs. It was found that these indicators are related to semen concentration and fertility.

5. Parkinsons data set\(^3\) - a range of biomedical voice measurements from 31 people, 23 with Parkinson's disease. Parkinsons can be detected through subtle changes to the voice, including tremor, breathiness and weakness.

2.2.2. Function approximation problem set

Neural networks can also be used to approximate functions which map inputs to outputs. In this research study a number of difficult optimization problems were selected and used to generate data-sets of inputs to the function output. White noise was added to 5% of the samples and datasets were generated for dimensions five and ten. The following optimization functions were selected for this research study:

1. Michalewicz function
2. Rosenbrock function
3. Schubert function
4. Schwefels function
5. Sphere function

For information on each one of these functions including function definitions, formulae and multi-dimensional graphs, please take a look at the optima page:
http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page364.htm

\(^3\) For more information on this ongoing initiative please visit the homepage of the parkinsons voice initiative:
http://www.parkinsonsvoice.org/index.php
2.3. Algorithm parameters

2.3.1. Particle Swarm Optimization

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swarm size</td>
<td>35</td>
</tr>
<tr>
<td>Inertia weight</td>
<td>0.729844</td>
</tr>
<tr>
<td>Social component weight</td>
<td>1.496180</td>
</tr>
<tr>
<td>Cognitive component weight</td>
<td>1.496180</td>
</tr>
<tr>
<td>Neighbourhood size</td>
<td>Local best of 5</td>
</tr>
<tr>
<td>Lambda (penalty coefficient)</td>
<td>0.6</td>
</tr>
<tr>
<td>Epsilon (threshold for non-zero weight)</td>
<td>0.05</td>
</tr>
<tr>
<td>Epochs</td>
<td>1000</td>
</tr>
<tr>
<td>Samples</td>
<td>30</td>
</tr>
</tbody>
</table>

2.3.2. Vector Evaluated Particle Swarm Optimization

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Swarm size</td>
<td>35</td>
</tr>
<tr>
<td>Inertia weight</td>
<td>0.729844</td>
</tr>
<tr>
<td>Social component weight</td>
<td>1.496180</td>
</tr>
<tr>
<td>Cognitive component weight</td>
<td>1.496180</td>
</tr>
<tr>
<td>Neighbourhood size</td>
<td>Local best of 5</td>
</tr>
<tr>
<td>Archive size</td>
<td>5000</td>
</tr>
<tr>
<td>Train : validate ratio</td>
<td>70 : 30</td>
</tr>
<tr>
<td>Epochs</td>
<td>1000</td>
</tr>
<tr>
<td>Samples</td>
<td>30</td>
</tr>
</tbody>
</table>
3. Results

This section contains the results for each classification and function approximation problem used in the study. These results are averaged over 30 samples. For each problem, three graphs are presented from the following page onwards. One shows the standard-deviations in fitness across those 30 samples, the second shows the average fitness over time (epochs), and the third one shows the size of the neural network architecture expressed as a percentage of non-zero weights. i.e. 0.72 means that 73% of the neural network’s weights are non-zero.
3.1. Classification results

3.1.1. Andersons Iris data set

![Graph 1: Iris Data Set Standard Deviations](image)
Comparison of standard deviations for the PSO and VPESO algorithms on the Iris classification data set

![Graph 2: Iris Data Set Training Errors](image)
Comparison of training errors for the PSO and VPESO algorithms on the Iris classification problem

![Graph 3: Iris Data Set Neural Network Architecture Size](image)
Comparison of the neural network sizes as optimized by the PSO and VPESO algorithms on the Iris classification data set
3.1.2. Breast cancer data set
3.1.3. Breast tissue data set
3.1.4. Fertility data set

**Fertility Data Set Standard Deviations**
Comparison of standard deviations for the PSO and VEPSO algorithms on the Fertility classification data set.

**Fertility Data Set Training Errors**
Comparison of training errors for the PSO and VEPSO algorithms on the fertility classification problem.

**Fertility Data Set Neural Network Architecture Size**
Comparison of the neural network sizes as optimized by the PSO and VEPSO algorithms on the Fertility classification data set.
3.1.5. Parkinsons data set

**Parkinsons Data Set Standard Deviations**
Comparison of standard deviations for the PSO and VEPSO algorithms on the Parkinsons classification data set.

**Parkinsons Data Set Training Errors**
Comparison of training errors for the PSO and VEPSO algorithms on the Parkinsons classification problem.

**Parkinsons Data Set Neural Network Architecture Size**
Comparison of the neural network sizes as optimized by the PSO and VEPSO algorithms on the Parkinsons classification data set.
3.2. Function approximation results

3.2.1. Michalewicz function

![Graph of Michalewicz Data Set Standard Deviations]

Comparison of standard deviations for the PSO and VEPSO algorithms on the Michalewicz function approximation data set.

![Graph of Michalewicz Data Set Training Errors]

Comparison of training errors for the PSO and VEPSO algorithms on the Michalewicz function optimization problem.

![Graph of Michalewicz Data Set Neural Network Architecture Size]

Comparison of the neural network sizes as optimized by the PSO and VEPSO algorithms on the Michalewicz function approximation problem.
3.2.2. Rosenbrock function

Comparison of standard deviations for the PSO and VEPSO algorithms on the Rosenbrock function optimization data set.

Comparison of training errors for the PSO and VEPSO algorithms on the Rosenbrock function optimization problem.

Comparison of the neural network sizes as optimized by the PSO and VEPSO algorithms on the Rosenbrock function optimization data set.
3.2.3. Schubert function

**Schubert Data Set Standard Deviations**
Comparison of standard deviations for the PSO and VEPSO algorithms on the Schubert function approximation problem

**Schubert Data Set Training Errors**
Comparison of training errors for the PSO and VEPSO algorithms on the Schubert function approximation problem

**Schubert Data Set Neural Network Architecture Size**
Comparison of the neural network sizes as optimized by the PSO and VEPSO algorithms on the Schubert function approximation data set
3.2.4. Schwefel's function

Schwefel Data Set Standard Deviations
Comparison of standard deviations for the PSO and VEPSO algorithms on the Schwefel function approximation data set

Schwefel Data Set Training Errors
Comparison of training errors for the PSO and VEPSO algorithms on the Schwefel function optimization problem

Schwefel Data Set Neural Network Architecture Size
Comparison of the neural network sizes as optimized by the PSO and VEPSO algorithms on the Schwefel function optimization problem
3.2.5. Sphere function

![Sphere Data Set Standard Deviations](chart1.png)

Comparison of standard deviations for the PSO and VEPSO algorithms on the Sphere function approximation data set

![Breast Tissue Data Set Training Errors](chart2.png)

Comparison of training errors for the PSO and VEPSO algorithms on the Breast tissue classification problem

![Breast Tissue Data Set Neural Network Architecture Size](chart3.png)

Comparison of the neural network sizes as optimized by the PSO and VEPSO algorithms on the Breast Tissues classification data set
3. Results continued

Because the results above looked quite promising for the classification problems, the following pareto front graphs were generated showing the balance between optimized neural network architectures and performance.

3.3. Pareto fronts

3.3.1. Anderson’s Iris data set

![Diagram of Anderson’s Iris data set pareto front graph]

3.3.2. Breast cancer data set

![Diagram of Breast cancer data set pareto front graph]
3.3.3. Breast tissue data set

Pareto front of last 200 candidate solutions

This graph shows the pareto front of optimal candidate solutions between architecture size and fitness.

3.3.4. Fertility data set

Pareto front of last 200 candidate solutions

This graph shows the pareto front of optimal candidate solutions between architecture size and fitness.
3.3.5. Parkinsons data set

3.4. Key observations in results

The following trends were observed in the graphs presented in the previous pages, these trends will be considered when presenting conclusions and recommendations for further research.

1. Neither of the two techniques could approximate the optimization problems well for five dimensions, and the results were worse at 10 dimensions.4
2. Regularization better at estimated the pareto front as VEPSO tended to plateau.
3. Standard deviations of regularization were higher than the VEPSO approach.
4. Standard deviations tended to increase when using the regularization approach.
5. The percentage of non-zero weights decreased better over time using the regularization approach than with using the VEPSO approach.
6. The training error of the neural networks were lower on average when using the VEPSO approach but were more likely to plateau.

4 The results were omitted from this report
4. Conclusions

On average the standard deviations in fitness of solutions found using the VEPSO approach was lower than those found using regularization. This side-effect could be the result of an ineffective knowledge transfer strategy (KTS) or not enough diversity in the swarms. This finding is consistent with some of the body of knowledge surrounding the performance of the VEPSO algorithm in general. That having been said, the fitness of solutions found using VEPSO approach were better on average than those found with the regularization approach. Despite challenged with the VEPSO algorithm, the performance of the regularization approach on the classification problems was impressive. These points are discussed in more detail below.

4.1. Diversity control in VEPSO

We refer to the findings in the paper by K.S. Lim, S. Buyamin, and Z. Ibrahim titled ‘Convergence and diversity measurement for Vector Evaluated Particle Swarm Optimization based on the ZDT Test Problems’. In this report these researchers assert that VEPSO is weak in solving non-convex, non-uniformity search space and low solution density near Pareto optimal front problems. They state that VEPSO has poor diversity due to no diversity control mechanism. The results in this research study show prolonged plateaus in performance indicators which could be indicative of a lack of diversity in the sub-swarms and or premature convergence to suboptimal solutions.

4.2. Knowledge transfer strategies in VEPSO

Knowledge transfer happens between the two sub-swarms because through selecting the global guide for a sub-swarm as the global best particle from the neighboring sub-swarm. This tactic may be the cause of poor results in the VEPSO as compared with a standard PSO using regularization to cater for the secondary objective.

4.3. Fitness of solutions found using VEPSO

In all likelihood the solutions found using the VEPSO approach are not truly better than those found using regularization. The reason for this statement is that by adding a penalty function onto solutions found using regularization, the results are distorted in favour of the VEPSO algorithm. In future studies, the actual fitness of the candidate solution should be calculated for reporting, and the regularized fitness used for updating positions internally to the PSO.
4.4. Performance of regularization

Despite poor performance by the VEPSO algorithm, the performance of the standard PSO algorithm on the classification problems was impressive. Below is a table showing the percentage reduction in non-zero connections in the neural network architecture:

<table>
<thead>
<tr>
<th>Classification problem</th>
<th>Reduction %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Breast cancer</td>
<td>~6%</td>
</tr>
<tr>
<td>Breast tissue</td>
<td>~25%</td>
</tr>
<tr>
<td>Iris flowers</td>
<td>~45%</td>
</tr>
<tr>
<td>Fertility</td>
<td>~30%</td>
</tr>
<tr>
<td>Parkinsons</td>
<td>~10%</td>
</tr>
</tbody>
</table>

These results are promising because they will result in improved performance of the neural network without sacrificing the quality and accuracy of the neural network’s classifications.

4.5. Optimization problems

As stated in the results sections, neither strategies were able to approximate the optimization problems given to them. In retrospect, this was perhaps an ambitious goal as optimization problems are specifically designed to be complex functions. Nevertheless, some performance was shown by the regularization technique in approximating the Schubert function.

5. Research recommendations

1. **Augment the traditional VEPSO with diversity controls on each of the sub-swarms by using either a charged particle swarm optimization (CPSO) or a quantum particle swarm optimization (QPSO) instead of a standard PSO.**

2. **Compare the performance of regularization against other algorithms capable of solving multi-objective optimization problems such as the MOPSO algorithm.**

3. **Assess the impact of alternative knowledge transfer strategies in the VEPSO algorithm.**