Using neural network to approximate macroeconomic forecast models for BRICS nations

Three strategies are used to train neural network economic forecast models on temporal data: a standard feedforward neural network (FFNN), two simple recurrent neural networks (SRNN’s), and a population of FFNN’s evolved using standard and dynamic particle swarm optimization (PSO) algorithms. Each forecast model is compared and used to draw conclusions on the effectiveness of each strategy.

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

Economic forecasting is the process of making predictions about the future state of an economy. Such predictions are significant from a societal, governmental, and financial perspective. Societally speaking, identification of economic bubbles (such as the credit bubble which caused the financial crisis of 2008) is a principal concern. When economic bubbles implode they adversely impact society through increased inequality, lower standards of living, increased unemployment levels, and other far-reaching effects. Economic forecasting is also one of the responsibilities of government. Governments actively influence economies through interest rates, deficits, current accounts, and regulatory policies. Poor governance, specifically around monetary policy, is one of the root causes of the hyperinflation which plagued Zimbabwe. As a result, governments must maintain strong economic models in order to predict potential impacts of their actions. Lastly, business owners and investors looking for hone competitive advantages in the global economy will maintain economic models to monitor and predict emergent economies where growth will, hopefully, outstrip inflation. Because of economic forecastings significant presence on the global stage, accurate quantitative models are highly sought after.

Unfortunately, traditional techniques for deriving economic forecast models are limited by their linearity and poor scalability. Neural networks, computational models which approximate non-linear regression functions between inputs and outputs, are being used to address the shortcomings of traditional, autoregressive, techniques. That having been said neural networks are not without their shortcomings. An example is that models approximated by neural networks are essentially black boxes which need to retrained when the environment changes. In a financial setting, the environment is inherently dynamic, as such, static neural networks often underwhelm. More advanced techniques can be used to train neural networks to perform better in dynamic environments. One commonly used approach is the use of recurrent neural networks. Another, less widely known, approach involves using a particle swarm optimization algorithm adapted to work in dynamic environments to train swarms of neural networks. In this research study the performance of static, recurrent, and pso-trained algorithms used to predict the economic swings of BRICS nations are compared.
Table of Contents

Introduction and problem statement
Table of Contents
Background information
    Neural Networks (NN’s)
        Feedforward neural networks (FFNN’s)
        Simple recurrent neural networks (SRNN’s)
            Elman SRNN
            Jordan SRNN
    Particle swarm optimization algorithms (PSO’s)
        Standard Particle Swarm Optimization
        Charged Particle Swarm Optimization (CPSO)
        Using PSO’s to train FFNN’s
    Economic Forecasting
        Socioeconomics
    Related Literature
Design and implementation
    Class diagram view of the implementation
Data processing
    Brazil
    Russia
    India
    China
    South Africa
    Feedforward neural network
    Elman recurrent neural network
    Jordan recurrent neural network
    Standard particle swarm optimization
    Charged particle swarm optimization
Experiment design
Research results
    Sum Squared Error (SSE) and Total Prediction Error (PE) results
        Table of graphs
        Brazil Results
Conclusions
    Conclusions pertaining to the algorithms
    Conclusions pertaining to the problem domain
Recommendations for further research
    Academic Resources used
    Web Resources used
Appendix A - Rest of Results (India, Russia, China, and SA)

Russia Results
India Results
China Results
South Africa Results
Background information

This section aims to introduce the reader to the paradigms and concepts used in this report. It begins with an introduction to neural networks followed by an introduction to macroeconomic modelling. Following this, a list of related literature is supplied.

Neural Networks (NN’s)

The unique abilities of the human brain have and continue to prompt much research into its internal functions and mechanisms. It has long been a goal of Artificial Intelligence to replicate the brain. The ability of computer scientists to model the human brain in its entirety may be a long way off, but great successes have nevertheless been achieved through simple constructions that closer resemble the brain of a 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:\footnote{1: Partial list taken from ‘Computational Intelligence - An introduction’ by Andries Engelbrecht}

- 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

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 topography or architecture of the neural network. In short the following topographies exist: single layer, multilayer feedforward, temporal, and self organized. The specific topographies used in this report will now we explained in more detail.
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.

Simple recurrent neural networks (SRNN’s)

SRNN’s have structures similar to FFNN’s, in that, there are still typically input, hidden, and output layers. They differ from FFNN’s in that an additional context later is presented. This allows recurrent neural networks to better approximate those mathematical functions representing time series where outputs from the network form inputs. In this research study the Elman and Jordan SRNN’s have been used.

Elman SRNN

In the elman network a context layer is added which feeds input to and received input from the hidden layers in a traditional feedforward neural network. The Elman network was first developed by Jefferey L. Elman in 1990. The architecture is illustrated below for clarity.

---

2 Image of feedforward neural network is courtesy of heaton research and can be found here: [http://www.heatonresearch.com/w/images/a/a4/Method-ff-1.png](http://www.heatonresearch.com/w/images/a/a4/Method-ff-1.png)
Jordan SRNN

The Jordan SRNN is similar to the Elman SRNN in that it consists of a context layer. The principal difference between the two is that the context layer in the Jordan network sends input to and received inputs from the output layer (as opposed to the hidden layer). This architecture is illustrated below for clarity.

---

3 This image of the Elman simple recurrent neural network is courtesy of heaton research and can be found here: [http://www.heatonresearch.com/w/images/8/8c/Method-elman-1.png](http://www.heatonresearch.com/w/images/8/8c/Method-elman-1.png)
Particle swarm optimization algorithms (PSO’s)

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.

Standard Particle Swarm Optimization

In the standard PSO algorithm a population, also referred to as a swarm, is initialized wherein each particle represents a candidate solution to some 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. In order to achieve this, an objective function (the function representing the quantity being maximized or minimized) needs to be defined. The choice of objective function has a huge impact on the effectiveness of the PSO algorithm. The below image illustrates the convergence of particles in a PSO in a two dimensional search space towards at least a local minimum (note, this may not be the global minimum).

4 This image can be found at https://www.groksolutions.com/static/images/technology/pso.jpg
Charged Particle Swarm Optimization (CPSO)

One problem facing PSO algorithms is a lack of diversity in the swarm as the particles converge onto a local minimum. As such a number of different adaptations to the basic PSO algorithm have been designed and implemented to combat premature convergence. This is especially important when searching for optima within a dynamic search space. Dynamic search spaces are ones which the search space is continuously changing. Temporal data can be dynamic if, over time, the inputs that define the landscape change.

The CPSO algorithm uses repulsion, wherein particles repel one another when they begin to converge, to introduce swarm diversity and prevent premature convergence. This algorithm was inspired by the electrostatic energies contained within charged particles. In the CPSO the traditional attractive force from each particle to the center of mass of the swarm is counteracted by interparticle repulsive forces. As such, the closer particles get to one another, the greater the repulsive forces which will result in divergence between those particles. In this research study all of the particles in the swarm were charged.

Using PSO’s to train FFNN’s

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. That having been said, this approach assumes that the search space is static. In an attempt to avoid premature convergence onto a sub-optimal set of weights for the neural network; dynamic PSO’s are used in favour of traditional PSO’s. This research study looks at the suitability of the CPSO algorithm and a traditional PSO algorithm against traditional machine learning training techniques for neural networks.
Economic Forecasting

Economic models are theoretical constructs that make use to variables and relationships between those variables to represent complex economic processes. These models are used by economists to make sense of complex macroeconomic and microeconomic trends as well as to select relevant data. Macroeconomics is the study of economies as a whole and deals with concepts such as unemployment, gross domestic product, inflation, and price levels. Microeconomics is the study the market behaviour of individual consumers and firms in an attempt to understand the decision making processes of firms and households. Such models are applied in numerous ways including, but not limited to: economic forecasting; proposing economic policies; strategy formulation; economic planning and allocation; investment management; and risk management.

There are three broad categories of economic models: stochastic models, mathematical models, and qualitative models. Models are classified according to whether they are discrete or continuous; their purpose or intent; their scope; whether they use general, partial, or non equilibria; and whether they are qualitative or quantitative. Stochastic models are built using stochastic processes that make use of statistical tools such as regressions. Autoregressive conditional heteroskedasticity (ARCH)\textsuperscript{5}, models are some of the most popular within this category. Since neural networks approximate linear and nonlinear regressions lines where calculation is impossible, economic models that use them can be classified as stochastic economic models.

New agent-based computational economics (ACE) models have demonstrated an ability to overcome assumptions of classical models. This quote captures the essence of this nicely.

\begin{quote}
"We think one of the most promising directions is to view financial markets from a biological perspective and, specifically, within an evolutionary framework in which markets, instruments, institutions, and investors interact and evolve dynamically according to the "law" of economic selection. Under this view, financial agents compete and adapt, but they do not necessarily do so in an optimal fashion." - Farmer and Lo, Frontiers in Science symposium, 1999
\end{quote}

\textsuperscript{5} For a nice overview of the stochastic processes found in ARCH models refer to ‘Introduction to ARCH and GARCH models’ by TA Roberto Perelli available at: http://www.econ.uiuc.edu/~econ472/ARCH.pdf
Socioeconomics

Socioeconomics is the social science that studies how economic activity affects social processes and vice versa. It concerns itself with societal progression, stagnation, and regression. In this field a number of indicators are used to analyze the current status and project the future status of an economy with respect to social development. The largest source of socioeconomic data is the world bank which provides a full api to all of its databases for free on its website, http://data.worldbank.org. Some popular indicators are listed below. The ones in bold will are included in this study:

<table>
<thead>
<tr>
<th>Social development</th>
<th>Economic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fertility rates</td>
<td>Agriculture value added</td>
</tr>
<tr>
<td>Children in employment</td>
<td><strong>Cash surplus / deficit</strong></td>
</tr>
<tr>
<td><strong>Labour participation rates</strong></td>
<td>Central government debt levels</td>
</tr>
<tr>
<td>Life expectancy</td>
<td>**Gross domestic product (GDP)**⁶</td>
</tr>
<tr>
<td>HIV statistics</td>
<td>Export % of goods and services</td>
</tr>
<tr>
<td>Male to female employment</td>
<td>Import % of goods and services</td>
</tr>
<tr>
<td><strong>Unemployment</strong></td>
<td>Gross national income (GNI)</td>
</tr>
<tr>
<td>Vulnerable employment</td>
<td><strong>Gross savings</strong></td>
</tr>
<tr>
<td>Secondary school enrolment</td>
<td>Inflation - value added, prices, etc.</td>
</tr>
<tr>
<td>Refugee populations</td>
<td><strong>Foreign direct investments</strong></td>
</tr>
<tr>
<td>Population brackets</td>
<td>Gross capital formation</td>
</tr>
<tr>
<td>Birth rates</td>
<td>Current account balances</td>
</tr>
<tr>
<td>Others …</td>
<td>Other …</td>
</tr>
</tbody>
</table>

These indicators are available for a hundreds of countries internationally within which the world bank and its affiliates have a presence. The following two paragraphs were taken from the world bank's website to explain the sources of their information in each one of the two datasets this report concerns itself with:

⁶ GDP will be the output, but also an input as this is temporal
Social development dataset

Data here cover child labor, gender issues, refugees, and asylum seekers. Children in many countries work long hours, often combining studying with work for pay. The data on their paid work are from household surveys conducted by the International Labour Organization (ILO), the United Nations Children’s Fund (UNICEF), the World Bank, and national statistical offices. Gender disparities are measured using a compilation of data on key topics such as education, health, labor force participation, and political participation. Data on refugees are from the United Nations High Commissioner for Refugees complemented by statistics on Palestinian refugees under the mandate of the United Nations Relief and Works Agency.

Economic policy and external debt dataset

Economic indicators measure outcomes in the structure and rates of change of output, trade, and aggregate demand, and in macroeconomic performance. The data here consist of national accounts, government finances, money supply, prices, balance of payments, and external debt. Data are gathered from national statistical organizations and central banks by World Bank missions and from the International Monetary Fund's data files.

In an attempt to reduce the scope of this research report and accuracy of the model, a small group of nations have been selected to comprise the data with which the neural networks will develop their predictive models. To remain locally relevant, the countries selected include the so-called, BRICS, nations: Brazil, Russia, India, China, and South Africa. These countries are still classed as developing nations or emerging economies. Some economists project that these nations will someday control more of the global economy than the G7, a group of seven highly industrialized nations namely the United States, United Kingdom, France, Germany, Italy, Canada and Japan. In addition to having great economic potential, what makes the BRICS nations similar to one another is that they all suffer from high amounts of social strife and all need to overcome great social development hurdles in order to realize their economic potential. China has already realized a lot of its potential but still suffers from great social inequality and has only just begun moving toward being a more consumer driven economy.
BRICS nations shown here in Green
Related Literature

It is beyond the scope of this research study to provide a full literature review of relevant works from related bodies of knowledge. That having been said, the following papers were used in the design and implementation of this research study and may be of interest to the reader:

Design and implementation

The design and implementation of the time series modelling framework was done in the Java programming language with the use of the Encog framework which provided the neural network functionality. In total the following five cases were implemented:

1. A standard feedforward neural network (FFNN)
2. An Elman recurrent neural network
3. A Jordan recurrent neural network
4. A standard PSO used to train standard FFNN’s and
5. A charged PSO (CPSO) used to train standard FFNN’s

These five algorithms were used to train neural networks to forecast two economic time-steps ahead of the current time measured in 6 month intervals. Two objective functions were used to compare the performances of the resulting neural networks:

1. The Sum Squared Error (SSE) of the Neural Network and
2. The total of prediction error across all time steps - this was the sum of the error bars between the forecasts made by the neural network and the ideal value

Additionally the predictions made after 5000 epochs were recorded and visualized. The rest of this section will go into more detail surrounding the implementation and the design decisions made during the research study.

Class diagram view of the implementation

The classes used to realize the implementation of this research study were allocated into one of four packages: main, data, recurrent, and pso. Each package had a separate set of concerns and all were associated in some way or another with the Encog library. The implementation as it stands is illustrated below:
Additionally, the approach taken was to implement the framework using a data-driven approach. That is to say the inputs into the framework were used to define the data-structures, reports, and neural networks. As such, the implementation could easily be used for problems outside of the economic forecasting domain.

**Data processing**

Data flowing into the framework was preprocessed using the Encog analyst tool. In this preprocessing step the data was normalized and segregated between training and test data. In total, 22 inputs flowed into each neural network implementation: 10 economic indicators from the the previous 2 years, plus the Gross Domestic Product, GDP (output), from the previous two years. During the normalization process the following useful statistics were generated:
### Brazil

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cash surplus</td>
<td>1.50796</td>
<td>-6.93011</td>
<td>-2.29408</td>
<td>1.3926</td>
</tr>
<tr>
<td>Consumer Price Inflation (CPI)</td>
<td>2949.68507</td>
<td>3.19859</td>
<td>361.68442</td>
<td>727.68936</td>
</tr>
<tr>
<td>Foreign Direct Investments</td>
<td>5.0844</td>
<td>0.21409</td>
<td>2.23964</td>
<td>1.41268</td>
</tr>
<tr>
<td>Gross Savings</td>
<td>21.63288</td>
<td>12.45576</td>
<td>17.02641</td>
<td>2.50256</td>
</tr>
<tr>
<td>Labour Participation</td>
<td>70.1</td>
<td>64.6</td>
<td>68.67326</td>
<td>1.15222</td>
</tr>
<tr>
<td>Secondary School Enrol</td>
<td>81.99038</td>
<td>65.78239</td>
<td>73.65304</td>
<td>5.07489</td>
</tr>
<tr>
<td>Unemployment</td>
<td>9.7</td>
<td>3.7</td>
<td>7.82209</td>
<td>1.30113</td>
</tr>
<tr>
<td>Vulnerable Employment</td>
<td>35.8</td>
<td>25.1</td>
<td>29.96628</td>
<td>3.6764</td>
</tr>
<tr>
<td>GDP Growth</td>
<td>7.53362</td>
<td>-4.3</td>
<td>2.82889</td>
<td>2.26924</td>
</tr>
</tbody>
</table>

### Russia

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cash surplus</td>
<td>9.88307</td>
<td>-4.20927</td>
<td>4.23467</td>
<td>2.62412</td>
</tr>
<tr>
<td>Consumer Price Inflation (CPI)</td>
<td>874.62185</td>
<td>6.85802</td>
<td>191.33698</td>
<td>318.59747</td>
</tr>
<tr>
<td>Exports of Goods and Services</td>
<td>62.32</td>
<td>13.27</td>
<td>33.3557</td>
<td>8.26399</td>
</tr>
<tr>
<td>Imports of Goods and Services</td>
<td>32.4</td>
<td>-46.41735</td>
<td>2.97617</td>
<td>22.17445</td>
</tr>
<tr>
<td>Foreign Direct Investments</td>
<td>4.51592</td>
<td>0.17465</td>
<td>1.63322</td>
<td>1.26904</td>
</tr>
<tr>
<td>Gross Savings</td>
<td>37.11668</td>
<td>17.93947</td>
<td>29.33912</td>
<td>3.63405</td>
</tr>
<tr>
<td>Labour Participation</td>
<td>67.7</td>
<td>57.1</td>
<td>61.58256</td>
<td>2.42129</td>
</tr>
<tr>
<td>Secondary School Enrol</td>
<td>96.3822</td>
<td>83.13237</td>
<td>89.80445</td>
<td>3.37416</td>
</tr>
<tr>
<td>Unemployment</td>
<td>13.3</td>
<td>5.2</td>
<td>8.21628</td>
<td>2.27171</td>
</tr>
<tr>
<td>Vulnerable Employment</td>
<td>8.2</td>
<td>0.8</td>
<td>4.36628</td>
<td>2.30411</td>
</tr>
<tr>
<td>GDP Growth</td>
<td>10</td>
<td>-14.53107</td>
<td>0.66099</td>
<td>6.90071</td>
</tr>
</tbody>
</table>

### India

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cash surplus</td>
<td>0</td>
<td>-5.4187</td>
<td>-3.17808</td>
<td>0.99318</td>
</tr>
<tr>
<td>Consumer Price Inflation (CPI)</td>
<td>13.87025</td>
<td>3.68481</td>
<td>7.79132</td>
<td>3.0696</td>
</tr>
<tr>
<td>Exports of Goods and Services</td>
<td>31.39607</td>
<td>-4.68544</td>
<td>13.52195</td>
<td>7.7235</td>
</tr>
<tr>
<td>Imports of Goods and Services</td>
<td>32.58765</td>
<td>-2.43999</td>
<td>14.16929</td>
<td>8.70866</td>
</tr>
<tr>
<td>Foreign Direct Investments</td>
<td>3.54598</td>
<td>0.02676</td>
<td>1.03398</td>
<td>0.85718</td>
</tr>
<tr>
<td>Gross Savings</td>
<td>36.75828</td>
<td>22.13093</td>
<td>28.37891</td>
<td>4.70316</td>
</tr>
<tr>
<td>Labour Participation</td>
<td>61</td>
<td>55.6</td>
<td>59.57907</td>
<td>1.59192</td>
</tr>
<tr>
<td>Secondary School Enrol</td>
<td>63.2145</td>
<td>43.44509</td>
<td>49.91055</td>
<td>6.42157</td>
</tr>
<tr>
<td>Unemployment</td>
<td>4.4</td>
<td>2.1</td>
<td>3.70233</td>
<td>0.64654</td>
</tr>
<tr>
<td>Vulnerable Employment</td>
<td>83.4</td>
<td>80.8</td>
<td>82.7686</td>
<td>0.89553</td>
</tr>
<tr>
<td>GDP Growth</td>
<td>10.54638</td>
<td>1.05683</td>
<td>6.53745</td>
<td>2.11619</td>
</tr>
</tbody>
</table>
### China

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cash surplus</td>
<td>24.70353</td>
<td>-0.60113</td>
<td>6.35779</td>
<td>6.90451</td>
</tr>
<tr>
<td>Consumer Price Inflation (CPI)</td>
<td>24.23709</td>
<td>-1.40789</td>
<td>4.77186</td>
<td>6.07894</td>
</tr>
<tr>
<td>Exports of Goods and Services</td>
<td>32.04121</td>
<td>-10.3248</td>
<td>16.68825</td>
<td>8.84712</td>
</tr>
<tr>
<td>Imports of Goods and Services</td>
<td>33.5502</td>
<td>-20.31167</td>
<td>15.35237</td>
<td>10.0935</td>
</tr>
<tr>
<td>Foreign Direct Investments</td>
<td>6.2463</td>
<td>0.97692</td>
<td>3.82145</td>
<td>1.18777</td>
</tr>
<tr>
<td>Gross Savings</td>
<td>53.35564</td>
<td>37.28703</td>
<td>44.53157</td>
<td>5.41455</td>
</tr>
<tr>
<td>Labour Participation</td>
<td>79.1</td>
<td>74.1</td>
<td>76.73954</td>
<td>1.71242</td>
</tr>
<tr>
<td>Unemployment</td>
<td>4.3</td>
<td>2.3</td>
<td>3.43023</td>
<td>0.66831</td>
</tr>
<tr>
<td>Vulnerable Employment</td>
<td>7.8</td>
<td>5</td>
<td>6.36744</td>
<td>1.01345</td>
</tr>
<tr>
<td>GDP Growth</td>
<td>14.2</td>
<td>3.8</td>
<td>10.19651</td>
<td>2.22157</td>
</tr>
</tbody>
</table>

### South Africa

<table>
<thead>
<tr>
<th>Indicator</th>
<th>Max</th>
<th>Min</th>
<th>Mean</th>
<th>Stdev</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cash surplus</td>
<td>1.07624</td>
<td>-5.12346</td>
<td>-1.76851</td>
<td>1.33513</td>
</tr>
<tr>
<td>Consumer Price Inflation (CPI)</td>
<td>15.33477</td>
<td>1.38538</td>
<td>7.66409</td>
<td>3.3518</td>
</tr>
<tr>
<td>Exports of Goods and Services</td>
<td>10.93789</td>
<td>-19.53168</td>
<td>3.38872</td>
<td>5.37507</td>
</tr>
<tr>
<td>Imports of Goods and Services</td>
<td>18.26108</td>
<td>-17.39122</td>
<td>5.78491</td>
<td>7.55094</td>
</tr>
<tr>
<td>Foreign Direct Investments</td>
<td>6.1364</td>
<td>-0.07035</td>
<td>1.22563</td>
<td>1.27167</td>
</tr>
<tr>
<td>Labour Participation</td>
<td>55.3</td>
<td>48.4</td>
<td>51.85698</td>
<td>1.74964</td>
</tr>
<tr>
<td>Secondary School Enrol</td>
<td>95.69964</td>
<td>66.07226</td>
<td>86.21275</td>
<td>8.02938</td>
</tr>
<tr>
<td>Unemployment</td>
<td>27.2</td>
<td>16.9</td>
<td>23.05233</td>
<td>2.64972</td>
</tr>
<tr>
<td>Vulnerable Employment</td>
<td>15.8</td>
<td>10</td>
<td>14.67674</td>
<td>2.08216</td>
</tr>
<tr>
<td>GDP Growth</td>
<td>5.60372</td>
<td>-2.13704</td>
<td>2.61786</td>
<td>2.06494</td>
</tr>
</tbody>
</table>
Feedforward neural network

The feedforward neural networks used in this research study were constructed within the Encog framework with the following properties:

★ Input neurons → 23 input neurons consisting of: 10 indicators for time period t and t-1, target outputs (GDP growth) from time periods t-1, and t-2, and one bias unit.
★ Hidden neurons → 31 hidden neurons consisting of: 30 standard hidden neurons using the Sigmoid activation function, and 1 bias unit
★ Output neurons → 2 output neurons for time t, and t+1.
★ Learning rule → back propagation learning

When encoded as a vector the feedforward neural networks consisted of arrays 752 elements. This encoded format was used by the PSO and Charged PSO.

Elman recurrent neural network

The Elman recurrent neural networks used in this research study were constructed within the Encog framework. Each had the same number of input, hidden, and output neurons as the Feedforward neural network but also had the following property:

★ Context neurons → 30 context neurons which are connect to and from the 30 hidden neurons in the network structure.

When encoded as a vector the feedforward neural networks consisted of arrays 812 elements, 752 from the standard architecture and an additional 60 for the context layer.

Jordan recurrent neural network

The Jordan recurrent neural networks used in this research study were constructed within the Encog framework. Each had the same number of input, hidden, and output neurons as the Feedforward neural network but also had the following property:

★ Context neurons → 2 context neurons which are connect to and from the 2 output neurons in the network structure.

When encoded as a vector the feedforward neural networks consisted of arrays 756 elements, 752 from the standard architecture and an additional 4 for the context layer.
Standard particle swarm optimization

It was noted that the number of particles in the swarm had a relatively large impact on computational complexity on the algorithm, with little impact on the effectiveness of the algorithm. As such, the particle swarms used in this research study consisted of only three particles, representing neural networks encoded as arrays of doubles. The following parameters and design decisions were made for the PSO:

- Population size → three feedforward neural networks (FFNN)
- Particle representation → weights of FFNN’s encoded as array of doubles
- Fitness function → Sum Squared Error (SSE) of FFNN
- Social component → 0.75
- Cognitive component → 0.25

Velocities of the particles were calculated using the standard velocity update equation described in a standard PSO algorithm. However, the position update equation was adapted in such a way as to make it more compatible with the encog framework and more sensitive to optima. The velocity was applied fractionally element per element in the encoded vector. This results in the particles converging toward the global best particle.

Particle position update equation

\[
p_{t+1}[x] = p_t[x] + s \cdot v
\]

In the above equation there is a parameter called \( s \). This is a scaling parameter used to slow down the converged rate when it is between 0 and 1. During the simulation it was set to 0.5.
Charged particle swarm optimization

The charged PSO algorithm is similar to the standard PSO algorithm with one notable exception: an additional repulsive force is introduced into the model. This force exists between the particles in the swarm. As the particles get closer to one another the repulsive force strengthens and the particles diverge from one another. This results in increased swarm diversity as the particles do not converge onto a single solution. The charged PSO algorithm has been successfully used in dynamic environments. The following parameters were set for the charged PSO:

- Population size → three feedforward neural networks (FFNN)
- Particle representation → weights of FFNN’s encoded as array of doubles
- Fitness function → Sum Squared Error (SSE) of FFNN
- Social component → 0.75
- Cognitive component → 0.25
- Similarity matrix → see below for explanation
- Perception limit → not applied, the entire population (3 particles) was perceptible
- Charge magnitude → determined using similarity matrix (this was not static)
- Core radius → similarity matrix >= 99.8%

One challenge in using the charged PSO is calculating the similarity between two or more particles (their proximity to one another). In this implementation an elegant solution was found to easily calculate the similarity of two particles in a matrix. This is illustrated below.

**Particle similarity calculation**

\[
similarity(X,Y) = \text{Average}(\text{Similarity of each element to one another})
\]

\[
similarity(X[i],Y[i]) = X[i] / (X[i] + Y[i]) \quad \text{--> when} \ 0.5, \text{perfectly similar}
\]
Experiment design

In producing the results presented in the next section of this report a number of different experiments were setup and run using the aforementioned framework. The following experiments were run:

<table>
<thead>
<tr>
<th>Data set</th>
<th>Algorithms tested</th>
<th>Samples</th>
<th>Epochs per sample</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brazil</td>
<td>Feed forward Neural Network (FFNN)</td>
<td>55 samples</td>
<td>5000 epochs</td>
</tr>
<tr>
<td>Russia</td>
<td>Jordan Recurrent Neural Network (RNN)</td>
<td>55 samples</td>
<td>5000 epochs</td>
</tr>
<tr>
<td>India</td>
<td>Elman Recurrent Neural Network (RNN)</td>
<td>55 samples</td>
<td>5000 epochs</td>
</tr>
<tr>
<td>China</td>
<td>FFNN’s trained with standard PSO</td>
<td>55 samples</td>
<td>5000 epochs</td>
</tr>
<tr>
<td>South Africa</td>
<td>FFNN’s trained with charged PSO</td>
<td>55 samples</td>
<td>5000 epochs</td>
</tr>
</tbody>
</table>

For each experiment three measures of fitness were captured:

1. Sum Squared Error (SSE) values
2. Total Prediction Error (PE) values
3. The predictions after 5000 epochs

In total more than 13.6 million data points were produced during this research study. The derived results presented in the next section represent averages across samples of at least 55 samples. As such, they can be considered to be ‘statistically relevant’ and empirically verified.
Research results

Sum Squared Error (SSE) and Total Prediction Error (PE) results

The results obtained across all five data sets representing each nation in the BRICS group were highly consistent with one another with respect to minimizing the sum squared error and total prediction error. As such, the results obtained using the Brazil data-set have been selected for presentation in this report. For completeness, a full set of results for each country is provided in the appendix of this report.

Table of graphs

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sum Squared Error (SSE) after 1000 iterations</td>
<td>Shows performance of each algorithm in minimizing the SSE objective function over 1000 epochs</td>
</tr>
<tr>
<td>Sum Squared Error (SSE) after 5000 iterations</td>
<td>Shows performance of each algorithm in minimizing the SSE objective function over 5000 epochs</td>
</tr>
<tr>
<td>Total Prediction Error (PE) after 1000 iterations</td>
<td>Shows the total prediction error made by the neural network for 1000 iterations</td>
</tr>
<tr>
<td>Total Prediction Error (PE) after 5000 iterations</td>
<td>Shows the total prediction error made by the neural network for 5000 iterations</td>
</tr>
<tr>
<td>Feedforward Neural Network Economic Forecasts after 5000 epochs</td>
<td>This graph shows the predictions made by a standard FFNN on the inputs in the data set. The last four time periods in the series are unseen</td>
</tr>
<tr>
<td>Jordan Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs</td>
<td>This graph shows the predictions made by a Jordan RNN on the inputs in the data set. The last four time periods in the series are unseen</td>
</tr>
<tr>
<td>Elman Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs</td>
<td>This graph shows the predictions made by an Elman RNN on the inputs in the data set. The last four time periods in the series are unseen</td>
</tr>
<tr>
<td>PSO trained FFNN’s Economic Forecasts after 5000 epochs</td>
<td>This graph shows the predictions made by the global best particle in the PSO. The last four time periods in the series are unseen</td>
</tr>
<tr>
<td>Charged PSO trained FFNN’s Economic Forecasts after 5000 epochs</td>
<td>This graph shows the predictions made by the global best particle in the Charged PSO. The last four time periods in the series are unseen</td>
</tr>
</tbody>
</table>
Brazil Results

Sum Squared Error (SSE) after 1000 iterations

In the above graph you can see that two best performing algorithms with respect to minimizing the sum-squared error of the neural network are the Elman recurrent neural network and the feed-forward neural network trained using the charged PSO. The performance of the Jordan network is comparable to that of a standard feed forward neural network. Additionally, the convergence of standard PSO is quite gradual. However, this is likely attributable to the relatively small parameter values.
In the above graph you can see that the elman network and charged PSO continue to outperform a Jordan neural network as well as a standard feed-forward neural network. Interestingly, the average SSE value of the global best particle starts to outperform the Elman network and the charged PSO toward the last quarter of the epochs. That having been said, the sum-squared-error does not indicate the performance of the neural network in making accurate economic forecasts.
Total Prediction Error (PE) after 1000 iterations

The above graph illustrates the total errors made in economic forecasting made by each of the algorithms. As with the sum-squared error the two most promising algorithms were the Elman recurrent neural network and the feedforward network trained using the charged PSO algorithm. Another interesting observation can be made surrounding the Elman network: the prediction error deteriorated in the beginning of the algorithm for a few hundred epochs before eventually improving.
Total Prediction Error (PE) after 5000 iterations

As with the SSE graphs, the performance of the FFNN particle eventually beats the performance of the Charged PSO trained FFNN and equals the performance of the Elman recurrent neural network.
Feedforward Neural Network Economic Forecasts after 5000 epochs

The above graph illustrates the forecasts made by a feedforward neural network trained to forecast the Brazilian economy after 5000 epochs. The last four data points are unseen.

Jordan Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

The above graph illustrates the forecasts made by a Jordan RNN trained to forecast the Brazilian economy after 5000 epochs. The last four data points are unseen.
Elman Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

The above graph illustrates the forecasts made by a Elman RNN trained to forecast the Brazilian economy after 5000 epochs. The last four data points are unseen.

PSO trained FFNN’s Economic Forecasts after 5000 epochs

The above graph illustrates the forecasts made by a FFNN trained by a PSO to forecast the Brazilian economy after 5000 epochs. The last four data points are unseen.
Charged PSO trained FFNN’s Economic Forecasts after 5000 epochs

The above graph illustrates the forecasts made by a FFNN trained by a Charged PSO to forecast the Brazilian economy after 5000 epochs. The last four data points are unseen.
Conclusions

The conclusions drawn from the results of this empirical study have been summarized below. Conclusions pertaining to the algorithms and those pertaining to economic forecasting have been separated below for clarity.

Conclusions pertaining to the algorithms

- The use of the charged particle swarm optimization algorithm in training feedforward neural networks on temporal data is a viable and sometimes better approach than using simple recurrent neural networks.
- For the problems investigated, the Elman simple recurrent neural network was the best performing approach on average in terms of both minimizing the sum-squared error as well as minimizing the total forecast error of the network.
- For the problems investigated, the charged particle swarm optimization was the second best performing approach on average and outperformed a standard PSO considerably in terms of total prediction error and the forecast graphs include in appendix A.
- In the majority of cases tested, the Jordan simple recurrent neural network performed worse than the elman simple recurrent neural network. Upon inspection the predictions of the Jordan RNN’s appear more ‘erratic’.

Conclusions pertaining to the problem domain

- Neural networks can be used to approximate non-linear economic forecast models, but, performance is problem dependent. Certain models performed better for some countries than others therefore careful consideration should always be taken when choosing a neural network architecture.
- On average the traditional feed-forward neural network best approximated the training data and showed the greatest sensitivity to the total number of decision boundaries in the data. This can be seen in the forecast graphs in appendix A.
- Different neural network architectures embody different characteristics and can be more or less sensitive to ‘turning points’ in temporal data. As such, the use of multiple neural network architectures in a quorum might produce more accurate results.
Recommendations for further research


2. Assess the hypothesis that multiple neural networks deployed within a quorum would improve the accuracy of economic forecasts. Include a comparison of quorums consisting of neural networks using the same architecture against quorums consisting of neural networks using different architectures.

3. Extend the current research topic to calculate the ‘significance’ of each economic indicator feeding data into the neural network. The aim of this would be to determine which economic indicators influence economic growth and which ones do not.
References

Academic Resources used


Web Resources used

Appendix A - Rest of Results (India, Russia, China, and SA)

Russia Results

Sum Squared Error (SSE) after 1000 iterations

Sum Squared Error (SSE) after 5000 iterations
Total Prediction Error (PE) after 1000 iterations

Total Prediction Error (PE) after 5000 iterations
Feedforward Neural Network Economic Forecasts after 5000 epochs

Jordan Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs
Elman Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

PSO trained FFNN's Economic Forecasts after 5000 epochs
Charged PSO trained FFNN's Economic Forecasts after 5000 epochs
India Results

Sum Squared Error (SSE) after 1000 iterations

![Graph showing Sum Squared Error (SSE) after 1000 iterations for India Data Set]

Sum Squared Error (SSE) after 5000 iterations

![Graph showing Sum Squared Error (SSE) after 5000 iterations for India Data Set]
Total Prediction Error (PE) after 1000 iterations

![Graph showing PE after 1000 iterations]

Total Prediction Error (PE) after 5000 iterations

![Graph showing PE after 5000 iterations]
Feedforward Neural Network Economic Forecasts after 5000 epochs

Jordan Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs
Elman Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

PSO trained FFNN’s Economic Forecasts after 5000 epochs
Charged PSO trained FFNN's Economic Forecasts after 5000 epochs

![Graph showing economic forecasts using Charged PSO trained FFNNs](image-url)
China Results

Sum Squared Error (SSE) after 1000 iterations

Sum Squared Error (SSE) after 5000 iterations
Total Prediction Error (PE) after 1000 iterations

Total Prediction Error (PE) after 5000 iterations
Feedforward Neural Network Economic Forecasts after 5000 epochs

![Feedforward NN Forecasts China](image1)

Jordan Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

![Jordan RNN Economic Forecasts China](image2)
Elman Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

PSO trained FFNN's Economic Forecasts after 5000 epochs
Charged PSO trained FFNN’s Economic Forecasts after 5000 epochs
South Africa Results

Sum Squared Error (SSE) after 1000 iterations

Sum Squared Error (SSE) after 5000 iterations
Total Prediction Error (PE) after 1000 iterations

![Graph showing Total Prediction Error (PE) after 1000 iterations.]

Total Prediction Error (PE) after 5000 iterations

![Graph showing Total Prediction Error (PE) after 5000 iterations.]

Page 50
Feedforward Neural Network Economic Forecasts after 5000 epochs

[Graph showing Feedforward NN Forecasts South Africa]

Jordan Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

[Graph showing Jordan RNN Economic Forecasts South Africa]
Elman Recurrent Neural Network (RNN) Economic Forecasts after 5000 epochs

PSO trained FFNN's Economic Forecasts after 5000 epochs
Charged PSO trained FFNN’s Economic Forecasts after 5000 epochs