Analysis of the viability of a Dynamic GA coupled with Fuzzy Membership functions in the Forex Market

Paulo Bernardo
paulo.bernardo@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

November 2021

Abstract

Trading services have never been in higher demand than in the recent years, with many investors turning their eyes to the Foreign Exchange Market due to its high volume and low volatility. This thesis explores the background of this market, as well as the background of some machine learning techniques applied to it throughout the last years. This work also proposes and presents an implementation of a Dynamic GA based in the technical analysis of the FOREX market. An hybrid approach using this Dynamic architecture and a Fuzzy Logic component is also proposed, with the objective of analysing its viability to forecast a financial market such as the FOREX. By testing both frameworks in three different testing periods of six months each the Dynamic implementation presented steady and profitable results in all three periods, while the hybrid approach presented itself as a non viable option in its current version to real time trading. Some promising results of this version do point to the possibility of creating a profitable and steady implementation.
Keywords: FOREX, Fuzzy Logic, Genetic Algorithms, Technical Indicators

1. Introduction

The Evolutionary Computation concepts that an algorithm such as the GA represents, are based in the natural selection and evolutionary processes present in Darwin's "On the Origin of Species", which are translated to the GA's mechanisms such as fitness based selection, mutation, and even crossover breeding. In recent years, several relevant implementations of these principles have been applied to the financial markets with positive results, in works by Gorgulho et al. [5], by Hirabayashi et al. [6] and by Almeida et al.[7].

The idea of Fuzzy Logic is more recent, having first been mentioned in a proposal by the Azerbaijani scientist Lotfi Zadeh [13] in 1965. The general concept associated with fuzzy logic is to discard binary/ternary signals and instead associate values that represent partial truth, as opposed to a Boolean truth. Some relevant works that employ the concept of fuzzy logic to financial markets are the works published by Juszczuk et al. [8] and Naranjo et al. [10].

The environment selected to test the performance of the implemented architecture was the Foreign Exchange Market (FOREX). This is the most traded financial market in the world [4], even larger than the stock market, with a daily volume of around \$6.6 trillion, according to the 2019 Triennial Central Bank Survey [1]. In this market the financial instruments sold and bought are the countries' currencies. To forecast this financial market, while using a machine learning architecture, two types of market analysis were considered. The Fundamental analysis and the Technical Analysis. In the context of this market, the first relates to the macro economic factors of the currency market, such as geopolitical factors as well as inflation rates. The second analysis of the bases itself solely on the values of the market to generate technical indicators that forecast the market's behaviour, making it perfect type of analysis to use in a machine learning framework such as the one that this work proposes.

To this effect several methods have already been proved to achieve profitable results while forecasting the FOREX market with the use of technical analysis of the market allied to Neural Networks, or coupled with an Evolutionary Algorithm. Within the same scope of these works, Fuzzy logic approaches have also proven to present more consistent results than some benchmarks such as the Buy & Hold strategy.

2. Background

2.1. Forex market analysis

Now that the intricate mechanisms of the Forex market are exposed, the necessity of analysing its state at a given time arises. To this effect, two schools of thought emerge, the Fundamental Analysis and the Technical Analysis.

Fundamental analysis studies the relationship between the evolution of exchange rates and economic indicators [2], of the currencies' country. This analysis focus heavily in the assessment of the economic, social and political forces surrounding the intervening economies.

Most fundamental studies rely heavily on macroeconomic indicators, such as economic growth rates, interest rates, inflation, and unemployment. Other criteria taken into account, while forecasting the FOREX market, are the geopolitical factors, such as, monetary and economic policy changes, and changes in governments or international relationships. The importance of the aforementioned indicators is evidenced by the fluctuations that are felt in the market during important economic meetings, such as meetings of the European Central Bank or the Federal Open Market Committee.

While fundamental analysis is used to forecast the long term evolution of the currency rates, based on the macro-economical indicators, technical analysis focus solely on the past events of the currencies' price rates, to forecast the future rate evolution and capitalize from it. Due to its focus on the past rates, the technical analysis has become the primary tool to successfully analyze and trade shorter-term price movements [2].

This type of study is based on three premises, enunciated by Ozturk et al. [11]:

- Market action discounts everything: any factor that can affect the prices is already reflected in the price.
- Prices move in trends: the purpose of the technical analysis is to detect a price trend in the early phases of development.
- History repeats itself: technical analysis uses patterns which have shown success in the past and assumes they will work in the future.

This type of analysis consists primarily on the study of technical indicators, some of which can be interpreted to predict market direction or to generate buy and sell signals, as well as to set profit targets and stop-loss safeguards, due to its ability to generate price-specific information and forecasts [2].

As Almeida et al. [7] states, exist two main types of technical indicators:

- Trend following - Used to understand trends, that is, to identify if a trend has begun or ended. These indicators are used usually to identify entry and exit points.
- Momentum oscillators - Predict sudden changes on the asset's behaviour, such as, the speed of the price movement variation, which is of major importance when considering the amount of leverage to be used.

2.2. Fuzzy Sets

When considering forecasting with indicators, the traditional approach to take would be a crisp one. In this approach a BUY/SELL signal is generated when conditions are satisfied, for example a BUY signal can be generated as the following:

$$f_{BUY} = true \quad , if(cond_{1,2,...,N_{BUY}} = true \tag{1}$$

These conditions are binary, taking only values of *True* or *False* (1 or 0), and are derived from the indicators' value. One such condition can be exemplified by the logic present in equation 2, that corresponds to a condition based in an indicator's value, in which, if this indicator exceeds tr, the condition becomes true.

$$cond_1 = true \quad , if(ind_{1_{value}} > tr). \tag{2}$$

On the other hand, in a multi-criteria fuzzy approach, such as the ones proposed by Naranjo et al. [10] and by Juszczuk et al. [8], the signals BUY/SELL are generated by analyzing a vector of the type $y_c = [y_1, y_2, .., y_n]$, where n is the number of indicators, and $y_1, y_2, ..., y_n$ are the "fuzzied" values of each indicator, which are computed by inserting each of the indicator's value in its corresponding membership function. These membership functions can be constructed in a way that the value which the indicator takes is non-binary. An example of a membership function, that corresponds to the fuzzy approach explained previously can be observed in figure 1.

This type of approach, while making the computation more difficult, tolerates uncertainty in decision making [8], and can be used to make a more flexible approach.

2.3. GA: Structure

A traditional GA, as the one described in [9], would have a structure as described by the flowchart present in figure 2. The first step is, as with several algorithms, initialization, in which an initial population is generated. This population is comprised of individuals that can be described by a data structure, called chromosome. These data structures

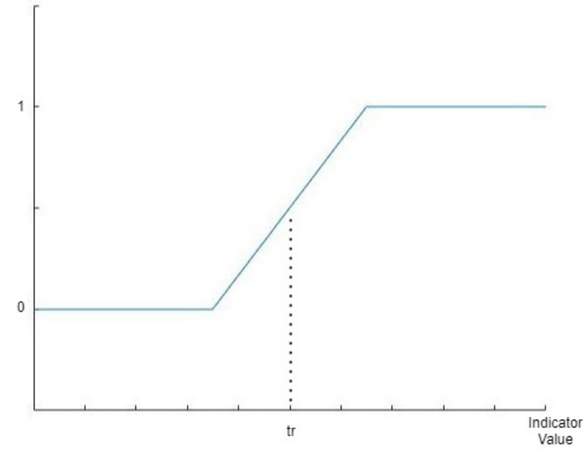

Figure 1: Fuzzy approach membership function.

are normally composed of binary, real or integer arrays.

Defining and designing the chromosome is a task that depends heavily on the problem at hand, since the solution of the algorithm will be the chromosome of the fittest individual when the termination criteria is met. After defining the chromosome structure to be used, the population is randomly initialized.

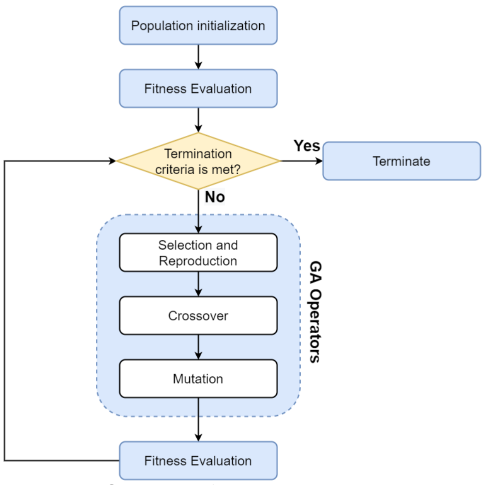

Figure 2: Structure of a genetic algorithm.

The next step is an initial fitness evaluation, which can be optional, with many researchers disregarding this step. A fitness evaluation should be made whenever changes to the population are made, with the objective of testing if an individual meets the termination criteria. In GA models tasked with forecasting a financial market, a very standard fitness evaluation is the Return on investment (ROI) measured for each individual in a simulated time series using training data, which can be formulated as in equation 3.

$$ROI(X) = \frac{Returns(X) - Investment(X)}{Investment(X)} \quad (3)$$

The individuals' fitness evaluation is also of great importance to the next step, selection. The objective of this step is to pick a set of individuals that will transit to the next "generation". There are several methods for selecting individuals, such as Truncation Selection [5] and Tournament Selection [6].

Afterwards, a new generation of individuals is generated, which is comprised of the set of selected individuals and a set of new individuals created through crossover. The process of crossover selects two individuals ("parents") from the selected pool, and through crossover techniques, generates new chromosomes ("offspring"). Most common techniques, such as Two-Cut-Point Crossover [6] or One-Cut Point Crossover [5], involve "cutting" the chromosome in one or more points and generating a different chromosome using parts of the "parents" chromosomes. The final GA operator procedure, before evaluating the population's fitness, is mutation. A mutation event is a low-probability occurrence, changing one or several genes (variable) inside a given chromosome.

This algorithm ends up converging to an optimal population, after several generations have passed. The best individuals from this population are then chosen to test their performance, which in this case, is to forecast the forex market.

2.4. GA:Diversity

The main problem of trying to obtain an optimal solution using a genetic algorithm is the population stagnation around local maxima. This problem is explored by MIT's Professor Patrick H. Winston in his lecture [12], in which several populations are initialized and updated during several generations with the objective of obtaining an optimal solution, ending up to stagnate around a local maximum without reaching this solution. The origin of the stagnation is the loss of diversity that affects the population after converging to one of these local maxima, severely reducing the search space. Such event happens due to the fact that the best individuals of the current generation, which are the ones present in this local maximum, will be the progenitors of the next.

A basic approach to help maintaining a population diverse is to increase the probability of mutation [9], which is the only operator inside a standard GA that introduces diversity to a pool of genes. Other solution for this inherent problem is the one carried out in [12], in which the evaluation of each individual is based on its fitness value coupled with how different it is from other individuals already selected, that is, the individuals that are most likely to be selected for the next generation are the ones that possess high fitness values and have less similarities between themselves. Other concept that may help improve diversity is ”immigrants”. This concept consists in the insertion of randomly created individuals in each generation [3].

2.5. GA: Adaptive Approach

Traditionally, GAs aim to solve static problems, whose solutions are precise and quick to obtain, on the other hand, when facing a real world problem, challenges arise. These challenges may be dealt with, using different and more effective approaches.

One such problem that arises from the work at hand is the fact that, although financial markets are proven to be cyclic in the long term, in the short term they are subject to continuous changes. These constant changes mean that an instance of a technical indicator that works well on a particular trend may fail when the fitness landscape changes [3]. To deal with this challenge the genetic algorithm must be able to find the optimal solution for each instance without having to restart the whole algorithm, giving the model an adaptive behaviour.

2.6. State of the Art

The use of technical indicators in conjunction with a Genetic Algorithm to forecast financial markets is an approach widely carried out by several academics, while forecasting financial markets.

An example of the use of TI is the approach carried out by Gorgulho et al. [5], that has the objective of trying to forecast the stock market. In this work a GA model generates BUY/SELL signals based on technical rules which are designed through the technical analysis of 7 indicators and through the Double Crossover method. Both of these works also explore a way to maximize profit using a portfolio composition module, which is achieved by choosing the best stocks in the market to invest in.

Literature more focused in the financial market at hand (FOREX) includes Almeida et al.[7] who use 5 different technical indicators in their SVM-Genetic Algorithm hybrid approach, in which both the technical indicators and a sequence of prices are used to train both the SVM and Genetic Algorithm models and to generate BUY/SELL signals. Other works in which technical indicators are used in conjunction with a genetic algorithm, are the one published by Hirabayashi et al. [6], in which 4 indicators are used, and the study made by Ozturk et al. [11]. The latter work uses a total of 24 indicators(parameterized using a GA), to capture the underlying ”rules” and testing them with several selection modules, namely a genetic algorithm and a greedy search heuristic.

A different use of technical indicators applied to the forecast of Forex, is present in the works of Naranjo et al. [10] (uses RSI, Average Directional Movement Index and a custom-made MACD) and Juszczuk et al. [8] (uses RSI, Commodity Channel Index, Stochastic Oscillator, DeMarker indicator, Bulls indicator and Moving Average of Oscillator). In both works, membership functions are created for each indicator ”fuzzing” the indicators’ values. This fuzzy information is used later to generate BUY and SELL signals through custom-made algorithms developed independently in each work.

3. Implementation

3.1. Overview

As stated in the introductory section, the main objective of this thesis is to create a model that predicts the FOREX market effectively, and study the viability of this method coupled with a fuzzy approach. With that objective in mind, several algorithms already applied to the subject by previous researches were considered as a starting point. Among them were algorithms such as Random Forests, Neural Networks and Genetic Algorithms. Due to its previous successes in the area, a Technical Analysis based Dynamic Genetic Algorithm was chosen as the fundamental idea.

A simple overview of the implemented algorithm is represented in figure 3, which highlights several key parts of the model. Each of these components deserve its own section, where they will be succinctly explored, while trying not to loose focus of the overall algorithm.

3.2. Chromosome Generation

To design the chromosome’s structure, some inspiration was drawn from the architecture of the work from Almeida et al. [7], in which chromosomes are composed by one weight and one parameter, for each trading rule implemented.

In this work however, chromosomes are created with one weight and several parameters (variables) for each trading rule. The general structure of the chromosomes generated during this work is represented in figure 4. From this figure one can conclude that the chromosome is split into several segments, one for each trading rule, which may not have the same number of parameters.

3.2.1 Individual Creation

After generating a number of chromosomes equal to the size of the population intended, the implemented architecture creates an individual for each of the chromosomes. In this architecture, an individual possesses three main components, which are the chromosome that characterises it, a data frame which details how the individual performs in the training data, and finally a fitness score that allows the individual to be compared with others. Since the chromosomes have already been cre-

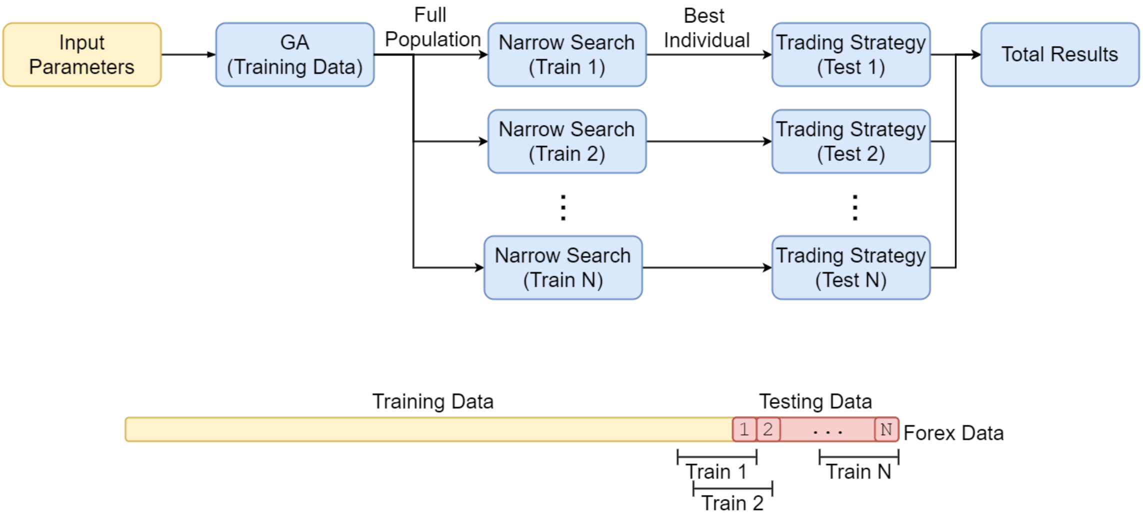

Figure 3: Simplified overview of the implemented model.

ated, the first component of each individual already exist.

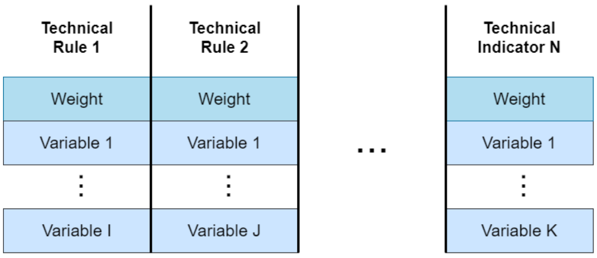

Figure 4: Representation of the general structure of this architecture's chromosome.

The second step is to create the data frame. This data frame was constructed to contain the trading signals of each trading rule, a general trading signal, and the simulation values of the general trading signal in the training data. To compute the trading signals of the selected trading rules, the model first uses a few of the parameters present in the chromosome and the data frame with the input data to compute the technical indicators, necessary to process the trading rules selected. The selected trading rules use as inputs the indicators, as well as the remaining parameters and the input data, to generate a trading signal composed by buy signals (1), sell signals (-1) and neutral signals (0), referent to the training period.

Each trading rule produces a signal, which are then combined to produce the general trading signal. The previously mentioned weight parameter enters in effect at this stage, being the main component in the combination of the trading signals, giving emphasis to some trading rules to the detriment of others. This follows the guideline in equation 4, which generates an array with positive and negative float values. This types of values are however incompatible with the previous notion of buy and sell signal, which are symbolised by the number 1 and -1.

$$general\_signal = \sum_{i=0}^{N} signal_i \cdot weight_i \tag{4}$$

Therefore this general signal has to be normalised prior to simulation. The devised method to normalise this signal follows the logic present in algorithm 1. As it can be deduced, the implemented architecture processes the signal array obtained previously into a quinary trading signal composed by integer values which may symbolise light/heavy buy or sell positions, or even a neutral position.

Algorithm 1: Method 2 to normalise the general trading signal.

$buy_1 = 1/3 * max(general\_signal)$
$buy_2 = 2/3 * max(general\_signal)$
$sell_1 = 1/3 * min(general\_signal)$
$sell_2 = 2/3 * min(general\_signal)$

for *i* **in** *general_signal* **do**
  if $i > buy_2$ **then**
    $i = 2$
  else if $buy_2 > i > buy_1$ **then**
    $i = 1$
  else if $i < sell_2$ **then**
    $i = -2$
  else if $sell_2 < i < sell_1$ **then**
    $i = -1$
  else
    $i = 0$

After obtaining the normalised trading signal, whose function is to state the trading strategy of its individual for the period at hand, the objective becomes to simulate the strategy represented by it and evaluate it.

3.3. Population Training

After creating the population, the next step in any GA is to train it.

As stated in section 2.3, the first step in a traditional GA is to evaluate every individuals' fitness, and rank them. In the case of the implemented architecture, this method is replaced by *Sort Population*, which as its name suggests, sorts the population created previously by the fitness values of its individuals (computed previously in the individual simulation). As input this process receives a list of individuals, which in the first iteration comes directly from the *Population Creation* process, as opposite to the following iterations, in which this list of individuals is created from the chromosomes that result from the GA operators.

After sorting the population, a *Selection* process takes place, in which the chromosomes of the top individuals, are extracted and two lists are made with them, as figure 5 demonstrates. The first, the *elite* list, is composed by the chromosomes from the individuals with the best fitness values, in this case the top 1-10% individuals.

The second list is the *crossover* list, which is composed by the top 50-60% individuals from the population. This list also contains the elites from the first list, and is used to perform the crossover operation described in the next section.

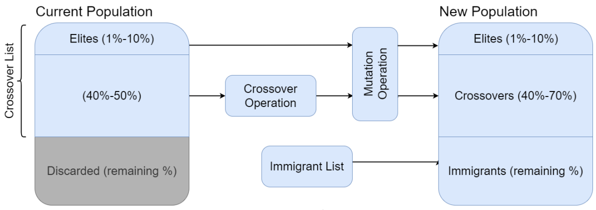

Figure 5: Representation of the selection process.

The final list of chromosomes resulting from the GA operators, is composed by the elites' chromosomes, the chromosomes generated from the crossover operation and immigrant chromosomes generated using the process described in section 3.2, following the proportions shown in figure 5.

3.4. Population Testing

One of the major objectives of this work was to reproduce a GA that optimizes the solution to the problem at hand dynamically. As such, a solution was envisioned in which a population would be traditionally trained with a given amount of training data, in which the diversity mechanisms would be tuned to generate the most diverse population. The model would subsequently retrain this population using a much smaller sample size and number of generations, narrowing the diversity of the original population, and use the best of the resulting individuals to predict the next few periods. The originally trained population would then be used in the retraining process several times before a reset happens and a new population is created. The implementation of this process, dubbed Narrow Search method, is further explained in its own section (3.4.1).

3.4.1 Narrow Search

The second method is more complex, since there several steps before arriving at a final trading strategy. A simplified representation of the method is presented in figure 6. In contrast with the first method of testing, it is observable that the second one has more inputs, adding every individual of the originally trained population, a series of training periods and a set of GA parameters, to the input list of the previous method.

The first step of the implemented procedure is to segment the total testing period into several shorter periods (hours/days), and to associate each of them to a shorter training period (weeks/months) that directly precedes them. An example of this would be to train the population in the period between 1/1/2015-28/2/2015 and generate a trading signal to the whole day of 1/3/2015.

After creating these time periods, a narrow training is performed with the previously trained population using one of the training periods. It should be noted that, from this point forward the first training is going to be called *wide* training and the training which occurs in this method of testing will be called *narrow* training, since the two types of training aim to achieve different things. The wide training aims to create the most diverse population, having a high number of generations and high ga parameters that increase the population diversity. Conversely, the narrow training aims to decrease the population's diversity, with fewer generations and low ga parameters.

The final step of this testing method is to use the best individual of the population that results from the narrow training and generate a trading signal for the testing period associated with the reduced training period. The whole process is repeated one time for each of the testing periods, resulting in a series of trading signals, that are consequently appended and simulated as a whole.

3.5. Fuzzy Implementation

One of the points of interest in this work is the comparison between a crisp logic model and a fuzzy logic model, the latter of which being the theme of the current section.

Firstly, it should be emphasised that both of the

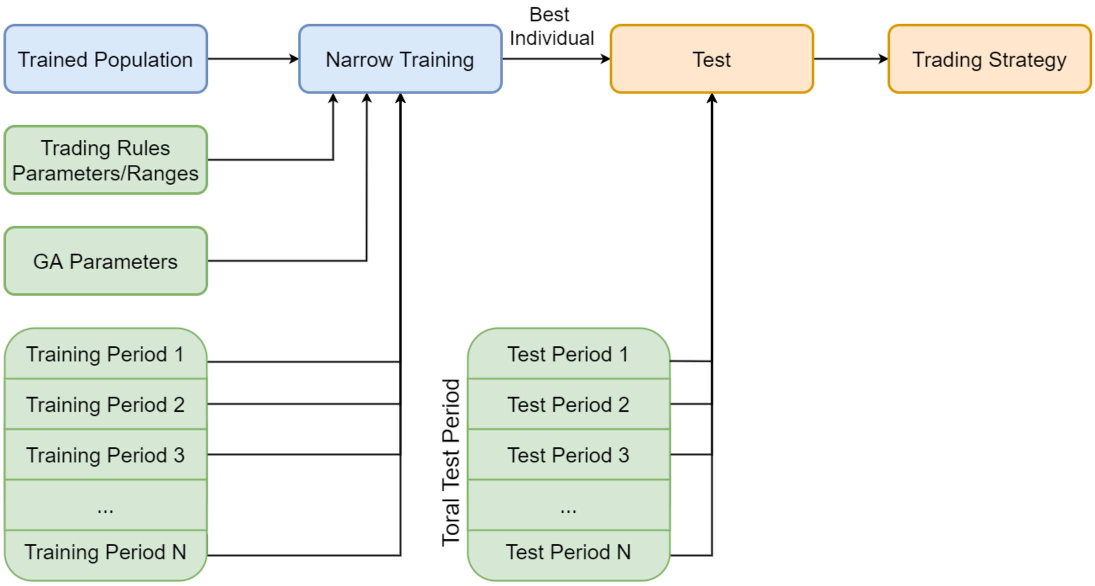

Figure 6: Representation of the Narrow Search method.

implemented models are almost identical. The only difference between both models occurs during the computation of the trading signals of each selected trading rule in the *Individual Creation* (3.2.1), in which the crisp model generates a ternary signal (either -1, 0 or 1), while the fuzzy implementation generates a signal with float values between -1 and 1.

It is evident that this implementation relies heavily in custom made membership functions, which receive as input the value of a given indicator and generate as output a float signal value. After generating a signal for each of the selected trading rules the implementation follows the same process as in the crisp model, creating a general trading signal using the same method, represented by the pseudo-code in algorithm 1. It should also be stated that, the trading signals that are generated during the training and testing processes of the algorithm, are also processed through these membership functions.

The logic behind this approach is to generate weaker signals during periods in which the used indicator marginally fulfils the trading rule. On the other hand, signals close to 1 or -1 should be employed an indicator fulfils the trading rule by a considerable margin. Using this logic, it should be possible for the trading rules to assign how strong a trend is.

4. Results & discussion

4.1. Methodologies

The implemented system's validation can be described by two different case studies. However in each of the case studies some inputs and configurations will be kept constant.

Such is the case of the input data was used to train and test the trading system throughout the validation process, consists in FOREX's historic data feed of the EUR/USD currency pair. From the total data that spans from the year 2014 until 2020, three periods were selected, and segmented into training and testing data. The three testing periods are consecutive and not overlapped, providing three time periods with different characteristics.

In addition to the input data, each of the case studies also uses the same input parameters used to configure the Wide training component of the GA. As stated previously this training component precedes the testing process, which may follow the guidelines present in section **??** to generate the trading strategy of a static GA, used as a comparison object throughout the Validation section. It may also follow the guidelines of the Narrow Search method described in section 3.4.1, to produce a trading strategy using a Dynamic approach.

In table 1 each of training parameters are presented. These parameters were selected through the analysis of the experiments described in the literature, that had been performed in this same market with a comparable framework.

The input parameters utilized to simulate trading strategies were also invariable throughout the Validation process, both during the training are the testing of the algorithm. These parameters, which are present in table 2, provide a constant strategical behaviour throughout the executions, which allows comparisons between the machine learning methods to be made, without the trading strategy being a factor.

Table 1: Configuration of the parameters of the Wide Search (GA training).

Parameters	Value
Population Size	100
Generations	100
Elites	4
Crossover List	50
Crossover Generated	50
Mutation Rate	10

Table 2: Default simulation parameters.

Parameters	Values
Initial Investment	100 000€
Default Position	5000€
Leverage	20
Stop Gain Factor	5
Stop Loss Factor	0.2
Trading Signal	Quinary

To finalize, it should be pointed out that, throughout the analysis of the case studies, several benchmarks will be present, namely the BH and SH strategies, and, as stated above, a trading strategy generated by a static GA.

4.2. Case Study A

The first study case presented aims to evaluate the results of the Narrow Search method, and to observe how this method fares against the selected benchmark strategies. To achieve that, the case study focuses in utilizing different configurations for the input parameters of this method, in order to maximize the performance of this method.

The baseline configuration for this method is presented in table 3. It is possible to observe that each of the parameter values of every population diversity setting is tuned down, when compared with the configuration of the Wide Search presented previously. Evidence of this is the low mutation rate, and the high number of elite individuals and individuals generated by crossover. The number of generations is also tuned down. The objective of this tuning is to not overtrain the population during the testing phase of the algorithm.

Table 3: Parameters of the baseline configuration.

Parameters	Value
Generations	2
Elites	15
Crossover List	50
Crossover Generated	60
Mutation Rate	2%
Training Period Size	1440 samples
Testing Period Size	6 Samples

Several variations were selected to be offered as subjects of comparison to the baseline configuration. Among these configurations, there are variations selected to study the implemented method's behaviour when it processes different sample sizes to train and test samples. These variations were obtained by doubling/halving the training period size and doubling the testing period size, and both at the same time. Other variations, such as increasing the number of generations, increasing the mutation rate and reducing the number of elites, were selected to study the effects of increasing the diversity factor in the testing method. The full set of configurations is presented in table 4, which also evidences which parameters are altered.

It should also be mentioned that during this case study the only trading rules employed are presented in table 5. These trading rules were selected due to being used with success in literature with comparable frameworks, such as Hirabayashi et al. [6], Almeida et al. [7] and Gorgulho et al. [5].

4.3. Case Study A : Results

From the results obtained by varying the sample sizes, it is possible to conclude that the configuration *3TS-2*ts* achieved an average roi value higher than the other configurations in the 15 runs executed. This configuration scored 21.24%, against the 20.14%, 19.99%, 19.42% and 19.30%, of the *½*TS*, *3TS*, *2*ts* and *½*TS_2*ts* configurations, respectively. The *Base* configuration also performed worse than the *3TS-2*ts* configuration, having obtained an average roi of 20.96%.

Table 5: Trading rules and ranges used in case study A.

Trading Rule	Variables	Value Range
RSI	N	[5, 30]
	$ceiling$	[65, 75]
	$floor$	[5, 30]
ROC	N	[15, 120]
EMA	N	[10, 150]
MACD	N_{signal}	[5, 15]
	N_{short}	[10, 25]
	N_{long}	[20, 40]

The results obtained by using the configurations in which the diversity parameters are altered, offer several conclusions. First of all, it is possible to observe that every configuration in which the diversity parameters were modified attained worse results than the baseline configuration. The *5Gen*, *10Mut*, *30Elites* and *5Elites* obtained as an average ROI, respectively, 13.50%, 11.50%, 14.25% and 14.46%, against the 20.96% obtained by the base configuration.

Table 4: Variations of the baseline configuration tested in the Narrow Search method.

Configuration Name	Changed Parameters	New Values
1/2*TS	Train Period Size	1 Month (30*24 Samples)
3TS	Train Period Size	3 Months (90*24 Samples)
2*ts	Test Period Size	12 Samples
1/2*TS-2*ts	Train Period Size Test Period Size	1 Month (30*24 Samples) 12 Samples
3TS-2*ts	Train Period Size Test Period Size	3 Months (60*24 Samples) 12 Samples
5Gen	Generations	5
10Mut	Mutation Rate	10 %
30Elites	Elites	30
5Elites	Elites	5

The main conclusion to take away is that the Narrow Search method presents better average ROI than the Static GA, independently of the configuration used, as evidenced by figure 7. It should also be stated that, if the only metric used was the position accuracy, the Static GA would be most viable solution, since it obtain the best average accuracy, with 63.39%.

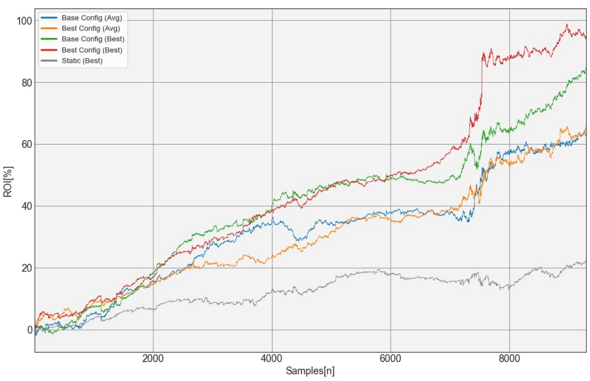

Figure 7: Evolution of ROI obtained by the relevant configurations of case study A throughout the totality of the testing period.

4.4. Case Study D

This final case study aims to validate the proposed fuzzy implementation coupled either with the Dynamic GA or the Static GA, and to compare it with the crisp approaches of both algorithms. To start, it should be stated that each of the default input settings are present in the Methodology Section (4.1), such as the training and testing periods utilized, the Wide Search parameters (table 1) and the investment parameters (table 2).

To test the fuzzy component coupled with the Dynamic ga, which utilizes the Narrow Search method as its testing module, one of the previously studied configurations was selected. The used configuration, which is present in table 3, is similar to the configuration utilized as baseline in Case Study A (section 4.2).

Two trading rule sets were selected in the validation process of this Case Study. Table 7 presents both of the sets.

Table 7: Sets of trading rules used during Case Study D.

Set 1	Set 2
RSI	RSI
ROC	ROC
EMA	EMA
MACD	MACD
	Bollinger
	Stochastic

4.4.1 Case Study D: Results

This section concludes Case Study D, from which several conclusions about the viability of a fuzzy approach can be pointed out.

The main conclusion to take away from this case study would be that the implemented fuzzy approach does not seem viable to implement in a real world setting. Its large computation time and the poor results of this approach coupled with rule set 1 are the main key contributors to this conclusion.

On the other hand, the same approach, while employing rule set 2, obtained slightly more profitable results than its crisp counterpart. In figure 8, the average ROI results of each of the Dynamic approaches are represented throughout the testing periods. As highlighted previously, the fuzzy approach coupled rule set 1 present similar results to the other approaches in the first testing period (0-3000 samples), and a decline in performance in the succeeding testing periods.

These results and the fact that the Static GA in conjunction with the fuzzy membership functions present slightly more profitable results than the their crisp counterparts, are promising indicators that an optimised version of this approach may be implementable.

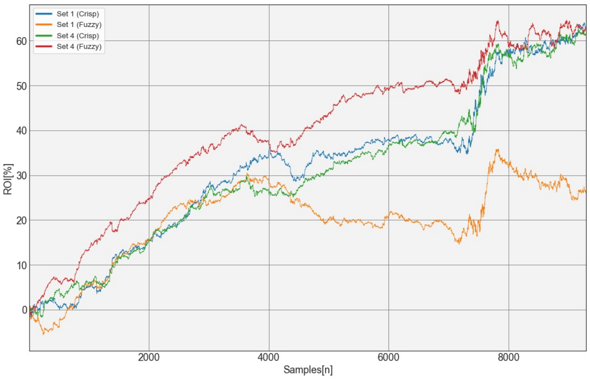

Figure 8: Evolution of ROI obtained by the crisp and fuzzy approaches throughout the totality of the testing period.

5. Conclusions

In this work a Dynamic approach that combines traditional GA with the proprieties of the Narrow Search method is proposed implemented. This model is based in the technical analysis principals and was tested in the FOREX market environment. This architecture was also coupled with a fuzzy logic approach to accomplish the objective of analysing the viability of an architecture such as this in a financial market such as the FOREX.

Throughout its testing several conclusions were drawn about the performance of the implemented approaches. First the robustness and efficiency of the Dynamic implementation were overwhelmingly positive, exceeding the profits of its main benchmark, a Traditional Static GA. It is important to mention that these profits are associated with low exposure values, presenting itself as relatively safe investment strategy generator.

This robustness and stability presented themselves throughout every case study in which the impact of parameter configurations and the use of trading rules was studied. This and the fact that this approach preformed better than several benchmarks and comparable frameworks from the state of the art indicate the accomplishment of one of the main objectives of this thesis.

The third and final conclusion of this work is that the fuzzy implementation coupled with the Dynamic approach created in context of this thesis does not present itself as viable to real time testing due to its long computation times and high exposure values, when compared with its crisp counterpart. This goes without saying that, in several executions this approach presented itself more profitable than the crisp version. These are very promising results which indicate that a profitable and stable fuzzy implementation coupled with the Dynamic approach may be possible.

References

[1] Foreign exchange turnover in april 2019. "[accessed 2-January-2021]".

[2] M. D. Archer. *Getting Started in Currency Trading: Winning in Today's Hottest Marketplace*. Getting Started In. Wiley, 2008.

[3] P. Fernández, D. J. Bodas Sagi, F. Soltero, and I. Hidalgo. Technical market indicators optimization using evolutionary algorithms. pages 1851–1858, 01 2008.

[4] M. Galant and M. Brian Dolan. *CURRENCY TRADING FOR DUMMIES*. Wiley India Pvt. Limited, 2007.

[5] A. Gorgulho, R. Neves, and N. Horta. Applying a GA kernel on optimizing technical analysis rules for stock picking and portfolio composition. *Expert Systems with Applications*, may 2011.

[6] A. Hirabayashi, C. Aranha, and H. Iba. Optimization of the trading rule in foreign exchange using genetic algorithm. In *Proceedings of the 11th Annual Conference on Genetic and Evolutionary Computation*, GECCO '09, page 1529–1536, New York, NY, USA, 2009. Association for Computing Machinery.

[7] B. Jubert de Almeida, R. Ferreira Neves, and N. Horta. Combining support vector machine with genetic algorithms to optimize investments in forex markets with high leverage. *Applied Soft Computing*, 64:596 – 613, 2018.

[8] P. Juszczuk and L. Kruś. Soft multicriteria computing supporting decisions on the forex market. *Applied Soft Computing*, 96:106654, 2020.

[9] L. Martí. Algoritmos genéticos, 2014. "[accessed 27-November-2020]".

[10] R. Naranjo, A. Meco, J. Arroyo, and M. Santos. An intelligent trading system with fuzzy rules and fuzzy capital management. *International Journal of Intelligent Systems*, 30(8):963–983, 2015.

[11] M. Ozturk, I. H. Toroslu, and G. Fidan. Heuristic based trading system on forex data using technical indicator rules. *Applied Soft Computing*, 43:170 – 186, 2016.

[12] P. H. Winston. Lecture 13: Learning: Genetic algorithms. "[accessed 14-December-2020]".

[13] L. Zadeh. Fuzzy sets. *Information and Control*, 8(3):338–353, 1965.