In Proceedings, Congress on Evolutionary Computation, Barcelona, Spain, 18-23 July, 2010, 2019-2026

EDDIE for Investment Opportunities Forecasting: Extending the Search Space of the GP

Michael Kampouridis and Edward Tsang
School of Computer Science and Electronic Engineering, University of Essex
Wivenhoe Park, Colchester CO4 3SQ, UK
{*mkampo, edward*}*@essex.ac.uk*

Abstract— **In this paper we present a new version of a GP-based financial forecasting tool called EDDIE. The novelty of this new version (EDDIE 8), is its enlarged search space, where we allow the GP to search in the space of the technical indicators, in order to form its Genetic Decision Trees. In this way, EDDIE 8 is not constrained in using pre-specified indicators, but it is left up to the GP to choose the optimal ones. We then proceed to compare EDDIE 8 with EDDIE 7, which is based on previous EDDIE versions; EDDIE 7 has a smaller space where the indicators are pre-specified by the user and are part of EDDIE 8's space. Results show that thanks to the bigger search space, new and improved solutions can be found by EDDIE 8. However, there are cases where EDDIE 8 can still be outperformed by its predecessor. Analysis shows that this depends on the nature of the solutions. If the solutions come from EDDIE 8's search space, then EDDIE 8 can find them and perform better; if, however, solutions come from the smaller search space of EDDIE 7, then EDDIE 8 is having difficulties focusing in such a small space and is thus outperformed by EDDIE 7.**

I. INTRODUCTION

Financial forecasting is an important area in computational finance [1]. There are numerous works that attempt to forecast the future price movements of a stock; several examples can be found in [2], [3]. Genetic Programming [4], [5] (GP) is an evolutionary technique that has widely been used for financial forecasting. Some recent examples are [6], [7], where GP was used for time series forecasting and for inflation prediction, respectively. In addition, other evolutionary methods have been used for forecasting. Such examples are, for instance, Genetic Network Programming [8] and Differential Evolution [9]. EDDIE [10], [11], [12], [13], is a machine learning tool that uses Genetic Programming, to make its predictions. In this paper, we present EDDIE 8 (ED8), which is the newest version. The novelty of this algorithm is in its rich, extended grammar. Instead of using a fixed number of pre-specified indicators from technical analysis [14], like the previous versions do, ED8 allows the GP to search in the space of these technical indicators and use the ones that it considers to be optimal. Thanks to its enlarged search space, ED8 is considered to be an improvement, because it has the potential, through the learning process, of discovering better solutions that its predecessors cannot. A similar approach to ours, where there is an attempt to address the problem of fixed number of pre-specified strategies, can be found in [15], [16], where Grammatical Evolution was used instead of GP.

In order to present the value of ED8, we compare it with EDDIE 7 (ED7), which is a re-implementation of Jin Li's EDDIE 4 [11], [17] (a.k.a. FGP-2), with the addition of some indicators that Martinez-Jaramillo [18] found helpful and used in his own version of the algorithm. The dataset is created from artificial data, because we consider that this is the best way to ensure that patterns exist in the data and that also we have control over their nature. The way the rest of this paper is organised is as follows: Section II presents and explains the differences between ED7 and ED8, section III presents the methodology used for creating the artificial dataset, section IV describes the experimental parameters, section V shows the results of the experiments, section VI discusses these results, and finally, section VII concludes this paper.

II. DIFFERENCES BETWEEN ED7 AND ED8

In this section we present the two versions, ED7 and ED8, and explain their differences. We first start by presenting the former, and the way it works.

A. EDDIE 7

EDDIE is a forecasting tool, which learns and extracts knowledge from a set of data. As we said in the previous section, ED7 is a re-implementation of Jin Li's FGP-2 with the only difference being that it uses some additional indicators that Martinez-Jaramillo used in his version of EDDIE.

The kind of question ED7 tries to answer is 'will the price increase within the n following days by r%'? The way ED7 works, and in fact all EDDIE versions, is that the user first feeds the system with a set of past data; EDDIE then uses this data and through a GP process, it produces and evolves Genetic Decision Trees (GDTs), which make recommendations of buy (1) or not-to-buy (0). It then evaluates the performance of these GDTs on a training set, for each generation. The GDT with the highest fitness at the last generation is finally applied to a testing set.

The set of data used is composed of three parts: daily closing price of a stock, a number of attributes and signals. Stocks' daily closing prices can be obtained online in websites such as `http://finance.yahoo.com` and also from financial statistics databases like *Datastream*. The attributes are indicators commonly used in technical analysis [14]; which indicators to use depends on the user and his belief

TABLE I

TECHNICAL INDICATORS USED BY THE GP. EACH INDICATOR USES 2 DIFFERENT PERIODS, 12 AND 50 DAYS, IN ORDER TO TAKE INTO ACCOUNT A SHORT-TERM AND A LONG-TERM PERIOD. FORMULAS OF OUR INTERPRETATION FOR THESE INDICATORS ARE PROVIDED IN THE APPENDIX.

Technical Indicators (Abbreviation)	Period
Moving Average (MA)	12 & 50 days
Trade Break Out (TBR)	12 & 50 days
Filter (FLR)	12 & 50 days
Volatility (Vol)	12 & 50 days
Momentum (Mom)	12 & 50 days
Momentum Moving Average (MomMA)	12 & 50 days

of their relevance to the prediction. Table I presents the technical indicators that our algorithm uses.[1]

The signals are calculated by looking ahead of the closing price for a time horizon of n days, trying to detect if there is an increase of the price by $r\%$ [10]. For this set of experiments, n was set to 20 and r to 4%. In other words, the GP was trying to use some of the indicators of Table I in order to forecast whether the daily closing price was going to increase by 4% within the following 20 days.

After we feed the data to the system, EDDIE creates and evolves a population of GDTs. Figure 1 presents the Backus Normal Form (BNF) [21] (grammar) of ED7. As we can see, the root of the tree is an If-Then-Else statement. Then the first branch is either a boolean (testing whether a technical indicator is greater than/less than/equal to a value), or a logic operator (and, or, not), which can hold multiple boolean conditions. The 'Then' and 'Else' branches can be a new Genetic Decision Tree (GDT), or a decision, to buy or not-to-buy (denoted by 1 and 0).

```
<Tree> ::= If-then-else <Condition> <Tree> <Tree> | Decision
<Condition> ::= <Condition> "And" <Condition> |
                <Condition> "Or" <Condition> |
                "Not" <Condition> |
                Variable <RelationOperation> Threshold
<Variable> ::= MA_12 | MA_50 | TBR_12 | TBR_50 | FLR_12 |
               FLR_50 | Vol_12 | Vol_50 | Mom_12 | Mom_50 |
               MomMA_12 | MomMA_50
<RelationOperation> ::= ">" | "<" | "="
Decision is an integer, Positive or Negative implemented
Threshold is a real number
```

Fig. 1. The Backus Normal Form of the ED7

We would also like to draw the reader's attention at the *Variable* symbol of Figure 1; here are the 12 indicators which we mentioned earlier in Table I that ED7 is using. They are pre-specified and should thus be considered as constants of the system. As we will see later, ED8 is not using these constants, but a function instead.

[1] We use these indicators because they have been proved to be quite useful in developing GDTs in previous works like [18], [19] and [20]. Of course, there is no reason why not use other information like fundamentals or limit order book information. However, the aim of this work is not to find the ultimate indicators for financial forecasting.

Each GDT's performance is evaluated by a fitness function, presented here. If the prediction of the GDT is positive (buy-1), and also the signal in the data for this specific entry is also positive (buy-1), then this is classified as True Positive (TP). If the prediction is positive (buy-1), but the signal is negative (not-buy-0), then this is False Positive (FP). On the other hand, if the prediction is negative (not-buy-0), and the signal is positive (buy-1), then this is False Negative (FN), and if the prediction of the GDT is negative (not-buy-0) and the signal is also negative (not-buy-0), then this is classified as True Negative (TN). These four together give the familiar confusion matrix [22], which is presented in Table II.

TABLE II

CONFUSION MATRIX

	Actual Positive	Actual Negative
Positive Prediction	True Positive (TP)	False Positive (FP)
Negative Prediction	False Negative (FN)	True Negative (TN)

As a result, we can use the metrics presented in Equations (1), (2) and (3).

Rate of Correctness

$$RC = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Rate of Missing Chances

$$RMC = \frac{FN}{FN + TP} \tag{2}$$

Rate of Failure

$$RF = \frac{FP}{FP + TP} \tag{3}$$

Li [17] combined the above metrics and defined the following fitness function, presented in Equation (4):

$$ff = w_1 * RC - w_2 * RMC - w_3 * RF \tag{4}$$

where w_1, w_2 and w_3 are the weights for RC, RMC and RF respectively. Li states that these weights are given in order to reflect the preferences of investors. For instance, a conservative investor would want to avoid failure; thus a higher weight for RF should be used. However, Li also states that tuning these parameters does not seem to affect the performance of the GP. For our experiments, we chose to include strategies that mainly focus on correctness and reduced failure. Thus these weights have been set to 0.6, 0.1 and 0.3 respectively.

The fitness function is a constrained one, which allows EDDIE to achieve lower RF. The effectiveness of this constrained fitness function has been discussed in [12], [17]. The

Procedure EDDIE ( )
Begin
Partition whole data into training data and testing data;
/ While training data is employed to train EDDIE to find the best-so-far-rule, the test data is used to determine the performance of predictability of the best-so-far-rule */*
Pop <- InitializePopulation (Pop); */* randomly create a population of GDTs.*/*
Evaluation (Pop); */* calculate fitness of each GDT in Pop */*
Repeat
Pop <- Reproduction (Pop) + Crossover (Pop); */*new population is created after genetic operators of reproduction (which reproduces M*Pr individuals) and crossover (which creates M*(1-Pr) individuals). Pr denotes the reproduction probability and M is the population size */*
Pop <- Mutation (Pop); */*Apply mutation to population */*
Evaluation (Pop); */* Calculate the fitness of each GDT in Pop */*
Until (TerminationCondition( )) */* determine if we have reached the last generation */*
Apply the best-so-far rule to the test data;
End

Fig. 2. Pseudo code for the procedure that EDDIE follows. (Based on [17], p.76)

constraint is denoted by R, which consists of two elements represented by percentage, given by

$$R = [Cmin, Cmax],$$

where $C_{min} = \frac{P_{min}}{N_{tr}} \times 100\%$, $C_{max} = \frac{P_{max}}{N_{tr}} \times 100\%$, and $0 \leq C_{min} \leq C_{max} \leq 100\%$. N_{tr} is the total number of training data cases, P_{min} is the minimum number of positive position predictions required, and P_{max} is the maximum number of positive position predictions required.

Therefore, a constrained of $R = [50, 65]$ would mean that the percentage of positive signals that a GDT predicts[2] should fall into this range. When this happens, then w_1 remains as it is (i.e. 0.6 in our experiments). Otherwise, w_1 it takes the value of zero.

During the evolutionary procedure, we allow three operators: crossover, mutation and reproduction. After reaching the last generation, the best-so-far GDT, in terms of fitness, is applied to the testing data.

Figure 2 summarises what we have said so far, by presenting the pseudo code that the EDDIE algorithms use for their experiments.

This concludes this short presentation of ED7. However, ED7 and its previous versions are considered to have a drawback: nobody can guarantee that the periods chosen for the indicators are the appropriate ones. Why is 12 days MA the right period for a short term period and it is not 10, or 14? As we mentioned earlier, choosing an indicator and as a consequence a period for this indicator, depends on the user of the algorithm and his belief of how helpful this specific indicator can be for the prediction. However, it can be argued that this is subjective and different experts could pick a different period for their indicators. In addition, this choice of indicators limits the patterns that ED7 can discover. This is hence the part of the focus of our research. We believe that allowing EDDIE to search in the space of the periods of the indicators would be advantageous and eliminate any possible weaknesses of the human decision process. For these purposes, we implemented a new version, ED8, which allows the GP to search in the search space of the periods of the indicators. The following section explains how ED8 manages this.

[2]As we have mentioned, each GDT makes recommendations of buy (1) or not-to-buy (0). The former denotes a positive signal and the latter a negative. Thus, within the range of the training period, which is t days, a GDT will have returned a number of positive signals

B. EDDIE 8

Let us consider a function $y = f(x)$, where y is the output, and x is the input. In our case, the input is the indicators and the output is the prediction made by our GP. The function f is unknown to the user and is the GDTs that the algorithm generates, in order to make its prediction. As we just said in the previous section, the input is fixed in ED7; it uses 6 indicators, with 2 different pre-specified periods (12 and 50 days). This limits ED7's capability in finding patterns that cannot be expressed in its vocabulary. ED8 uses another function $y = f(g(z))$, where $x = g(z)$; in other words, g is a function that generates indicators and periods for EDDIE to use. ED8 is not only searching in the space of GDTs, but also in the space of indicators. It can thus return Genetic Decision Trees (GDTs) that are using any period within a range that is defined by the user.

```
<Tree> ::= If-then-else <Condition> <Tree> <Tree> | Decision
<Condition> ::= <Condition> "And" <Condition> |
                <Condition> "Or" <Condition> |
                "Not" <Condition> |
                VarConstructor <RelationOperation> Threshold
<VarConstructor> ::= MA period | TBR period | FLR period |
                     Vol period | Mom period | MomMA period
<RelationOperation> ::= ">" | "<" | "="
Terminals:
        MA, TBR, FLR, Vol, Mom, MomMA are function symbols
        Period is an integer within a parameterised range, [MinP, MaxP]
        Decision is an integer, Positive or Negative implemented
        Threshold is a real number
```

Fig. 3. The Backus Normal Form of ED8

As we can see from the new syntax at Figure 3, there is no such thing as a *Variable* symbol in ED8. Instead, there is the *VarConstructor* function, which takes two children. The first one is the indicator, and the second one is the Period. Period is an integer within the parameterised range [MinP, MaxP] that the user specifies.

As a result, ED8 can return decision trees with indicators like 15 days Moving Average, 17 days Volatility, etc. The period is not an issue anymore, and it is up to ED8, and as a consequence up to the GP and the evolutionary process, to decide which lengths are more valuable for the prediction.

The immediate consequence of this is that now our new version is not restricted only to the 12 indicators that ED7 uses (which are still part of ED8's search space); on the contrary, it now has many more options available, thanks to this enlarged search space.

III. Methodology

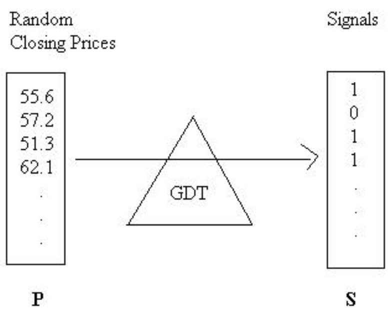


Fig. 4. Methodology for creating an artificial dataset. The random closing prices (P) use a GDT previously derived by EDDIE, in order to create the set of signals S.

As we said earlier, in order to evaluate the performance of ED7 and ED8, we test them by using an artificial dataset. The advantage of using such a dataset, as mentioned in Section I is twofold: first of all, we can make sure that patterns exist in our data. In this way, it is meaningful for EDDIE to attempt to make forecasts. In addition, we have control over the nature of these patterns. This is very important, because it enables us to study the weaknesses and strengths of the algorithms, i.e. in what kind of data would ED7 or ED8 perform better. We do not neglect the fact that it would also be interesting to compare the two algorithms under real data; this is actually something we have done, and will be reported in another occasion. Here we want to focus on testing in artificial datasets, for the reasons just mentioned. Let us now continue by explaining our methodology for creating the datasets.

It was explained earlier that in traditional experiments for EDDIE [10], [11], [23], a dataset would consist of three parts: the daily closing prices, the technical indicators, and the buy/not-to-buy signals. In order to create the artificial data set, we need to replicate these three parts. First of all, we generate a set of random prices, which is represents the daily closing prices. We then calculate the technical indicators for this set. Finally, in order to create the signals for the random prices, we apply to them a GDT that was previously evolved with EDDIE. After the application of the GDT, a new set of signals is created. Basically the difference here from the traditional approach is that we do not use the question "will the price of the stock increase by $r\%$ in the next n days". The signals are created in a new way, based on a given GDT, which should be considered as a hidden function; ED7 and

TABLE III
EDDIE Parameters

EDDIE Parameters	Value
R	[50,65]
n	20
r	4
period	[2,65]

ED8 are therefore asked to rediscover this hidden function. Therefore, after these three steps, we create a dataset like the ones EDDIE uses for its traditional experiments. Figure 4 shows the procedure we have just explained. The first column is the random prices, which is fed into a GDT for generating a set of signals.

It should also be mentioned that the evolved GDT which acts as the hidden function could be obtained either from ED7 or from ED8. In this way, the patterns could come from ED7's search space only, or from a larger search space (ED8). As mentioned above, this is the strength of this approach. Not only are we sure that patterns exist in our dataset, we are also able to determine which search space these patterns come from. We come back to the argument we mentioned at the begin of this paper, that having an artificial dataset allows us to control the nature of the patterns. And of course, being able to control the nature of the patterns allows us to observe the differences in the behaviour of the two versions.

Finally, let us introduce some important terminology. As mentioned, the evolved GDT which acts as the hidden function can be obtained either from ED7 or from ED8. Thus, when it is obtained by ED7, this GDT is called GDT-7, whereas when it is obtained by ED8, this GDT is called GDT-8. In addition, when we present results from ED7, we are going to denote these results as EDDIE 7_GDT-7, if the patterns come from ED7's search space, or EDDIE 7_GDT-8, if the patterns come from EDDIE 8's space. Equivalently, ED8's results will be denoted either as EDDIE 8_GDT-7 or EDDIE 8_GDT-8, depending on which search space the patterns come from.

IV. Experiments

The aim of the experiments is to test the impact of searching a larger space. Therefore, we have limited our comparison between ED8 (large space) and ED7 (smaller space).

As we said in the previous section, the prices of the data were randomly generated. The training period was 1000 days and the testing period 300.

Moreover, Table III presents the parameters of the our algorithm. R is set in the range of [50,65], with n and r being 20 days and 4%, respectively. The last entry of this Table, $period$, refers to ED8 and the range of the indicators' periods; it is set in the range of 2-65 days.

The GP parameters are also presented at Table IV. The values of these parameters are the ones used by Koza [4]. The results seem to be insensitive to these parameters. For

TABLE IV
GP PARAMETERS

GP Parameters	Value
Max Initial Depth	6
Max Depth	17
Generations	50
Population size	500
Tournament size	6
Reproduction probability	0.1
Crossover probability	0.9
Mutation probability	0.01

statistical purposes, we run the GP for 50 times. We then calculate the averages of our performance measures over these 50 runs and we present them in the next session.

Finally, we should mention that a single run of either version does not last for more than a few minutes. ED8 is slightly slower than ED7 of course, due to its big search space, but this fact does not seem to significantly affect its runtime.

V. RESULTS

This section is divided into two parts. The first part presents the results for signals generated by GDT-7, and the second one results for signals generated by GDT-8. We should also say that apart from the main metrics RC, RMC, and RF (Equations (1), (2) and (3) above), we also use two additional performance metrics: Average Annualised Rate of Return (AARR), and Rate of Positive Return (RPR). However, as these two metrics are not part of the fitness function, they should are only used as a reference. The formulas for these two additional metrics are presented in the Appendix.

TABLE V
SUMMARY RESULTS FOR THE TESTING PERIOD, OVER 50 RUNS, FOR ED7 AND ED8. THE PATTERNS WERE CREATED BY GDT-7. THE RESULTS ARE SHOWN IN % PERCENTAGES.

EDDIE 7_GDT-7					
	RC	RMC	RF	AARR	RPR
Mean	97.36	2.4	1.3	47691.23	90.42
St.Dev.	1.58	1.97	1.51	1146.75	0.2
Max	99.66	4.8	6.36	49246.4	90.73
Min	94.66	0.4	0	100	90.1
EDDIE 8_GDT-7					
	RC	RMC	RF	AARR	RPR
Mean	77.66	17.11	15.36	46730	90.48
St.Dev.	10.28	10.75	6.8	4885.08	1.03
Max	91	35.57	25.32	51648.92	92.85
Min	63.66	2.4	6.03	34138.56	88.29

A. GDT-7

Table V presents the summary results for the testing period over 50 GP runs. As we can observe, EDDIE 7_GDT-7 (ED7 with patterns that have been created by GDT-7) is doing significantly better in all performance measures and is very close finding a perfect solution[3] (RC=97.36, RMC=2.4, RF=1.3). It is also interesting to observe that the standard deviation of EDDIE 7_GDT-7's results is small, which basically indicates that the values for RC, RMC and RF are very similar among the 50 runs. This however does not happen with EDDIE 8_GDT-7 (ED8 with patterns that have been created by GDT-7), where the standard deviation is bigger for all RC, RMC and RF. As we can also see from Table V, the values of all RC, RMC and RF have worsen to 77.66, 17.11 and 15.36, respectively. Furthermore, we can also observe that the Min and Max values of the above three metrics are in a much bigger range for ED8. Also, ED7 has higher AARR, whereas the RPR is quite similar, for both ED7 and ED8.

In addition, Figure 5 presents the training fitness of two GDTs, one for EDDIE 7_GDT-7 (Figure 5a) and one for EDDIE 8_GDT-7 (Figure 5b), over 50 generations. The individuals chosen for this observation were the ones that had the highest performance[4] at the testing period, among all 50 runs. To be more specific, each time we train ED7 or ED8, the evolutionary procedure returns the best-so-far individual (GDT); at the end of the 50 generations, this GDT is tested against a testing dataset and returns a performance. This procedure happened for 50 times, for both ED7 and ED8. We then chose the best GDT from ED7 and ED8, in terms of its performance. As we can see, EDDIE 7_GDT-7 comes very fast to a solution, which is actually very close to the optimal one (i.e. $fitness = 1$). On the other hand, EDDIE 8_GDT-7 does not seem to reach to fitness levels as high as EDDIE 7_GDT-7 does. It only manages to reach around 80%, which is quite high, but not as high as EDDIE 7_GDT-7's.

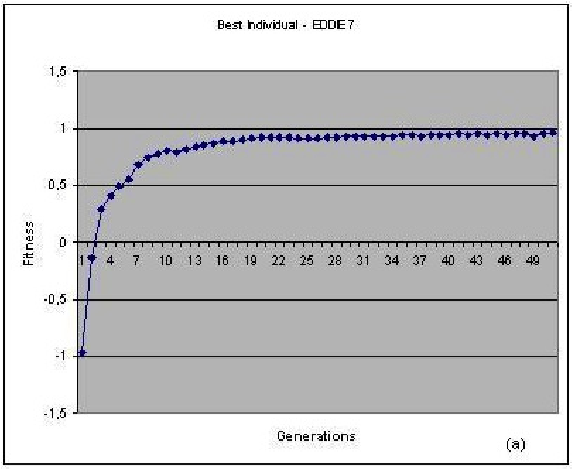
The poor results could be explained by the exponential increase in the search space of EDDIE 8_GDT-7. For this reason, we tested EDDIE 8_GDT-7's performance with a bigger population (1500 individuals) and more generations (100). The reasoning in this was that because of the big search space, EDDIE might have needed more candidate solutions or more time in order to perform better. However, as we can see from Table VI, EDDIE 8_GDT-7's summary results did not seem to have any significant improvement (mean of RC was improved from 77.66 to 78.72, mean of RMC improved from 17.11 to 15.86 and mean of RF 15.36 to 14.91).

B. GDT-8

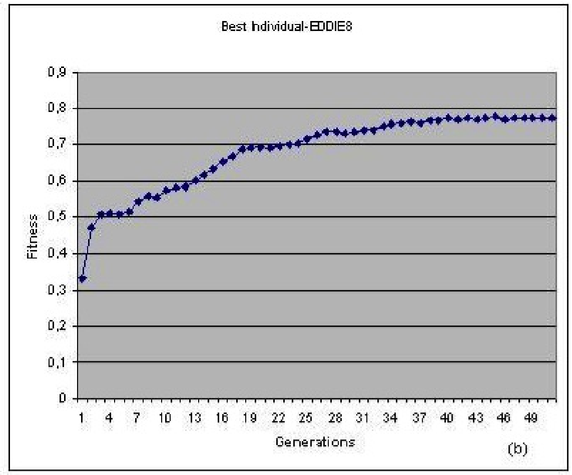
The results in this section are quite different. As we can see from Table VII, none of EDDIE 7_GDT-8 (ED7 with patterns that have been created by GDT-8) or EDDIE 8_GDT-8 (ED8 with patterns that have been created by GDT-8) seem to be able to find solutions very close to the optimal one. In addition, this time EDDIE 8_GDT-8 is performing better

[3]A perfect solution can be defined as any GDT that fits the testing dataset perfectly. This essentially means that RC would be 100%, and RMC=RF=0

[4]Performance is equivalent to fitness



(a) EDDIE 7



(b) EDDIE 8

Fig. 5. Training fitness of a single individual for EDDIE 7_GDT-7 [figure 5(a)] and EDDIE 8_GDT-7 [figure 5(b)]. The individuals presented here are the ones that had the highest performance during the testing period, among all 50 runs.

TABLE VI
SUMMARY RESULTS FOR THE TESTING PERIOD, OVER 50 RUNS, FOR EDDIE 8_GDT-7. THE RESULTS ARE SHOWN IN % PERCENTAGES. THE NUMBERS OF GENERATIONS AND POPULATION HAVE CHANGED TO 100 AND 1500, RESPECTIVELY.

EDDIE 8_GDT-7					
	RC	RMC	RF	AARR	RPR
Mean	78.72	15.86	14.91	47460.18	90.74
St.Dev.	7.67	10.48	4.35	4857.71	0.9
Max	91.33	35.57	21.21	57879.55	93.29
Min	64.66	0.48	6.17	38344.04	89.44

than EDDIE 7_GDT-8, in terms of summary statistics (all RC, RMC and RF are better). Furthermore, EDDIE 8_GDT-8's maximum value for RC (92.67) and minimum values for RMC (8.25) and RF (0) are significantly better than the ones of EDDIE 7_GDT-8 (74.33, 29.13 and 19.42 respectively). Finally, ED8's AARR is significantly better; RPR is also slightly better for ED8.

In order to see whether the difference in the performance measures is indeed significant, we run a two-sample Kolmogorov-Smirnov non-parametric test. The null hypothesis is that the two samples come from the same continuous distribution; it is rejected if the value obtained by the test is greater than the critical value. Table VIII shows us that H_0 is rejected for all performance measures at 5% significance level. The critical value at this significance level is 0.136.

VI. DISCUSSION

From the above experiments, we have shown that both ED7 and ED8 have been able to rediscover the hidden functions (see Figure 4). This is very important and proves the effectiveness of these two methods. Also, it should not be considered as something trivial, since it cannot be assumed that other, arbitrary methods would be able to do this.

TABLE VII
SUMMARY RESULTS FOR THE TESTING PERIOD, OVER 50 RUNS, FOR ED7 AND ED8. THE PATTERNS WERE CREATED BY GDT-8. THE RESULTS ARE SHOWN IN % PERCENTAGES.

EDDIE 7_GDT-8					
	RC	RMC	RF	AARR	RPR
Mean	72.09	23.49	18.30	22569.97	90.02
St.Dev.	72.0996	23.495	18.30	22569.97	190.02
Max	74.33	29.13	20.30	34438.01	92.27
Min	68.33	19.42	16.67	10000	88.71
EDDIE 8_GDT-8					
	RC	RMC	RF	AARR	RPR
Mean	81.91	21.14	5.83	34741.70	91.14
St.Dev.	5.48	3.73	6.52	4547.87	0.70
Max	92.67	32.04	20.85	48613.98	92.07
Min	68.33	8.25	0	10000	88.73

TABLE VIII
KOLMOGOROV-SMIRNOV TEST FOR TESTING WHETHER THE DIFFERENCES BETWEEN ED7 AND ED8 ARE SIGNIFICANT AT 5% SIGNIFICANCE LEVEL. THE CRITICAL VALUE IS 0.136.

	RC	RMC	RF	AARR	RPR
K-S test	0.84	0.54	0.88	0.9	0.56

Furthermore, our work has also shown that ED8 has a value over ED7. The reason for this is first of all because it has richer grammar, which allows to search in an extended space. As a result, ED8 is able to discover functions that its predecessor cannot.

However, our analysis also showed that ED8 cannot always perform better than ED7. It seems that there is a trade-off between 'searching in a bigger space' and 'search effectiveness'. It is obvious that the results are affected by the patterns in the dataset. If these patterns come from ED8's search space, ED8 can find better solutions. This is something we anticipated, since ED7 cannot search for these solutions. From Figure 6, a look into the components of the trees that ED8 used during the evolutionary procedure of a single run would show us that ED8 indeed took advantage of its big search space and came up with solutions that it is impossible for ED7 to find. The x-axis of this figure presents the range of the periods (2-65 days) that the 6 technical indicators are using. The y-axis shows the occurrence of these indicators, in the logarithmic scale, after 50 generations of a single run. As we can see, all indicators are used and they use many different periods within the range of 2-65 days.

However, a question arises, whether just using a bigger number of indicators is enough to get better prediction results. This point becomes even clearer in cases where the patterns in the dataset come from a very small search space, like the one of ED7's. It then seems very hard for ED8 to find as good solutions as ED7 does. The solutions are indeed in its search space, but because they come from a very small area of it, it seems that ED8 cannot search effectively enough to find them. The search space has increased exponentially and there is an obvious trade-off between the more expressive language that ED8 provides and the search efficiency of ED7.

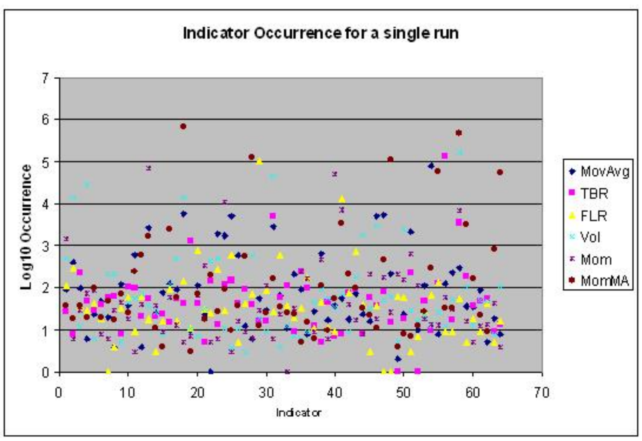


Fig. 6. Indicators occurrence after 50 generations for a single run. This occurrence is presented in the y-axis and is in a logarithmic scale with a base of 10. The range of the period for the 6 indicators is from 2 to 65 days, and is presented in the x-axis. There are 6 different colours in the graph, each one denoting a different technical indicator.

Future research should focus on finding new operators that would allow EDDIE to search the search space more effectively. EDDIE 8_GDT-7 performed well, but there is no reason why it should not perform as well as EDDIE 7_GDT-7 did. Therefore, the new operators should allow ED8 to perform always at least as good as ED7. Furthermore, another path that could be followed could be a constrained fitness function, which would improve ED8's search effectiveness. Finally, we also intend to present comparative results of the two algorithms under empirical data, so that we can generalize our conclusions from this paper.

VII. Conclusion

In this paper we presented EDDIE 7 (ED7) and EDDIE 8 (ED8); the former is a re-implementation of previous EDDIE versions, whereas the latter is a new version, which has an extended search space and allows the GP to search in the space of technical indicators. We then presented the results of our experiments, after comparing them under an artificial dataset, in which we know patterns exist. These patterns could contain indicators that are in the vocabulary of ED8 or in the vocabulary of ED7. In the first instance, where patterns contain indicators that appear in the vocabulary of ED8, ED8 performs better. However, should all patterns contain indicators that appear only in the vocabulary of ED7, then ED7 can outperform ED8. It seems that ED8 is having difficulties in searching effectively in this case. Future research could focus on improving the search efficiency of ED8.

Appendix

A. Technical Indicators

The following section presents the technical indicators that the GP is using, along with their formulas. We performed a sort of standardization in order to avoid to have a very big range of numbers generated by GP, because this would increase the size of the search space even more. Given a price time series $[P(t),\ t \geq 0]$, and a period of length L, Equations (5), (6), (7), (8), (9) and (10) present these formulas.

Moving Average (MA)

$$\text{MA(L,t)} = \frac{P(t) - \frac{1}{L}\sum_{i=1}^{L} P(t-i)}{\frac{1}{L}\sum_{i=1}^{L} P(t-i)} \quad (5)$$

Trade Break Out (TBR)

$$\text{TBR(L,t)} = \frac{P(t) - max\{P(t-1), \ldots, P(t-L)\}}{max\{P(t-1), \ldots, P(t-L)\}} \quad (6)$$

Filter (FLR)

$$\text{FLR(L,t)} = \frac{P(t) - min\{P(t-1), \ldots, P(t-L)\}}{min\{P(t-1), \ldots, P(t-L)\}} \quad (7)$$

Volatility (Vol)

$$\text{Vol(L,t)} = \frac{\sigma(P(t), \ldots, P(t-L+1))}{\frac{1}{L}\sum_{i=1}^{L} P(t-i)} \quad (8)$$

Momentum (Mom)

$$\text{Mom(L,t)} = P(t) - P(t-L) \quad (9)$$

Momentum Moving Average (MomMA)

$$\text{MomMA(L,t)} = \frac{1}{L}\sum_{i=1}^{L} Mom(L, t-i) \quad (10)$$

B. Additional Performance Measures

Here we present the formulas for the two additional metrics AARR and RPR, as presented in [17]. We would once again like to remind the reader that these metrics should be used for reference only, since they are not part of the fitness function.

***Hypothetical Trading Behaviour**: We assume that when a positive position is predicted by a GDT, one unit of money is invested in a stock reflecting the current closing price. If the closing price does rise by r% or more at day t within the next n trading days, we then sell the portfolio at the closing price of day t. If not, we sell the portfolio on the n_{th} day, regardless of the price.*

Given a positive position predicted, for example, the i_{th} positive position, for simplicity, we ignore transaction cost, and annualise its return by the following formula, presented in Equation (11):

$$ARR_i = \frac{255}{t} * \frac{P_t - P_0}{P_0} \quad (11)$$

Where P_0 is the buy price, P_t is the sell price, t is the number of days in markets, 255 is the number of total trading days in one calendar year. Given a GDT that generates N_+ number of positive positions over the period examined, its average ARR is shown in Equation (12):

$$AARR = \frac{1}{N}\sum_{i=1}^{N_+} ARR_i \quad (12)$$

RPR (Equation (13)) refers to the ratio of the number of signals, which turn out to achieve positive returns, to the total number of positive positions predicted, where a specific GDT is invoked for a finite period

$$RPR = \frac{1}{N_+}\sum_{i=1}^{N_+} I_i \quad (13)$$

where

$$I_i = \begin{cases} 1 & if ARR_i \geq 0 \\ 0 & otherwise \end{cases}$$

and

$$0 < i \leq N_+$$

where N_+ is the number of positive positions generated by the GDT, and ARR_i is an annualised rate of return for the i_{th} signal.

References

[1] E. Tsang and S. Martinez-Jaramillo, "Computational finance," *IEEE Computational Intelligence Society Newsletter*, pp. 3–8, 2004.

[2] S.-H. Chen, *Genetic Algorithms and Genetic Programming in Computational Finance*. Springer-Verlag New York, LLC, 2002.

[3] V. Sharma and D. Srinivasan, "Evolutionary computation and economic time series forecasting," in *Proceedings of the IEEE Conference on Evolutionary Computation*, Singapore, 25-28 September 2007, pp. 188–195.

[4] J. Koza, *Genetic Programming: on the programming of computers by means of natural selection*. Cambridge, MA: MIT Press, 1992.

[5] R. Poli, W. Langdon, and N. McPhee, *A Field Guide to Genetic Programming*. Lulu.com, 2008.

[6] M. Bernal-Urbina and A. Flores-Méndez, "Time series forecasting through polynomial artificial neural networks and genetic programming," in *Proceedings of the IEEE Conference on Evolutionary Computation*, Hong Kong, 1-6 June 2008, pp. 3324–3329.

[7] H. Xie, M. Zhang, and P. Andreae, "Genetic programming for new zealand CPI inflation prediction," in *Proceedings of the IEEE Conference on Evolutionary Computation*, Singapore, 25-28 September 2007, pp. 2538–2545.

[8] Y. Chen, S. Mabu, K. Hirasawa, and J. Hu, "Genetic network programming with sarsa learning and its application to creating stock trading rules," in *Proceedings of the IEEE Conference on Evolutionary Computation*, Singapore, 25-28 September 2007, pp. 220–237.

[9] C. Worasucheep, "Trading index mutual funds with evolutionary forecasting," in *Proceedings of the IEEE Conference on Evolutionary Computation*, Hong Kong, 1-6 June 2008, pp. 430–435.

[10] E. Tsang, J. Li, S. Markose, H. Er, A. Salhi, and G. Iori, "EDDIE in financial decision making," *Journal of Management and Economics*, 2000.

[11] J. Li and E. Tsang, "Improving technical analysis predictions: an application of genetic programming," in *The 12th International FLAIRS Conference*, USA, 1999, pp. 108–112.

[12] E. Tsang, S. Markose, and H. Er, "Chance discovery in stock index option and future arbitrage," *New Mathematics and Natural Computation, World Scientific*, vol. 1(3), pp. 435–447, 2005.

[13] J. Li and E. Tsang, "Reducing failures in investment recommendations using genetic programming," in *The 6th International Conference on Computing in Economics and Finance, Society for Computational Economics*, Barcelona, 2000.

[14] R. Edwards and J. Magee, "Technical analysis of stock trends," *New York Institute of Finance*, 1992.

[15] M. O'Neill, T. Brabazon, C. Ryan, and J. Collins, "Evolving market index trading rules using grammatical evolution." in *Proceedings of EvoIASP*, 2001.

[16] I. Dempsey, M. O'Neill, and A. Brabazon, "Live trading with grammatical evolution," in *Proceedings of the Grammatical Evolution Workshop*, 2004.

[17] J. Li, "FGP: A genetic programming based financial forecasting tool," Ph.D. dissertation, Department of Computer Science, University of Essex, 2001.

[18] S. Martinez-Jaramillo, "Artificial financial markets: An agent based approach to reproduce stylized facts and to study the red queen efect," Ph.D. dissertation, CFFEA, University of Essex, 2007.

[19] F. Allen and R. Karjalainen, "Using genetic algorithms to find technical trading rules," *Journal of Financial Economics*, vol. 51, pp. 245–271, 1999.

[20] M. Austin, G. Bates, M. Dempster, V. Leemans, and S. Williams, "Adaptive systems for foreign exchange trading." *Quantitative Finance*, vol. 4(4), pp. 37–45, 2004.

[21] J. Backus, "The syntax and semantics of the proposed international algebraic language of Zurich," in *International Conference on Information Processing*. UNESCO, 1959, pp. 125–132.

[22] F. Provost and R. Kohavi, "Glossary of terms," *Journal of Machine Learning*, vol. 30(2-3), pp. 271–274, 1998.

[23] E. Tsang, J. Li, and J. Butler, "EDDIE beats the bookies," *International Journal of Software, Practice & Experience, Wiley*, vol. 28(10), pp. 1033–1043, 1998.