2018 IEEE International Conference on Big Knowledge

Stochastic Optimization for Market Return Prediction Using Financial Knowledge Graph

Xiaoyi Fu[1], Xinqi Ren[1], Ole J. Mengshoel[2] and Xindong Wu[1]
[1]MAS Academy of Sciences
Mininglamp Software Systems, Beijing, China
[2]Dept. of Electrical and Computer Engineering
Carnegie Mellon University
[1]{fuxiaoyi,renxinqi,wuxindong}@mininglamp.com
[2]ole.mengshoel@sv.cmu.edu

***Abstract*— Interactive prediction of financial instrument returns is important. It is needed for asset managers to generate trading strategies as well as for stock exchange regulators to discover pricing anomalies. In this paper, we introduce an integrated stochastic optimization technique, namely genetic programming (GP) with generalized crowding (GC), GP+GC. GP+GC is as an integrated method for market return prediction, using a financial knowledge graph (KG). On the one hand, using time-series data for twenty-nine component stocks of the Dow Jones industrial average, we show that our stochastic optimization method can give strong prediction performance by providing a comparison of its return performances with two traditional benchmarks, namely a Buy & Hold strategy and the Moving Average Convergence Divergence (MACD) technical indicator. On the other hand, we use features extracted from a time-evolving knowledge graph constructed from fifty component stocks of the Shanghai Stock Exchange SSE50 index. These features are used by our GP+GC variant and then expression learnt by GP+GC are extracted into a KG. Overall, this work demonstrates how to integrate GP+GC with KGs in a powerful manner.**

Keywords*— *Genetic Programming, Generalized Crowding, Stochastic Optimization, Knowledge Graph, Finance

I. Introduction

Understanding stock market trends plays a vital role in the daily work of asset managers as well as stock exchange regulators. For example, the China Securities Regulatory Commission monitors hundreds of thousands of stocks trades in real-time through a supervision platform deployed at Shanghai Stock Exchange. The goal is to discover abnormal stocks and accounts. A human-readable prediction model is essential to such an interactive system, to support exploration and investigation of abnormal events. Nevertheless, the problem of predicting stock prices is very challenging, as the stock market behavior is often chaotic and volatile[14]. Various techniques based on genetic algorithms and neural networks have been proposed to forecast stock market prices [1][2][3][5][6][8]. Genetic programming is among the most promising approaches to explainable stock prediction due to several specific properties, which we now highlight.

Genetic programming (GP) algorithms can find a symbolic structure model that characterizes the dynamic behavior of a sequential (or time-series) data set [4][17]. Unfortunately, the performance of GP algorithms, and evolutionary computation more generally, is often reduced due to premature convergence [18]. In previous research, niching techniques, including crowding techniques [9][10][15][16][25], have been successfully employed to counter-act such premature convergence. However, these techniques have primarily been investigated for genetic algorithms (GAs), not for GP. In this paper, we integrate generalized crowding (GC) [9] with GP. This creates what we believe is a novel integration of GP and generalized crowding. This integrated approach, which we denote GP+GC, maintains population diversity to improve prediction performance. Furthermore, we show the synergy of GP+GC with an interactive system built on a financial knowledge graph (KG).

In experiments, we validate the performance of our novel GP+GC stochastic optimization technique on market return prediction using stock market time-series data. We use Moving Average Convergence Divergence (MACD) and Buy & Hold as baselines. By introducing features from a financial knowledge graph built on 50 stocks traded in Shanghai Stock Exchange, we also give a statistical answer to the following question: Which stocks cause the index to deviate from normal and what are the correlations between key factors? We answer this question by means of knowledge extracted from expressions learned by our GP+GC method.

The rest of this paper is organized as follows. In Section 2 we provide a brief overview of previous work on financial data mining, with emphasis on market return forecasting and financial knowledge graphs. In Section 3 we discuss the GP model as well as its performance on a real data set with and without our integration of GC. In Section 4 we experimentally compare the performance of GP-based algorithms in market return prediction on the return of investment (ROI) with a benchmark technical analysis strategy. Then we show how the GP-based model provides a synthesis approach for prediction of financial investment return, using a knowledge graph with application to market volatility attribution. In Section 5 we give brief concluding remarks.

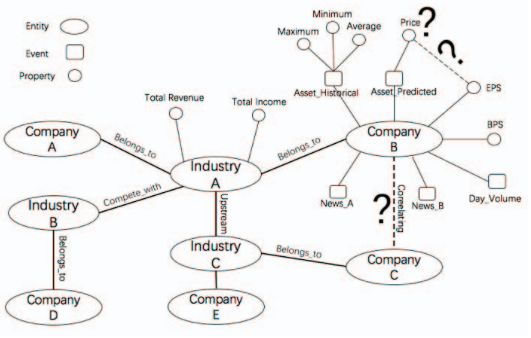

Fig. 1. An example data model for a financial knowledge graph

978-1-5386-9125-0/18/$31.00 ©2018 IEEE
DOI 10.1109/ICBK.2018.00012

II. Previous Work

In this section, we review previous work on knowledge graphs and financial prediction models separately.

A. Financial Knowledge Graph

The idea of knowledge graph goes back in time: in artificial intelligence and psychology there was related research back in the 1960s, 1970s, 1980s, and 1990s under titles such as semantic memory, semantic networks, knowledge-based systems, knowledge bases, semantic web, and so forth.

At Mininglamp, we construct a time-evolving financial knowledge graph of stocks traded on the Shanghai Stock Exchange (SSE), using heterogeneous information sources including public sentiments, public disclosures, industrial macro data, financial ratios of companies traded, as well as daily stock prices and weekly price statistics. Raw Chinese financial data are downloaded from the WIND Financial Terminal (WFT) and processed into a graph schema, see Figure 1. Different from other financial knowledge graphs [24], we use vendor-provided attributes including sentiment classification and financial ratios (thus, natural language processing tasks such as news sentiment classification and information retrieval from financial fillings are out of the scope of this paper).

The usage of knowledge graph as a default data model benefits our prediction system in at least three ways. First, it enriched the feature pool of our model thus, the overall robustness of the prediction is enhanced. Second, the knowledge graph enables explainable feature extraction by representing a subgraph as a symbolic structure when training GP models. (This is not studied in this paper, but constitutes an interesting topic for future study.) Finally, as shown in Figure 2, our knowledge graph visualization platform (SCOPA) enables interactive presentation of event data to human analysts in near real time. SCOPA is incorporating knowledge about correlations extracted from expressions learned by our GP model back into the knowledge graph.

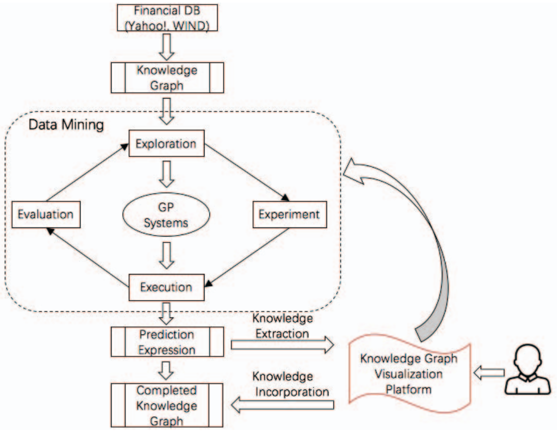

Fig. 2. An interactive system for market prediction and price anomaly discovery using financial knowledge graph

As result, prediction results and expressions learned augment the knowledge graph, both as entity properties and relations. Also, by showing statistical histories and correlations between graph components, the interactive system provides evidence for data mining.

B. Financial Market Prediction

We now discuss some previous research on predicting financial investment returns.

Genetic programming (GP) has been applied successfully in the field of stock price analysis [12][13][21][28]. GP benefits from direct explanation via complex expressions that the GP model learns. Such explanation is crucial to the problem we face. Allen and Karjalainen used GP to generate technical trading rules for the S&P 500 index, using daily prices from 1928 to 1995 [19]. They found that when transaction costs were considered, the rules did not earn consistent returns on investments over a simple Buy & Hold strategy in the out-of-sample test periods. Potvin et al. [20] carried out experiments that tested the viability of GP-based trading rules against a simple Buy & Hold strategy for 14 Canadian companies listed on the Toronto Stock Exchange, and showed that the trading rules generated by GP are generally beneficial when the market falls or when it is stable. On the other hand, these rules do not match the Buy & Hold approach when the market is rising. Grosan and Abraham have applied a linear GP (LGP) to stock market analysis and found that an LGP hybrid (with multi-expression systems) outperformed neural networks and neuro-fuzzy systems for inter-day prediction of stock prices for the NASDAQ and Nifty indices [17]. Standard GP-generated trading rules have been tested on 30 stocks from the Dow, resulting in more than 100% excess returns compared to a Buy & Hold strategy in a bearish market [8]. Excessive returns over market indices have also been obtained using a computational system that combines a conventional market analysis method (technical analysis), genetic programming, and multi-objective optimization tested in six historical time series of representative assets from the Brazil stock exchange market (BOVESPA) [27].

A variety of explainable models have also been investigated for time series data analysis more broadly. For example, grammar-based decision tree [4] is proposed to learn models that are human interpretable, with a focus on mining of heterogeneous multivariate data such as aircraft and sign language time series. Recently, progress has been made in applying deep neural network architectures such as LSTM [26] to stock price prediction. However, as a black box model, neural networks [23] are limited in our practical use, because it is hard to explain the model an ANN has learned.

All these previous efforts, while testing many different algorithms for prediction in interesting ways, give no clear picture of the relationships between profitability and prediction accuracy, nor human readable interpretation for model learned by algorithms, particularly based on a financial knowledge graph. Based on previous work, stochastic optimization variants of GP are now introduced and experiments are conducted using features picked from heterogeneous financial knowledge expressed as a data model [11]. Tests are performed on time series stock price forecasting for two different markets, and we discuss the relationships between profitability, market volatility, and prediction accuracy.

III. AGORITHMS

Evolutionary algorithms have been a powerful tool in financial forecasting [22]. We focus in this paper on time-series prediction of stock prices using genetic programming (GP), and investigate empirically the difference in performance between (i) GP without generalized crowding and (ii) GP with generalized crowding [9]. As our starting point, we use a standard GP framework [7].

A. Generalized Crowding

Crowding is a technique for survivor selection in genetic algorithms introduced by De Jong [25]. Crowding's goal is (i) to preserve diversity in the population to prevent premature convergence to local optima or (ii) to find a diverse set of (local and global) optima. Crowding consists of two phases [9] [10]: a pairing phase and a replacement phase. In the paring phase, children are paired with their parents (in the current population) per a similarity metric. In the replacement phase, a decision is made for each pair of individuals as to which of them will remain in the next generation's population.

Traditionally, there are three approaches for the replacement phase: deterministic, probabilistic, and simulated annealing. Generalized crowding is a relative new technique [9] for the replacement phase; it is discussed in more detail in Section III.C.

B. Evolutionary Operations

Our mutation and crossover operations are based on the literature. Point mutation mutates a single node in a copy of a tree from the prior population before the mutated tree is being added to the next generation. Branch mutation mutates a branch in a copy of a tree from the prior population before it is being added to the next generation. For crossover, 2 trees are selected as parents to produce 2 offspring. Within each parent tree a branch is selected. For child A, parent A is copied, with its selected branch deleted. Parent B's branch is then copied to the former location of parent A's branch, and inserted. This process is reversed for child B.

The original generalized crowding approach [9] was developed for genetic algorithms, where the distance metric used to match a parent and a child is defined as the Hamming distance between the two genomes. In our GP+GC algorithm, Hamming distance cannot be applied in a straightforward way. Instead, the difference in the number of nodes between the trees is used as a distance metric in our experiments. More complex similarity metrics [29] are left for future investigations.

C. The Basic Procedures

The *deterministic crowding* approach of Mahfoud [16] includes the following steps:

Step 1: The individuals (bitstrings) in the current population are randomly paired.

Step 2: With probability Pc, the parents in each pair (p_1, p_2) is recombined. The two resulting children (c_1, c_2) are mutated with probability P_M.

Step 3: Each child competes with one of its two parents to be included (as new parents q_1 and q_2) in the population of the next generation. Let $d(i_1, i_2)$ denote the (Hamming) distance between two individuals, i_1 and i_2.

If $d(p_1,c_1) + d(p_2,c_2) < d(p_1,c_2) + d(p_2,c_1)$, q_1 becomes the winner among p_1 and c_1, q_2 becomes the winner among p_2 and c_2; else q_1 becomes the winner among p_1 and c_2, q_2 becomes the winner among p_2 and c_1.

In *generalized crowding*, the replacement phase is further divided into two steps: first, matching parent-child pairs are computed, minimizing a distance metric. Then, local tournaments are held by means of a replacement rule. A detailed pseudo-code description of how generalized crowding is integrated into GP is shown in Figure 3. The replacement rule in *generalized crowding* is based on a scaling factor φ, which allows a broad range of replacement rules to be used by simply varying φ. Specifically, the probability of a child being decided as the winner of a competition between a matched parent p and a child c is given by:

$$P(child) = \begin{cases} \frac{f(c)}{f(c)+\varphi\times f(p)} & if\ f(c)>f(p) \\ 0.5 & if\ f(c)=f(p) \\ \frac{\varphi\times f(c)}{\varphi\times f(c)+f(p)} & if\ f(c)<f(p) \end{cases} \quad (1)$$

In (1), $f(c)$ and $f(p)$ denote, respectively, the fitness of the child and the parent. The introduction of φ means that generalized crowding can emulate both deterministic ($\varphi = 0$) and probabilistic ($\varphi = 1$) crowding. More generally, the non-negative φ parameter enables trading off between exploitation (small φ) and exploration (large φ) in a uniform framework.

D. Implementation Details

The GP system uses historical data to evolve a non-linear function that uses market data to predict the stock prices in the coming days. Attributes from the past 5 days are applied to predict the next day and the fitness function is proportional to the accumulated mean squared error of prediction in training periods. The multiplier to prediction error is set to be 10 to promote evolution of GP structures. The mutation rate is set to be 4% as a compromise between previous works [25]. A relatively large population (1000) is chosen [8]. The basic experimental setup is shown in Table I.

When generating new symbolic sub-tree structures, the probability for a sub-tree structure to be a leaf is 25%. The probability of a leaf being a parameter value (the attributes we used as input for each GP regression model) is 50% and the probability of a leaf being a random constant is 50%. More detailed parameters used for experiments are shown in Table II, all of them are optimized within common settings. The following constraints are put on the complexity of the GP structure:

- A maximum buffer size (1000) for the number of tree nodes in GP structure (Genome).
- Linear penalty equal to the number of nodes in GP structure.

Figure 6 shows how tree length can be restricted by a linear penalty in the fitness function.

Fig. 3. Pseudo-code for GP with Generalized Crowding (GP+GC)

```
GP+GC($n,S,P_M,P_C,G_N,F$)
Input: $n$ population size;
$S$ number of parents in tournament;
$P_M$ probability of mutation;
$P_C$ probability of crossover;
$G_N$ number of generations;
$F$ fitness function.
Output: newPop final population of individuals.
Begin
 $G_C = 0$; {initialize current generation counter}
 Randomly initialize a population (oldPop) of GP structure;
 While $G_C < G_N$
   While SIZE(oldPop) > 1
     Evaluate fitness of each individual;
     Randomly select parents from oldPop;
     Update SIZE(oldPop);
     Perform Crossover with Probability $P_C$;
     Perform Mutation with Probability $P_M$;
     Perform generalized crowding:
       Matching parent and child based on distance metric;
       Replacement based on rules discussed in Section IV.A.
       Update newPop;
   End
   Update oldPop with newPop;
   $G_C = G_C + 1$;
 End
 Return newPop
End
```

TABLE I. PARAMETERS OF SUB-TREE STRUCTURES

Buffer Size for Genomes	1000
Max. Size, Randomly Generated Sub-tree	10
Max. Number of Function Parameters	4
Probability of Leaf During Tree Creation	25%
Probability of Using Function instead of Leaf	20%
Probability of Leaf being Parameter Value	50%
Probability of Leaf Being Floating Point Value	34%
Probability Leaf Being Integer Value	16%
Min. Initial Value, Leaf	0
Max. Initial Value, Leaf	10

IV. EXPERIMENTS

A. Profitability Based on Application of Time Series Prediction

The training and testing data sets were downloaded from Yahoo! Finance. The data sets contain five attributes for each stock and each date: the open price, close price, high price, low price, and volume. For time series prediction, we choose Mean, Max, Min of close price and volume of the stock in the previous 5 days as input values, and 3 runs per stock for GP with generalized crowding (with the empirically optimized scaling factors $\varphi = 1.0$, $\varphi = 1.2$, and $\varphi = 1.5$).

The trading rules used in our experiments are:

- If the prediction price of a certain stock index is higher than its average of the past 20 days and the share of this index is held, sell one share.
- Else if the prediction price of a certain stock index is lower than its average of the past 20 days and the share of this index is not held, buy one share.
- Otherwise, do nothing.

Metrics comparing the profitability of trading rules based on GP+GC and traditional technical analysis approaches are calculated and are discussed below. The prediction prices of 29 stocks (out of 30) are reasonably close to their real prices.

As benchmarks, we apply two technical analysis techniques to generate trading strategies: simple Buy & Hold and moving average convergence divergence (MACD) [8].

We test GP with generalized crowding (GP+GC) using the parameters shown in Table I and Table II on the first stock data in the Dow. Three different scaling factors, namely $\varphi = 1.0$, $\varphi = 1.2$, and $\varphi = 1.5$, were tested and results are shown in Figure 6 (the best from 6 runs of each scaling factor setting was chosen). The fitness values and the tree lengths of GP structures for each generation are plotted.

Comparing the curve of GP with or without generalized crowding (having scaling factor $\varphi = 1.2$), we see that GP without generalized crowding tends to keep improving in terms of fitness value in early generations but is beaten by GP+GC in later generations. This suggests that the performance of GP can be restricted by premature convergence, while the generalized crowding of GP+GC can help with preventing premature convergence, and thus achieve a better performance at the end of a run.

TABLE II. BASIC PARAMETERS OF GP WITH GENERALIZED CROWDING USED IN STOCK MARKET TRADING

Terminal Set	Value: Mean, Max, Min of close price and volume of the stock in last 5 days; Real Functions		
GP value	Open price of the predicted day		
Criterion of Fitness (F)	-10* Accumulated MSE + No. of nodes in GP structure		
Elitism	Yes	**No. of Parents in tournament(S)**	2
Crossover Probability (P_C)	95%	**Population Size (n)**	1000
Mutation Probability (P_M)	4%	**Generation (G_N)**	100
Training Data Set	January 2, 2003 to December 30, 2005 of e.g. AIG		
Testing Data Set	January 3, 2005 to December 29, 2006 of e.g. AIG		
Complexity Penalty	Number of nodes in GP structure		

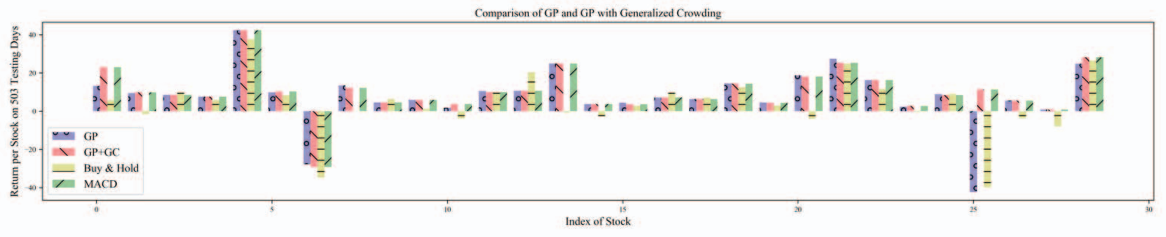

Fig. 4. Return per stock performance for GP and GP with Generalized Crowding

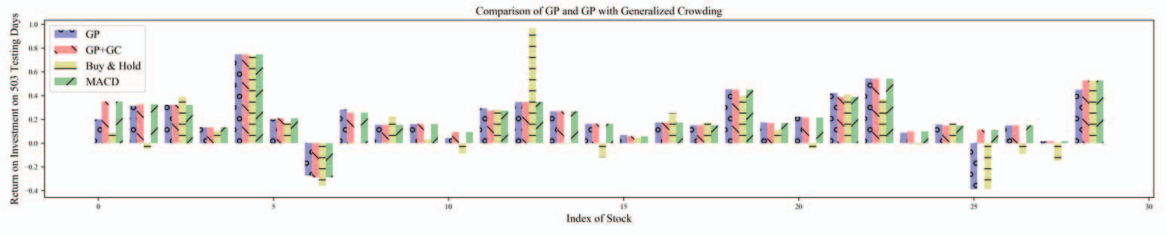

Fig. 5. ROI performance for GP and GP with Generalized Crowding

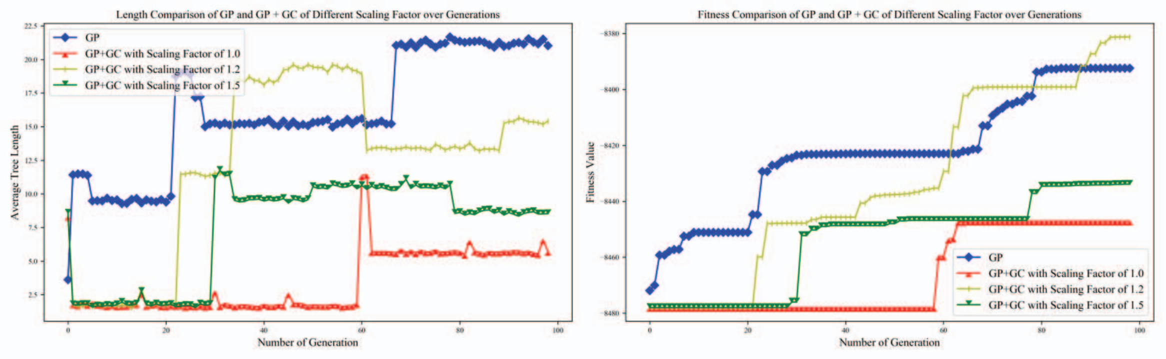

Fig. 6. Comparison of fitness value and average tree length

Figure 4 and Figure 5 show, respectively, the returns on investment and total returns on the testing period for all 29 stocks of all four strategies including GP, GP+GC, Buy & Hold, and MACD. As we can see from the trend in the above figures, GP-based trading rules outperform Buy & Hold or MACD for most of the stocks, while statistically, GP+GC often performs better than GP.

From a statistical perspective, the mean, deviation and sharp ratio based on a risk-free ROI of 0.05 for GP, GP+GC, and Buy & Hold are calculated and shown in Table III, which indicates the same conclusion as we discussed above.

TABLE III. MEAN AND STD OF ROI FOR DIFFERENT STRATEGIES

Indicator Strategy	GP	GP with GC	Buy & Hold	MACD
Average ROI	0.2075	0.2311	0.1421	0.1343
STD of ROI	0.2184	0.1913	0.2947	0.215
Sharpe Ratio	0.7212	0.9470	0.3124	0.3921

By scrutinizing each of the 29 stocks, we find that the GP algorithm is especially effective in bearish markets and not necessarily in bullish ones. This supports the conclusion in previous work [8].

In Table III, we notice that both GP and GP+GC in our implementation outperform the traditional techniques. The mean, deviation and sharp ratio based on a risk-free ROI of 0.05 for GP are calculated and shown in Table III. These results show that, in terms of average of ROI and deviation of ROI, the three strategies in descending performance order are GP+GC, GP and Buy & Hold.

B. Hypothesis Testing on Model Prediction

To find out the reason why GP+GC often outperforms traditional technical analysis in terms of ROI, a paired-sample *t*-test is conducted on the model trained using the parameter settings listed in Table V. Raw Chinese financial data are downloaded from the WIND Financial Terminal (WFT).

Under reasonable assumption, the return on equity of each stock is approximately normally distributed and the predicted return on equity is independent from true market observations.

In the paired sample *t*-test, returns on equity are measured twice, one time from the true market value we observed, and another time forecasted from genetic programming, resulting in pairs of observations. The null hypothesis is as follow:

- H0: The true mean difference between the true daily possible maximum unrealized gain on equity and the predicted daily possible maximum unrealized gain on equity is zero.

To show the maximum volatility of the daily stock price, if short selling is possible, we use the daily maximum unrealized gain on equity (MUGOE) to represent the return:

$$MUGOE = (P_{max} - P_{min}) / P_{close}$$

P_{max} is the maximum price of the current trading day, P_{min} is the minimum price of the current trading day, and P_{close} is the close price of the latest trading day. From the test statistics and *p*-value shown in Table IV, with a significance level of 5%, we failed to reject H0 that the prediction mean equals to the observed mean for more than half of the stocks in SSE 50 Index.

For those stocks for which H0 is rejected, we find it statistically significant that GP+GC overall underestimates the maximum possible return on equity, as 37 out of 50 mean differences are below zero. Nevertheless, GP+GC gives a very good estimate of the future stock return, attributed to the tiny mean square error and standard deviation.

The firm but conservative forecast of market return might be the key to a higher overall profit gained from the GP+GC model in the long run, as losses are prevented in bearish markets. Calculating a confidence interval for the mean difference would tell us within what limits the true difference is likely to lie.

TABLE IV. PARAMETER SETTINGS FOR GENETIC PROGRAMMING USED IN HYPOTHESIS TESTING

Terminal Set	Mean, Max, Min of close price and volume of the stock in last 5 days;		
	Total number of total/ negative news in the last 5 days;		
	Indicators and growth rate of the stock: ROE (Return on Investment), BPS (Book Value Per Share), EPS (Earnings Per Share), OCFPS (Operating Cash Flow Per Share);		
	Industry Indicators and growth rate: Total number of enterprises, Total Revenue, Total Net Income		
Function Set	Plus, Minus, Multiply, Divide, Square of		
GP Value	(Max price – Min price) / close price		
Population Size (*n*)	100	**Generation (G_N)**	10
Maximum Tree Depth	4	**Minimum Number of Nodes for Tree**	3
Training Data Set	Dec 1, 2015 to Jun 30, 2016 of e.g. 600000.SH		
Testing Data Set	July 1, 2016 to Sep 30, 2016 of e.g. 600000.SH		

Taking stock 601377.SH as example, we can observe that the 0.1% point of the two-tailed *t*-distribution with 63 degrees of freedom (with 64 trading days as the testing set) is around 3.46, and the 99.9% confidence interval for the true mean difference is therefore:

$$0.013 \pm (3.46\times0.0026) = (0.004,0.022)$$

This confirms that although the difference in scores is statistically significant, it is relatively small. We can be 99.9% sure the true mean increase lies somewhere between nearly zero and just over 2 points for our prediction model on stock index 601377.SH. In the most extreme scenario, the average deviation of prediction on maximum possible market return is just over two percent which implies a precise prediction of market volatility.

TABLE V. T-TEST RESULT OF FIFTY STOCKS OF SSE 50 INDEX ON TEST DATASET

Index	Mean	Standard Error	T-test Statistic	P-Value
600000	-0.0016	0.0017	-0.9567	0.3427
600016	-0.0003	0.0024	-0.1395	0.8898
600010	-0.008	0.0028	-2.902	0.0054
600028	-0.0024	0.0012	-1.933	0.0577
600029	0.0018	0.0025	0.7351	0.4650
600030	-0.0002	0.0016	-0.6841	0.4965
600036	-0.0028	0.0015	-1.871	0.661
600048	-0.0098	0.0056	1.764	0.08251
600050	-0.0049	0.0023	-2.161	0.345
600104	0.006	0.0018	3.264	0.0178
600111	-0.0062	0.0027	-2.313	0.024
600518	-0.0023	0.0026	-0.8693	0.3881
600519	-0.0007	0.0029	-0.2529	0.8011
600637	-0.0069	0.0023	-3.003	0.0038
600795	-0.0028	0.0017	-1.61	0.1124
600893	0.0104	0.0026	4.007	0.0017
600837	-0.0075	0.003	-2.5338	0.0138
600887	0.0041	0.0018	2.272	0.0265
600893	-0.0072	0.0034	-2.142	0.0361
600958	-0.0178	0.0026	-0.684	0.4965
600999	-0.0063	0.0029	-2.134	0.0367
601006	0.0003	0.0019	0.1315	0.8958
601088	0.0000	0.0013	0.0027	0.9979
601166	-0.004	0.0014	-2.957	0.0044
601169	-0.0029	0.0015	-1.878	0.0651
601186	0.0017	0.0014	1.186	0.2401
601211	-0.0044	0.0013	-3.266	0.0018
601288	-0.0013	0.001	-1.275	0.2070
601318	0.0017	0.0014	1.186	0.2401
601328	-0.0026	0.0014	-1.876	0.0653
601336	0.003	0.0025	1.178	0.2432

601377	0.013	0.0026	5.084	0.0000
601390	-0.0006	0.0023	-0.2757	0.7835
601398	-0.0022	0.0006	-3.874	0.0003
601601	-0.0021	0.0012	-1.766	0.0822
601628	0.0021	0.003	0.7105	0.48
601668	-0.0013	0.0018	-0.718	0.4754
601669	-0.0025	0.0018	-1.342	0.1844
601688	-0.0097	0.0033	-2.931	0.0047
601727	0.0006	0.0033	0.1753	0.8614
601766	0.0048	0.0022	2.221	0.0299
601788	-0.0072	0.0022	-3.278	0.0017
601800	-0.0079	0.0038	-2.066	0.0429
601818	-0.0004	0.0012	-0.3266	0.7448
601857	-0.0096	0.0014	-6.466	0.0000
601919	-0.0076	0.0032	-2.381	0.0203
601985	0.0056	0.0022	2.511	0.0146
601988	-0.0002	0.0009	-0.215	0.8305
601989	0.006	0.0025	2.384	0.0202
601998	-0.0069	0.0028	-2.475	0.016

C. *Volatility Attribution Based on Genetic Programming*

Market regulators are particularly concerned about abnormal volatility in stock markets. Now we propose a framework for volatility attribution, which is the analysis of which stock contributes the most to the market movement in a certain analysis period and why they do so.

First, a genetic programming model is learned using features and parameter settings shown in Table IV. The experimental results are shown in Table V Notably; the stock trading volume is in the millions of shares. Used as features are a company's financial ratios (up to the trading day) measuring the company's earnings, assets, and cash flow. Other financial features from the knowledge graph include the industrial indicator the company belongs to. As discussed in Section IV.B, the genetic programming model gives accurate prediction on market volatility with a bias toward underestimating the potential return. Based on this observation, we propose the following two-step framework to provide a quantified criterion for regulators' stock screening:

- When an abnormal movement of SSE 50 Index is observed, stocks with top maximum possible daily return are picked.
- Leave out those stocks with actual return below the lower bound of model prediction.

Using the above framework, a case study consisting of the top 5 volatile stocks during the time from Aug 11th, 2016 to Aug 19th, 2016 are shown in Table VI. With a true return below the lower bound of the model predicted return, 600048.SH and 601390.SH are eliminated from the abnormal stock list which consolidates the fundamental analysis results.

TABLE VI. TRUE AND MODEL PREDICTED RETURNS OF TOP FIVE VOLATILE STOCKS DURING AUG 12TH 2016 AND AUG 19TH 2016

Index	Date	Expression Learned	Predicted Return	True Return
601336.SH	16/08/15	Volume / (BPS * Min Price)	4%	13.39%
600999.SH	16/08/15	BPS/ (Min Price ^ 2) + 0.0001 * Max Price	2.97%	10.67%
601390.SH	16/08/19	BPS * 0.0001 * Volume	18.4%	10.48%
600048.SH	16/08/12	BPS * 0.0001 * Volume / ROE	14.7%	9.98%
600030.SH	16/08/15	- EPS * 0.0001 + 0.0002 * Volume	5.1%	9.56%

As the expressions shown above, correlations of indicators such as trading volume, book value per share, and historical minimum/maximum prices of a stock with predicted return can be incorporated as triplets into the financial KG as we discussed in Section II. This is reflecting the impact of changes of financial ratios, liquidity, and historical price statistics on current price.

V. CONCLUSIONS

This paper introduced a stochastic optimization algorithm GP+GC, integrating genetic programming (GP) with generalized crowding (GC).

In experiments, the performance of trading rules based on our stochastic optimization algorithm was evaluated on 29 component stocks of the Dow Jones Industrial Average index. Statistical evidence supported that our algorithm can result in better short-term prediction of future stock prices, thus resulting in a better return on investment compared to a classic GP technique. Our GP+GC method also resulted in better average profits compared to technical analysis such as MACD.

By training GP models on features from a financial knowledge graph built for each component stock of the SSE 50 index, our experiments suggested that when used as a prediction tool for financial investment return, our integrated GP+GC algorithm tends to underestimate the market return within a very small range. Thus, excessive returns on investment can be obtained by making trade decisions simply relying on signals generated by comparison of the predicted result and previous periodical averages.

Finally, we propose a quantified framework for volatility attribution, demonstrating a potentially powerful tool for market regulators.

It is an interesting question for future research to investigate the similarity metrics for comparing tree structures in generalized crowding for GP. How to build automatic feature extractors by deeply integrating the symbolic structure of a genetic programming model with a knowledge graph model shall also be studied in the future.

ACKNOWLEDGMENT

We thank regulators from Shanghai Stock Exchange for providing insight and motivation to this work. Xiaoyi Fu would also like to thank his teammates from Mininglamp Software Systems for their generous help on data preparation.

REFERENCES

[1] Yumlu, S., Gurgen, F. and Okay, N. *A comparison of global, recurrent and smoothed-piecewise neural models for Istanbul stock change (ISE) prediction*. Pattern Recognition Letters, 2005, Volume 26, Issue 13, pp. 2093-2103.

[2] Yao, J. and Poh, H. L. *Prediction the KLSE Index Using Neural Networks*. Proceedings of IEEE International Conference on Artificial Neural Networks, 1995, Volume 2, pp. 1012-1017.

[3] Soni, S. *Applications of ANNs in Stock Market Prediction: A Survey*. International Journal of Computer Science & Engineering Technology (IJCET), 2011, Volume 2, Issue 3, pp. 71-83.

[4] Lee, R., Kochenderfer, M. J., Mengshoel, O. J., and Silbermann, J. Interpretable Categorization of Heterogeneous Time Series Data. Proceedings of SDM, May 2018, pp. 216-224.

[5] Lanzi, P. L., Stolzmann, W., and Wilson, S. W. (Eds.). *Advances in Learning Classifier Systems*. Lecture Notes in Artificial Intelligence, 2001, Volume 1996, pp. 37-51.

[6] Rao, Z. and Alvarruiz, F. *Use of an Artificial Neural Network to Capture the Domain Knowledge of a Conventional Hydraulic Simulation Model*. Journal of Hydro informatics, 2007, Volume 9, Issue 1, pp. 15-24.

[7] Staats, K., Pantridge, E., Cavaglia, M., Milovanov, J., and Aniyan, A. *TensorFlow Enabled Genetic Programming*. Proceedings of GECCO, July 2017, pp. 1872-1879.

[8] Devayan, M., Lee, V. C. S., and Yew, S. O. *An Empirical Study of Genetic Programming Generated Trading Rules in Computerized Stock Trading Service System*. IEEE International Conference on Service Systems and Service Management, June 30-July 2, 2008.

[9] Galan, S. F. and Mengshoel, O. J. *Generalized Crowding for Genetic Algorithms*. Proceedings of GECCO, July 2010, pp. 775-782.

[10] Mengshoel, O. J and Goldberg, D. E. *The Crowding Approach to Niching in Genetic Algorithms*. Evolutionary Computation, 2008, Volume 16, Issue 3, pp. 315-354.

[11] Wilcke, X., Bloem, P., and de Boer V. *The Knowledge Graph as the Default Data Model for Learning on Heterogeneous Knowledge*. Data Science, 2017, Volume 1, no. 1-2, pp. 39-57.

[12] Wilson, G. and Banzhaf, W. *Algorithmic Trading with Developmental and Linear Genetic Programming*. Genetic Programming Theory and Practice VII, Genetic and Evolutionary Computation, pp. 119.

[13] Mahfoud, S. and Mani, G. *Financial Forecasting Using Genetic Algorithms*. Applied Artificial Intelligence, 1996, Volume 10, Number 6, pp. 543-565.

[14] Lebaron, B., Arthur, W. B., and Palmer, R. *Time Series Properties of an Artifical Stock Market*. Journal of Economic Dynamics & Control, 1999, Volume 23, Issue 9-10, pp. 1487-1516.

[15] Yan, W., Sewell, M., and Clack, C. D. *Learning to Optimize Profits Beats Predicting Returns – Comparing Techniques for Financial Portfolio Optimization.* Proceedings of GECCO, July 2008, pp. 1681-1688.

[16] Mahfoud, S. W. *Crowding and Preselection Revisited.* Proceedings of the 2nd International Conference on Parallel Problem Solving from Nature, 1992, pp. 27–36.

[17] Eiben, A. E. and Smith, J. E. *Introduction to Evolutionary Computing*, 2003, Springer. Chapter 5.

[18] Kinnear, K. E., Angeline, P. J. *Advances in Genetic Programming*, 1994, MIT Press Cambridge. Volume 1, pp. 111-127.

[19] Allen, F. and Karjalainen, R. *Using Genetic Algorithms to Find Technical Trading Rules*. Journal of Financial Economics, 1999, Volume 51, Issue 2, pp. 245-271.

[20] Potvin, J. Y., Soriano, P., and Vall, M. *Generating Trading Rules on the Stock Markets with Genetic Programming*. Computers & Operations Research, 2004, Volume 31, Issue 7, pp. 1033-1047.

[21] Grosan, C. and Abraham, A. *Stock Market Modeling Using Genetic Programming Ensembles.* Genetic Systems Programming Theory and Experiences, 2006, Volume 13, Issue 2, pp. 133-148.

[22] Hoos, H. H. and Stützle, T. *Stochastic Local Search: Foundations & Applications*, 2005, Elsevier Inc. Chapter 1.

[23] Bishop, C. M. *Pattern Recognition and Machine Learning*, 2006, Springer. Chapter 5.

[24] Pujara, J. *Extracting Knowledge Graphs from Financial Filings: Extended Abstract*. Proceedings of the 3rd International Workshop on Data Science for Macro, Modeling with Financial and Economic Datasets, Article No. 5.

[25] De Jong., K. A. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1975.

[26] Chen, K., Zhou, Y., and Dai., F. *A LSTM-based Method for Stock Returns Prediction: A Case Study of China Stock Market*. Proceedings IEEE International Conference on Big Data, October 2015, pp. 2823-2824.

[27] Pimenta, A., Ciniro A. L., Nametala, F. G. G., and Carrano, E. G. *An Automated Investing Method for Stock Market Based on Multiobjective Genetic Programming.* Computational Economics, June 2018, Volume 52, Issue 1, pp. 125–144.

[28] Aguilar-Rivera, R., Valenzuela-Rendón, M., and Rodríguez-Ortiz, J. J. *Genetic Algorithms and Darwinian Approaches in Financial Applications: A Survey*. Expert Systems with Applications, November 2015, Volume 42, Issue 21, pp. 7684-7697.

[29] Yang, R., Kalnis, P., and Tung, A. K. H. *Similarity Evaluation on Tree-structured Data*. Proceedings of SIGMOD, June 2005, pp. 754-765.