Enhanced Strongly typed Genetic Programming for Algorithmic Trading

Eva Christodoulaki, Michael Kampouridis, Maria Kyropoulou
{ec19888,mkampo,maria.kyropoulou}@essex.ac.uk
School of Computer Science and Electronic Engineering, University of Essex, Wivenhoe Park, United Kingdom

ABSTRACT

This paper proposes a novel strongly typed Genetic Programming (STGP) algorithm that combines Technical (TA) and Sentiment analysis (SA) indicators to produce trading strategies. While TA and SA have been successful when used individually, their combination has not been considered extensively. Our proposed STGP algorithm has a novel fitness function, which rewards not only a tree's trading performance, but also the trading performance of its TA and SA subtrees. To achieve this, the fitness function is equal to the sum of three components: the fitness function for the complete tree, the fitness function of the TA subtree, and the fitness function of the SA subtree. In doing so, we ensure that the evolved trees contain profitable trading strategies that take full advantage of both technical and sentiment analysis. We run experiments on 35 international stocks and compare the STGP's performance to four other GP algorithms, as well as multilayer perceptron, support vector machines, and buy and hold. Results show that the proposed GP algorithm statistically and significantly outperforms all benchmarks and it improves the financial performance of the trading strategies produced by other GP algorithms by up to a factor of two for the median rate of return.

KEYWORDS

Technical Analysis, Sentiment Analysis, Genetic Programming, Algorithmic Trading

ACM Reference Format:
Eva Christodoulaki, Michael Kampouridis, Maria Kyropoulou. 2023. Enhanced Strongly typed Genetic Programming for Algorithmic Trading. In *Genetic and Evolutionary Computation Conference (GECCO '23), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3583131.3590359

1 INTRODUCTION

Algorithmic trading is the execution of orders using pre-programmed trading strategies to generate profit. These systems have been used in trading for many years with their popularity increasing constantly, as the number of services and companies to trade increase. The topic of algorithmic trading is popular amongst researchers, too, who use Machine Learning (ML) implementations to maximise profits. ML algorithms examine historical information of the stock market and identify patterns, "learning" how certain indicators are associated with certain trends. Then, when they recognise such a pattern, the algorithms generate signals indicating an upcoming change in trend, which can be used to generate profit.

Technical analysis is a financial technique that uses price trends and patterns to identify trading opportunities. Sentiment analysis corresponds to recognising events relevant to stocks, identifying their importance towards influencing their price and using that for predicting stock prices. Researchers have mainly utilised Technical Analysis (TA) indicators, such as volatility and moving average, for algorithmic trading, but sentiment analysis (SA) indicators, such as sentiment polarity, have also been successfully considered in the more recent years. The benefits observed by the two individual analyses have now brought about the promise of achieving an improved performance by their combination. Indeed, [15] and [5], very recently provided initial evidence supporting this promise, by creating financially advantageous tradings strategies that utilise both analysis types.

In this paper we combine TA and SA indicators in the context of genetic programming (GP) algorithms. Our proposed algorithm, STGP-SATA-sum, uses a strongly typed GP structure, where TA and SA indicators are handled in separate parts of the model (subtrees/branches of the tree). As a result, each GP tree has a dedicated branch that deals only with TA indicators and another branch that deals only with SA indicators. This has the advantage of letting the algorithm focus on the search space of each individual indicator type and encourages better exploration and exploitation. As seen in other studies [4, 5], combining the indicators into a GP algorithm both enhances the financial advantages and assists the exploration of the indicators.

The fitness function of the proposed STGP-SATA-sum takes into account not only the performance of a given individual (tree), as it usually happens in evolutionary algorithms, but also the performance of the TA and SA subtrees. As a result, the GP evolves individuals that ensure that both the technical and sentiment analysis indicators contribute to the overall performance of a GP individual. This is particularly important, because it guarantees good performance for each component of an individual (TA and SA subtrees), and also good performance for the overall individual, which contains these TA and SA subtrees.

The purpose of the research is to showcase that combining TA and SA indicators in the terminal set of the strongly typed GP can be used to create financially advantageous trading strategies. Five years' data on 35 international companies' stocks were used to evaluate the performance of the proposed GP algorithm and four other GP-variants with respect to three financial metrics (Sharpe ratio, rate of return, risk). Furthermore, the proposed GP algorithm

GECCO '23, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0119-1/23/07…$15.00
https://doi.org/10.1145/3583131.3590359

is compared against a financial benchmark, buy and hold, and two algorithmic benchmarks, multilayer perceptron (MLP) and support vector machine (SVM). Buy-and-Hold (BnH), is a common financial strategy, where the investor buys stocks and holds them for a long period of time, regardless of uncertainty and volatility.

The paper is organised as follows. We begin by introducing relAtive research work in Section 2. The methodology of the research can be found in Section 3 and the experimental setup in Section 4. The results and the analysis of the study are presented and discussed in Section 5. We conclude the paper in Section 6.

2 LITERATURE REVIEW

Technical analysis (TA) indicators and machine learning have long been combined. Artificial neural networks are widely applied in finance for forecasting and algorithmic trading. For example, Mostafa [14] used technical analysis indicators with linear models and Nelson et al. [16] used a long short-term memory (LSTM) model to forecast future stock trends. For genetic programming, one of the first papers to utilise technical analysis (TA) indicators for financial forecasting was by Li and Tsang [11], where the algorithm was able to outperform commonly used, non-adaptive, individual technical rules. In the last decade, more studies have achieved similar results [8, 9]. Berutich et al. [2] and Brabazon et al. [3] showed that genetic programming (GP) algorithms can evolve trading strategies by generating solutions that endure extreme market conditions, as well as, create new solutions and optimize the solution parameters.

Kohara et al. [10] used neural networks for Sentiment Analysis (SA) and studied how to increase the accuracy of prediction of multivariate models using prior knowledge from newspaper headlines. Xie et al. [20] generalized from sentences to scenarios and Ding et al. [7] produced an event-driven stock model by feeding news into a deep convolutional neural network (CNN). Christodoulaki et al. [6] studied the individual properties and financial advantages of TA and SA under a GP structure individually for algorithmic trading.

Peng and Jiang [18] used deep neural networks (DNN) to predict stock price movements, combining prices and online financial news, increasing the accuracy of the model. Vargas et al. [19] used text mining on news from Reuters regarding the S&P500 index, along with technical indicators, in a recurrent neural network (RNN) and CNN hybrid model. Nan et al. [15] created a reinforcement learning approach, utilising price data and news headline sentiments.

Yang et al. [21] considered combining TA and SA indicators and compared the performance of individual and combined approaches. Christodoulaki et al. [4] present a simple GP that combines the two financial indicators. In [5] the authors considered the combination of TA and SA indicators for algorithmic trading using a strongly typed GP algorithms to enhance the exploration properties.

All of the above works have shown that both TA and SA indicators can be used for profitable algorithmic trading. However, as we can observe, there are limited works that have combined TA and SA indicators. In addition, simply combining indicators under an algorithm might not take full advantage of the potential of both indicator types, as the search might focus on one type only. This thus motivates us to use a strongly-typed GP, which enforces its individuals (trees) to always contain dedicated TA and SA nodes. In addition, to avoid individuals with weaker subtrees (e.g. most trading actions could be coming from the TA subtree, while the SA subtree having minimal or no actual contribution to trading), we propose using a fitness function that also rewards the performance of the TA and SA subtres. We present more details about this in the next section.

3 METHODOLOGY

The methodology of our research has been divided in four parts. Section 3.1 presents the two types of analysis and relevant indicators (indicators) that we are going to consider. Section 3.2 covers the GP methodology, including model representation and GP operators. Section 3.3 discusses the trading algorithm utilised by the GP, while Section 3.4 presents the fitness function and metrics that will be considered.

3.1 Financial analysis processes

This section covers the processes of technical analysis and sentiment analysis in two separate subsections.

3.1.1 Technical analysis. Technical analysis (TA) is a popular tool in financial forecasting and algorithmic trading, with researchers using financial metrics to calculate and create new technical analysis indicators, in order to recognize trends in the stock market and generate higher profits.

Our study uses six widely-adopted technical analysis indicators, namely the Moving Average, the Momentum, the Rate of Change, the Williams %R, the Midprice and the Volatility, defined in Equations (1) - (6) below. These are calculated based on historical data on (adjusted) close prices, highest and lowest daily prices of selected companies, available on Yahoo! Finance (more details on our datasets are presented in Section 4.1). Each indicator is considered with respect to look-up windows of $n = 5$ and $n = 10$ days, giving rise to 12 TA indicators summarized in Table 1.

Table 1: Technical Analysis indicators. Each indicator is considered for two different lookup windows (n).

lookup windows $n = 5$ and $n = 10$
Moving Average
Momentum
ROC
Williams %R
Volatility
Midprice

The Moving Average is defined as follows, and is used to smooth out the data and helps with noise elimination towards identifying trends. p_j denotes the adjusted closing price of the j-th day in our dataset for a corresponding stock.

$$\text{Moving Average}(n, j) = \frac{\sum_{i=j-n}^{j} p_i}{n}, \text{ for } j \geq n. \quad (1)$$

The Momentum captures the difference between the most recent adjusted closing price and the adjusted closing price n days ago, as follows.

$$\text{Momentum}(n, j) = p_j - p_{j-n}, \quad (2)$$

while the Rate of Change (ROC) normalizes the momentum.

$$\mathrm{ROC}(n, j) = \left(\frac{p_j}{p_{j-n}} - 1\right) \cdot 100. \tag{3}$$

The volatility is a statistical measure of the dispersion of returns over a given period of time. We calculate the following relevant indicator.

$$\mathrm{Volatility}(n, j) = \sqrt{\mathrm{Var}\left(\left\{\frac{p_{j-i}}{p_{j-n}} - 1\right\}_{i\in\{0,\ldots,n-1\}}\right)}, \tag{4}$$

where Var defines the sample variance over a dataset.

The Williams %R indicator, defined in Equation (5), reflects the level of most recent closing price, cl_j (at day j), to the highest high price, $hh_{n,j}$, of all values in the lookup window ending at day j. $ll_{n,j}$ denotes the lowest low price over all days in the lookup window ending at day j.

$$\text{Williams \%R}(n, j) = -100 \cdot \frac{hh_{n,j} - cl_j}{hh_{n,j} - ll_{n,j}} \tag{5}$$

$Midprice$, defined in Equation 6, returns the midpoint value of the highest high price, $hh_{n,j}$, and the lowest low price, $ll_{n,j}$, over all days in the lookup window ending at day j.

$$\mathrm{Midprice}(n, j) = \frac{hh_{n,j} - ll_{n,j}}{2} \tag{6}$$

All TA indicators were normalised between $[-1, 1]$.

3.1.2 *Sentiment analysis.*

As financial markets get influenced by events and stocks' prices increase/decrease along with people's decisions on online information, there is a surge of studies using sentiment analysis indicators in the areas of financial forecasting and algorithmic trading. Sentiment analysis (SA) is the process of extracting the sentiment out of articles and online comments and utilising into increasing the accuracy of stock estimation and trading strategies' profits.

Two widely adopted sentiment analysis indicators are the sentiment polarity and subjectivity of given texts. The former, captures the inclination of sentiment, and the relative text is classified as positive, negative or neutral. The latter captures the extent to which the respective text expresses a personal opinion rather than a fact. In our analysis we use indicators based on the above indicators, while distinguishing between the method of calculating them (definitions of respective methods appear below). In particular, we consider 12 SA indicators summarized in Table 2. All SA indicators were normalised between $[-1, 1]$.

Table 2: Sentiment Analysis Indicators

textBlob		**SentiWordNet**	**AFINN**
TEXTpol,	TEXTsub	TEXTsenti	TEXTafinn
TITLEpol,	TITLEsub	TITLEsenti	TITLEafinn
SUMMpol,	SUMMsub	SUMMsenti	SUMMafinn

In sentiment analysis classification research, it is popular to use specialized SA programs, namely TextBlob [13], SentiWordNet [1] and AFINN sentiment [17] for calculating the polarity and/or subjectivity of given texts. *TextBlob* is a Python library, offering a simple API assisting in calculating the polarity and subjectivity of the text. *SentiWordNet 3.0* is an enhanced lexical resource, based on lexical taxonomy, *WordNet*, of the English language, explicitly devised to support sentiment classification and opinion mining. It contains a list of words classified as positive, negative, or neutral and an overall percentage of the sentiment of a given text is calculated as the weighted average of the relevant words. *AFINN* sentiment is a popular lexicon for sentiment analysis that contains more than 3300 words with a polarity score to each one of them, developed by Finn Årup Nielsen. In our research, the in-built function for the lexicon is being utilised, which is available in Python.

Our sentiment analysis indicators (Table 2) consider the polarity and subjectivy levels extracted by TextBlob, as well as the sentiment polarity extracted by *SentiWordNet* and *AFINN*. The relevant articles, their titles and their summaries are considered separately, thus giving rise to 12 SA indicators.

Our analysis included downloading articles relevant to selected companies and associating their sentiment with the corresponding date and price changes. We developed a scrapper that uses the Google Search Console API in Python to download the first twenty pages of daily Google Search results, using the name of each company as a keyword. The articles were downloaded for the same period as for the TA indicators.

We narrowed our attention to articles of at least 500 characters long, that included both the name of the corresponding company and its stock market ticker. This helped ensure that we only consider articles relevant to the companies that were downloaded correctly.

We matched the dates of the articles' appearance with the relevant stock price data. For articles appearing on weekends, when the stock market is closed, the sentiment was included to that of Friday's, in order to capture their influence on the stock price of the following day (Monday). In cases where more than one articles were appearing for the same company on the same date, we found the average sentiment value of the articles. For the days where no articles were published, a sentiment of 0 was assigned to indicate neutrality and/or no action, to ensure continuity of our datapoints.

3.2 Genetic programming and the STGP-SATA-sum algorithm

We propose the STGP-SATA-sum algorithm; a novel strongly-typed Genetic Programming algorithm whose fitness function considers both the overall tree performance, as well as the performance of the subtrees of Sentiment Analysis (SA), and Technical Analysis (TA). We argue that both the strongly typed structure of our algorithm and the proposed fitness function assist in the exploration process of the vast search space.

Note that the models are found in the training process and then they are implemented in the testing process/set to find the results of the trees in unseen data.

3.2.1 *Model representation.*

Our model representation requires solutions to be presented in a tree structure consisting of a root node, function nodes and terminal nodes. Part 1 of Figure 1 shows a sample tree that the STGP-SATA-sum algorithm can create. The strongly typed structure of our algorithm enforces that the root will have two children, where each one allows for a different indicator type. The root is always an AND function that unites the two branches; the first branch of the AND function is enforced to be SA-related and the second branch is forced to be TA-related.

The function nodes are based on the logical functions AND, OR, Greater than (GT) and Less than (LT), with different variants allowing for different indicators. In particular, our algorithm uses SA_AND, SA_OR, SA_GT, SA_LT function nodes in the SA branch and it uses TA_AND, TA_OR, TA_GT, TA_LT function nodes in the TA branch. The function set for all GP variants is summarised in Table 3.

Table 3: Function set

Function set	
Function set (STGP-SATA-sum)	AND, SA_AND, TA_AND, SA_OR, TA_OR, SA_GT, TA_GT, SA_LT, TA_LT

With respect to terminal sets, the SA branch allows only for SA terminals and the TA branch allows only for TA terminals. The SA terminal set includes indicators summarized in Table 2 as well as an Ephemeral Random Constant (ERC) that acts as a threshold value to the indicators and takes a random value between -1 and 1. Similarly, the TA terminal set includes indicators summarized in Table 1, as well as the ERC.

3.2.2 GP operators. We use subtree crossover, which occurs with probability p, and point mutation, which occurs with probability $1-p$. When performing *subtree crossover* in a strongly-typed setting we exchange both the SA and the TA branches, *but not between themselves*. The nodes need to be of the same type (e.g. a terminal node with another terminal node) and of the same data type (SA subtree with SA subtree). To ensure the legality of the tree exchange, first we crossover the SA subtrees of the two selected parents and when this process is completed, we crossover the TA subtrees of the two trees.

When performing *point mutation* in a strongly-typed setting there also are some constraints that need to be met. For example, function node SA_OR can only be changed to SA_AND, function node TA_GT can be replaced only with TA_LT (similarly for other function nodes), an ERC can be only replaced with another ERC and a terminal variable can only be replaced with another variable from the same terminal set. The algorithm thus ensures that valid data types replaced the mutated nodes.

3.3 Trading algorithm

The STGP-SATA-sum trees above are used to generate trading signals. In particular, the binary outcome of the root AND function is passed on to the STGP-SATA-sum algorithm and is used towards making a recommendation to buy or hold a stock, as follows.

Every GP model that is being evolved is embedded into another tree, which has an If-Then-Else (ITE) statement as its root, see Figure 1. We note that only Part 1 of Figure 1 evolves through GP operations. The second and third branches of the latter tree are *fixed* and correspond to buy (1) and hold (0) recommendations, respectively; so there is no need for them to be part of the evolutionary process.

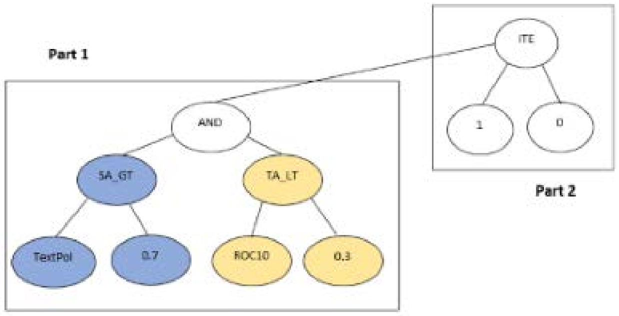

Figure 1: Sample tree of STGP-SATA-sum. The first child of the AND function is forced to be SA-related and the second child to be TA-related. This tree checks if the TEXTpol indicator is greater than 0.7 (ERC) and if ROC is less than 0.3 (ERC). If both of them are true, the recommendation will be to buy (1), otherwise it will be to hold (0).

The trading algorithm takes as input the 0/1 signals, as well as two more parameters, d and r, as follows: when the signal is 1, the algorithm buys *one* stock per trade, which it will sell when the price increases by more than the rate of reference r, or after d days have passed, whichever happens sooner. The algorithm won't buy a new stock if it still owns a previously bought one. Parameters d and r are optimized during the validation phase and are the same for all GP algorithms considered but different across different companies (see Section 4.3).

3.4 Fitness function and Metrics

Our analysis considers the metrics of return, risk and Sharpe ratio, which are defined as follows.

The *return*, R, of a trade captures the profit made as a percentage of the amount invested. The calculation of the profit takes into account transaction costs 0.025% of the selling price (c_t). In particular, the return is calculated as shown in Equation (7), where V_f denotes the final value, or the price the stock was sold, and V_i denotes the initial value, or the price the stock was bought.

$$R = \frac{(1 - c_t)V_f - V_i}{V_i}. \tag{7}$$

The rate of return, *RoR*, denotes the sample mean of the returns of all trades in a corresponding period of time in question.

The *risk* is captured as the standard deviation of the returns, that is $\sqrt{var[R]}$.

The *Sharpe ratio*, S_r, is defined as the ratio of the expected value of the excess return compared to the risk free return, R_f, over the risk. Formally,

$$S_r = \frac{\mathrm{E}[R - R_f]}{\sqrt{var[R]}}, \tag{8}$$

where R_f is the risk free return.

The *fitness function*, f_{sum}, of STGP-SATA-sum is defined as the summation of the Sharpe ratio, S_r^C, of the complete tree, which combines SA and TA indicators; the Sharpe ratio, S_r^{SA}, of the subtree that considers only SA indicators; and the Sharpe ratio, S_r^{TA}, of the substree that considers only TA indicators, with weights w_c, w_{sa}, and w_{ta}. Formally,

$$f_{sum} = w_c \cdot S_r^C + w_{sa} \cdot S_r^{SA} + w_{ta} \cdot S_r^{TA}. \tag{9}$$

For example, in Figure 1, S_r^C corresponds to the subtree with root node AND (i.e. Part 1), S_r^{SA} corresponds to the subtree with root node SA_GT (blue-coloured nodes), and S_r^{TA} corresponds to the subtree with root node TA_LT (yellow-coloured nodes).

The advantage of using this fitness function is that it allows us to evolve trees that maximise all three Sharpe ratios; as a result, the GP can guide the search towards trading strategies that have strong performance across all three components of the fitness function. This is particularly important, because if, for example, the fitness function was only the Sharpe ratio of the whole tree (as it usually happens in such cases in the literature), the GP would be able to identify well-performing trees, but would not necessarily take advantage of its strongly typed nature that ensures that there are always both TA and SA indicators present.[1]

Thus, in this paper, the proposed GP algorithm (STGP-SATA-sum) is a strongly typed GP whose fitness function is the maximisation of the Sharpe ratio produced by the weighted sum of the TA subtree, the SA subtree and the complete tree that contains the above two subtrees. The strongly typed structure of the GP allows better exploration and exploitation of both TA and SA indicators, and the fitness function enhances the contribution of each of these indicators in the trading performance of the algorithm.

4 EXPERIMENTAL SETUP

4.1 Data

Our analysis is based on data on 35 companies, and datasets include historical stock prices and relevant news articles. The companies were selected based on their popularity, in order to ensure that a sufficient amount of news articles is available. The analysis spans a 5-year period, between 1st January 2015 and 31st January 2020. The period excludes the pandemic of COVID-19, because that would make the train/validation sets too different from the test set, and the parameter tuning would not be reliable.

The daily closing price data were downloaded from Yahoo! Finance. Regarding sentiment analysis, news articles, their titles and summaries, were downloaded by a web scrapper that was developed in the context of this project. We could generate 24 relevant indicators based on this data; see Section 3 for details. Finally, the companies' datasets were partitioned into three sets in sequence: 60% used for training, 20% for validation, and 20% for testing.

4.2 Benchmarks

The proposed STGP-SATA-sum is benchmarked against four other GP algorithms:

- **GP-TA** is a (non-strongly typed) GP algorithm that only has technical analysis indicators on its terminal set; the fitness function is the maximisation of the Sharpe ratio of a given tree.
- **GP-SA** is a (non-strongly typed) GP algorithm that only has sentiment analysis indicators on its terminal set; the fitness function is the maximisation of the Sharpe ratio of a given tree.
- **GP-SATA** is a (non-strongly typed) GP algorithm that combines indicators of technical and sentiment analysis; the fitness function is the maximisation of the Sharpe ratio of a given tree.
- **STGP-SATA** is a strongly typed GP algorithm that combines indicators of technical and sentiment analysis; the fitness function is the maximisation of the Sharpe ratio of a given tree.

In addition, the following two algorithmic benchmarks were considered:

- **Multilayer perceptron (MLP)** is a fully connected class of feedforward artificial neural networks.
- **Support vector machine (SVM)** is a supervised learning model.

The two benchmarks have been used widely in the relevant literature and, in this research, the built-in models of the scikit-learn library in Python were utilised. We use these two algorithms to tackle a binary classification problem in the form of "Is the stock price going to increase by r% within the next n days?". Class 1 denotes a buy action, and Class 0 denotes a hold action. The sell action takes again place as a part of the trading strategy that was described earlier in Section 3.3.

Finally, STGP-SATA-sum is evaluated against the following financial benchmark:

- **Buy-and-Hold$_{d,r}$ (BnH$_{d,r}$)**: Buy at the beginning of every trading period. Sell when the price increases by more than the rate of reference r, or after d days have passed, whichever happens sooner.

We use a novel alternative version of BnH as we are interested to compare all financial metrics and not just the rate of return. Furthermore, we compare the financial benchmark during the same period as the test set of STGP-SATA-sum for a fair comparison.

4.3 Parameter tuning

The parameter tuning took place in two steps.

First, we selected appropriate values for the GP parameters of population size, crossover probability (p), number of generations, tournament size, and maximum depth of the trees, while keeping the trading parameters d and r constant.[2] A combination of parameters was identified that performed equivalently well, and without any statistical differences, for all GP variants (see Section 4.2); using the same parameters for all GP models also enables a fair comparison.

[1] In fact, early experiments have shown exactly this: when the fitness function was only the Sharpe ratio of the whole tree, very frequently the best tree would have a large and well-performing subtree on the SA side, but a small and bad-performing subtree on the TA side. In addition, the performance of the overall tree was no better than the performance of a non-strongly typed GP that allowed the presence of both TA and SA indicators, thus making the use of the strongly-typed architecture, redundant. Our proposed fitness function overcomes this limitation.

[2] The mutation probability is $1 - p$, thus it was not necessary to include it in the parameter tuning process.

These GP parameters are summarized in Table 4 and are the same in all runs, for all GP algorithms and across all companies. This step was completed by a grid search using the validation data-set. We also considered different values of the weights of the fitness function components, i.e. w_c, w_{sa}, and w_{ta}. We found that equal weights (i.e., a weight of 0.33 for each component) offers the best trading performance in the validation set.

Table 4: GP Parameters for GP-TA, GP-SA, GP-SATA. STGP-SATA.

GP Parameters	
Population size	1000
Crossover probability	0.95
Mutation probability	0.05
Generations	50
Tournament size	4
Maximum tree depth	6

The second step involves optimizing over the trading strategy parameters d and r. These parameters are company-specific to enable for better trading performance, while their tuning utilised the validation set. They stay constant since the differences between the validation and the test set are minimal.

The parameter tuning for MLP and SVM is performed separately using binary classification, where one class corresponds to a price increase for the next day, and the other corresponds to the price decreasing or staying the same. Later, the model with the best predictive ability on the validation set is chosen. This is the predicted class for the test set and it is later used as signals, fed into the trading strategy. The trading strategy parameters are set to be the same d (days) and r (percentage increase) values as in the GP-variants. The tuning process for these two machine learning algorithms for trading purposes is based on [12].

5 RESULTS

This section presents results of our experiments on the performance of the STGP-SATA-sum algorithm against all other GP benchmarks, along with a brief discussion. For each algorithm, 50 independent runs were performed on the training set for each one of the 35 companies. Each run results in a tree/model which corresponds to a trading strategy that is then evaluated in the test set. This takes place after the tuning of the GP parameters that took place using the validation set.

Our analysis only considers runs where the corresponding algorithm performs at least two trades. This is because including runs with zero or one trades would skew the statistical analysis, as risk (and rate of return and Sharpe ratio, in the case of zero trades) would be 0. In addition, risk being 0 means that the Sharpe ratio cannot be calculated, as its denominator is equal to 0.

To increase confidence in our findings, a two-sample Kolmogorov-Smirnov (KS) test was performed on all comparisons of the best performing algorithm against the remaining GP algorithms, for all the runs of each algorithm for each company, resulting in at least two trades. The null hypothesis is that the respective two distributions being compared each time, come from the same continuous distribution. The KS test was chosen due to its sensitivity to differences in the shape of the empirical cumulative distribution of two samples. To account for the multiple comparisons (multiple benchmarks), the Holm-Bonferroni correction was performed. In particular, the minimum acceptable p-value for a statistical significance at a 5% significance level is equal to $\alpha(rank) = \frac{0.05}{4-rank+1}$, which is different for the different ranks of the p-values calculated; $rank \in \{1, 2, 3, 4\}$. The 4 in the denominator corresponds to the number of different comparisons, i.e. number of GP algorithms STGP-SATA-sum is compared against, in each financial metric individually. Rank corresponds to the rank of the p-values, with 1 corresponding to the smallest p-value and 4 to the largest. In other words, the first ranked p-value should be less than 0.0125, the second less than 0.0166, the third less than 0.025 and the forth less than 0.05 to show that the two distributions are statistically different.

5.1 Summary statistics on financial metrics

5.1.1 GP algorithms - Sharpe ratio. Table 5 presents the mean, median, standard deviation (StDev), maximum (Max) and minimum (Min) Sharpe ratio values for each algorithm over the 50 independent GP runs on all companies. STGP-SATA-sum has the highest mean, median, maximum, as well as the lowest minimum Sharpe ratio values, while having the highest standard deviation value.

Table 5: Statistical analysis on Sharpe ratio values. Best values denoted in boldface.

Algorithm	*Mean*	*Median*	*StDev*	*Max*	*Min*
GP-SATA	2.98	1.36	10.7	44.8	-38.3
GP-SA	2.92	1.49	4.49	17.8	-1.09
GP-TA	2.72	1.46	6.3	17.6	-20.9
STGP-SATA	3.21	1.8	**4.71**	18	-9.73
STGP-SATA-sum	**7.44**	**2**	13.55	**72.5**	**-0.41**

Table 6 presents the KS test p-values for the comparisons against the best ranking algorithm (i.e. STGP-SATA-sum). When p-value is below its corresponding significance level (indicating a statistically significant difference between the two distributions), we denote this by putting the relevant p-value in bold face. As we can observe, the average Sharpe ratio values of STGP-SATA-sum statistically outperform those of the other algorithms in the different significance levers based on Holm-Bonferroni correction. The only exception is GP-SA, where the p-value is marginally above the 5% level, thus making their difference statistically significant at the 10% level.

Table 6: KS test p-values on mean Sharpe ratio. Statistical significance changes based on the Holm-Bonferroni correction.

Algorithm	*STGP-SATA-sum p-values*	Rank	*Significance level*
GP-SATA	**6.42E-08**	2	0.016
GP-SA	0.051	4	0.05
GP-TA	**3.61E-17**	1	0.0125
STGP-SATA	**0.0001**	3	0.025

5.1.2 GP algorithms - Rate of Return. Table 7 presents our results on the rate of return (RoR) per algorithm. STGP-SATA-sum has, again, the highest mean, median, maximum, as well as the lowest minimum values, while GP-TA has the highest standard deviation. The advantage of combining SA and TA analyses' indicators is particularly evident in the median RoR column, where STGP-SATA-sum performs almost twice as well as the other algorithms.

Table 7: Statistical analysis on rate of return values. Best values denoted in boldface.

Algorithm	*Mean*	*Median*	*StDev*	*Max*	*Min*
GP-SATA	0.012	0.007	0.02	0.094	-0.02
GP-SA	0.009	0.008	0.022	0.064	-0.04
GP-TA	0.010	0.009	0.03	**0.09**	-0.08
STGP-SATA	0.014	0.008	0.019	**0.09**	-0.006
STGP-SATA-sum	**0.017**	**0.016**	**0.018**	**0.09**	**-0.004**

The KS test p-values in Table 8, show that the mean RoR results of STGP-SATA-sum are statistically significant and they statistically outperform the mean values of the other algorithms. The only exception is again in the comparison with GP-SA, where their difference is statistically significant at the 10% level.

Table 8: KS test p-values on mean rate of return. Statistical significance changes based on the Holm-Bonferroni correction.

Algorithm	*STGP-SATA-sum p-values*	Rank	*Significance level*
GP-SATA	**1.89E-06**	2	0.016
GP-SA	0.051	4	0.05
GP-TA	**9.87471E-16**	1	0.0125
STGP-SATA	**0.005**	3	0.025

5.1.3 GP algorithms - Risk. Table 9 summarizes the results on the risk of each of the algorithms. STGP-SATA-sum has the lowest mean, median, standard deviation and maximum risk values, while each algorithm exhibits minimum risk equal to 0.

Table 9: Statistical analysis on risk values. Best values denoted in boldface.

Algorithm	*Mean*	*Median*	*StDev*	*Max*	*Min*
GP-SATA	0.029	0.022	0.023	0.09	**0**
GP-SA	0.028	0.021	0.021	0.07	**0**
GP-TA	0.026	0.021	0.021	0.09	**0**
STGP-SATA	0.025	0.025	0.018	0.063	**0**
STGP-SATA-sum	**0.017**	**0.015**	**0.014**	**0.058**	**0**

Again, the STGP-SATA-sum algorithm is the best performing algorithm with respect to risk. Table 10 presents the p-values for the KS tests and the significance level thresholds based on the Holm-Bonferroni correction. STGP-SATA-sum manages to statistically outperform all other four algorithms in terms of risk.

Table 10: KS test p-values on mean risk. Statistical significance changes based on the Holm-Bonferroni correction.

Algorithm	*STGP-SATA-sum p-values*	Rank	*Significance level*
GP-SATA	**0.0002**	2	0.016
GP-SA	**0.01**	4	0.05
GP-TA	**1.42E-13**	1	0.0125
STGP-SATA	**0.003**	3	0.025

5.2 STGP-SATA-sum - Weights of Fitness Function

In this section, we present indicative scenarios of different weight combinations of the three components of the fitness function, to better understand how the weights affect the trading performance. In particular, we considered the case of equal weights and three cases where the algorithm that combines the SA and TA indicators is set as the main component with 50% weight: in one, the two individual components are considered with equal weights, while in the other two one of them is prioritised. These results in Tables 11 - 13 demonstrate that equal weights have better performance across the three metrics of Sharpe ratio, rate of return, and risk.

Table 11: Sharpe ratio of weights

Weights (w_c, w_{sa}, w_{ta})	*Mean*	*Median*	*StDev*
0.33, 0.33, 0.33	**7.44**	**2**	13.55
0.5, 0.25, 0.25	4.85	0.73	13.1
0.5, 0.15, 0.35	4.48	0.90	**9.8**
0.5, 0.35, 0.15	3.77	0.56	12.2

Table 12: Rate of returns of weights

Weights (w_c, w_{sa}, w_{ta})	*Mean*	*Median*	*StDev*
0.33, 0.33, 0.33	**0.017**	**0.016**	**0.018**
0.5, 0.25, 0.25	0.008	0.006	0.031
0.5, 0.15, 0.35	0.007	0.005	0.024
0.5, 0.35, 0.15	0.011	0.010	0.025

Table 13: Risk of weights

Weights (w_c, w_{sa}, w_{ta})	*Mean*	*Median*	*StDev*
0.33, 0.33, 0.33	**0.017**	**0.015**	**0.014**
0.5, 0.25, 0.25	0.029	0.021	0.024
0.5, 0.15, 0.35	0.030	0.021	0.023
0.5, 0.35, 0.15	0.029	0.029	0.020

More specifically, STGP-SATA-sum using the equal weights of 0.33 has a higher average Sharpe ratio and median; while its standard deviation is high due to outliers. Furthermore, it statistically outperforms the other weight variations with p-values of 0.000344, 0.00702 and 0.00046 as they are introduced in Table 11. The α levels are again based on the Bonferroni-Holm correction.

With respect to rate of return, the proposed algorithm has double the average and median of the other weight combinations, while it has the lowest standard deviation, too. STGP-SATA-sum with the equal weights statistically outperforms the algorithms shown in Table 12 0.00107, 0.00476 and 0.00074.

For risk, it again, showcases the least risk value in mean, median and standard deviation. This time, it does not statistically outperform the weight variations in Table 13, as the p-values are 0.059, 0.0218 and 0.039.

5.3 STGP-SATA-sum VS algorithmic and financial benchmarks

The average values of the three metrics for MLP, SVM and the financial benchmark BnH$_{d,r}$ on the 35 companies appear in Table 14, while the median values can be found in Table 15.

Table 14: MLP, SVM and BnH$_{d,r}$ average values

	STGP-SATA-sum	MLP	SVM	BnH$_{d,r}$
Sharpe ratio	**7.44**	0.31	0.32	0.12
RoR	**0.017**	0.01	0.01	0.0075
Risk	**0.017**	0.043	0.043	0.073

Table 15: MLP, SVM and BnH$_{d,r}$ median values

	STGP-SATA-sum	MLP	SVM	BnH$_{d,r}$
Sharpe ratio	**2**	0.19	0.20	0.10
RoR	**0.016**	0.007	0.007	0.004
Risk	**0.015**	0.04	0.04	0.04

As we can observe, STGP-SATA-sum has noticeably higher average and median values than MLP, SVM, and BnH$_{d,r}$. The mean rate of return (RoR) values are similar among all algorithms, but the median RoR for STGP-SATA-sum is at least double when compared to the benchmarks. Lastly, our proposed algorithm introduces risk reductions in both average and median values in factor of at least two. The above results are also confirmed by a Kolmogorov-Smirnov test, which returns a p-value of $8.86E-11$ (Sharpe ratio), 0.0020 (RoR), and $1.11E-05$ (Risk) between STGP-SATA-sum and MLP. Similarly, the p-values between STGP-SATA-sum and SVM were $8.86E-11$ (Sharpe ratio), 0.010 (RoR) and $3.65E-05$ (Risk). Thus in both cases there was statistical significance between the differences between the distributions at the 5% significance level. Lastly, the K-S test p-values when comparing the distributions of STGP-SATA-sum and BnH$_{d,r}$ are $5.96E-14$ for Sharpe ratio, $4.32E-06$ for rate of return and $1.35E-08$ for risk, which again confirm that there is statistically significant difference between the two distributions. Note that the Holm-Bonferroni correction was again accounted for the above statistical differences.

5.4 Discussion

We can summarise our findings as follows.

Combining technical and sentiment analysis indicators leads to profitable trading strategies, while at the same time maintaining low risk levels. In fact, all variants that combined technical analysis and sentiment analysis (i.e. GP-SATA, STGP-SATA, and STGP-SATA-sum) gave better mean results for Sharpe ratio and rate of return, when compared to the individual TA and SA strategies, i.e. GP-TA and GP-SA. In terms of risk, STGP-SATA and STGP-SATA-sum had the lowest values.

The strongly-typed GP framework ensures that both technical analysis and sentiment analysis indicators are represented in solutions. A weakness of the non-strongly-typed GP-SATA is that it can return individuals with no indicators of SA or TA type. As a result, it's not able to fully take advantage of the fact that its indicator set allows both types of indicators. This has negative effects in its mean performance, as it always performs worse to the STGP variants.

The proposed fitness function ensures that the GP evolves individuals with strong performing SA and TA subtrees, significantly improving the trading performance. This is particularly evident in rate of return (Table 7), where we can observe that STGP-SATA-sum's median value of 0.016 is approximately the sum of GP-SA's median value (0.008) and GP-TA's median value (0.009); indicating that the fitness function has managed to take full advantage of the strengths of the technical and sentiment analysis indicators and essentially 'combine' the performance of thes two algorithms.

6 CONCLUSION

To conclude, we presented a novel strongly typed GP that combines technical and sentiment analysis indicators under a fitness function that takes into account the Sharpe ratios of the individual subtrees. Our algorithm was compared to four other GP algorithms, as well as other financial and machine learning benchmarks, across 35 datasets. The findings showed that the proposed GP statistically and significantly outperforms all other algorithms.

Our results demonstrate the significance of combining indicators from technical and sentiment analysis, towards enhancing the models' knowledge and achieving financially advantageous trading strategies. *It is also evident that simply combining the indicators is not enough and it is important to effectively search the spaces of both types of indicators with a strongly typed architecture.* Finally, we observed how an appropriately defined fitness function can lead to a clearly improved performance.

Future work will focus on adding a third type of terminal set in the strongly-typed GP setting, namely fundamental analysis. This type of analysis forms a different 'school of thought' to technical analysis, and views the performance of a company by looking into its financial statements. It also considers the macroeconomic factors such as interest rates, unemployment rates, and GDP (gross domestic product). It is not common to produce trading strategies that include both fundamental and technical analysis (let alone all three types of analysis), and we believe that doing so under the above GP framework can lead to an even better financial performance. Furthermore, we will include multi-objective optimisation in our research, as a form of comparison and extension of our current novel algorithms.

REFERENCES

[1] Stefano Baccianella, Andrea Esuli, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In *LREC*, Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, Mike Rosner, and Daniel Tapias (Eds.). European Language Resources Association. http://nmis.isti.cnr.it/sebastiani/Publications/LREC10.pdf

[2] José Manuel Berutich, Francisco López, Francisco Luna, and David Quintana. 2016. Robust technical trading strategies using GP for algorithmic portfolio selection. *Expert Systems with Applications* 46 (2016), 307–315.

[3] Anthony Brabazon, Michael Kampouridis, and Michael O'Neill. 2020. Applications of genetic programming to finance and economics: past, present, future. *Genetic Programming and Evolvable Machines* 21, 1 (2020), 33–53.

[4] Eva Christodoulaki and Michael Kampouridis. 2022. Combining Technical and Sentiment Analysis under a Genetic Programming algorithm. In *UK Workshop of Computational Intelligence (UKCI)*.

[5] Eva Christodoulaki and Michael Kampouridis. 2022. Using strongly typed genetic programming to combine technical and sentiment analysis for algorithmic trading. In *2022 IEEE Congress on Evolutionary Computation (CEC)*. 1–8. https://doi.org/10.1109/CEC55065.2022.9870240

[6] Eva Christodoulaki, Michael Kampouridis, and Panagiotis Kanellopoulos. 2022. Technical and Sentiment Analysis in Financial Forecasting with Genetic Programming. In *2022 IEEE Symposium on Computational Intelligence for Financial Engineering and Economics (CIFEr)*. 1–8. https://doi.org/10.1109/CIFEr52523.2022.9776186

[7] Xiao Ding, Yue Zhang, Ting Liu, and Junwen Duan. 2015. Deep learning for event-driven stock prediction. In *Twenty-fourth international joint conference on artificial intelligence.*

[8] Michael Kampouridis, Abdullah Alsheddy, and Edward Tsang. 2013. On the investigation of hyper-heuristics on a financial forecasting problem. *Annals of Mathematics and Artificial Intelligence* 68 (2013), 225–246.

[9] Michael Kampouridis and Fernando EB Otero. 2017. Heuristic procedures for improving the predictability of a genetic programming financial forecasting algorithm. *Soft Computing* 21, 2 (2017), 295–310.

[10] Kazuhiro Kohara, Tsutomu Ishikawa, Yoshimi Fukuhara, and Yukihiro Nakamura. 1997. Stock price prediction using prior knowledge and neural networks. *Intelligent Systems in Accounting, Finance & Management* 6, 1 (1997), 11–22.

[11] Jin Li and Edward PK Tsang. 1999. Improving Technical Analysis Predictions: An Application of Genetic Programming.. In *flairs Conference*. 108–112.

[12] Xinpeng Long, Michael Kampouridis, and Delaram Jarchi. 2022. An in-depth investigation of genetic programming and nine other machine learning algorithms in a financial forecasting problem. In *IEEE Congress on Evolutionary Computation (CEC).*

[13] Steven Loria. 2018. Textblob Documentation. *Release 0.15* 2 (2018), 269.

[14] Mohamed M Mostafa. 2010. Forecasting stock exchange movements using neural networks: Empirical evidence from Kuwait. *Expert Systems with Applications* 37, 9 (2010), 6302–6309.

[15] Abhishek Nan, Anandh Perumal, and Osmar R Zaiane. 2022. Sentiment and knowledge based algorithmic trading with deep reinforcement learning. In *International Conference on Database and Expert Systems Applications*. Springer, 167–180.

[16] David MQ Nelson, Adriano CM Pereira, and Renato A de Oliveira. 2017. Stock market's price movement prediction with Long Short-Term Memory neural networks. In *2017 International joint conference on neural networks (IJCNN)*. IEEE, 1419–1426.

[17] Finn Årup Nielsen. 2011. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. In *Proceedings of the ESWC2011 Workshop on 'Making Sense of Microposts': Big things come in small packages (CEUR Workshop Proceedings, Vol. 718)*, Matthew Rowe, Milan Stankovic, Aba-Sah Dadzie, and Mariann Hardey (Eds.). 93–98. http://ceur-ws.org/Vol-718/paper_16.pdf

[18] Yangtuo Peng and Hui Jiang. 2015. Leverage financial news to predict stock price movements using word embeddings and deep neural networks. *arXiv preprint arXiv:1506.07220* (2015).

[19] Manuel R Vargas, Beatriz SLP De Lima, and Alexandre G Evsukoff. 2017. Deep learning for stock market prediction from financial news articles. In *2017 IEEE International Conference on Computational Intelligence and Virtual Environments for Measurement Systems and Applications (CIVEMSA)*. IEEE, 60–65.

[20] Boyi Xie, Rebecca Passonneau, Leon Wu, and Germán G Creamer. 2013. Semantic frames to predict stock price movement. In *Proceedings of the 51st annual meeting of the association for computational linguistics*. 873–883.

[21] Steve Y Yang, Sheung Yin Kevin Mo, Anqi Liu, and Andrei A Kirilenko. 2017. Genetic programming optimization for a sentiment feedback strength based trading strategy. *Neurocomputing* 264 (2017), 29–41.