IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 23, 2021, accepted July 13, 2021, date of publication August 2, 2021, date of current version August 12, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3101952

Reducing Manual Effort to Label Stock Market Data by Applying a Metaheuristic Search: A Case Study From the Saudi Stock Market

MOHAMMAD ALSULMI
Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11451, Saudi Arabia
e-mail: malsulmi@ksu.edu.sa

This work was supported by the Research Center of College of Computer and Information Sciences, Deanship of Scientific Research, King Saud University, Saudi Arabia.

ABSTRACT Computational intelligence and machine learning techniques have been widely considered for a variety of domains, including financial and data analysis applications. Stock market trading, as a part of the financial domain, has benefited from these techniques in learning models to forecast the direction of stock prices. Traders typically rely on these models, trained using historical stock market data, for monitoring market events to make the right decisions, which can result in more profits while exchanging stocks. One observed limitation of the practices currently being utilized for learning stock market models is the use of manually labeled datasets. This reduces the effectiveness of the data labeling and can cause a lower model prediction accuracy. To address this limitation, this paper proposes an automatic labeling approach that exploits a metaheuristic search to perform the labeling task for stock market data. The results of empirical experiments demonstrate that this approach is very promising as it outperforms the current manual approaches for stock data labeling and achieves higher labeling effectiveness.

INDEX TERMS Automatic labeling, machine learning, metaheuristic search algorithms, stock data labeling, stock market trading.

I. INTRODUCTION

The recent advancements in computational intelligence techniques have led to their wide adaptation for a variety of domains, including the financial analytics domain, typically known as FinTech. Stock market trading, as an instance of FinTech applications, has benefited from emerging technologies in artificial intelligence (AI) and machine learning (ML) to assist financial institutions and stock investors make the right decisions [1]–[3], [13]. Conventionally, the stock trading process involves the frequent monitoring of market events and the movement of stock prices to find good probabilities for purchasing stocks at low prices, and then making profits by selling them at high prices. The process is time-consuming and requires stock traders and investors to conduct extensive research to locate potentially profitable stocks. Consequently, traders typically spend several hours a day monitoring stock movements. However, by leveraging ML models to forecast the stock price directions, stock investors can rely on these models to predict whether a stock price will go up or down the next day. This makes the trading process timely and potentially profitable [4]–[7].

Typically, the models used for stock market prediction tasks are trained using large sets of historical stock market data to predict the price movements. One underlying challenge to the production of these huge datasets is having methods to label the data effectively as the accuracy of the learned models is dependent on the quality of the generated datasets. Although several approaches have been proposed to address this challenge (e.g., in [11]–[13]), many of the approaches are manually realized. In this study, we aimed to address the limitations of the currently used labeling methods by defining the stock data labeling task as an optimization problem. To this end, we first examined several solutions to these problems by employing known classic metaheuristic search algorithms, including hill-climbing [16], [17] and simulated annealing [19]–[21]. Subsequently, we formulated a proposed general framework for automatic labeling that utilizes on these optimization algorithms. Using a case study for the automatic labeling of the Saudi Stock Market (Tadawul) data,

The associate editor coordinating the review of this manuscript and approving it for publication was Huaqing Li.

this approach is shown to be very effective for this task. This is because it improves the labeling performance by over 80% increase in comparison to the currently used manual approaches.

The remainder of this paper is organized as follows. Section II provides an overview of the learning problem to predict the stock price direction while focusing more on the current manual approaches used for labeling the datasets for this task. Section III describes a general framework for automatic labeling considering two classic metaheuristic search algorithms and explains how they can be applied. Section IV presents a case study to evaluate the proposed approach using Saudi Stock Market data and provides an analysis and discussion of the results. Finally, Section V concludes the paper by summarizing the main contributions followed by the future directions of research work.

II. BACKGROUND

ML methods have been widely adopted to forecast the direction of stock market prices [1], [13]. This is achieved using a variety of ML algorithms such as decision trees [2], [3], support vector machines (SVMs) [4], [5], [7], and artificial neural networks (ANNs) [4], [6], [8], [9]. These algorithms generate models that are trained on datasets that represent domain-specific features and labels extracted from past stock trading periods [10], [11]. The learned models are then used to forecast the future stock directions. For example, the models can predict whether a stock price will go up or down the next day. Examining these learners is beyond the scope of this paper; however, this topic will be reviewed for clarity when defining the data labeling problem. Normally, the datasets that are utilized to learn these models are represented as shown below.

$$\boldsymbol{F_i} = \begin{bmatrix} t_1 & f_{1,1} & f_{1,2} & f_{1,3} & \cdots & f_{1,m} \\ t_2 & f_{2,1} & f_{2,2} & f_{2,3} & \cdots & f_{2,m} \\ t_3 & f_{3,1} & f_{3,2} & f_{3,3} & \cdots & f_{3,m} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ t_{n-1} & f_{n-1,1} & f_{n-1,2} & f_{n-1,3} & \cdots & f_{n-1,m} \\ t_n & f_{n,1} & f_{n,2} & f_{n,3} & \cdots & f_{n,m} \end{bmatrix}$$

$$\boldsymbol{Y_i} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ \vdots \\ y_{n-1} \\ y_n \end{bmatrix}$$

Here, $\boldsymbol{S_i} = [\boldsymbol{F_i Y_i}]$ (the merging of the two matrices $\boldsymbol{F_i}$ and $\boldsymbol{Y_i}$) can be considered an entire dataset for stock i, where the size of this set corresponds to a consecutive time series of length n (e.g., n days). A row in the dataset represents a data instance that consists of a vector of features $\left[f_{x,1}\, f_{x,2}\, f_{x,3}\, \cdots f_{x,m}\right]$ and a target label y_x. Each feature $f_{x,y}$ captures a single property of stock i for a given time t_x. It usually represents a statistic about stock i on a given day (e.g., opening price and closing price) or a technical indicator (e.g., moving average, MACD, RSI, and ADX) capturing a trend about the stock that is computed based on the previous stock prices and volume values (see [10], [11] for more details about these technical indicators).

The labels in $\boldsymbol{Y_i}$ indicate the expected direction of the stock price and they are used as the target values for learning. For instance, assuming that this is a binary classification problem, label $y_x \in \{\boldsymbol{UP}, \boldsymbol{DOWN}\}$ can be used to indicate whether the stock price will move up or down for the following day. For the multi-classification problems, y_x can be used to indicate a decision on whether to purchase, sell, or keep monitoring the price of the stock for the following day (hence, $y_x \in \{\boldsymbol{BUY}, \boldsymbol{SELL}, \boldsymbol{MONITOR}\}$). This formulation is generalized beyond daily trading periods as an instance of the dataset that can be used to represent stock movements per hour or per minute within a given time frame (e.g., a day, a week, or a month).

A. PROBLEM FORMULATION

Considering the previous formulation, one underlying challenge is having an adequate scheme for generating the labels that are needed for learning stock prediction models. A method is required to maximize the effectiveness of the generated labels in terms of the profits and losses for a given time. However, this method should be feasible when considering the constraints in computational and storage resources. Thus, this study defines the stock market data labeling problem as follows. Given a dataset for stock i (unlabeled) with a time of length n, this study seeks to label this dataset in such a manner that the generated labels, which are represented by the vector $\boldsymbol{Y_i} = \{y_1, y_2, y_3, \ldots, y_{n-1}, y_n\}$, can maximize an objective function f on the data for stock i.

Assuming that there are k potential values for a given label, the problem is reduced to finding the best sequence of labels of length n while maximizing the objective function f. It is an NP-hard problem in which determining the best solution (i.e., the optimal sequence of the labels) requires examining $\boldsymbol{k^n}$ potential solutions [15]. By having very low n and k values, this problem is managed through the application of an exhaustive search; however, as the values of these two parameters increase, these solutions may become unrealistic and infeasible.

One potential solution to our stock data labeling problem is to conduct a fully manual labeling process by relying on human annotators, which is a common strategy for different ML tasks. Typically, this is achieved by conducting user studies where the human assessors and domain experts are asked to annotate each data instance that satisfies the predefined criteria. For instance, assume that we are conducting multi-class labeling for a given stock (e.g., the labels are BUY, SELL, MONITOR, as explained above). Hence, each assessor is demonstrated with a chart that shows the movement of the stock for a specific time frame. The assessor is then asked to identify the potential buying and selling points from the chart (Figure 1 below illustrates this process), and then each instance is annotated with a suitable label.

FIGURE 1. Illustration of the fully manual process to identify the buying, monitoring, and selling points with green, blue, and red arrows, respectively. The figure was produced by a commercial tool for Saudi stocks trading.

This solution seems to work well for a different category of ML tasks [14]. Nevertheless, because of the volume and velocity in which the stock data are produced, considering such a strategy is becoming difficult and labor-intensive, thus making it impractical and non-scalable for large stock data.

Several attempts have been made to automate this task, as demonstrated in previous studies [11], [12], [27], [28]. In particular, the researchers have proposed annotating the data instances by defining a sliding window with a size of w days. Subsequently, each instance (e.g., a day in the dataset) is annotated based on the accumulated returns (the sum of the daily change in the stock price) for the next w days [11]. In other words, the labeling of a data instance is performed based on a specified threshold v value and the accumulated stock price change for a window in the following days. Although this strategy is broadly applied in the literature, one of its disadvantages is the need to manually fine-tune the w and v values on a target dataset. For instance, in [27], the authors found a window size of four to be a good estimate for this parameter, whereas [28], [11], and [12] suggested five, 20, and 11, respectively, to be more suitable window size values. Throughout this paper, this method is referred to as the manual labeling strategy because of the manual effort involved in setting of the two parameters w and v.

Alternatively, we propose a more attainable solution than the latter two. By considering the underlying labeling problem in this paper as an optimization problem, the application of metaheuristic search algorithms is proposed. This includes applying hill-climbing and simulated annealing to annotate the stock data. Later, these algorithms are described in terms of how they are applied to address this problem, and then they are assessed by comparing them to the manual labeling approaches.

B. METAHEURISTIC SEARCH ALGORITHMS

Metaheuristic search algorithms are a category of algorithms that are adopted to tackle various optimization problems spanning a wide range of domains, including engineering design and computational intelligence domains. In these problems, locating an optimal solution or sub-optimal solution to a given problem can be a difficult task as the size of the search space of the potential solutions can grow rapidly compared to the increase in the input size of the problem (i.e., exponential growth or beyond) [29]. This makes applying exhaustive search by exploring all the potential solutions in the search space of a problem infeasible given the constraints of computational resources. This limitation is overcome by applying metaheuristic search algorithms that provide strategies for guiding the search process to explore large spaces of candidate solutions in an efficient manner.

There are several metaheuristic search approaches, such as hill-climbing [16], simulated annealing [19], tabu search [30], and genetic programming [31]. In this study, we consider two algorithms: hill-climbing and simulated annealing [16], [19], [29]. The simplicity of both algorithms (i.e., few parameters need to be tuned) yet their effectiveness in finding optimal and sub-optimal solutions, as shown in [18], [20], [22], make them very attainable for the stock data labeling problem. We surmise that other metaheuristic approaches such as genetic algorithms can be used as well to optimize our underlying problem, however that is left for future exploration.

The two algorithms, hill-climbing and simulated annealing, are applied throughout an iterative process that seeks to locate solution that is an improvement to a current explored solution [18]. Hill-climbing typically starts with a random state, then it moves to the best neighboring state maximizing a given objective function f, and it continues until no neighboring state is better than the current solution. However, this property of hill-climbing makes it susceptible to getting stuck in a local optimum solution, hence it is not guaranteed to find global optimum solutions [17], [18]. One way to avoid this limit is by applying random restarts in

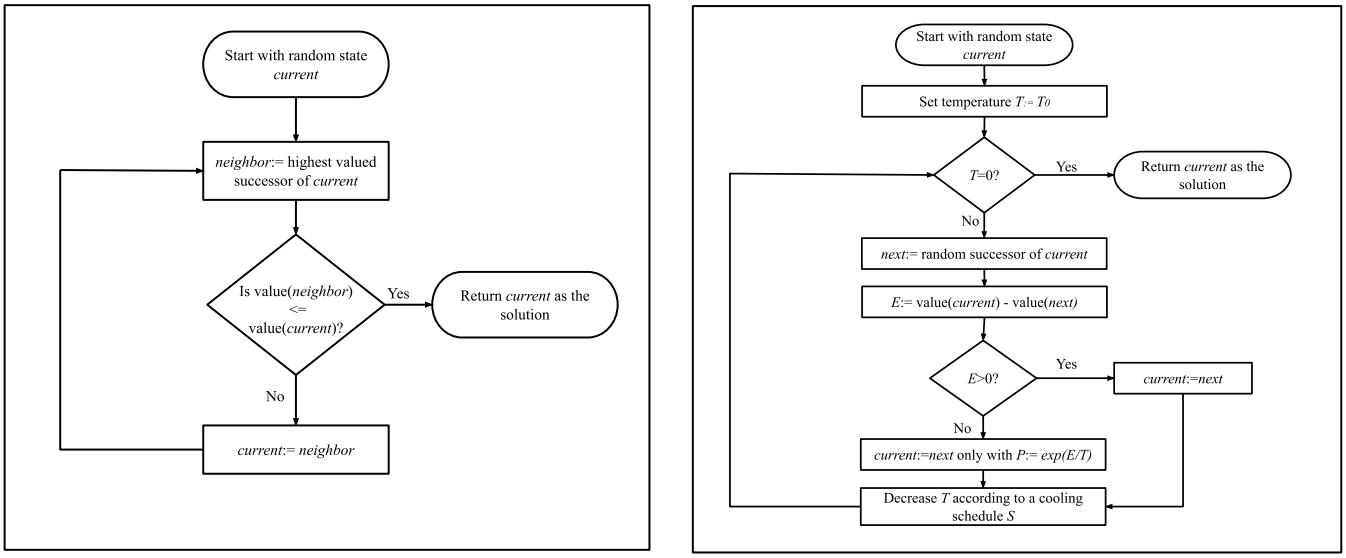


FIGURE 2. Illustration of the respective metaheuristic search procedure for finding solutions to optimization problems by the two algorithms: hill-climbing (left flowchart) and simulated annealing (right flowchart).

which the algorithm is executed for several rounds and stops when no more improvement is observed beyond the best solution.

Simulated annealing, on the other hand, escapes the local optimum states by allowing some random moves (i.e., random walks); however, over time, the frequency of these moves is reduced using the notion of temperature and cooling scheduling (i.e., more frequent moves when the temperature is warm and less frequent moves as the temperature cools) [18]. Simulated annealing has been shown to converge to the global optimum [20]. The detailed search and optimization procedure for each of the two algorithms is illustrated in Figure 2. Later, in Section III, we describe how both hill-climbing and simulated annealing are applied to generate optimum labeling sequences for stock market data.

III. PROPOSED METHODOLOGY

The underlying problem of this study concerns finding a sequence of labels $\boldsymbol{Y_i} = \{\boldsymbol{y_1}, \boldsymbol{y_2}, \boldsymbol{y_3}, \ldots, \boldsymbol{y_{n-1}}, \boldsymbol{y_n}\}$ ($\boldsymbol{y_x}$ can have one of the potential values for k) that maximizes an objective function for the given stock data. For simplicity it is assumed that $k = 3$ such that $\boldsymbol{y_x} \in \{\boldsymbol{BUY}, \boldsymbol{SELL}, \boldsymbol{MONITOR}\}$. It is also assumed that daily trading is considered; thus, $\boldsymbol{Y_i}$ represents labels for n consecutive days. For more efficiency, this study proposes to explore the adaptation of two search algorithms, hill-climbing and simulated annealing, and it applies them in a divide-and-conquer manner. The general framework of this approach is defined as follows.

Step 1 (partitioning) For a given dataset for stock i with successive time series that has the size n (e.g., n days), it is divided into $\frac{n}{l}$ equivalent partitions, where l is the length of a single partition (note that the data for an entire year could be divided into partitions that each represents a month).

Step 2 (initialization). For each partition, a corresponding sequence of labels is generated by randomly assigning label values to one of the three values: $\{\boldsymbol{BUY}, \boldsymbol{SELL}, \boldsymbol{MONITOR}\}$. For instance, if a single partition represents the days within a month period (e.g., the partition size is 30), the randomly generated sequence of labels could look like $\{\boldsymbol{MONITOR}, \mathbf{SELL}, \mathbf{SELL}, \boldsymbol{BUY}, \boldsymbol{MONITOR}, \ldots.., \boldsymbol{BUY}\}$. This is performed by representing this sequence as a vector of integer values where each element in this vector is in the set of (1, 2, and 3) which corresponds to BUY, SELL, and MONITOR labels, respectively. Thus, each element in the vector is initialized by randomly generating a value between 1 and 3 (inclusive). Note that because of this formulation, the boundaries for the generated solutions will be vectors of length l (e.g., 30) such that $\{\mathbf{1}, \mathbf{1}, \mathbf{1}, \mathbf{1}, \mathbf{1}, \ldots.., \mathbf{1}\}$, a vector of all BUY labels, represents the lower bound for the generated solution whereas the vector $\{\mathbf{3}, \mathbf{3}, \mathbf{3}, \mathbf{3}, \mathbf{3}, \ldots.., \mathbf{3}\}$, a vector of all MONITOR labels, is representing the upper bound.

Step 3 (search and optimization) Apply a metaheuristic search algorithm (e.g., hill-climbing or simulated annealing) to each partition and use the randomly generated labels from Step 2 as the initial values. The objective function f is maximized on that partition considering a feature value for the corresponding partition (e.g., using the closing price for the consecutive days as a feature).

Step 4 (aggregation) Aggregate the $\frac{n}{l}$ sequences and merge them into a single labeling sequence while maximizing the objective function f on the partitioning boundaries. Hence, the dataset for n days is ready and labeled.

Steps 1 and 2 are followed by specifying the value of l; they split the data according to l, and then it generates an initial set of labels that are random for each split. For the remaining steps, a further explanation is provided later in this section,

but first, the objective function used in this study is defined below.

A. OBJECTIVE FUNCTION FOR STOCK DATA LABELING

To design an effective function for labeling stock data, the objective function should accurately model the goals of short-term and long-term investors while exchanging stocks. Clearly, investors intend to maximize their profits and minimize the instances for making losses. Thus, this goal should be achievable in that maximizing the value of the objective function correlates with maximizing the goals for stock traders and investors. In this paper, we present a simple way to model this goal as in the objective function shown in Equation (1).

The function $\boldsymbol{f}$ is applied on a sequence of labels given their corresponding stock prices (e.g., the closing price) and it computes the accumulative profitability of that sequence assuming that an entire stock trading process is performed following the labels in that sequence. To evaluate a given sequence of labels using $\boldsymbol{f}$, one first needs to extract each buying–selling pair from the same sequence (i.e., look for each BUY followed by SELL in the sequence and consider it as a pair), match them with their corresponding stock closing price (i.e., extract the price of BUY and SELL in a pair), and then compute the value of $\boldsymbol{f}$. Note that a given sequence of labels $\boldsymbol{Y_i}$ consists of the buying–selling regions to simulate entering a stock, holding it, and then exiting the stock by selling it. Equation (1) is defined as follows:

$$\boldsymbol{f}\ (\boldsymbol{P}) = \sum_{j=1}^{|\boldsymbol{P}|} \left(\boldsymbol{SP_j} - \boldsymbol{BP_j}\right) \tag{1}$$

P is a set of pairs that represent stock buying and selling prices that correspond to the buying–selling regions in a labeled sequence, i.e., $\boldsymbol{P} = \{(\boldsymbol{BP_1}, \boldsymbol{SP_1}), (\boldsymbol{BP_2}, \boldsymbol{SP_2}), \ldots (\boldsymbol{BP_{|P|}}, \boldsymbol{SP_{|P|}})\}$. BP_1 is the price of the stock at the first occurrence of the BUY label in the sequence $\boldsymbol{Y_i}$, whereas SP_1 is the price of the stock at the first occurrence of the SELL label after BP_1. Note that this formulation does not account for any BUY label succeeding BP_1 and comes before SP_1 (or any SELL follows SP_1). This simulates an investor's action of entering and exiting stocks at the earliest opportunities; thus, any pair $\boldsymbol{BP_j}, \boldsymbol{SP_j}$ is computed based on the labels succeeding the pair $\boldsymbol{BP_{j-1}}, \boldsymbol{SP_{j-1}}$. The function provides a good estimate of how well a sequence of labels is obtained by computing the accumulative profits/losses for a given sequence of labels. Hence, the sequence with the highest f value should be the best for labeling the stock data that correspond to this sequence.

A more realistic model of stock trading than Equation (1) can be formulated by averaging the stock buying price at multiple points instead of the first one. This is done to simulate a more conservative trading scheme in which an investor enters a stock during several points of the BUY labels (i.e., splitting the risk of buying at several points instead of one) and then exits by selling at the earliest point. Equation (2) describes this model:

$$\boldsymbol{f}\ (\boldsymbol{P}) = \sum_{j=1}^{|\boldsymbol{P}|} \left\{\boldsymbol{SP_j} - \left(\sum_{t \in \boldsymbol{BP_j}} \frac{\boldsymbol{BP_{j,t}}}{|\boldsymbol{BP_j}|}\right)\right\}. \tag{2}$$

Here, P is represented and defined as the following set: $\left\{\left(\boldsymbol{BP_{1,1}}, \boldsymbol{BP_{1,2}}, .., \boldsymbol{BP_{1,t}}, \boldsymbol{SP_1}\right), .., (\boldsymbol{BP_{|P|,1}}, \boldsymbol{BP_{|P|,2}}, .., \boldsymbol{BP_{|P|,t}}, \boldsymbol{SP_{|P|}})\right\}$. Therefore, f is computed based on an average of the several consecutive buying price points followed by a single selling point for all buying–selling regions in the sequence $\boldsymbol{Y_i}$, as shown in P above. In addition, f can be further extended to model taking some risk while selling a stock (i.e., exiting a stock at multiple selling points instead of one), as demonstrated in Equation (3).

$$\boldsymbol{f}\ (\boldsymbol{P}) = \sum_{j=1}^{|\boldsymbol{P}|} \left\{\left(\sum_{t \in \boldsymbol{SP_j}} \frac{\boldsymbol{SP_{j,t}}}{|\boldsymbol{SP_j}|}\right) - \left(\sum_{t \in \boldsymbol{BP_j}} \frac{\boldsymbol{BP_{j,t}}}{|\boldsymbol{BP_j}|}\right)\right\}, \tag{3}$$

B. LABELING WITH HILL-CLIMBING

By having a partition of labels with the initially assigned values, these were randomly generated as described in Step 2 above. A hill-climbing algorithm [16], [17] was applied to search for an optimum labeling sequence that maximizes the objective function f. The procedure used in the hill-climbing algorithm for the labeling of stock data is as follows.

Step 1. Start with an initial sequence of labels $\boldsymbol{I}$.

Step 2. Generate set S with all the neighbors of $\boldsymbol{I}$. This can be performed by permuting the value of each label in sequence $\boldsymbol{I}$ with one of the labels BUY, SELL, and MONITOR (if $\boldsymbol{I}$ has a length of n, then this roughly generates $2n$ distinct neighbors for $\boldsymbol{I}$).

Step 3. Evaluate the neighbors on the objective function $\boldsymbol{f}$, then find the neighbor with the highest value, assuming it is $\boldsymbol{I'}$.

Step 4. If the fitness value of $\boldsymbol{f}$ on $\boldsymbol{I'}$ is higher than $\boldsymbol{I}$, then assign $\boldsymbol{I} = \boldsymbol{I'}$ and go to Step 2 again; otherwise, return $\boldsymbol{I}$ as the optimum labeling sequence.

As described in Section II.B, the greedy nature of the hill-climbing algorithm imbues it with the potential of escaping the global optimum by being stuck in a local optimum point [17], [18]. It may also fail to find a solution beyond its initial state. To avoid this, this study considered applying a hill-climbing algorithm with random restarts [18] in which the algorithm could be executed for a fixed number of rounds. During each round, a sequence of initial values different from those in the previous rounds is started. This procedure proceeds until there is no change in the returned solution for several rounds or until the maximum number of rounds is reached.

C. LABELING WITH SIMULATED ANNEALING

Simulated annealing (SA) [19]–[21] is yet another algorithm that can be used to find a labeling sequence that maximizes the objective function f. Compared to hill-climbing, SA escapes the local optimum points by allowing some random moves to the neighboring sequences (see Figure 2 for

more details). However, over time, the frequency of these moves gradually decreases. This is maintained by the principle of the initial starting temperature and cooling schedule [18], [21]. The application of simulated annealing to a partition of stock data is accomplished via the following steps.

Step 1. Start with a temperature $\boldsymbol{T} = \boldsymbol{T_0}$ and with an initial sequence of labels $\boldsymbol{I}$.

Step 2. If $\boldsymbol{T}$ equals 0, then $\boldsymbol{I}$ is returned as the optimum labeling sequence.

Step 3. Randomly select one neighbor of $\boldsymbol{I}$ from the set S (containing all the neighbors of $\boldsymbol{I}$) and assume that this neighbor is $\boldsymbol{I'}$.

Step 4. Assume that $\boldsymbol{\Delta E}$ is the fitness value of $\boldsymbol{f}$ on $\boldsymbol{I}$ (the fitness value of $\boldsymbol{f}$ on $\boldsymbol{I'}$).

Step 5. If $\boldsymbol{\Delta E}$ is higher than 0, then assign $\boldsymbol{I} = \boldsymbol{I'}$; otherwise, assign $\boldsymbol{I} = \boldsymbol{I'}$ with a probability value of $\boldsymbol{p} = \boldsymbol{e}^{\frac{\Delta E}{T}}$.

Step 6. Decrease the temperature by assigning $\boldsymbol{T} = \boldsymbol{\alpha T}$, then move to Step 2 again.

The parameters $\boldsymbol{T_0}$ and $\boldsymbol{\alpha}$ in the algorithm above are used to control the speed at which the temperature $\boldsymbol{T}$ is decreased. If $\boldsymbol{T}$ is decreased sufficiently slowly, it is guaranteed to converge to a global optimum [20]. It has been hypothesized that a hill-climbing algorithm with enough random restarts can be as good as applying simulated annealing [18], [22]. This hypothesis is examined later while considering the stock data.

D. AGGREGATION OF THE LABELED PARTITIONS

As indicated in Step 1 in the general framework, the long sequence of n consecutive periods is divided into $\frac{n}{l}$ partitions, and each one is maximized using either hill-climbing or simulated annealing. This is applied to speed up the optimization process, especially when dealing with very long periods. To aggregate those partitions back after generating a sequence of corresponding labels for each, the following process is applied. The partitions are combined according to their timely order, and hill-climbing (with random restarts) is applied again, but this time, only the boundary points of those partitions are maximized. For instance, if a sequence is split into three partitions, then it can be merged by combining the partitions and applying hill-climbing by permuting the values of the four labels that occur at the end of each partition and the start of the next partition. This step is needed to guarantee consistency across the entire sequence and perhaps to maximize the objective function of this sequence even further.

IV. EVALUATION: A CASE STUDY WITH THE SAUDI STOCK MARKET

This section reports and examines the experiments conducted for the automatic annotation of the stock market datasets. The experimental settings are described, which include the dataset and choices made in the experiments. The results of the experiments are then presented and compared with the manual labeling approaches adopted in previous work.

A. EXPERIMENTAL SETTINGS

This study used the Saudi Stock Market (Tadawul) dataset, which is publicly available in [26]. It contains daily historical data, from 1994 to the end of 2017, for over 200 companies listed in the Tadawul market, as well as the data for the TASI market index (the main index for the Saudi Stock Market). Each instance in the dataset is represented by the following attributes: the stock identifier, date, opening price, closing price, high price, low price, and volume. This study followed automatic and manual labeling procedures on the entire dataset using the daily closing price as the main feature. For demonstration purposes, this paper reports only the results for the TASI index (1010) and three companies: Alrajhi Bank (1120), Sabic (2010), and Nadec (6010) (two large companies and one mid-sized company, respectively). This study implemented the labeling procedure for the two algorithms, hill-climbing and simulated annealing, as described in Section III, using the Java programming language. The experiments were conducted on a standard Intel 3.2 GHz CPU with 8 GB of RAM and running the Mac operating system. The function described in Equation (2) was used as the objective function for both algorithms. For hill-climbing, the random start value was set to cn, where n is the length of the sequence that was being optimized, and c is a tunable parameter to maximize the convergence of the algorithm (it was tuned from 1 to 100). For simulated annealing, for simplicity, T_0 was set to a constant value (i.e., 100), and α was set such that $\boldsymbol{\alpha} = \boldsymbol{1} - \frac{\boldsymbol{1}}{\boldsymbol{cn}}$ where c is tuned to control how slowly the temperature is decreased, which leads to the convergence of the global optimum labeling sequence (the value of c was tuned between 1 and 1,000). Notably, this study considered the data instances that correspond to a single year as a dataset for the years from 2014 to 2017. Because this approach is performed in a divide-conquer manner by splitting the data into smaller partitions, each dataset was split into 12 roughly equivalent partitions, and then it was merged back after applying the optimization search for each partition.

To compare the proposed automatic labeling with manual labeling as explained in Section II.A, this investigation implemented a manual labeling procedure as described in [11], [12]. In addition, to achieve more consistency with the automatic labeling, the function defined in Equation (2) was used. The manual labeling procedure relies on two parameters, the window size w and threshold v; thus, both parameters can be tuned by setting their values in such a manner that maximizes the effectiveness of the objective function for each dataset (w is tuned from 4 to 20 and v from 0.01 to 0.05, and the values that maximize the function for a given dataset were chosen).

B. RESULTS AND ANALYSIS

This study was conducted to examine the effectiveness of automatic labeling using a metaheuristic search when it is applied to stock market data. In addition, this investigation sought to evaluate the performance of the proposed approach

in comparison to the approaches included in prior work. Thus, this study attempted to answer the following research questions.

RQ1: Whether automatic labeling based on the metaheuristic search is more effective than manual labeling, which is widely adopted in the literature.

RQ2: Whether partitioning the datasets (i.e., a divide-and-conquer version of the approach) while applying a search for the optimization has a performance impact in comparison to applying no data partitioning.

RQ3: Whether there is a noticeable difference between the two algorithms, hill-climbing and simulated annealing, while considering the task of labeling the stock data.

1) RQ1: COMPARISON WITH MANUAL LABELING

This investigation first addressed the question of whether automatic labeling is more effective than manual labeling, which was adopted in the previous studies [11], [12], [27], [28]. Table 1 shows the performance results of the automatic and manual labeling for four stocks: TASI (1010), Alrajhi (1120), Sabic (2010), and Nadec (6010) using the value of the objective function as the main evaluation metric (i.e., the accumulative profitability in %). The results include the daily trading instances for four years of the dataset (excluding holidays and weekends). Note that for both automatic algorithms, the reported values were averaged over 20 trials.

As Table 1 indicates, automatic labeling that uses an optimization search is far better than labeling with manual approaches. This is true, even when the parameters for manual labeling w and v are maximized for each dataset. Considering the averages of both algorithms for automatic labeling, the improvement over the manual approach is noticeable, ranging between a 63% to 106% increase in effectiveness (on average, automatic labeling leads to an improvement of more than 80%). By conducting hypothesis testing with a pairwise one-sided t-test, it can be observed that this improvement is statistically significant for both algorithms, where $p < 0.0001$. The manual labeling procedure can be executed faster than both automatic labeling algorithms (the slowness of both algorithms is due to the optimization step). Despite this, applying a partitioning-based version of these algorithms still delivers an acceptable execution time, which is no more than 1.3 K milliseconds for each data instance in Table 1. Thus, the minimal increase in the execution time can be tolerated as more gain for the labeling effectiveness is delivered.

We also analyzed the resulting labels from each of the approaches to examine the distribution of the labels for the following three categories: BUY, SELL, and MONITOR. Table 3 presents the results by considering the generated labels for all the datasets. Table 3 shows that hill-climbing and simulated annealing have a comparable distribution of the three categories, with no one category dominating the other categories. More labels were observed as assigned to SELL and MONITOR (almost evenly distributed between the two), while the remaining labels are assigned to the BUY category. Nevertheless, for manual labeling, almost half of the labels are assigned to the SELL category, whereas very few labels (i.e., less than 20%) are assigned to BUY. This may make the data labeled manually inadequate for stock prediction tasks (owing to the unbalanced classes); therefore, data must be balanced before applying learning.

We further analyzed how well each labeling approach is in capturing the movements and trending events of stocks by looking into the resulting labels from these approaches. Figure 3 illustrates the results by plotting the stock price for the entire period of four years and highlighting each data instance with either BUY, SELL, or MONITOR labels for Alrajhi (1120) stock. From Figure 3, we see that indeed automatic labeling with either hill-climbing or simulated annealing correlates very well with the movements of a stock price and results in accurate labels reflecting the stock trending events. It outperforms manual labeling by being able to detect short trading periods and label them with their correct labels (e.g., looking at the first 50 days of 2014, manual labeling failed to detect several buying opportunities and labeled them with SELL and MONITOR only). Overall, the analysis performed in this study suggests that automatic labeling is indeed more effective than manual labeling because it generates datasets with higher labeling effectiveness; therefore, they can be more suitable for ML tasks.

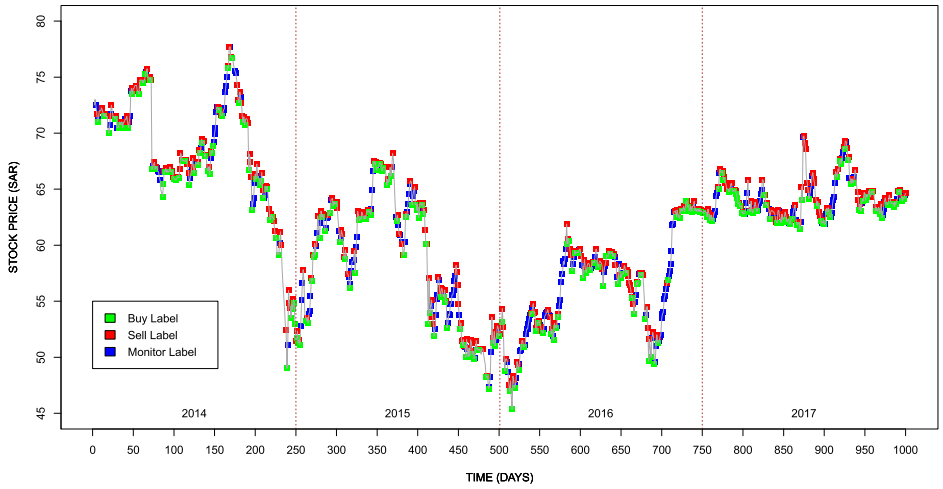
2) RQ2: PARTITIONING VERSUS NO PARTITIONING

Further, this study addressed the question of whether a divide-and-conquer version for this approach, which was performed by splitting each dataset into smaller partitions and applying optimization to each partition, is more effective than applying optimization with no data partitioning. Tables 1 and 2 present the results of the two cases. By observing the results from both tables, there is a slight improvement in the non-partitioning case over data partitioning. More specifically, in terms of the labeling effectiveness, the improvement, on average, is no more than 4% for hill-climbing and simulated annealing, but this improvement is statistically significant. This suggests that although the difference in effectiveness between the two cases is minimal, it is consistent across all the datasets reported in Tables 1 and 2.

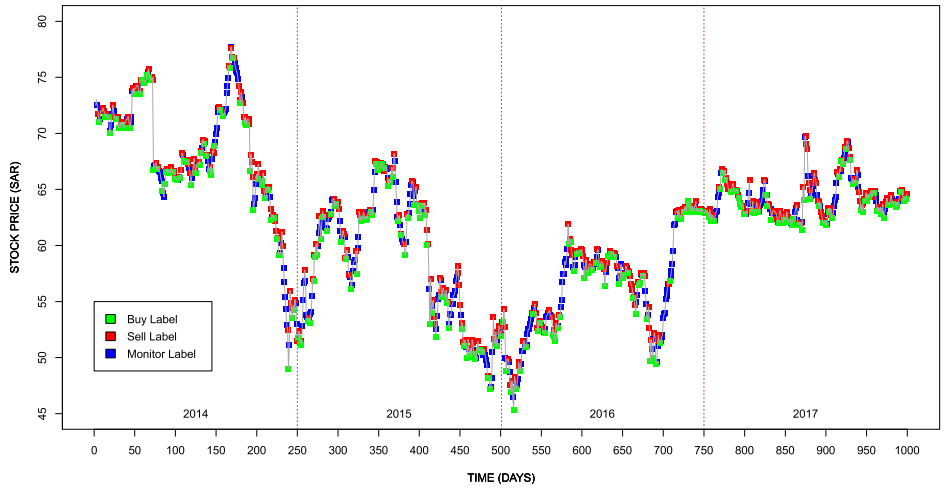
In terms of the efficiency of both cases, the outcomes are slightly different. By conducting a small-scale study where the execution times of the two versions of the algorithms are compared, it seems that labeling with data partitioning has a noticeable advantage because it is approximately 11 times faster than labeling without partitioning the data. It can be observed that the hill-climbing algorithm takes about 1.3 K milliseconds to label a given dataset if the data are partitioned, whereas it takes over 14 K milliseconds if no partitioning is applied. This analysis suggests that there is a tradeoff between the effectiveness and efficiency for automatic labeling because gaining has more effectiveness in labeling, which has some associated costs. Thus, although there is a slight improvement in the case of labeling without partitioning the data, it can be traded for a greater increase in the speed of the labeling procedure execution.

TABLE 1. Effectiveness results for the automatic labeling (with data partitioning) and manual labeling approaches. Bold-faced values indicate the approach that results in the best labeling sequence and earned the highest value on the objective function.

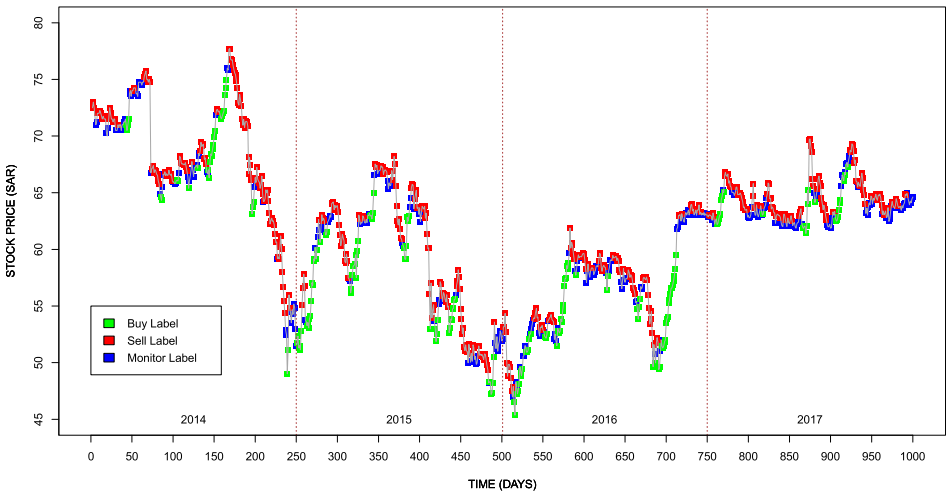
Stock Identifier	Number of Instances	Year	Results	Hill-Climbing	Simulated Annealing	Manual Labeling
1010	250	2014	Best	**136.1**	**136.1**	71.9
			Worst	**132.6**	131.0	___
			Average	**134.1**	133.6	___
1010	251	2015	Best	88.1	**89.1**	48.2
			Worst	86.1	**86.2**	___
			Average	87.4	**87.6**	___
1010	249	2016	Best	**89.4**	**89.4**	43.8
			Worst	85.4	**86.2**	___
			Average	87.7	**88.0**	___
1010	250	2017	Best	**76.3**	76.2	73.9
			Worst	73.6	**74.2**	___
			Average	75.1	**75.3**	___
1120	250	2014	Best	90.7	**91.4**	56.0
			Worst	88.6	**88.9**	___
			Average	**89.9**	**89.9**	___
1120	251	2015	Best	167.4	**168.1**	96.1
			Worst	**162.7**	161.4	___
			Average	**165.9**	165.5	___
1120	249	2016	Best	**132.1**	**132.1**	80.1
			Worst	**128.1**	**128.1**	___
			Average	**131.5**	131.4	___
1120	250	2017	Best	**86.2**	86.1	44.1
			Worst	**83.6**	83.1	___
			Average	84.9	**85.0**	___
2010	250	2014	Best	**115.4**	115.1	71.1
			Worst	113.2	**113.3**	___
			Average	**114.5**	**114.5**	___
2010	251	2015	Best	202.2	**202.6**	109.5
			Worst	**195.3**	194.9	___
			Average	199.2	**199.7**	___
2010	249	2016	Best	**156.9**	156.7	90.0
			Worst	**153.8**	153.6	___
			Average	**155.6**	155.5	___
2010	250	2017	Best	66.0	**66.2**	32.9
			Worst	62.1	**62.9**	___
			Average	64.1	**64.5**	___
6010	250	2014	Best	**197.3**	197.2	116.5
			Worst	**190.3**	189.9	___
			Average	195.4	**195.7**	___
6010	251	2015	Best	205.2	**205.4**	114.0
			Worst	195.3	**197.6**	___
			Average	200.5	**201.6**	___
6010	249	2016	Best	196.6	**198.4**	130.2
			Worst	**187.7**	185.1	___
			Average	**192.8**	192.5	___
6010	250	2017	Best	183.7	**184.2**	118.7
			Worst	**177.2**	176.7	___
			Average	**181.3**	181.2	___



(a) Labeling using Hill-Climbing.



(b) Labeling using Simulated annealing.



(c) Labeling using manual labeling.

FIGURE 3. Resultant labels for the Alrajhi (1120) stock datasets (from 2014 to 2017) with Buy, Sell, and Monitor labels using automatic labeling (hill-climbing and simulated annealing), and manual labeling.

TABLE 2. Effectiveness results for automatic labeling (no data partitioning). Bold-faced values indicate the approach that results in the best labeling sequence and earned the highest value on the objective function.

Stock Identifier	Number of Instances	Year	Results	Hill-Climbing	Simulated Annealing
1010	250	2014	Best	137.4	**137.6**
			Worst	**137.0**	136.3
			Average	**137.1**	136.7
1010	251	2015	Best	**89.3**	88.9
			Worst	**88.2**	88.0
			Average	**88.6**	**88.6**
1010	249	2016	Best	**91.0**	90.7
			Worst	**90.1**	89.9
			Average	**90.5**	90.4
1010	250	2017	Best	**77.4**	**77.4**
			Worst	**76.7**	76.3
			Average	**77.1**	76.9
1120	250	2014	Best	**95.3**	**95.3**
			Worst	**94.6**	94.3
			Average	**95.0**	94.7
1120	251	2015	Best	**168.2**	167.7
			Worst	**166.5**	166.4
			Average	**167.3**	167.2
1120	249	2016	Best	136.1	**136.2**
			Worst	**134.7**	133.7
			Average	**135.2**	134.5
1120	250	2017	Best	**88.3**	87.4
			Worst	**87.5**	86.7
			Average	**87.8**	87.1
2010	250	2014	Best	119.7	**120.0**
			Worst	**119.2**	119.1
			Average	**119.4**	**119.4**
2010	251	2015	Best	**201.0**	200.5
			Worst	**199.1**	198.5
			Average	**200.1**	199.6
2010	249	2016	Best	159.3	**159.9**
			Worst	**158.0**	157.7
			Average	**158.7**	158.4
2010	250	2017	Best	**68.4**	68.2
			Worst	**67.8**	67.3
			Average	**68.1**	67.7
6010	250	2014	Best	**200.4**	200.3
			Worst	**199.5**	199.3
			Average	**200.0**	199.8
6010	251	2015	Best	**206.4**	206.1
			Worst	**204.7**	204.1
			Average	**205.4**	205.2
6010	249	2016	Best	**217.8**	217.4
			Worst	**216.3**	214.2
			Average	**217.1**	215.8
6010	250	2017	Best	194.3	**194.4**
			Worst	**193.9**	193.1
			Average	**194.1**	193.7

TABLE 3. Distribution of the generated labels among BUY, SELL, and MONITOR for each labeling approach.

Label Distribution (%)			Approach
BUY	SELL	MONITOR	
25.38%	39.55%	35.07%	Hill-Climbing
25.72%	37.05%	37.23%	Simulated Annealing
19.02%	48.87%	32.11%	Manual Labeling

3) RQ3: HILL-CLIMBING VERSUS SIMULATED ANNEALING

Lastly, this investigation attempted to address the question of whether there is a major difference between the two optimization algorithms considered in this study when it is applied to label stock data. Generally, it is believed that hill-climbing with random restarts is as effective as simulated annealing when it is used to solve a variety of optimization problems [18], [22]. Some other factors may affect the performance of these algorithms, such as how the parameter values of the algorithms are assigned. Several techniques have been introduced to guide the setting of the initial temperature parameter value T_0 [23] for simulated annealing, as well as some suggestions on how the temperature reduction schedules [24], [25] are initialized. Nevertheless, for simplicity and to allow easy comparison between the two algorithms, this study tuned the parameters of the two algorithms as indicated in Section IV.A, (tuning c for hill-climbing shows no impact when it is set to a value more than one, whereas for the simulated annealing, the improvement stops as the c value reaches 1,000).

The results reported in Tables 1 and 2 suggest that both algorithms are comparable for stock labeling tasks. In fact, for some instances, the first algorithm outperforms the second, whereas for some other instances, the second outperforms the first (the difference between the two algorithms is statistically significant for Table 1, and it can be noted that some are consistent but there is a very minor improvement in hill-climbing as shown in Table 2). Moreover, by looking at Figure 3, it can be noticed that the two algorithms appear to result in almost identical labels for the period of four years. Therefore, a similar conclusion can be drawn to that reported previously in the literature [18], [22], in that if the parameters for the two algorithms are finely tuned, they tend to act similarly.

V. CONCLUSION

Various manual approaches have been proposed for the stock data labeling problem. They are widely utilized in the literature to label stock datasets, which are then used in training models for predicting stock movements in future times. This study defined the stock data labeling problem and formulated it as an NP-hard problem. An automatic labeling solution was then proposed that uses metaheuristic search algorithms such as hill-climbing and simulated annealing. This solution demonstrates that it is very promising and outperforms manual approaches with high effectiveness.

One limitation of this study is that it does not address the question of whether the improvement in labeling effectiveness will correlate into higher prediction accuracy when learning ML prediction models using the data labeled with this automatic approach. Therefore, the future directions of our research will address this limitation and include the conducting of further analyses of this approach. More specifically, the resulting labeled datasets from this approach will be used to train the ML models to predict the future stock movements and then their performance in terms of profitability and making higher investment returns will be evaluated. We are optimistic that the conclusions drawn from this study will aid in the stock movement prediction learning task.

REFERENCES

[1] J. Patel, S. Shah, P. Thakkar, and K. Kotecha, "Predicting stock market index using fusion of machine learning techniques," *Expert Syst. Appl.*, vol. 42, no. 4, pp. 2162–2172, Mar. 2015.
[2] M. Miró-Jul, G. Fiol-Roig, and A. P. Isern-Deyà, "Decision trees in stock market analysis: Construction and validation," in *Proc. IEA-AIE*, Berlin, Germany, 2010, pp. 185–194.
[3] S. Basak, S. Kar, S. Saha, L. Khaidem, and S. R. Dey, "Predicting the direction of stock market prices using tree-based classifiers," *North Amer. J. Econ. Finance*, vol. 47, pp. 552–567, Jan. 2019.
[4] A. F. Sheta, S. E. Ahmed, and H. A. Faris, "Comparison between regression, artificial neural networks and support vector machines for predicting stock market index," *Soft Comput.*, vol. 7, no. 8, p. 2, 2015.
[5] C.-Y. Yeh, C.-W. Huang, and S.-J. Lee, "A multiple-kernel support vector regression approach for stock market price forecasting," *Expert Syst. Appl.*, vol. 38, no. 3, pp. 2177–2186, Mar. 2011.
[6] E. Chong, C. Han, and F. C. Park, "Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies," *Expert Syst. Appl.*, vol. 83, pp. 187–205, Oct. 2017.
[7] M. Sedighi, H. Jahangirnia, M. Gharakhani, and S. F. Fard, "A novel hybrid model for stock price forecasting based on metaheuristics and support vector machine," *Data*, vol. 4, no. 2, p. 75, May 2019.
[8] X. Pang, Y. Zhou, P. Wang, W. Lin, and V. Chang, "An innovative neural network approach for stock market prediction," *J. Supercomput.*, vol. 76, no. 3, pp. 2098–2118, Jan. 2018.
[9] M. Qiu and Y. Song, "Predicting the direction of stock market index movement using an optimized artificial neural network model," *PLoS ONE*, vol. 11, no. 5, May 2016, Art. no. e0155133.
[10] G. R. Weckman and S. Lakshminarayanan, "Identifying technical indicators for stock market prediction with neural networks," in *Proc. IIE Annu. Conf.*, Houston, TX, USA, 2004, pp. 1–6.
[11] Y. Alsubaie, K. E. Hindi, and H. Alsalman, "Cost-sensitive prediction of stock price direction: Selection of technical indicators," *IEEE Access*, vol. 7, pp. 146876–146892, 2019.
[12] O. B. Sezer and A. M. Ozbayoglu, "Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach," *Appl. Soft Comput.*, vol. 70, pp. 525–538, Apr. 2018.
[13] A. Gupta and D. S. D. Sharma, "A survey on stock market prediction using various algorithms," *Int. J. Comput. Technol. Appl.*, vol. 5, no. 2, pp. 530–533, Apr. 2014.
[14] Y. Roh, G. Heo, and S. E. Whang, "A survey on data collection for machine learning: A big data—AI integration perspective," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 4, pp. 1328–1347, Apr. 2021.
[15] J. V. Leeuwen, "Algorithms and complexity," in *Handbook of Theoretical Computer Science. Cambridge, MA, USA: MIT Press*, 1990.
[16] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Oper. Res.*, vol. 21, no. 2, pp. 498–516, 1973.
[17] B. Selman and C. P. Gomes, "Hill-climbing search," *Encyclopedia Cognit. Sci.*, vol. 81, no. 1, pp. 333–335, 2006.
[18] P. Norvig and S. J. Russell, *Artificial Intelligence: A Modern Approach*. Upper Saddle River, NJ, USA: Prentice-Hall, 2020.

[19] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, May 1983, Art. no. 6710680.
[20] V. Granvillem, M. Krivanek, and J.-P. Rasson, "Simulated annealing: A proof of convergence," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, no. 6, pp. 652–656, Jun. 1994.
[21] D. Bertsimas and J. Tsitsiklis, "Simulated annealing," *Statist. Sci.*, vol. 8, no. 1, pp. 10–15, 1993.
[22] S. H. Jacobson and E. Yücesan, "Analyzing the performance of generalized Hill climbing algorithms," *J. Heuristics*, vol. 10, no. 4, pp. 387–405, Jul. 2004.
[23] H. H. Shakouri, K. Shojaee, and M. T. Behnam, "Investigation on the choice of the initial temperature in the simulated annealing: A mushy state SA for TSP," in *Proc. 17th Medit. Conf. Control Autom.*, Jun. 2009, pp. 1050–1055.
[24] J. Stander and B. W. Silverman, "Temperature schedules for simulated annealing," *Statist. Comput.*, vol. 4, no. 1, pp. 21–32, Mar. 1994.
[25] R. K. Samuel and P. Venkumar, "Optimized temperature reduction schedule for simulated annealing algorithm," *Mater. Today, Proc.*, vol. 2, nos. 4–5, pp. 2576–2580, 2015.
[26] *Money Expert Institute, Saudi Market Dataset*. Accessed: Apr. 4, 2021. [Online]. Available: https://data.mec.biz/saudi_mec
[27] S. O. Olatunji and M. S. Alahmadi, "Forecasting the Saudi Arabia stock prices based on artificial neural networks model," *Int. J. Intell. Inf. Syst.*, vol. 5, no. 2, pp. 77–86, Oct. 2013.
[28] M. Usmani, S. H. Adil, K. Raza, and S. S. A. Ali, "Stock market prediction using machine learning techniques," in *Proc. 3rd Int. Conf. Comput. Inf. Sci. (ICCOINS)*, Aug. 2016, pp. 322–327.
[29] X. Yang, "Metaheuristic optimization," *Scholarpedia*, vol. 6, no. 8, pp. 1–15, Jan. 2011.
[30] F. Glover, "Tabu search—Part I," *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, Aug. 1989.
[31] J. H. Holland, "Genetic algorithms and adaptation," in *Adaptive Control of Ill-Defined Systems*, vol. 16. Boston, MA, USA: Springer, 1984, pp. 317–333.

MOHAMMAD ALSULMI received the B.S. degree in computer science from King Saud University, Riyadh, Saudi Arabia, in 2011, and the M.S. and Ph.D. degrees in computer and information sciences from the University of Delaware, Newark, DE, USA, in 2014 and 2018, respectively. He is currently an Assistant Professor of computer science with the Department of Computer Science, College of Computer and Information Sciences, King Saud University. Prior to joining King Saud University, he was a Graduate Student Research Assistant with the Information Retrieval Laboratory, University of Delaware, where he worked in several research projects involving high performance computing, search systems, and clinical decision support. He has published several articles in a number of venues, including IEEE BIBM, ACM IUI, NIST TREC, and ICCS. His research interests include machine learning applications in financial and medical domains, text mining analytics, and search and information retrieval.