SYSTEMS AND METHODS FOR ALLOCATING CAPITAL TO TRADING STRATEGIES FOR BIG DATA TRADING IN FINANCIAL MARKETS

CROSS-REFERENCE TO RELATED APPLICATIONS

[0001] The present application claims the benefit and priority of U.S. Provisional Application Serial No. 61/949,938 filed on March 7, 2014, titled "Systems and Methods for Big Data Trading in Financial Markets," which is hereby incorporated by reference.

[0002] The present application is related to U.S. Non Provisional Application Serial No. __/___,___ filed concurrently herewith, titled "Systems and Methods for Generating and Selecting Trading Algorithms for Big Data Trading in Financial Markets," which is hereby incorporated by reference.

FIELD OF THE TECHNOLOGY

[0003] The embodiments disclosed herein are related to systems and methods for using big data analysis for trading in financial markets.

SUMMARY

[0004] Provided herein are exemplary systems and methods for allocating capital to trading strategies, including generating a virtual machine by a master virtual machine server for a trading strategy in a historical server, obtaining historical performance data for the trading strategy from the historical server, transforming by the virtual machine the historical performance data into metrical data, transforming by the virtual machine the historical performance data and metrical data into a neural network usable data set, creating by the virtual machine a neural network base, and forming by the virtual machine a neural network.

[0005] Exemplary systems and methods may further comprise training by the virtual machine a neural network for one or more data points, calculating by the virtual machine an error rate for the one or more data points until the error rate stops converging or cannot converge, saving by the virtual machine the neural network, testing by the virtual machine the saved neural network against updated metrical and historical data, training by the virtual machine the saved neural network, saving by the virtual machine the neural network as a binary object, transmitting by the virtual machine the binary object to the historical server, activating by a fusion server the neural network, obtaining by the fusion server historical metrical and historical performance data, calculating by the fusion server a confidence value, determining by the fusion server whether to execute a trade, performing by the fusion server a survey, determining by the fusion server an order to send to an exchange, sending by the fusion server the order to the exchange and updating by the fusion server to reflect an executed order.

[0006] An exemplary system for allocating capital to trading strategies may include a means for generating a virtual machine for a trading strategy in a historical server, a means for obtaining historical performance data for the trading strategy

from the historical server, a means for transforming the historical performance data into metrical data, a means for transforming the historical performance data and metrical data into a neural network usable data set, a means for creating a neural network base, and a means for forming a neural network. According to further exemplary embodiments, the system may include a means for training by the neural network for one or more data points, a means for calculating an error rate for the one or more data points until the error rate stops converging or cannot converge, a means for saving the neural network, a means for testing the saved neural network against updated metrical and historical data, a means for training the saved neural network, a means for saving the neural network as a binary object, a means for transmitting the binary object to the historical server, a means for activating the neural network, a means for obtaining historical metrical and historical performance data, a means for calculating a confidence value, a means for determining whether to execute a trade, a means for performing a survey, a means for determining an order to send to an exchange, a means for sending the order to the exchange, and a means for updating to reflect an executed order.

BRIEF DESCRIPTION OF THE DRAWINGS

[0007] FIGS. 1A-1C are a flowchart of an exemplary method for allocating capital to trading strategies.

[0008] FIG. 2 is a diagram of an exemplary system for implementing embodiments of the present technology.

DETAILED DESCRIPTION OF EXEMPLARY EMBODIMENTS

[0009] FIGS. 1A-1C are a flowchart of an exemplary method for allocating capital to trading strategies.

[0010] At step 101, a virtual machine is generated by a master virtual machine server for a trading strategy in a historical server.

[0011] According to various exemplary embodiments, a virtual machine may comprise an emulation of a particular computer system. Virtual machines operate based on the computer architecture and functions of a real or hypothetical computer, and their implementations may involve specialized hardware, software, or a combination of both.

[0012] In certain exemplary embodiments, a master virtual machine server may comprise a single server responsible for generating all of or most of the herein described virtual machines.

[0013] A historical server according to some exemplary embodiments may comprise trading algorithms, trading strategies or "bots" that meet minimum standards as stored in a database and/or in a data warehouse.

[0014] For example, a cloud manager may be a custom application that manages trading strategies or algorithms. The cloud manager is configured to the cluster of cloud computing instances for processing large amounts of data. The cloud manager serves as the user interface to handle the ordering and cancelling of virtual computing instances. Additionally, the cloud manager may allow for detailed customization of the virtual machines. For example, Random Access Memory ("RAM"), processor speed, number of processors, network details, security/encryption, and/or memory may be detailed for each virtual machine and/or all virtual machines. Once the cluster of cloud computing instances is

ordered and running, the cloud manager is "listening" for idle machines and "assigning" any idle machine a trading strategy for analyzing.

[0015] A cloud-based computing environment is a resource that typically combines the computational power of a large grouping of processors and/or that combines the storage capacity of a large grouping of computer memories or storage devices. For example, systems that provide a cloud resource may be utilized exclusively by their owners; or such systems may be accessible to outside users who deploy applications within the computing infrastructure to obtain the benefit of large computational or storage resources.

[0016] For example, from a 3rd party cloud provider, an order is placed to create virtual machine (vm) based of an image of a stored template vm with required specifications and name it "VM1."

[0017] At step 102, each virtual machine generated at step 101 obtains historical performance data for its respective strategy from the historical server.

[0018] Historical performance data may include price and volume data at any previous point in time for a tradable item in any market or exchange.

[0019] At step 103, each virtual machine transforms the historical performance data into metrical data.

[0020] Metrical data includes total profit over 1 year, percentage of profitable trades over a time period, how much was gained or lost in each trade, percentage of profitable trades in a bear or bull market, correlation to other indexes, ratio of profitable trades to cover largest loss, etc. For instance, % of profitable trades (= profitable trades / total number of trades) may be used as metrical data.

[0021] At step 104, each virtual machine transforms the historical performance data and metrical data into a neural network usable data set.

[0022] In some exemplary embodiments, a neural network usable data set comprises an aggregation of data points organized into one or more sets.

[0023] For example, for a particular time period, such as each day (or each minute, hour, month, year(s), decades, bull cycle, bear cycle, etc.), for any particular tradable item (for example, gold), the historical data and metrical data will be grouped together as a data set. Thus, for the tradable item of gold, in a particular market or exchange, for a particular calendar day, one may have a neural network usable data set comprising the historical data of $300 per ounce, volume traded, day high price and day low price, tick by tick changes in said historical data and the metrical data of year to date price performance of 25%.

[0024] At step 105, each virtual machine creates a neural network base. The neural network base, along with the neural network usable data set created at step 104, forms a neural network.

[0025] Creating a neural network base, according to some exemplary embodiments, includes processing a layer of input data and then processing that data algorithmically to compare the output of the algorithms against known output.

[0026] A neural network base may comprise of set of algorithms used to approximate against data inputs. These algorithms are able to store tested values within itself and store an error rate.

[0027] A neural network may comprise a neural network base and its underlying algorithms in tandem with a neural network usable data set. The neural network usable data set may function as a data feeder for the neural network base to calculate and/or otherwise interact with.

[0028] In various exemplary embodiments, feeding of a data point may be performed by the neural network iteratively requesting to process the next data point in the usable data set.

[0029] Data points, according to many exemplary embodiments, may include but is not limited to metrical data and historical data that the neuro network has access to as part of its useable data set.

[0030] At step 106, each virtual machine trains its respective neural network (and the neural network learns) for the one or more data points.

[0031] The neural network base, in some exemplary embodiments, uses the data points to systematically approximate solutions to the internal algorithms and then outputs an error rate for each (new) data point of the useable data set.

[0032] According to some embodiments, a neural network base comprises various algorithms designed to interact with a data set in order to determine if a particular trading strategy should be employed during the next trading period. For example, the neural network base may determine a progressively rising price pattern for a particular tradable item when a particular trading strategy is employed. Continuing this example, suppose the particular trading strategy resulted in profits in trading gold for the last three trading periods comprising $300, $325 and $350. The neural network base will determine there is a rising profit pattern and indicate that the particular strategy should be employed for the next trading period.

[0033] Training, according to certain exemplary embodiments, includes improving the ability of a neural network to generalize outputs and responses through use of "noisy" data that includes small random variations from previously learned solutions. Note that an over trained network gets better at matching the training set of patterns to the desired output, but it may do a poorer job of generalization.

[0034] An example of training by each virtual machine: Imagine a strategy that involves a teacher that is smarter than the network itself. For example, facial recognition. The teacher shows the network a bunch of faces, and the teacher already knows the name associated with each face. The network makes its guesses, then the teacher provides the network with the answers. The network can then compare its answers to the known "correct" ones and make adjustments according to its errors.

[0035] Most of the algorithms used in training artificial neural networks employ some form of gradient descent, using backpropagation to compute the actual gradients. This is done by taking the derivative of the cost function with respect to the network parameters and then changing those parameters in a gradient-related direction. Evolutionary methods, gene expression programming, simulated annealing, expectation-maximization, non-parametric methods and particle swarm optimization are methods for training neural networks.

[0036] For example, a neural network for handwriting recognition is defined by a set of input neurons which may be activated by the pixels of an input image. After being weighted and transformed by a function (determined by the network's designer), the activations of these neurons are then passed on to other neurons. This process is repeated until finally, an output neuron is activated. This determines which character was read.

[0037] Like other machine learning methods - systems that learn from data - neural networks have been used to solve a wide variety of tasks that are hard to solve using ordinary rule-based programming, including computer vision and speech recognition.

[0038] In the present case, a strategy with a known historical performance is being run in real time to determine if past performance (and/or under what conditions) will result in the same or a better future performance.

[0039] An example of step 106, with reference to step 105, where the particular trading strategy resulted in profits in trading gold for the last three trading periods respectively comprising $300, $325 and $350, the neural network running on the virtual machine predicted a profit for the next trading period of $375. In the next trading period, however, the actual profit turned out to be $360. Accordingly, the neural network will adjust downward (i.e. learn from the virtual machine) the projected profit for the day following the day when the actual profit was $360.

[0040] At step 107, an error rate for the one or more data points is observed until the error rate stops converging or cannot converge.

[0041] In some exemplary embodiments, an error rate is defined as closeness of the approximation of the neural network base where 0 is an exact approximation and greater than or equal to 1 is an incorrect network. Where the error rate is updated for each (new) data point.

[0042] Stopping of the converging of the error rate or the inability of the error rate to converge may be observed if the previous error rate is larger than the currently calculated error rate or the error rate has not changed.

[0043] An example of step 107 may include providing a solution to the neuro network where the first iteration produces an error rate of 0.01. The second iteration produces an error rate of 0.001. The third iteration produces an iteration of 0.0001. However, the fifth iteration produces an error rate of 0.01—which is larger than the previously calculated error rate.

[0044] Another example of step 107, with reference to steps 105 and 106, when the profit for trading gold using a particular trading strategy produced profits of $300, $325, $350 and predicted a profit of $375, when the profit actually turned out to be $365, an error rate of ($375-$365 = $10)/(projected profit = $25) = 40% results. Continuing this example, the error rate will continue to be calculated for subsequent trading days until the error rates stops going down.

[0045] At step 108, if the observed error rate does not converge, the neural network is dropped and marked as "unusable" for the one or more historical data and metrical data points for the tradable item employing a particular strategy.

[0046] According to some exemplary embodiments, dropping and marking of the neural network as unusable may include disposing of the neural network and the neural network usable data set, and reporting to the data provider that the neural network has failed.

[0047] If the error rate does converge, the neural network is saved and marked as "usable."

[0048] At step 109, the saved usable neural network is tested against updated historical and metrical data set. According to some embodiments, an updated historical and metrical data set comprises historical and metrical data collected since the time the virtual machine created the neural network at step 105.

[0049] At step 110, for each time-frame in the updated historical and metrical data, the neural network is retrained until it stops converging. In various exemplary embodiments, retraining may include using the updated historical and metrical data as new inputs to the neural network for it to create an updated learning pattern.

[0050] For example, in the facial recognition example from above, giving the neuro network a new set of faces (i.e. data points) will allow the neural network to learn, classify, and update its knowledge of facial characteristics.

[0051] For example, referring to step 107, when the actual trading day profit produced by using a particular trading strategy in gold comprised $300, $325, $350 for a sequence of trading days, and the trading day profit was predicted to be $375 for the next trading day, and when the actual profit turned out to be $365, the neural network base may be retrained to lower its future expectations. Accordingly, the error rate will for the retrained neural network will be observed for convergence patterns as discussed herein.

[0052] At step 111, the neural network is saved with the one or more data points as a binary object. According to certain exemplary embodiments, a binary object may be defined as digital data stored in any unspecified format.

[0053] An example of performing step 111 may comprise generating an output of information and storing it digitally as a .jpg, .exe, .nc, .wav, .tar or otherwise known digital format.

[0054] At step 112, each virtual machine transmits the binary object to the historical server.

[0055] At step 113, a fusion server activates the neural network as stored in the historical server as the binary object.

[0056] In various exemplary embodiments, a fusion server may comprise a set of algorithms and/or databases, which reside on one server machine that is designed to select, access, and then combine sets of information from multiple neuro networks.

[0057] Activating a neural network, in certain exemplary embodiments, includes accessing a specific neuro network for components such as its input data, test values, stored weights, and other needed data items.

[0058] An example of performing step 113 may comprise accessing a neuro network for its internally stored weighted values.

[0059] At step 114, the neural network obtains updated historical and/or metrical data from the historical server.

[0060] In various exemplary embodiments, updated historical and/or metrical data is defined as data or data sets that exist externally to the historical server. According to other exemplary embodiments, updated historical and/or metrical data is historical data and/or metrical data obtained after the neural network was saved as a binary object at step 111.

[0061] At step 115, the neural network calculates a confidence value based upon the updated historical and/or metrical data.

[0062] In certain exemplary embodiments, a confidence value may be defined as a real number in the interval set of [0, 1] where 0 indicates no confidence and 1 indicates perfect confidence.

[0063] The calculation of the confidence value, according to some exemplary embodiments, is performed by the neural network base.

[0064] An example of performing step 115 includes a *Strategy 1* receiving a confidence value of 1 indicating trading confidence, so it will be considered as a component in a basket to be allocated trading capital on the next available trading session. In contrast, *Strategy 2* received a confidence value of 0 indicating no trading confidence, so it will not be considered as a component in a basket to be allocated trading capital on the next trading available session.

[0065] At step 116, the Fusion server obtains the calculated confidence value and determines whether to execute a trade.

[0066] According to various exemplary embodiments, the Fusion server accesses the calculated confidence value through activation of the neuro network.

[0067] Determining whether to execute a trade, according to some exemplary embodiments, is performed by algorithmic calculations performed by the Fusion server.

[0068] An example of performing step 116 includes the fusion server changing a trading strategy's exposure from FLAT to LONG when the activated output is high enough.

[0069] At step 117, the Fusion server performs a survey.

[0070] According to some exemplary embodiments, the survey comprises selecting a tradable item, determining a net position based on each position for the tradable item, and determining a percentage for the net position based on the total of all positions.

[0071] For example, assume the tradable item or market is gold. A trader has a maximum of 10 contracts (or investable units, shares, etc. not limited to futures contracts) that he or she may invest in the gold market. Presently, the trader has 30 contracts that are long (Buy) in the gold market and 70 contracts that are short (Sell) in the gold market. Accordingly, the trader's net present position is 40 contracts that are short out of a total of (30 + 70) 100 contracts, which computes to a survey of 40%

short. Each algorithm is not limited to representing 1 contract. Weighting could be different for each bot, but the net result of the basket survey (Longs + Shorts) is still the same.

[0072] At step 118, the Fusion server compares the survey against the maximum number of contracts that the trader may invest in the tradable time to determine an order to send to an exchange.

[0073] For example, with reference to step 118, the survey is 40% short with a maximum of 10 contracts that may be invested in the gold market. Accordingly, the current account position in Gold is compared with the expected position in Gold (-4). The automated order management system (Fusion Server) will send an order to the exchange to bring the account position in Gold to (Short) -4 contracts.
For example, the account position in Gold was Long 2 contracts. The Gold market suddenly moves down in price. This move triggered some of the algorithms to switch their signals from Long to Short, therefore changing the net result of the basket. The new survey reveals a 40% short position, and the maximum contracts to be traded in Gold is 10 contracts. The account must be adjusted to the current survey of 40% short, which is Short 4 or -4. If the account was Long 2 and it should now be Short 4, the Fusion Server will send an order to the exchange to Sell 6 Gold contracts (Sell 2 to bring the position to 0 from long 2, then Sell 4 more to bring the account position to -4 in Gold).

[0074] At step 119, the real time trading server is updated to reflect execution of the sent order to the exchange.

[0075] For example, with reference to steps 117 and 118 above, after the purchase of 4 contracts that are short in the gold market, the real time trading server will be updated to reflect the purchase and a new total position of 74 contracts that are short in the gold market.

[0076] FIG. 2 is a diagram of an exemplary system 200 for implementing embodiments of the present technology.

[0077] Shown in FIG. 2 are historical server 201, master virtual machine server 202, neural network 203, fusion server 204, historical database 205, virtual machines 206, market exchange 207, market data server 208 and trading server 209.

[0078] According to various exemplary embodiments, a virtual machine 206 is generated by a master virtual machine server 202 for a trading strategy in a historical server 201. Each virtual machine 206 generated obtains historical performance data for its respective strategy from the historical server 201. Each virtual machine 206 transforms the historical performance data into metrical data. Each virtual machine 206 transforms the historical performance data and metrical data into a neural network usable data set. Each virtual machine 206 creates a neural network base. The neural network base, along with the neural network usable data set forms a neural network 203. Each virtual machine 206 trains its respective neural network 203 (and the neural network learns) for one or more data points. An error rate for the one or more data points is determined and observed by each virtual machine 206 until the error rate stops converging or cannot converge. If the observed error rate does not converge, the neural network 203 is dropped and marked as "unusable" for the one or more historical data and metrical data points for the tradable item employing a particular strategy. If the observed error rate converges, the neural network 203 is saved and marked as "usable." The saved usable neural network 203 is tested against an updated historical and metrical data set. For each time-frame in the updated historical and metrical data, the neural network 203 is retrained until it stops converging. The neural network 203 is saved with the one or more data points as a binary object. Each virtual machine 206 transmits the binary object to the historical server 201. A fusion server 204 activates the neural network 203 as stored in the historical server 201 as the binary object. The

neural network 203 obtains updated historical and/or metrical data from the historical server 201. The neural network 203 calculates a confidence value based upon the updated historical and/or metrical data. The fusion server 204 obtains the calculated confidence value and determines whether to execute a trade. The fusion server 204 performs a survey. The fusion server 204 compares the survey against the maximum number of contracts that the trader may invest in the tradable time to determine an order to send to an exchange 207. The real time trading server 209 is updated to reflect execution of the sent order to the exchange 207.

[0079] The exemplary systems and methods described herein may be performed in a secure computing environment including the use of firewalls and encryption technology. Given the potentially high value of the information being generated, and the potential magnitude of the resulting investment decisions, measures may be taken to perform some or all of the steps herein in a secure manner, with emphasis on such steps as the determination of strategy and execution of trades. For example, in addition to an optimal strategy, non-optimal strategies may purposely be added in the same string or digital data environment of the optimal strategy to confuse any unwanted hackers intercepting such information. As another example, in addition to a desired trade to be executed, undesired trades may purposely be added in the same string or digital data environment of the desired trade to confuse any unwanted hackers intercepting such information. Further, the desired trade may receive funding for execution, whereas the undesired trades may not receiving funding for execution.

[0080] While various embodiments have been described above, it should be understood that they have been presented by way of example only, and not limitation. The descriptions are not intended to limit the scope of the technology to the particular forms set forth herein. Thus, the breadth and scope of a preferred embodiment should not be limited by any of the above-described exemplary

embodiments. It should be understood that the above description is illustrative and not restrictive. To the contrary, the present descriptions are intended to cover such alternatives, modifications, and equivalents as may be included within the spirit and scope of the technology as defined by the appended claims and otherwise appreciated by one of ordinary skill in the art. The scope of the technology should, therefore, be determined not with reference to the above description, but instead should be determined with reference to the appended claims along with their full scope of equivalents.

CLAIMS

1. A method for allocating capital to trading strategies comprising:

generating a virtual machine by a master virtual machine server for a trading strategy in a historical server;

obtaining historical performance data for the trading strategy from the historical server;

transforming by the virtual machine the historical performance data into metrical data;

transforming by the virtual machine the historical performance data and metrical data into a neural network usable data set;

creating by the virtual machine a neural network base;

forming by the virtual machine a neural network; and

further comprising the virtual machine and the historical server comprising separate entities, the virtual machine generated and dedicated for a specific trading strategy.

2. The method of claim 1, further comprising:

training by the virtual machine neural network for one or more data points.

3. The method of claim 2, further comprising:

calculating by the virtual machine an error rate for the one or more data points until the error rate stops converging or cannot converge.

4. The method of claim 3, further comprising:

saving by the virtual machine the neural network.

5. The method of claim 4, further comprising:

testing by the virtual machine the saved neural network against updated metrical and historical data.

6. The method of claim 5, further comprising:

training by the virtual machine the saved neural network.

7. The method of claim 6, further comprising:

saving by the virtual machine the neural network as a binary object.

8. The method of claim 7, further comprising:

transmitting by the virtual machine the binary object to the historical server.

9. The method of claim 8, further comprising:

activating by a fusion server the neural network.

10. The method of claim 9, further comprising:

obtaining by the fusion server historical metrical and historical performance data.

11. The method of claim 10, further comprising:

calculating by the fusion server a confidence value.

12. The method of claim 11, further comprising:

determining by the fusion server whether to execute a trade.

13. The method of claim 12, further comprising:

performing by the fusion server a survey.

14. The method of claim 13, further comprising:

determining by the fusion server an order to send to an exchange.

15. The method of claim 14, further comprising:

sending by the fusion server the order to the exchange.

16. The method of claim 15, further comprising:

updating by the fusion server to reflect an executed order.

17. A system for allocating capital to trading strategies comprising:

a means for generating a virtual machine for a trading strategy in a historical server;

a means for obtaining historical performance data for the trading strategy from the historical server;

a means for transforming the historical performance data into metrical data;

a means for transforming the historical performance data and metrical data into a neural network usable data set;

a means for creating a neural network base;

a means for forming a neural network; and

further comprising the virtual machine and the historical server comprising separate entities, the virtual machine generated and dedicated for a specific trading strategy.

18. The system of claim 17, further comprising:

a means for training by the neural network for one or more data points.

19. The system of claim 18, further comprising:

a means for calculating an error rate for the one or more data points until the error rate stops converging or cannot converge.

20. The system of claim 19, further comprising:

a means for saving the neural network;

a means for testing the saved neural network against updated metrical and historical data;

a means for training the saved neural network;

a means for saving the neural network as a binary object;

a means for transmitting the binary object to the historical server; a means for activating the neural network;

a means for obtaining historical metrical and historical performance data;

a means for calculating a confidence value;

a means for determining whether to execute a trade;

a means for performing a survey;

a means for determining an order to send to an exchange;

a means for sending the order to the exchange; and a means for updating to reflect an executed order.