Financial Signal Representation and Trading using Deep Neural network

Gayatri Panchwagh
Department of Computer Engineering
Siddhant College of Engineering Sadumbre,
Savitribai Phule, Pune University
Pune, Maharashtra
gayatrimujumdar1@gmail.com

Dr. Deepak Gupta
Department of Computer Engineering
Siddhant College of Engineering Sadumbre,
Savitribai Phule, Pune University
Pune, Maharashtra
deepak_gpt@yahoo.com

***Abstract*:** The stock markets all over world have many uncertainties which contain a large amount data with jumps, noises, and movements in leading to the highly non-stationary time series. Here, we introduce real-time financial signal representation and trading techniques as RDNN (Recurrent Deep Neural Network) for the environment sensing and recurrent decision making for the online financial assert trading. In which the RDNN have two parts, first one is DNN (Deep neural network) function of this DNN is feature learning and the second one is RNN (Recurrent Neural Networks) to predict rapidly changing market condition. This proposed model based on two biological related learning concepts such as Reinforcement Learning (RL) and Deep Leaning and DL (Deep Learning) is used to represent financial signals and self-taught reinforcement trading. Here, RL is used to interact with deep representations and makes trading decisions to accumulate the ultimate rewards in an unknown environment. It improves the robustness of market summarization using fuzzy learning concepts it reduces the uncertainty of input data. Here, used K-Means for making data static and labeled data, so it makes easy for the MLP to the predictes result. Fuzzy MLP will be using to provide the prediction of stock market.

Keywords: Deep Neural Network, Direct reinforcement Learning, Fuzzy logic, Multilayer Perceptron Model.

I. Introduction

Real-time health data collection is very common nowadays. This data is processed by various signal processing and machine learning algorithms. The procedure of mining and reasoning are similar in different applications. Researchers and engineers working with real-time signals perform similar preprocessing and processing steps prior to derivation. The collected data can be used to get real-time offline multiple results of the condition of the patient [2]. However, the use of health parameters is very limited, due to the processing of the network requirements of the infrastructure. The real health app requires real-time analysis of high-resolution sensor data, data from other sources as well as data from many users at the same time and local processing of all the data on a single computer can be achieved by calculation constraints, reliability, recovery scalability, fault/power supply problems, etc. which is not practical.

Recently, there has been a great interest in optimizing algorithms and increasing the efficiency of the system through system implementation [3]. But these methods only suggest a solution to a particular problem. They include design decisions that are difficult to generalize due to certain assumptions in the problem. The application of these challenges requires the implementation of a dependent machining platform efficient enough to work under real-world hardware and software constraints.This work and the construction of this problem will solve the intelligent distribution of the computational load.

The analysis of health data generally involves comparing pre-defined basis with extracted parameters. Symptoms can be detected when the readings are higher or lower than the threshold. Early detection of symptoms of heart failure can support the prediction of heart failure stroke and so they can be avoided. Therefore, the most important task is to define the “Accurate” threshold. The accuracy of the analysis depends strongly on the accuracy of the threshold used.

The cardiologist defines and updates the thresholds based on the measurements of the patient and the conducted interview with the patient [4]. In fact, cardiologists confirm that the values of the thresholds are not the same for all patients and can vary even for the same patients. As a result, there are 2 goals for this work. Firstly, propose a monitoring approach to remotely extract health parameters from patients suffering from heart failure will then define an analytical approach to automatically calculate and update the health threshold at the run time.

A smart home monitoring system for tracks vital signs in patients with CHF. The system collects vital signs (SpO2, Electro Cardio Graph (ECG), Blood Pressure (BP), and Body Weight) and sends them to a hospital information system that is evaluated by a physician. The aim of this system is to reduce the number of face-to-face visits of each patient with help of CHF.

In this paper study about the Literature Review done, in section II, the Proposed Approach Modules Description, Mathematical Modeling, Algorithm and Experimental setup in section III .Finally we provide a conclusion in section IV.

II. LITERATURE REVIEW

Portfolio management is the process of making decisions about the constant redistribution of the Fund amount in a number of different financial investment products in order to maximize profits while limiting risk. Traditional methods of portfolio management can be divided into four categories: “follow the winner”, “follow the loser”, “pattern matching” and “meta-learning” [1]. The first two categories are based on previously built financial models, although some machine learning techniques for parameter determination may also help. The effectiveness of these methods depends on the reliability of the models in different markets. The Pattern-Matching algorithms predict the next market distribution based on a sample of historical data and explicitly optimize the portfolio based on a sample distribution. The last class, the method of “meta-learning”, combines several strategies of other categories to achieve more consistent work [2].

There are deep machine learning approaches to trading financial markets. However, many of them try to predict price movements or trends [3]. A history of prices the neural network can output the predicted vector of asset prices from all assets as input data for the next period. Then the sales agent can act on this forecast. This idea easy to implement because it is supervised learning, or more specifically regression problem. However, the performance of these algorithms is based on price forecasting, strongly depends on the degree of accuracy of the forecast, but it turns out that the future market prices are difficult to predict. In addition, price forecasts are not market actions, converting them into actions requires an additional layer of logic. If this layer is a manual code, then the whole approach is not fully machine learning, and thus not very open and adapted. For example, for a network-based prediction, it is difficult to consider the transaction value as a risk factor.

Previous successful attempts in Model-free and complete machine learning schemes for algorithmic trading problems without predicting future prices include this recent deep use by Moody and Sachel, denpster and Rayman, caming, and RL Deng et al. (2017). These RL algorithms output a separate trading signal to the asset. Limited to single asset trade, they do not apply to the general problem of portfolio management; where distributors manage multiple assets.

In this paper study about the Literature Review done, in section II, the Proposed Approach Modules Description, Mathematical Modeling, Algorithm and Experimental setup in section III .and at final we provide a Conclusion in section

III. PROPOSED APPROACH

Propose recurrent deep neural learning techniques for the real-time financial signal representation and trading and this technique based on the two biological related learning concepts Deep Learning (DL) and Reinforcement Learning (RL) it automatically senses dynamic marketing condition and provides a prediction for the market conditions.

A. Proposed System Overview

In this present techniques RDNN (Recurrent Deep Neural Network) Structure for the environment sensing and recurrent decision making for the online financial assert trading. In which the RDNN using a two parts DNN (Deep neural network) it is used for the feature learning and the second one is RNN (Recurrent Neural Networks) for the RL (Reinforce Learning). Improve the robustness of market summarization using fuzzy learning concepts it reduces the uncertainty of input data.

The DDR trading system is using data of the real financial market for future contracts trading. In detail, we accumulate the historic prices of both the stock-index future (IF) and commodity futures. These real market data will be directly used for performance verifications. The deep RL system will be compared with other trading systems under diverse testing conditions. The comparisons show that the DDR system and its fuzzy extension are much robust to different market conditions and could make reliable profits on various future markets.

Read Dataset:

In this module, it read financial data. In this, we are used real word financial data such as stock index and commodity futures contracts.

Fuzzy Extensions:

In this module for data, uncertainty problem is considered. Financial sequences contain the high amount of unpredictable uncertainty due to the random gambling behind trading. Besides, a number of other factors, e.g., global economic atmosphere and some company rumors may also affect the direction of the financial signal in real time. Therefore, reducing the uncertainties in raw data is an important approach to increase the robustness for financial signal mining.

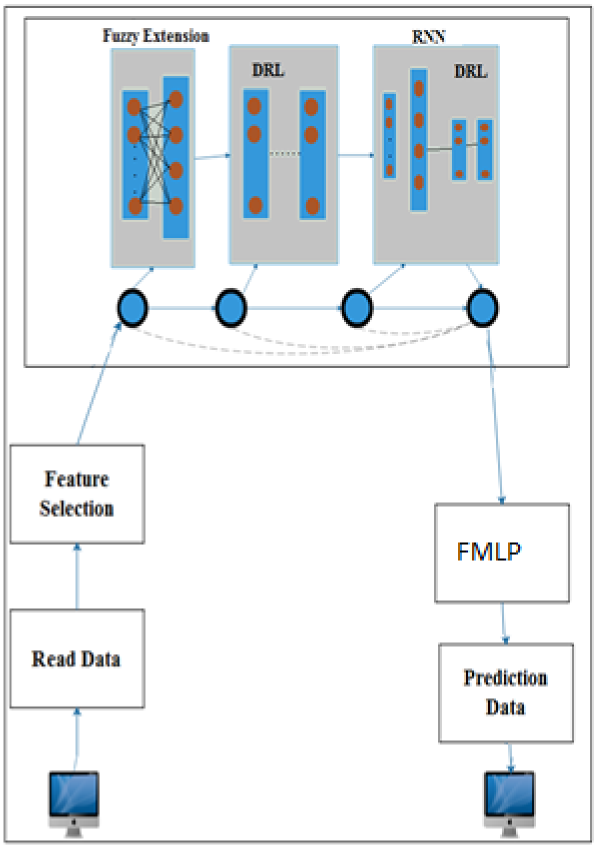

Figure 1.Proposed System Architecture

Membership functions of fuzzy sets are never unique. Different individuals might define various F A ' S for the same fuzzy set. For example, the membership function of a fuzzy TALL defined by an American for Americans would probably be different from that defined by an Oriental for Orientals.

Measures of fuzziness estimate the average ambiguity in fuzzy sets in some well-defined sense. We begin by considering properties that seem plausible for such a measure.

The fuzziness of a crisp set using any measure should be zero, as there is no ambiguity about whether an element belongs to the set or not. If a set is maximally ambiguous (p ~ (z =) 0.5 V x), then its fuzziness should be maximum. HzE(A, P) is a measure associated with fuzzy set A and When a membership value approaches either 0 or 1, the ambiguity about belongingness of the argument in the fuzzy set decreases.

Extensions to Membership Functions on Real Intervals of the multiplicative and additive classes are easily extended to fuzzy sets on any real interval. Let A be a fuzzy subset of X c R. Then we define a measure of fuzziness (under the multiplicative class) for A as:

$$H_C(A) = K\int_x g(\mu A(x))dx$$

Where K E R+ is a constant and g is a continuous function whose form is defined in. H, reduces to H , at whenever X is discrete.

For the reducing uncertainty of data, we are using a mostly used artificial intelligent techniques fuzzy learning. In which is used input as linguist values. In it compare a real word data with the number of fuzzy-rough sets and then deriving corresponding fuzzy membership degrees. Fuzzy-rough sets are naturally defining according to basic movements of stock price. Fuzzy sets are defined by increasing decreasing and no trend groups. The parameters in fuzzy membership function are predefined.

Direct Reinforcement Learning(DRL):

DRL (Direct Reinforcement Learning) is the review of the Moodys DRL frameworks and typical DRL is the one-layer RNN. In this we consider the P1, P2… Pn is the sequences released from exchange center. The return at time point t is determined by the Z = Pt - Pt-1 his value is based on the current market condition real-time trading decision policy.

The profit Rt made by the trading model is obtained by:

$$R_t = \delta_t - Z_t - C|\delta_t - \delta_{t-1}|$$

Recurrent Deep Neural Network (RDNN):

In this module, we are using practical learning strategy to train the DNN using two steps of system initialization and fine-tuning. For the learning of the DNN use three learning parts, First one is fuzzy representation part is it the easy process of initialization. In this only parameter are specified in the fuzzy center. Here K-means algorithms are applied for clustering and creating k classes of input data.

The parameter k is fixed as 3, because each input node is connected with three membership functions.

B. Algorithm

Algorithm 1: RNN Algorithm

```
Data: A set of sequence with their corresponding context
Output: RNN optimized for generation
Initialize RNN at random and set NXENT , NXE+R
and$\Delta$;
For s =T, 1, and – $\Delta$ do;
If s==T then;
Train RNN for NXENT epochs using XENT only;
Else;
Train RNN for NXE+R  epochs;
Use XENT loss in the first s steps; and REINFORCE
(Sampling from the model) in the remaining T –s steps;
End;
End;
```

```
Algorithm 2: K-Means Algorithm
Input the initial set of k cluster Centers C
Set the threshold THmin
    while k is not stable
    generate a new set of cluster centers CΘ by k-means
or every cluster centers CΘλ
get the minimum relevance score: min (Si)
if the min (Si) <THmin
add a new cluster center: k = k + 1
go to while
until k is steady.
```

Algorithm 3: Fuzzy Multi-Layer perceptron Model:

In Fuzzy MLP Algorithm, The concept of fuzziness is inherited from FRNN algorithm. In this algorithm it uses fuzzy logic to handle uncertities. Clustering stock data into 3 clusters like Losser, Gainer, and Neutral. So it makes easy for MLP to handle data to recall error back propagation (EBP) training.

If a submitted pattern provides an output far from desired value, the weights and thresholds are adjusted s.t. the current mean square classification error is reduced. The training is continued/repeated for all patterns until the training set provides an acceptable overall error. Usually the mapping error is computed over the full training set. EBP training is working in two stages:

The trained network operates feed-forward to obtain output of the network

The weight adjustment propagates backward from output layer through hidden layer toward input layer.

Error Back-Propagation Training Algorithm

Initialization of k means algorithm.

Use Fuzzy membership function to calculate

Values of each record in dataset.

Given P training pairs $\{Z_1, D_1, Z_2, D_2, \dots, Z_p, D_p\}$ where Zi is (l * 1), Di is (K*1), i=1,……….,P

The lth component of each Zi is of the value -1 since input vector are augmented.

Size J-1 of the hidden layer having output y is selected.

Jth component of y is -1, since hidden layer have also been augmented.

Y is (J * 1) and o is (k * 1)

In the following, q is training step and p is step counter within training cycle.

Choose $\eta > 0, E_{max} > 0$

Initialized weight at small random values, W is (k*j), V is (j*l),

Initialize counters and error: $q \leftarrow 1, p \leftarrow 1, E \leftarrow 0$

Training cycle begins here:

Set $\leftarrow Z_p, D \rightarrow D_p$;

$$y_i \leftarrow f(V_j^t Z), j = 1, \dots\dots, j$$

$$(V_j acolumnvector, jthrowofV)$$

$$o \leftarrow f(w_k^t y), K$$

$$(w_k acolumnvector, Kthrowofw)$$

$$(f(net) issigmoidfunction)$$

Find error : $E \leftarrow \frac{1}{2}(d-o)^2 + E$

$$fork = 1, \dots\dots, k$$

Error signal vectors of both layers are computed.

$$\delta_0 (outputlayererror) isk * 1,$$

$$\delta_y (hiddenlayererror) isj * 1$$

$$\delta_{ok} = \frac{1}{2}(d_k - o_k)(1 - o_k^2),$$

$$fork = 1, \dots\dots, k$$

$$\delta_{yj} = \frac{1}{2}(1 - y_j^2) \sum_{k=1}^{k} \delta_{ok} w_{kj},$$

$$forj = 1, \dots\dots, j$$

Update weight:

Output:

$$w_{kj} =\leftarrow w_{kj} + \eta \delta_{ok} y_i,$$

$$k = 1, \dots, k, j = 1, \dots, j$$

Hidden layer:

$$V_{ij} \leftarrow V_{ij} \eta \delta_{yj} z_j$$

$$jk = 1, \dots, j, i = 1, \dots, l$$

Update weight:

If p<P then p←p+1, q→q+1

Go to step iv, otherwise, go to step ix.

If $E < E_{max}$ the training is terminated, otherwise $E \leftarrow 0, p \leftarrow 1$ go to step iv for new training cycle.

C. Mathematical Mode:

Let S be system such that

S = Let s be the system such that

S = {I, P, O, Sc, Fc}

Where;

I = Input of system

P = Process in system

O = Output of System

Sc = Success case of output of system

Fc = Failure case of output of system

I = {I1, I2...In};

Where;

I = Finance dataset

Process:

P1 = {I1}   // Read dataset

P2 = {P1};

Auto encoder:

$$\phi = X \rightarrow F$$
$$\Psi = X \rightarrow F$$

$$|(\phi, \Psi)X| \vee \quad {}^{2}$$
$$X =$$

P3 = {P2 };

Direct reinforcement learning:

$$\Pi : S \times A \rightarrow [0,1]$$

$$\Pi(a|s) = P(a_t = a \vee s_t)S$$

P4 = {P3};

Recurrent neural Networks:

$$T_i, Y_i = -Y_i + \sum_{j=1}^{n} w_{ij}\sigma(Y_i - \Theta_i) + I_i(t)$$

P4 = {P3};

Fuzzy membership function:

$$0_i^{\gamma} = V_i\left(q_i^1 = e^{\frac{-o_i^1 - m_i}{0_i^1 \Psi_i}}\right)$$

P6 = {P3, P4, P5 };

Prediction of finance data

Sc = {When successfully prediction is given}

Fc = {When Prediction is Fails}

Dataset

.

IV.RESULTS AND DISCUSSION

A. Experimental Setup

Hardware and software of proposed system given below:

Software Technology:

1. Technology: Core Java
2. Tools: JDK 1.8, Netbeans 8.0.2
3. Operating System: Windows 7

Hardware Technology

1. Processor: 1.0 GHz
2. RAM: 1 GB
3. Hard Disk: 730 GB

B. Dataset

Here we use online dataset such as Yahoo finance historical database of S&P BSE SENSEX (BSESN). Time period: from 01 July 1997-08-Jan-2019.

C. Expected Result

Table 1 shows that memory comparison between existing system RNN algorithm and proposed system MLP algorithm. Proposed algorithm required 20 bytes memory and existing system required 30 byte memory.

Algorithm	Memory in byte
RNN	30
MLP	20

Table 1: Memory Comparison of RNN and MLP Algorithm

Figure 2 shows that graph of memory comparison between existing system RNN algorithm and proposed system MLP algorithm. Result graph shows that the proposed system required less memory than the existing system.

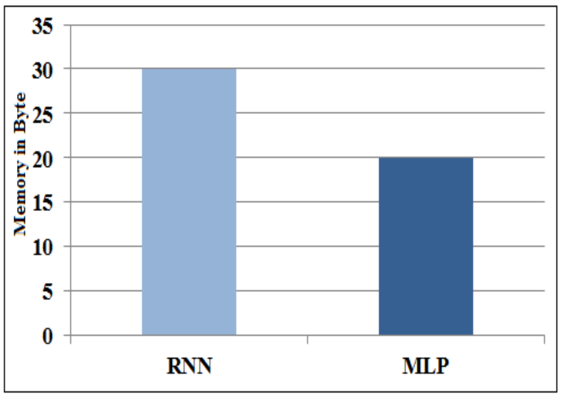

Fig 2: Memory Comparison Graph of RNN and MLP Algorithm

Table 2 shows that time comparison between existing system RNN algorithm and proposed system MLP algorithm. Proposed algorithm required less time as compared to existing system.

Algorithm	Time in ms
RNN	1800
MLP	1400

Table 2: Time Comparison of RNN and MLP Algorithm

Fig 2 shows that graph of time comparison between existing system RNN algorithm and proposed system MLP algorithm. Result graph shows that the proposed system required less time than the existing system.

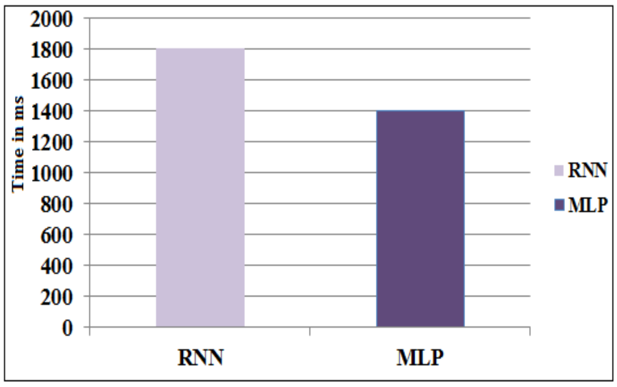

Fig 3: Time Comparison Graph of RNN and MLP Algorithm

Table 3 shows that comparison between existing system RNN algorithm and proposed system MLP algorithm. A proposed technique is more accurate than the existing techniques.

Algorithm	Accuracy in %
RNN	97
MLP	99

Table 3: Accuracy Comparison of RNN and MLP Algorithm

Figure 2 shows that graph of accuracy comparison between existing system RNN algorithm and proposed system MLP algorithm. Result graph shows that the proposed system is more accurate than the existing system.

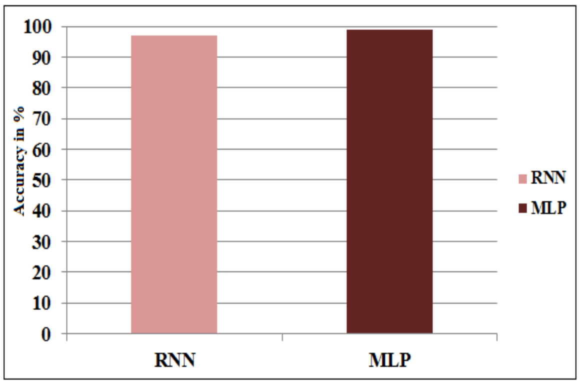

Fig 3: Accuracy Comparison Graph of RNN and MLP Algorithm

V. CONCLUSION AND FUTURE SCOPE

Here, we introduce RDNN techniques for the recurrent decision making for the online financial assert trading and environment sensing. This technique is made up two parts DNN and second one is RNN. It improves the robustness of market summarization by the use of fuzzy learning concepts, and it reduces the uncertainty of input data. The deep RL system will be compared with other trading systems under diverse testing conditions. The comparisons show that the DDR system and its fuzzy extension are much robust to different market conditions and could make reliable profits on various future markets. K-Means algorithm is used for the making data statics and labeled data, so it makes easy for the MLP to the predicted result. Fuzzy MLP is will be using to the predicted result. FRNN has some limitations like it takes long computation time because of complex calculations. So FRNN also provides lower accuracy while handling Uncertainties. FMLP overcomes both limitation of FRNN in computation time and accuracy of stock market prediction.

ACKNOWLEDGMENT

The authors would like to thank the researchers as well as publishers for making their resources available and teachers for their guidance. We are thankful to the authorities of Savitribai Phule University, Pune and concern members of the cPGCON-2019 conference, organized by, for their constant guidelines and support. We are also thankful to the reviewer for their valuable suggestions.

REFERENCES

1.D. Wierstra,J. Hunt and D. Silver “Continuous Control with Deep Reinforcement Learning”, arXiv- 1509.02971v5, [2016].
2.David Lu, “Agent Inspired Trading Using Recurrent Reinforcement Learning and LSTM Neural Networks”, arXiv: 1707.07338 [2016].
3.James Cumming, “An investigation into the use of reinforcement learning technique within the algorithmic trading domain”; Knowledge Information System, Master’s thesis, Imperial College London, United Kiongdom, [2015].
4.Dongbin Zhao and Yuanheng Zhu, “MEC: A near-optimal online reinforcement learning algorithm for continuous deterministic systems”, IEEE Transaction on Neural Networks and Learning Sys., Vol-26, no-2, [2015].
5.Volodymyr Mnih, Daan Wierstra and Shane Legg, “Human Level Control through Deep Reinforcement Learning”;Macmillan Publishers Limited, Vol- 5 1 8; Nature, [2015].
6.Yue Deng, Risheng Liu, and Sanqing Hu, "Low-rank structure learning via nonconvex heuristic recovery”;IEEE Transaction on Neural Network and Learning System, Vol-24, no.-3, [2013].
7. Alex Graves,Geoffrey Hinton and Abdel-rahman Mohamed, “Speech Recognition with Deep Recurrent Neural Network”; IEEE International Conference on Acoustics, Speech and Signal Processing, [2013].
8.Yue Deng, Zengke Zhang, and Qionghai Dai, "Noisy depth map fusion for multiview stereo via matrix completion”; IEEE Journal of Selected Topics in Signal Processing, Vol-6, no.-5, [2012].
9.George Dahl, Li Deng, and Dong Yu, “Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition”; IEEE Transaction on Audio Speech and Language Processing, Vol:20, no.:1, [2012].