On Estimation of Value-at-Risk with Recurrent Neural Network

Zhichao Du*
Department of Statistical Science
University of Toronto
Toronto, Ontario, Canada
zhichao.du@mail.utoronto.ca

Menghan Wang*
Department of Social Science
University of Western Ontario
London, Ontario, Canada
mwang584@uwo.ca

Zhewen Xu*
Department of Statistical Science
University of Toronto
Toronto, Ontario, Canada
joanne.xu@mail.utoronto.ca

***Abstract*—In this paper, we propose a new method for predicting Value-at-Risk using recurrent neural network. We show that our new approach provide us a more flexible semi-parametric framework for forecasting VaR, and we obtain improved results comparing with other forecasting methods.**

***Index Terms*—recurrent neural network, Value-at-Risk, risk management**

I. Introduction

Risk management is an essential part in the financial market, for investors, fund managers, and all other financial institutes [1]. It is always important for practitioners to predict the risk of the future market, and one of the most commonly used measure of risk is called Value-at-Risk.

Value-at-Risk, usually referred to as VaR, measures the potential risk that may happen in an investment or a portfolio of investments according to a specific period of market history period and volatility. Financial institutions use VaR to calculate the cash reserve they need to cover potential loss. VaR contains three variables, a time frame, a confidence level, and a loss amount [2]. VaR is now one of the most commonly used and widely accepted measures of risk.

Statistically, VaR is the α^{th} quantile of the future return. For random variable X, the VaR at level α is defined as

$$\text{VaR}_\alpha(X) = \inf\{x : P(X \geq x) \leq \alpha\}.$$

To calculate VaR, one of the most commonly used method is the historical simulation approach, which is simple and nonparametric. The basic idea behind historical simulation approach is that the distribution of past market is stationary, and therefore we can estimate the quantile by using empirical quantile of the historical data. However, the stationarity assumption is usually too strong and often unrealistic, as pointed out in [3]. Also, it is always questionable about how much past information should be included when estimating the empirical quantile. Another off-the-shelf method is the Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model proposed by [4]. Although it was proposed a few decades ago, nowadays it is still one of the most robust and reliable method for modeling the volatility process of financial returns. It is also very easy to adopt this model to estimate the Value-at-Risk.

With the recent development of machine learning and artificial intelligence, researchers start to apply techniques from artificial neural network (ANN) to predict Value-at-Risk in the hope that neural networks can help to improve the forecasting accuracy [5]. ANN has been studied since the 80s. Inspired by human brain neural networks, ANN contains a sophisticated multilayer structure to model the nonlinear relationships between the data [6]. In [8], the authors propose a forecasting method based on the naive ANN model, and present some promising results. A similar approach using ANN for estimating Value-at-Risk can be found in [9]. However, one should notice that ANNs are designed to process independent observations. That is, we are assuming that the input covariates are independent to each other. This assumption is clearly violated here, since the financial market returns shall be a time dependent process. Researchers have turned to recurrent neural network (RNN), a specialized neural network to hand dependent data, to model financial markets. Some recent progress can be found in [11], where the authors successfully apply RNN to predict market returns. For the specific problem of forecasting Value-at-Risk, the authors of [10] propose a hybrid model by combining RNN with GARCH model to predict future VaR.

In this paper, we propose a semi-parametric approach to predict Value-at-Risk using RNN. We utilize RNN's ability to handle the time dependent data, and avoid the model assumptions imposed by GARCH process. While the method proposed by [10] also implement the RNN, in their paper the inputs are estimated parameters from different types of GARCH models, while in our paper we intend to propose a more data-drive method by using the original data as the inputs. We show that our approach is flexible, and the results are comparable with some existing methods that are commonly applied in practice.

The rest of this paper is organized as follows. In Section 2 we review recurrent neural network and discuss the details of our new approach. In Section 3 we present the predicting results. Section 4 concludes.

* Equal contribution.

978-1-7281-4087-2/19/$31.00 ©2019 IEEE
DOI 10.1109/AI4I46381.2019.00034

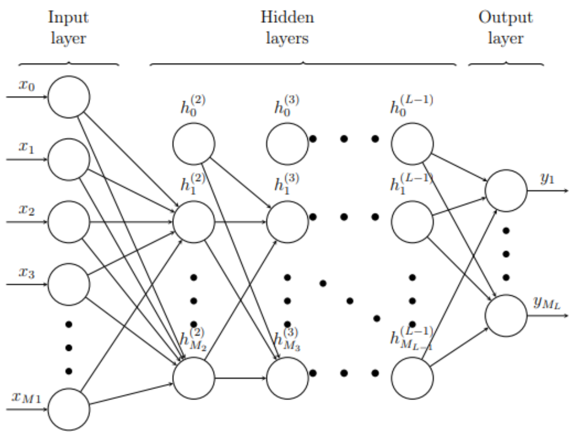

Fig. 1. Example of a common neural network architecture. [12]

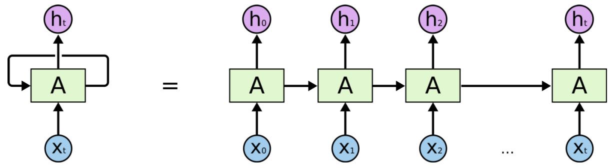

Fig. 2. Structure of a Recurrent Neural Network.

II. Methodology

A. Artificial Neural Network

Artificial neural network (ANN) is arguably the most common structure of neural network. A typical fully-connected ANN contains an input layer, several hidden layers, and an output layer. Each layer contains many nodes that are often referred to as neurons, and all neurons are interconnected with each other. There are also cases where the neurons are sparsely connected. This multilayer structure will help ANN to approximate many nonlinear relationships with reasonable accuracy. See [14] and [13] for more details. An example of the ANN can be found in Figure 1.

B. Recurrent Neural Network and LSTM

As mentioned in Section 1, the conventional ANN is not intended for data with dependence. Therefore, we would like to apply recurrent neural network (RNN) to forecast the VaR. RNN processes inputs one at a time and then return output in the output layer. In the meantime, the output of intermediate layers of the previous step will be delivered to the next layer as part of the input. This ongoing recursive process continues for every element of a sequence until we obtain a prediction of the final output. The following diagram illustrates the structure a typical RNN: See Fig. 2 for a graphic illustration[1].

Theoretically, RNN can include information from arbitrarily long sequences. However, in practice RNN will suffer from gradient vanishing when the process is too long. In [7], the authors propose the RNN with Long Short-term Memory cell

[1]https://colah.github.io/posts/2015-08-Understanding-LSTMs/

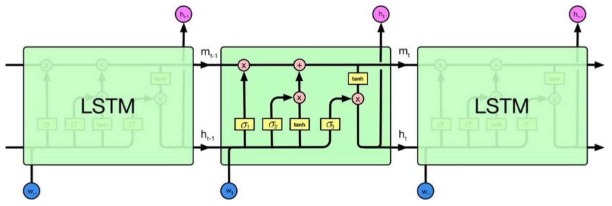

Fig. 3. Long Short-term Memory cell.

in response to this challenge. The intuition behind the idea of LSTM is to add a latent process that is dedicated to store the long term memory while the remaining part is similar to a regular RNN. Nowadays most RNN models are built with LSTM cells or its variants (see, for example, GRU from [15]). Also see Fig. 3 for an illustration[2]. In this plot, σ denotes the logit function, $tanh$ denotes the hyperbolic tangent function, and $+$ or $\times$ denotes the corresponding mathematical operations. See the link in footnote for more details.

C. Details of Implementation

In this paper, we follow the similar approach proposed in [8]. We model the market returns as

$$\begin{aligned} r_t &= \mu_t + z_t\sigma_t \\ \mu_t &= f_1(\Omega_{t-1};\theta) \\ \sigma_t^2 &= f_2(\Omega_{t-1};\theta), \end{aligned}$$

where r_t is the log return at time t, μ_t is the corresponding conditional expected mean of log return, σ_t is the conditional volatility of the log return, and z_t is an innovation random variable with zero mean and unit standard deviation. We denote Ω_{t-1} as the set of information available at time $t-1$ and θ as the set of parameters, then under this framework the conditional mean μ_t and conditional volatility σ_t can be estimated by two nonlinear functions $f_1(\Omega_{t-1};\theta)$ and $f_2(\Omega_{t-1};\theta)$ which can be approximated by two RNN trained separately.

With the estimated $\hat{\mu}_t$ and $\hat{\sigma}_t$, we then estimate the VaR at time t at α level as

$$\text{VaR}_\alpha^t = \hat{\mu}_t + q_\alpha \times \hat{\sigma}_t,$$

where q_α is the α^{th} quantile of the assumed innovation distribution.

In this project, we study the market returns of S&P 500 Index that is publicly available on Yahoo Finance[3]. The data we study contain the closing price of S&P 500 Index from May 11, 2009 to May 9, 2019. In total 2517 trading days are included, and 2516 log returns can be calculated from them. We then normalize the data using min-max normalization so that the impact of the scale would be minimized. We train the neural networks using the market returns up to October 05, 2018 and using the remaining 146 observations as the test set. In the training process, we set a rolling window of size 100

[2]https://colah.github.io/posts/2015-08-Understanding-LSTMs/
[3]https://finance.yahoo.com/

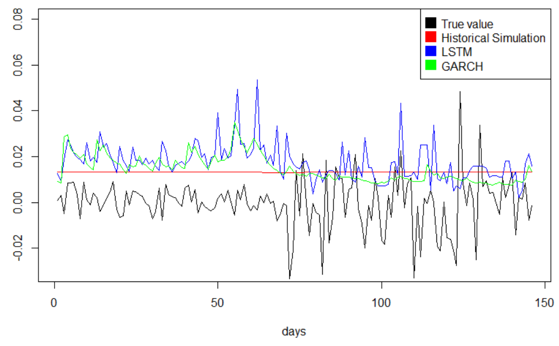

Fig. 4. The estimated Value-at-Risk for S&P 500 Index from 2018-10-08 to 2019-05-09.

days as the input and train our neural networks to predict the next value. Therefore, we trained two RNNs for return and squared return separately. Each RNN contains three LSTM layers with 4096, 1028, and 512 nodes respectively. Following the LSTM layers are three dense layers with 128, 32, and 16 layers. The input size of the neural network is 100 and the output size is 1.

III. Results

The most common method to measure the VaR prediction is back testing discussed in [17], In [16], the authors propose an alternative score function to evaluate the Value-at-Risk prediction. The score function has the form

$$S_\alpha(x,y) = \begin{cases} \alpha|x-y|, & x \le y \\ (1-\alpha)|x-y|, & x \ge y \end{cases}$$

where x is the predicted Value-at-Risk, y is the actual market return on that day, and α is the test level. The intuition behind this scoring function is that we put a larger penalty on the underestimated Value-at-Risk. Since our goal is to predict the future risk, it is reasonable to obtain a more conservative estimation. The smaller the mean score is, the better our model performs to predict the Value-at-Risk.

In this project, we forecast 146 days of Value-at-Risk using three different methods: RNN with LSTM cells as discussed section, the GARCH model, and historical simulation. The results are shown in Figure 4. The rejection rates and mean scores are shown in Table I.

TABLE I
Comparison of mean scores (at the level of 10^{-3}) with $\alpha = 0.95$, and rejection rate of Value-at-Risk predictions.

Method	Mean Score	Rejection Rate
RNN-LSTM	1.51	7.5%
GARCH	1.61	9.6%
Historical Simulation	1.30	6.2%

Here are some observations from our results. First we see that the historical simulation method has the smallest mean score and the rejection rate is closest to 5%, the target level. However, one should notice from the plot that the estimated VaR from historical simulation method is essentially a horizontal line. This is because the VaR from historical simulation is estimated as the empirical quantile of the previous observations. This empirical quantile usually does not change dramatically with a few extremal observations added to the data. Therefore, the VaR estimated from historical simulation may not really reflect the fluctuation of the market returns.

When comparing our new method with GARCH model, we see that the RNN approach has smaller mean score and the rejection rate is also closer to 5%. While the estimated Value-at-Risk from these two methods are close, we can see that results from RNN method are relatively conservative, and hence the mean score would be smaller. Therefore, we can safely conclude that our RNN model predict the VaR better than the traditional parametric approach. This is not surprising since the parametric assumptions from GARCH model is usually not realistic, and the mis-specification will be reflected on the forecasting results.

We should also notice that our RNN is relatively simple. In the general practice one could train a deep neural network with tens or hundreds of hidden layers. A deeper RNN may capture the micro-structure of the market and hence improve the forecasting accuracy.

IV. Conclusion

In this paper, we propose a new forecasting method for Value-at-Risk using recurrent neural network. We show that our new approach can forecast the future risks as well as, if not better, than traditional methods. Furthermore, this machine learning approach is more flexible and bears less model assumptions. Therefore, we believe our new approach could have more potentials than other methods. Future research could focus on including external information other than market return and training a deeper neural network to improve the forecasting accuracy.

References

[1] S. W. Rawls and C. W. Smithson, Strategic Risk Management, Journal of Applied Corporate Finance, vol. 2, no. 4, pp. 618, 1990.
[2] D. Duffie and J. Pan, An Overview of Value at Risk, The Journal of Derivatives, vol. 4, no. 3, pp. 749, 1997.
[3] T. Linsmeier and N. Pearson, "Risk measurement: an introduction to value at risk," ACE Reports 14796, University of Illinois at Urbana-Champaign, Department of Agricultural and Consumer Economics, 1996.
[4] T. Bollerslev, Generalized autoregressive conditional heteroskedasticity, Journal of Econometrics, vol. 31, no. 3, pp. 307327, 1986.
[5] H. Locarek-Junge and R. Prinzler, Estimating Value-at-Risk Using Neural Networks, Informationssysteme in der Finanzwirtschaft, pp. 385397, 1998.
[6] Mehra P., Wah B. W.: Artificial Neural Networks: Concepts And Theory. IEEE Computer Society Press, 1992.
[7] S. Hochreiter and J. Schmidhuber, Long Short-Term Memory, Neural Computation, vol. 9, no. 8, pp. 17351780, 1997.
[8] X. Chen, K. K. Lai, and J. Yen, A Statistical Neural Network Approach for Value-at-Risk Analysis, 2009 International Joint Conference on Computational Sciences and Optimization, 2009.
[9] S. Cheung, Z. Chen, and Y. Li, Comparing the Estimations of Value-at-Risk Using Artificial Network and Other Methods for Business Sectors, Proceedings of the International Neural Networks Society Recent Advances in Big Data and Deep Learning, pp. 267275, 2019.

[10] H. Y. Kim and C. H. Won, Forecasting the volatility of stock price index: A hybrid model integrating LSTM with multiple GARCH-type models, Expert Systems with Applications, vol. 103, pp. 2537, 2018.

[11] T. Fischer and C. Krauss, Deep learning with long short-term memory networks for financial market predictions, European Journal of Operational Research, vol. 270, no. 2, pp. 654669, 2018.

[12] M. T. Hagan, H. B. Demuth, and M. Beale, Neural network design. Boston: PWS-Publishers, 1995.

[13] K. Hornik, M. Stinchcombe, and H. White, Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks, Neural Networks, vol. 3, no. 5, pp. 551560, 1990.

[14] H. White, Nonparametric Estimation of Conditional Quantiles Using Neural Networks, Computing Science and Statistics, pp. 190199, 1992.

[15] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, Learning Phrase Representations using RNN EncoderDecoder for Statistical Machine Translation, Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.

[16] W. Ehm, T. Gneiting, A. Jordan, and F. Krger, Of quantiles and expectiles: consistent scoring functions, Choquet representations and forecast rankings, Journal of the Royal Statistical Society: Series B (Statistical Methodology), vol. 78, no. 3, pp. 505562, 2016.

[17] Christoffersen, P. Backtesting Value-at-Risk: A Duration-Based Approach. Journal of Financial Econometrics, vol. 2, no. 1, 2004, pp. 84108., doi:10.1093/jjfinec/nbh004.