Multi-Layer Allocated Learning Based Neural Network for Resource Allocation Optimization

P. C. Ko\textsuperscript{1}, P. C. Lin\textsuperscript{2}, J. A. You\textsuperscript{2}, Y. J. Tien\textsuperscript{1}, H. C. Wang\textsuperscript{1}

\textsuperscript{1}Department of Information Management
\textsuperscript{2}Institute of Finance and Information
\textsuperscript{1,2}National Kaohsiung University of Applied Sciences, Taiwan ROC.

Abstract
The investor’s asset allocation choice deeply depends on the trade-off between risk and return. The well-known mean variance method requires predetermined risk and expected return to calculate optimal investment weights of portfolio. The artificial neural network (ANN) with nonlinear capability is proven to solve large-scale complex problem effectively. However, the traditional ANN model cannot guarantee to produce reasonable investment allocation, because the summation of investment weight may not preserve 100\% in output layer. This article introduces a multi-layer allocated learning based neural network model and takes financial portfolio as an example to optimize assets allocation weights. This model dynamically adjusts the investment weight as a basis of 100\% of summing all of asset weights in the portfolio. The experimental results demonstrate the feasibility of optimal investment weights and superiority of ROI of buy-and-hold trading strategy compared with benchmark TSE (Taiwan Stock Exchange).

Keywords: Resource allocation, neural network, allocated learning based NN, portfolio, investment weight.

1. Introduction
Most investment professionals or investors consider asset allocation as the most important part of portfolio construction. Asset allocation of portfolio is concerned with the percentage of the overall portfolio value allocated to each portfolio individual. The mean variance model for the portfolio asset allocation is one of the best known models. This model requires satisfying the two conflicting optimization criteria which minimizes risk with predetermined the expected return of portfolio. Besides, the mean variance model may be desirable to restrict the number of assets in a portfolio and the percentage of the portfolio attends to any specific asset. The searching of efficient set frontier became much difficult, if that restriction exit.

In portfolio applications, using ANNs to portfolio management has gained interest in recent years. Hung, Liang and Liu [11] integrate the arbitrage pricing theory (APT) and ANNs to extracting risk factors and generating individual in portfolio. The empirical results indicate the integrated method beats the benchmark and ARIMA model. Chapados [9] demonstrated the success of ANNs with asset allocation framework according to a Value-at-Risk adjusted profit criterion for making asset allocation decisions. Both the forecasting and decision models are significantly outperforming the benchmark market performance. Eakins and Stansell [12] examine whether superior investment returns can be earned by using ANNs to perform forecasts based on a set of financial ratio to determine the intrinsic value of assets to enter the property portfolio. They find that the value ratio provides useful information that permits the selection of portfolio s that provide investment returns superior to the DJIA and S&P500. Hung, Cheung and Xu [5] present an extended adaptive supervised learning decision EASLD trading system to enhance portfolio management. Their researches take a balance between the expected returns and risks. Plikynas, Salalauskas and Poliakova [3] using ANNs to control nonlinear dynamics of heterogeneous foreign investment impact on national capitalization structure. The results of their research show better than multidimensional linear regression forecasting performance. Ellis and Wilson [2] applied ANNs to the Australian property sector stocks to construct a variety of value portfolios. Their risk-adjusted performances show the value portfolios outperform the benchmark by as much as 7.14\%.

Unfortunately, the traditional ANN model cannot produce reasonable allocation ratios, i.e. the summation of produced allocation ratios cannot retain 100\%, during the learning process. The aim of this paper is to introduce an allocated learning based neural network model to optimize the assets allocation in portfolio that will outperform the market. This approach suggests using quadratic programming to obtain the expected maximum variance of risk factor to risk assets returns of portfolio, and proposes a dynamic weight modification as a basis of 100\% of summing all of allocation ratios in the portfolio. Through the experimental results, the complex portfolio asset allocation management would be solved effectively by our model.

2. Multi-Layer Allocated Learning (MLAL) based Neural Network
In this section, an allocated learning algorithm applied to multi-layer neural network (MLALNN) with n inputs, one hidden-layer with h neurons and m outputs shown in Fig 1 is introduced. Let 
\[ a^l = \{a_1^l, a_2^l, \ldots, a_n^l\}^T \] denote the input signal, 
\[ a^H = \{a_1^H, a_2^H, \ldots, a_H^H\}^T \] denote the hidden signal and 
\[ y = \{y_1, y_2, \ldots, y_m\}^T \] denote the output signals. \( w^l \) and 
\( w^H \) represents the weight matrix shown in Equation (1) and 
Equation (2), where \( w_{i,j}^l \) refers to the synaptic weight connecting 
the hidden layer of neuron i to the input layer of neuron j, \( w_{i,j}^H \) 
refers to the synaptic weight connecting the output layer of neuron j to 
the hidden layer of neuron r. \( q(\cdot) \), \( b_r^H \) and \( y_r \) are the 
activation function, the bias applied to neuron r in hidden layer, the 
bias applied to neuron j in output layer and the output signals, respectively. \( a_{i,j}^H \) 
and \( v_j \) is denoted as the weighted sum of all synaptic inputs 
plus bias.

\[ w^l = \begin{bmatrix}
    w_{1,1}^l & w_{1,2}^l & \cdots & w_{1,n}^l \\
    w_{2,1}^l & w_{2,2}^l & \cdots & w_{2,n}^l \\
    \vdots & \vdots & \ddots & \vdots \\
    w_{n,1}^l & w_{n,2}^l & \cdots & w_{n,n}^l 
\end{bmatrix} \]  
(1)

\[ w^H = \begin{bmatrix}
    w_{1,1}^H & w_{1,2}^H & \cdots & w_{1,h}^H \\
    w_{2,1}^H & w_{2,2}^H & \cdots & w_{2,h}^H \\
    \vdots & \vdots & \ddots & \vdots \\
    w_{h,1}^H & w_{h,2}^H & \cdots & w_{h,h}^H 
\end{bmatrix} \]  
(2)

Fig 1. Multi-Layer Combinatorial Learning based 
Neural Network

Assume \( d = \{d_1^1Bd_1B \cdots d_mBd_m\} \) and 
\( e = \{e_1^1Bd_1B \cdots e_mBd_m\} \) represents the desired 
responses and error signal at the output layer. The 
instantaneous value \( \xi(k) \) of the total error energy at 
itration \( k \) is defined as the summation over all 
neurons in the output layer in Equation (3).

\[ \xi(k) = \frac{1}{2} \|d - y\|^2 = \frac{1}{2} \sum_{j=1}^{m} e_j^2(k) \]

where \( e = d - y \) 

(3)

To do the minimization, a manner similar to the 
Least Mean Square (LMS) algorithm is adopted to 
apply a correction \( \Delta w_{j,r}^H(k) \) shown in Equation (4) to its 
corresponding synaptic weights \( w_{j,r}^H(k) \), where \( \eta_{j,r} \) is 
the learning rate. To obtain better assets allocation in a 
portfolio problem, the summation of all output signals 
must be 100% (i.e. \( \sum_j y_j(k) = 1 \)) in learning epoch. \( \eta_{j,r} \) is 
not a fixed value commonly used in the conventional 
network. Especially, \( \eta_{j,r} \) is a variable calculated 
to maintain the summation of outputs always equal to 
100% in our allocated learning based algorithm.

\[ \Delta w_{j,r}^H(k) = -\eta_{j,r} \frac{\partial \xi(k)}{\partial w_{j,r}^H(k)} \]

(4)

It means

\[ w_{j,r}^H(k + 1) = w_{j,r}^H(k) + \eta_{j,r} \frac{\partial \xi(k)}{\partial w_{j,r}^H(k)} \]

(5)

According to the chain rule of calculus, we may 
express this gradient as

\[ \frac{\partial \xi(k)}{\partial w_{j,r}^H(k)} = \frac{\partial(\frac{1}{2} \|d - y\|^2)}{\partial e_j(k)} \frac{\partial e_j(k)}{\partial v_j(k)} \frac{\partial v_j(k)}{\partial w_{j,r}^H(k)} \]

where

\[ \frac{\partial \xi(k)}{\partial e_j(k)} = \frac{\partial(\frac{1}{2} \|d - y\|^2)}{\partial e_j(k)} = e_j(k) \]

(7)

\[ \frac{\partial e_j(k)}{\partial v_j(k)} = -1 \]

(8)

\[ \frac{\partial v_j(k)}{\partial w_{j,r}^H(k)} = q'(v_j(k)) \]

(9)

\[ \frac{\partial \xi(k)}{\partial w_{j,r}^H(k)} = \sum_{i=1}^{m} \frac{\partial w_{i,j}^H}{\partial w_{j,r}^H(k)} = a_{i,j}^H \]

(10)

Let \( \delta_j(k) \) denote the local gradient and be defined 
in Equation (11)

\[ \delta_j(k) = -\frac{\partial \xi(k)}{\partial v_j(k)} = e_j(k) \cdot q'(v_j(k)) \]

(11)

Equation (4) should be rewritten as Equation (12)

\[ \Delta w_{j,r}^H(k) = \eta_{j,r} \cdot \delta_j(k) \cdot a_{i,j}^H \]

(12)

After adjusting each weight \( w_{j,r} \), the summation of 
output signals would be 100%. It means that:

\[ \sum_{j=1}^{m} \sum_{r=1}^{l} q(w_{j,r}^H(k + 1) \cdot a_{i,j}^H) = 1 \]

(13)

where \( w_{j,0} = b_j \) and \( a_{i,j}^H = 1 \)

Substituting Equation (5), (12) into Equation 
(13), we obtain

\[ \sum_{j=1}^{m} \sum_{r=1}^{l} q((w_{j,r}^H(k) + \eta_{j,r} \cdot \delta_j(k) \cdot a_{i,j}^H) \cdot a_{i,j}^H) = 1 \]

(14)

Multiplying \( \eta_{row,col} \) (row=1, 2, ..., m and col=1, 
2, ..., l) with Equation (14) independently, we obtain 
following general equation form.
The data encompasses the entire period from January 1, 2000 to September 30, 2005. The training phase \( (k) \), the validation phase \( (v) \), and the test phase \( (l) \) are one period in each sliding window \( (SW) \) as shown in Fig 2 to cross-validate our model.

![Sliding windows simulation process](image)

**3.3 Neural Networks Parameters**

Effective neural network construction required employment of various optimizations setting. The root mean square error (RMSE) and expected return are considered as our key performance indexes in difference training iterations. RMSE is defined in Equation (20), where \( \hat{y} \) refers to the real testing data and \( y \) is produced by our MLALNN model. Minor changes in these parameters seem not to have a major effect on the performance in our experimental results.

\[
RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^{N} (y_i - \hat{y}_i)^2}
\]

The suitable number of iterations is set to 95 and number of neuron in hidden layer is set to 5 shown in Fig 3 and Fig 4, respectively, where the right side and left side of y-axis denote expected return \( E_{ROI} \).

![E_{ROI} in different training iterations](image)

![E_{ROI} with different number of neuron in hidden layer](image)

**3.4 Analysis of Results**

The performances of our proposed MLALNN model are summarized into two parts: (1) the comparisons of investing returns in each sliding window, and (2) analysis of ROI contributed by different asset allocations in each sliding window. Each sliding window is done five times and their mean values are taken to obtain more general results. Let \( ROI_{MLALNN} \) and \( ROI_{TSE} \) refer to the obtained ROI by using MLALNN model or simply TSE indexes. From Table 3, it is shown that the average of \( ROI_{MLALNN} \) (14.2945%) is greater than average of \( ROI_{TSE} \) (0.5291%). Moreover, \( ROI_{MLALNN} \) is superior to \( ROI_{TSE} \) in each sliding window. Even if the \( ROI_{MLALNN} \) in \( SW_1(-0.9845\%) \) and \( SW_2(-3.7299\%) \) are negative, they are also higher than the corresponding \( ROI_{TSE} \) (-5.1540%) and (-9.8803%). These results are also shown in Fig 5. It demonstrates that MLALNN obtain better asset allocations effectively.

Table 3. Comparisons of \( ROI_{MLALNN} \) and \( ROI_{TSE} \) in each sliding window
4. Conclusions

This paper introduces a novel allocated learning based neural network model to optimize investment weight of portfolio that will outperform the market. This model proposes a dynamic weight modification as a basis of 100% of summing all of asset weights in the portfolio. Results on 21 companies selected to be our testing targets from Taiwan 50 Index Constituents demonstrate the feasibility of optimal investment weights and superiority of ROI based on buy-and-hold trading strategy. Through the experimental results, the complex portfolio asset allocation management would be solved effectively by our model. From our experiments, it appears that using allocated learning based neural network model will converge to two dimensions (the highest expected return and the lower RMSE), simultaneously. Using MLALNN model produces better return of investment than TSE in each sliding window. Our MLALNN model recommends increasing/decreasing investment weight of optimistic/pessimistic prospects of assets effectively.

5. References


