Multi-Dimensional Asset Allocation Strategy with DA-RNN

T. Y. Lee

Abstract

Most RoboAdvisors reflect the perspective of investment banks, which differs from commercial banks in Korea.
Most customers who use commercial banks have a conservative approach.
In customer-focused thinking, the more you design your Robo Advisor, the more important it is to minimize customer losses.
It was designed with the belief that the guarantee of principal through defense of the bear market would be a solid foundation to be returned to profits from the bear market.
The traditional asset allocation model is dedicated to simple predictions that take risk as a parameter of volatility and draw the expected return to calculate the optimal share of the asset. This is not a big problem when the market is good, but it's going to cause a loss of the customer's principal in a booming market. This is not in line with the bank's robovisor idea, and we have created deep learning algorithms to defend against the bear market.

Introduction

RoboAdvisor's favorite traditional asset allocation model is Black-litterman, which is typically implemented by human input in the form of a house view. In order to use the results of deep learning in the traditional asset allocation model, the first thing that needs to be done is the Data-Transform and the data architecture that will accommodate it.

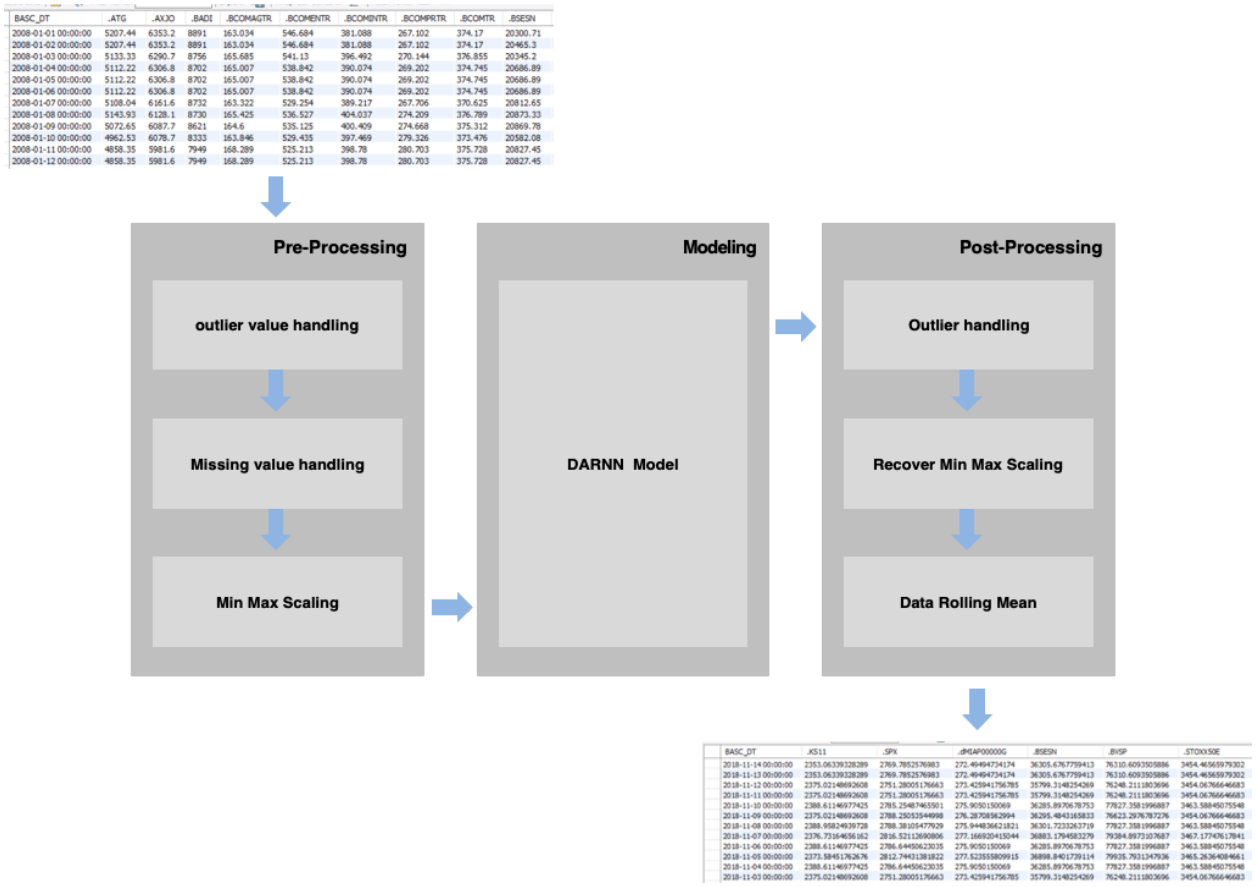
The ultimate differentiation is that it is designed to seamlessly combine the periodicity of deep learning data with existing data processes.
We introduced the Data Fusion Layer through analysis of existing batch data and introduced the Data Fusion Pipeline for the first time in the financial sector, creating an architecture to support it and creating a service model.

After deploying the traditional asset strategy allocation model in a multiple form, we applied the value as a prediction of the DA-RNN model generated from the index data.

As a result, the market flow was reflected and the portfolio was generated by calculating the share of investment by asset through this algorithm.

Deep Learning Model Adaptation Layer

The most important thing to apply deep learning is to check the periodicity of the data and the batch scheduler of the existing system. This allows us to reduce the size of a large amount of data, which is a significant contribution to infrastructure cost reduction and performance improvement.
The most important thing for customers in real service is the fast response speed.
We have configured the pretreatment and posttreatment processes as follows to maximize this.



Input Data

Input data consists of the closing price of major national indices, MSCI benchmark indices, exchange rates, bond yields and economic indicators, including investment universes. Currently, 130 indexes are used as of 08-20-20, and indexes can be added /

removed through user view or empirical simulation. The 130 indices mentioned were selected with reference to the prop trader views / reserch reports.

Research on Input Data

Financial Research Team Enhances Data Prediction Using FFT Algorithm

Pre-Processing

Outlier value handling

Outlier value elimination is used for faster convergence when training data on the network. Allow up to 6 months of volatility (standard deviation) * 2 if the change with the previous data exceeds that number, then interpolate with the average of the next and previous data. However, as the 2018-08-10 Turkish Lira exchange rate fluctuations can be normalized, outliers will be removed for less than five-year bonds and national economic indicators.

Missing value handling

Missing values basically use backward fill for daily data and interpolation for quarterly/monthly data such as economic indicators. However, unlike the general application, in the DaRNN model described below, when the economic indicator data was also backward filled, the simulation result was 2 ~ 3% better on average based on Hit Ratio. The exact reason is not known, but interpolation is likely to act as a noise because short-term forecasts of 1 to 3 months are more affected by exchange rates, bonds, and major indices than economic indicators. Therefore, rather than simply using the economic index, it is necessary to develop complementary economic indicators such as the Bank of Korea's BOK-COIN.

Min-Max Scaling

Min-Max scaling is used to stabilize the network's loss calculation and to achieve faster convergence. Among the cost functions, Euclidean distance is used when MSE Loss is used. When the scale range of a particular feature is wide, the cost can be deflected. It is also known that the learning convergence speed of the data scaled by the gradient decent is faster. [Ioffe *et al.*, 2015].

$$X_{sc} = \frac{X - X_{min}}{X_{max} - X_{min}}.$$

The formula uses the above scaling expression that is used by default.

Modeling

A Dual-Staged Attention-Based Recurrent Neural Network for Time Series Prediction [Qin et al., 2017] was used. This model is basically an autoregressive model, and predicts the target item value by inputting exogenous input and time series data of the target item.

$$y_t = F(y_{t-1}, y_{t-2}, y_{t-3}, \ldots, u_t, u_{t-1}, u_{t-2}, u_{t-3}, \ldots)$$

y_t : Target event closing price at time t

u_t : Exogenous Input at time t

F : Neural Network

The Layer and Model implementations are the same, but the DataLoader implementations differ.

Batch – Normalization

In many NN models such as CNN, GAN, and ANN, adding Batch Normalization Layer is doing better than not. And prevents Gradient Vanishing / Gradient Exploding from happening, enabling more stable and faster learning [Ioffe et al., 2015].

The normalization of data is done with Min-Max Scaling before data enters the network, but there is a problem that the average and the variance of the data change as it passes through many layers. To solve this problem, batch batch normalization layer between layers helps to keep data normalized.

Encoder

The Encoder section aims to learn abstract patterns of the exogenous input time series. Through Attention Layer, we learn variables to be weighted per sequence, and learn abstract patterns through LSTM.

Attention Layer

It is a variable learning method that combines the whole sequence data and the previous sequence to determine which variable weight is added at each time point. For example, let's say the stock prices of four stocks a, b, c, and d are predicted for 30 days and predict the stock price of stock a after 20 days. At this point, 3-vectors, excluding a, will be entered as inputs until the first, second, and nth days. Learning the weights wa, wb, wc, and wd for each item a, b, c, and d for the input of the k th day and passing [wat at, wbt bt, wct ct, wdt dt] to the input of the LSTM cell.

LSTM Layer

Learn the abstract pattern of sequential data with weighted inputs of the form [wat at, wbt bt, wct ct, wdt dt] created in the Input Attention Layer.

Decoder

Based on the abstract pattern of exogenous input learned in the encoder, the abstract pattern of endogenous input is studied.

Attention Layer

In the sequential pattern of exogenous input learned in the Encoder, we learn the weights of which sequence is important. Then, it is compressed with the self-attention vector by combining with the pattern of exogenous input.

Fully Connected Layer

Create a context vector that combines the information of the self-attention vector and the endogenous input.

LSTM Layer

The abstract pattern of the entire sequence is learned by combining the context vector containing the information of the endogenous input and the abstract pattern of the exogenous input learned in the encoder.

Fully Connected Layer

The target values of t are predicted by combining the result values of the LSTM layers.

Post-Processing

Outlier value handling

In order to increase the Hit Ratio of the relatively recent data, the students re-learn using the previous 60-90 days of data every 10-20 days. Lower network convergence For this reason, outlier values often occur at the beginning of the initial results.

Recover Min-Max Scaling

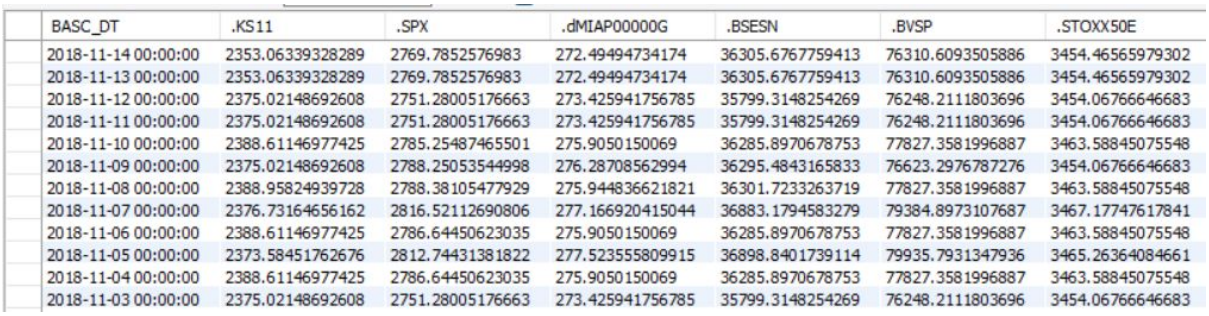
Since Min-Max Scaling was used in the pretreatment, the results are also generally between [0-1]. However, for the intuitive result analysis of the user, it is easy to come out in the form of the existing index, so it is restored to the original score.

Data Rolling Mean

Due to the backward filling of the missing weekend / holiday, the oscillating part of the

result is observed. To correct this, we get a smooth prediction graph through rolling mean (5 days).

OUTPUT DATA



BASC_DT	.KS11	.SPX	.dMIAP00000G	.BSESN	.BVSP	.STOXX50E
2018-11-14 00:00:00	2353.06339328289	2769.7852576983	272.49494734174	36305.6767759413	76310.6093505886	3454.46565979302
2018-11-13 00:00:00	2353.06339328289	2769.7852576983	272.49494734174	36305.6767759413	76310.6093505886	3454.46565979302
2018-11-12 00:00:00	2375.02148692608	2751.28005176663	273.425941756785	35799.3148254269	76248.2111803696	3454.06766646683
2018-11-11 00:00:00	2375.02148692608	2751.28005176663	273.425941756785	35799.3148254269	76248.2111803696	3454.06766646683
2018-11-10 00:00:00	2388.61146977425	2785.25487465501	275.9050150069	36285.8970678753	77827.3581996887	3463.58845075548
2018-11-09 00:00:00	2375.02148692608	2788.25053544998	276.28708562994	36295.4843165833	76623.2976787276	3454.06766646683
2018-11-08 00:00:00	2388.95824939728	2788.38105477929	275.944836621821	36301.7233263719	77827.3581996887	3463.58845075548
2018-11-07 00:00:00	2376.73164656162	2816.52112690806	277.166920415044	36883.1794583279	79384.8973107687	3467.17747617841
2018-11-06 00:00:00	2388.61146977425	2786.64450623035	275.9050150069	36285.8970678753	77827.3581996887	3463.58845075548
2018-11-05 00:00:00	2373.58451762676	2812.74431381822	277.523555809915	36898.8401739114	79935.7931347936	3465.26364084661
2018-11-04 00:00:00	2388.61146977425	2786.64450623035	275.9050150069	36285.8970678753	77827.3581996887	3463.58845075548
2018-11-03 00:00:00	2375.02148692608	2751.28005176663	273.425941756785	35799.3148254269	76248.2111803696	3454.06766646683

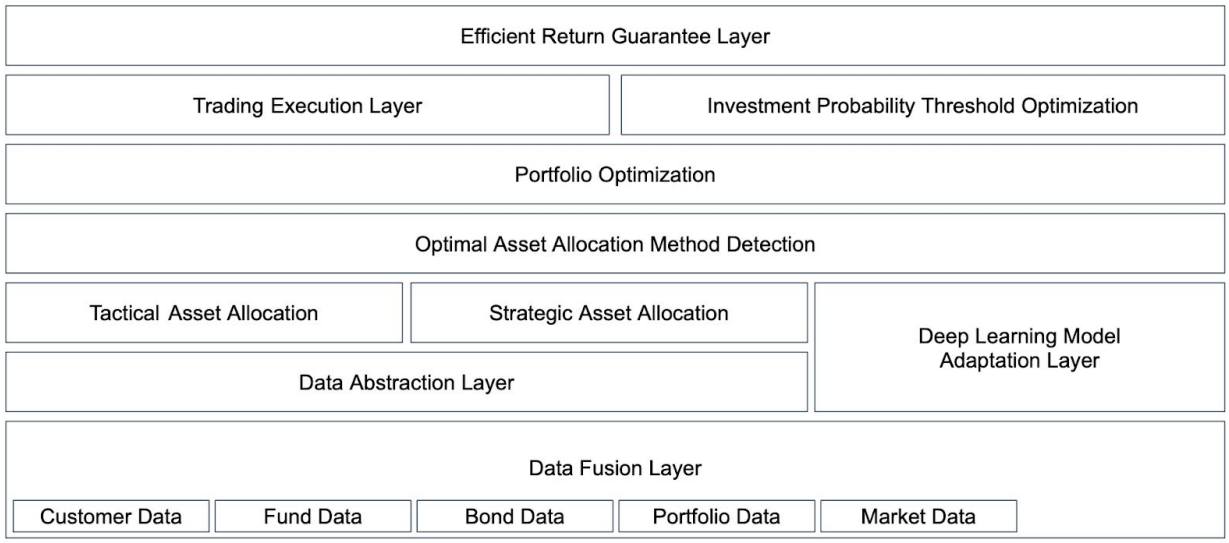
As shown in the above figure, data is created with the date as the index value and the individual index as the column.

Optimization

Multi-Dimensional Choice of Traditional Asset Allocation Models for the Market (Black-litterman, Risk-Parity, Momentum, etc.)
Leveraging deep-market forecasts as input to traditional asset allocation models
Prototype verification of various models using Pytorch to detect optimal model generation parameters and auto-ml implementation
In order to commercialize, stable verification of the output value and generated parameters after source conversion with Tensorflow is performed.

Conclusion

As a result, I constructed the following architecture.
It is a configuration that can flexibly apply data flexibility and new technology.


Efficient Return Guarantee Layer
Trading Execution Layer
Investment Probability Threshold Optimization
Portfolio Optimization
Optimal Asset Allocation Method Detection
Tactical Asset Allocation
Strategic Asset Allocation
Deep Learning Model Adaptation Layer
Data Abstraction Layer
Data Fusion Layer
Customer Data
Fund Data
Bond Data
Portfolio Data
Market Data