Multi-Task Learning for Stock Selection - Technical Updates

Kunj Golwala
golwala.1@iitj.ac.in

Indian Institute of Technology
Jodhpur
Rajasthan

Technical Updates

Multi Task Learning (MTL) is a subfield of machine learning in which multiple tasks are performed simultaneously and hidden layers are shared based on the commonalities of the tasks. This process helps in improving the generalization performance and also makes the learning and training process faster since all neural networks run parallely.

Free Parameter selection - According to the method used in the paper, one needs to specify which parameters are free and which parameters are shared/const The basic idea is to select which parameters and hidden layers should be shared on the basis of some measure specified by the learner. If this is not done, then one would have to select the parameters and hidden layers to be shared each and every time we use a new set of stocks/assets from different stock exchanges, since these stocks would be interrelated differently and have different commonalities as compared to the previous set of assets. The basic idea is to re parametrize the parameters.Let θ_i be the parameters of the i^{th} model and θ_i belongs to R^{n_1} i.e. has n_1 dimensions. For n models, one can re parametrize the parameters as $\theta_i = f(p_i,w)$ where $p_i \in R_2^n$, $w \in R_3^n$, with the constraint condition as n X n_1 < (n X n_2) + n_3. If f is a linear function, all the parameters of the n models will lie on the same linear space. The position of a point on this linear space will be given by a n_2 dimensional vector p_i and the linear space is characterized by n_3 parameters of w. Now which parameters should be shared and which parameters should be free will be determined by the distance of p_i from the parameters of other models, p_j's. Now one does not have to specify which parameters are free and which parameters are shared. Instead one needs to specify how many parameters are free per model and the shape of the space of f. Using this method, we can extend the networks to any dataset without having to check the sharing of parameters.

Borrowing and Lending - In our experiment, short-selling,borrowing and lending of stocks is not allowed. In real-life scenarios, these practices are used by wholesale traders and large investment companies. In terms of our model, it means that the weights of the different assets of the portfolio can only be a non-negative value and the sum of all these weights must be equal to one. When one uses practices like borrowing or lending, the sum of all the weights will not be equal to one since the total assets are changing along with the investment amount. When shares are borrowed, the total of the weights would be greater than one and the weights of all other stocks will have to be adjusted so as to bring the total back to one. This means that the parameters will have to be changed. Therefore, the trading module should be designed in a way such that it can assign negative weights also to the assets.

Checking for overfitting - In our experiment, we are training the ANN for 120 epochs. However, we are not checking the model for overfitting on the training data. One can check the values of training accuracy and validation/test accuracy for each epoch. We can use the monthly data of the time , 6 months before we began recording data for our experiment as validation or test data. If the accuracy for training accuracy is significantly higher than the test accuracy for a particular epoch, it means that the model is getting overfitted on the training data from that epoch onwards. By using this technique, we can determine the optimal number of epochs to be used for training the model by using early stopping(stopping at the epoch where there is large difference between training and validation/test accuracy) with a weight constraint(already used in the model). We can also improve regularization by adjusting the hyperparameters like number of hidden layers,number of nodes in each layer,etc to keep a check on overfitting. We can change the network complexity by changing the number of weights or the value of weights. A technique called dropout may also be used which involves probabilistically removing some inputs during training.

The model implemented in the paper can be improved by fine tuning the parameters and carefully selecting the appropriate hyperparameters. The fact that the model performed so well on such a large data set shows that it is already very refined, but the above stated points can possibly make the model even better for use in real life.