Gated Recurrent Unit Hierarchical Architecture for Fundamental Stock Analysis and Forecast

Gabriel Adriano de Melo
Department of Electrical and
Computer Engineering
Instituto Tecnológico de Aeronáutica
São José dos Campos, Brasil
Email: gam@ita.br

Paulo Marcelo Tasinaffo
Department of Electrical and
Computer Engineering
Instituto Tecnológico de Aeronáutica
São José dos Campos, Brasil
Email: tasinaffo@ita.br

***Abstract*—This work proposed a novel technique for fundamental analysis using a shared, hierarchical neural network model, based on the state-of-the-art Gated Recurrent Unit (GRU). The model uses the quarterly reports to predict the mean stock price at the next report, a three month ahead forecast based on past performance. There are 3 hierarchical models in the proposed architecture: the first is a general recurrent feature extractor, that can be interpreted as an encoder, the second is a specialized feedforward layer that captures the information from companies in the same sector, the third is a highly specialized logistic regression performed on the activations from the second model, trained separately for each company. This structure provides a solid background for transfer learning in which multiple neural networks, one for each stock, has most of it weights shared between its instances.**

I. Introduction

As a way to apply novel Deep Learning techniques to the financial field, an extensive research was carried out in order to prospect the commonest problems, from which, time series forecasting from the stock prices was found. A deeper review has found that many researchers use only the stock prices data to make a prediction. It turns out that this time series with a high frequency sampling has a high autocorrelation, and a frequency distribution that is similar to a Brownian Motion, which implies that the best estimator is simply the last known price of the times series with a factor of the drift.

Another common pitfall for researchers that have trained Machine Learning Models with more than thousands of parameters, specially neural networks models, is to have a biased estimative for the model's performance given the evaluation on the test set. For instance, [1] argues that most claimed research findings in financial economics are likely false. The reason for this is that the test set has also been used to fine-tune those parameters, and so the models were sampled and discarded based on the same set.

In order to avoid the short-term unpredictability that the stock prices show, whose best estimator is the current price, a longer quarterly sampling frequency was chosen, in which fundamental indicator's for the companies are released. Therefore, the analysis is predicated on those firm characteristics and under the assumption that those metrics govern long-term stock returns, as stated by [4].

An Artificial Neural Network architecture was proposed using a Gated Recurrent Unit layer followed by a hierarchical arrangement of fully connected layers, in which the last two layers were fine tunned according to the company and its segment. This model can be think of an ensemble of Neural Networks, for each company, that shares the same parameters for the initial layers. This feature extraction is thought to be invariant for the company and for the time period considered,

This paper is divided in the following manner: Section II brings a literature review from papers that also have applied neural networks in financial forecasting; Section III provides a basic understanding of neural networks and the architectures used as base for developing the proposed model; Section IV describes the network's development discussing the range of feasible hyper-parameters; Section V provides the implementation details and execution results for this work; Finally, Section VI summarize this work with suggestions for future research.

II. Related Work

Forecasting data series using neural networks has been widely use in many different areas such as electric load forecasting [5] using recurrent neural networks with the Long-Short Term Memory (LSTM) cell.

Most of the research for financial market forecasting takes into consideration only the stocks prices and its technical indicators (functions of the price series) as seen on [2], [3], [6]. These researches also use only a training and a test set on which their models are evaluated for hyper-parameter tunning, giving a biased estimate of the performance.

Deep feedforward neural networks were employed by [7] using only technical indicators (price based) in high-frequency having a performance that was close to random chance. Convolutional neural networks (CNN) were also used to model high frequency trading [8] using a spatio-temporal modeling by spatial-time diagrams. Auto-encoders were employed by [9] to predict the stock prices in order to have a portfolio that would outperform the market index, but the returns give in the test set were worst than the index.

Reinforcement learning with deep neural network was applied by [10] with results that outperformed the Dow Jones Industrial average from 2016 to 2018, but the agent had only

30 stocks to choose from. This limitation on the stock number and which criterion were used to choose these stocks weren't discussed in that paper, a finding that undermines the research.

The work of [11] applied several architectures for price time-series forecasting such as linear algorithms (auto regressive moving averages) and various neural networks architecture: convolutional neural network (CNN), recurrent neural network (RNN), Long Short-Term Memory (LSTM) and Multilayer Perceptron (MLP). The work did not provide a validation set and it reported a forecast that was close to the naive approach of predicting the future price as the current price. The approach used directly the undifferentiated time series with mean square error (MSE), which may be misleading since these time series has a high autocorrelation.

Factorization and empirical mode decomposition based neural networks were also employed by [12], who reported good results but it didn't have a validation set.

III. Theoretical Background

The Artificial Neural Networks are universal function approximators whose optimizing objective (in supervised learning) is to minimize the loss (the error) of its outputs with respect to the ground truth. Biology has provided the inspiration for developing this field in which neurons are the basic unit of computation and the information is said to be distributed across the synaptic connections between the neurons. In the computation model, a neuron represents only a node where its inputs are linearly summed with a bias and then fed to an activation function. It is crucial that the activation function should be non-linear so that the whole network can compute a non-linear function from its inputs.

The most common optimization technique is backpropagation, in which the partial derivatives of the loss function with respect to the activations are successively calculated from the last layer to the first layer.

A. Recurrent Neural Networks

Recurrent neural networks were developed in order to receive a sequence of data as input using its hidden states activations as an extra input for the next sequence step. This characterizes a feedback loop inside the neural network, that can be interpreted as a memory. The outputs of this network not only depends on its inputs but it also depends on its past activations, as shown on the Equation 1. In this equation, $A^{<t>}$ is a matrix that represent the activations in a layer of the recurrent network at the instant t, σ is an activation function that is applied element-wise, $X^{<t>}$ is the input vector for this layer at time t, W_{ax} and W_{aa} is the weight matrix for the inputs and the past activations respectively.

$$A^{<t>} = \sigma(W_{ax}X^{<t>} + W_{aa}A^{<t-1>}) \tag{1}$$

This property has made recurrent neural networks perform well on sequence tasks as machine translation and speech recognition. It is important to note that the simple fully connected recurrent neural network suffers from the vanishing and exploding gradient problems, caused by the multiplicative nature of backpropagating the error signals. This leads to an exponential increase or decrease of the gradient, making learning difficult for deep networks or for long time dependencies in the backpropagation through time.

B. Gated Recurrent Units

The most recent improvement for the Long-Short Term Memory (LSTM) [13] has been the Gated Recurrent Unit (GRU) that simplifies the cell architecture while maintaining similar or even better performance [14]. The Figure 1 shows the dataflow diagram of a GRU cell.

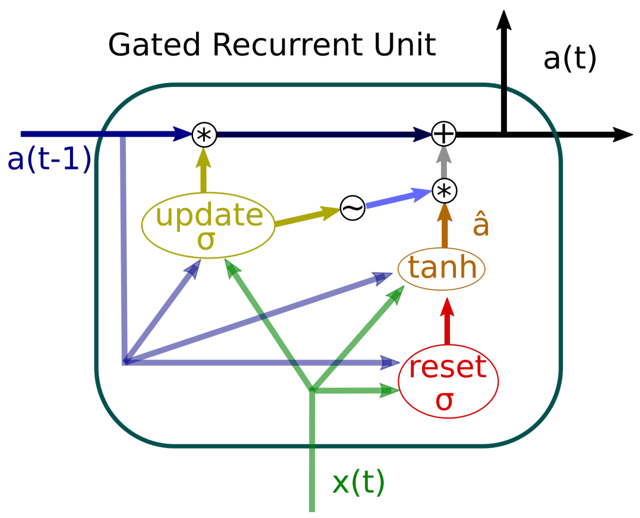

Fig. 1. Gated Recurrent Unit (GRU) cell represented by its dataflow diagram.

The GRU cell state is summarized by the Equation 2, where W are the weight matrix, Γ_u, Γ_r are the coefficients calculated from the Update and Reset gates, respectively.

$$\begin{aligned}\hat{A}(t) &= \tanh(W_{aa}(\Gamma_r * A(t-1)) + W_{ax}x(t))\\ \Gamma_r &= \sigma(W_{ra}\Gamma_r A(t-1) + W_{rx}x(t))\\ \Gamma_u &= \sigma(W_{ua}\Gamma_r A(t-1) + W_{ux}x(t))\\ A(t) &= \Gamma_u * \hat{A(t)} + (1-\Gamma_u) * A(t-1)\end{aligned} \tag{2}$$

IV. Proposed Model

In this section, the Hierarchical Architecture using a layer of Gated Recurrent Units (GRU) is presented. This model is used to predict the price variation of a stock in a 3-month time-step given its fundamentals and past price variation. This characterize a regression task.

This network has 3 different hierarchies in which distinct features are extracted. The innermost representation is responsible for generalizing between all company fundamentals data, being a sequence-to-sequence model. The idea is that there are fundamental companies characteristics between the dataset that can be summarized across all training cases. Therefore, this first model should trained across all companies.

The second hierarchy defines the sector to which an specific company pertains. In this way, every sector has its own weights that are shared between all companies within the same sector. This was developed because different sectors have distinct

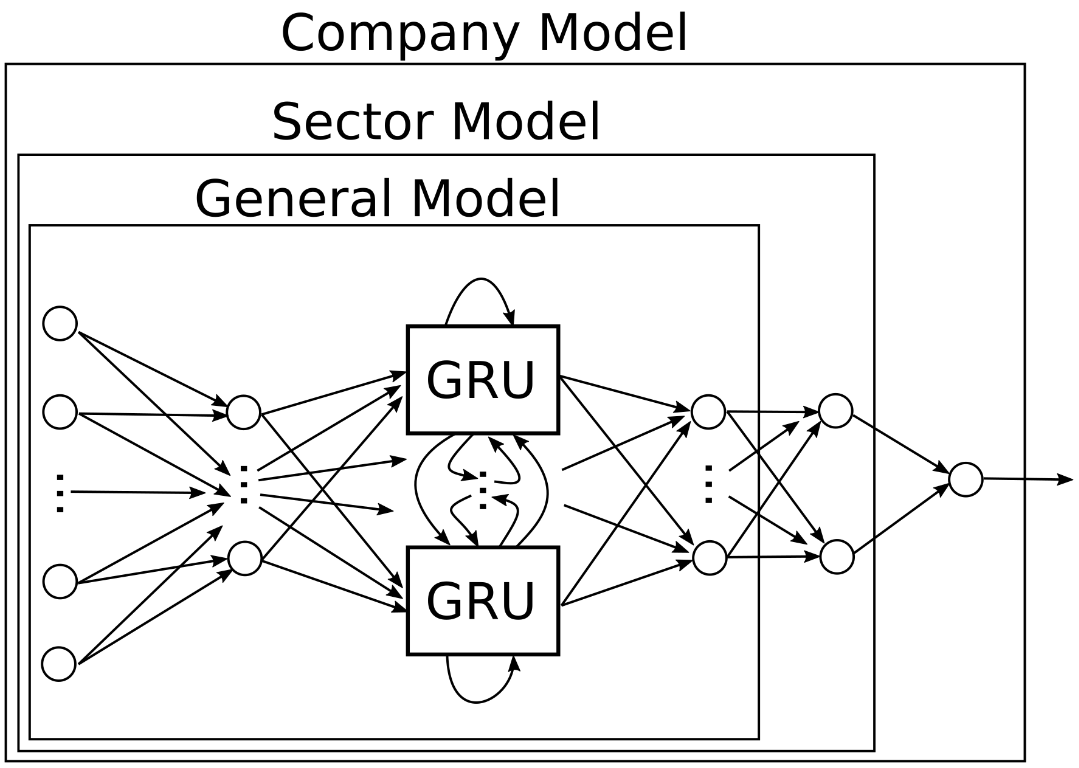

Fig. 2. A model for the proposed Gated Recurrent Unit Hierarchical Architecture.

fundamental ratios that should be taken in consideration when comparing the company data from those sectors.

Finally, the last hierarchy is just a logistic regression of the past hierarchical features, for which the prices estimates are made. This last layer depends only on one company data, therefore it has a minimal amount of parameters in order to avoid overfitting. This description is summarized in the Figure 2.

V. Implementation

The Python framework Keras [15] was used to build the neural network architecture proposed. The mean squared error (MSE) was defined as the loss function, observed in Equation 3, where Y_i is the target value at the training case i and $\hat{Y}_i$ is the network's prediction for this case. The batch size was defined as 32.

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(Y_i - \hat{Y}_i)^2 \tag{3}$$

As the optimization heuristics, the adaptive moment estimation (Adam) [16] has been selected due to its efficiency and robustness provided by the estimates of the average first moment and second moments of the gradients. In other words, this estimation combines the momentum of the gradient, defined as its exponentially weighted sum (exponential moving average) observed in the equation 4, and also the normalization factor that is the momentum of the square of the gradient (element-wise) observed in the Equation 5. Its hyper-parameters are the learning rate α, whose initial value was set as 0.01; β_1, the smoothing factor for the first moment, with default value 0.9; β_2, the smoothing factor for the second moment, with default value 0.999; and ϵ, a small constant that guarantees numeric stability in the update equation's denominator, seen in Equation 6, avoiding division by zero.

In the Equations 4, 5, 6, W denotes the weight vector that represents all trainable parameters in the neural network, V is the exponentially weighted sum from the gradient and S is from the square (element-wise) of the gradient. These equations are evaluated at every mini-batch iteration in the training phase, as the gradients estimates are evaluated.

$$V_{updated} = \beta_1 V + (1-\beta_1)\nabla W \tag{4}$$

$$S_{updated} = \beta_2 S + (1-\beta_2)(\nabla W)^2 \tag{5}$$

$$W_{updated} = W - \alpha\frac{V_{updated}}{\sqrt{S_{updated}}+\epsilon} \tag{6}$$

A. Data Overview

The data was acquired at the *Corporate fundamental data* provided freely by the Tilden Group [17]. The Tilden Group states that this dataset has come from over 20 years of 10-Q and 10-K filings made by public companies with the U.S. Securities and Exchange Commission. They affirmed to have extracted the data from both text and XBRL filings,

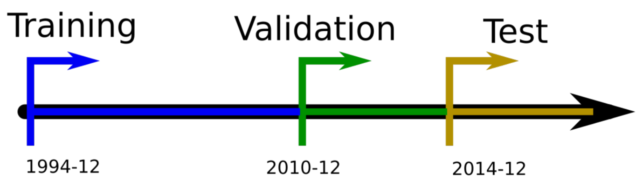

Fig. 3. Dataset segmentation in training, validation and test sets.

normalizing the data into quarterly time series of final restated values.

The data from 763 available companies was initially verified by discarding stocks that were early than 1994 or that had all data of a indicator missing. In order to get information about the sectors each company was categorized, new data from another plataform, SimFin [18], was freely acquired. Only companies that had a category in this dataset was kept, while the others were discarded. After those steps of discarding data that was unattainable, only 209 companies had remained.

It is important to note that some values were missing from some fundamental data entries for almost all companies, so it was necessary to interpolate those values. A linear interpolation was used. Besides missing values, there were also outliers that have been found in the data that were the result of some mistake, by having a value as big as 10^18. Using the median deviation to the median, those outliers were filtered and theirs values were also interpolated. Those problems found in the dataset may harm the model performance by breaking some of the structural patterns of the data.

The following indicators were used directly as input, after being normalized: EPS basic, EPS diluted, Dividend per share, ROE (Return on equity), ROA (Return on assets), P/B ratio (the ratio of Price to Book value of equity per share), P/E ratio (the ratio of Price to EPS diluted TTM as of the previous quarter), Dividend payout ratio, Long-term debt to equity ratio, Equity to assets ratio, Net margin (the ratio of Earnings TTM to Revenue TTM), Asset turnover (the ratio of Revenue TTM to TTM average Assets), Free cash flow per share, Current ratio.

The following indicators were divided by its value at the previous time-step (ratio normalization): Assets, Current Assets, Liabilities, Current Liabilities (at the end of a quarter), Shareholders equity (includes both common and preferred stockholders), Goodwill & intangibles, Long-term debt, Revenue, Earnings, Earnings available for common stockholders, Cash from operating activities, Cash from investing activities, Cash from financing activities, Cash change during period, Cash at end of period, Capital expenditures, Price (the medium price per share of the company common stock during a given quarter as reported, not adjusted for subsequent dividends), Book value of equity per share, Cumulative dividends per share, Non-controlling interest, Preferred equity. This normalization is shown by Equation 7 where the $R(t)$ is the ratio normalization performed, $V(t)$ is the value to be normalized at time-step t and ϵ is a small constant to avoid division by

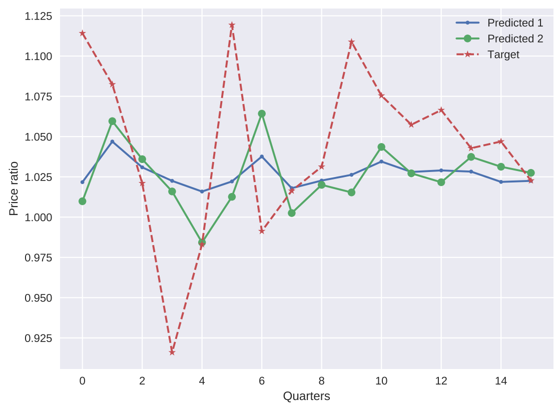

Fig. 4. The mean ratio of the price predicted by the models.

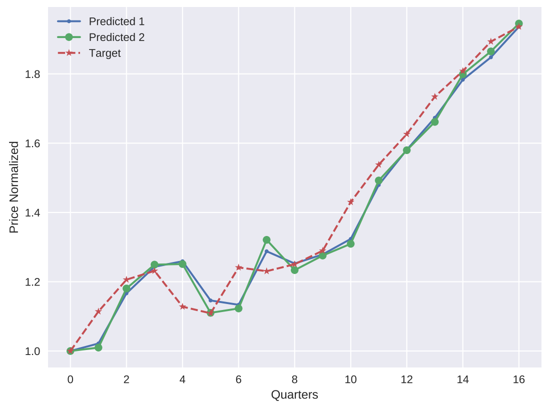

Fig. 5. The mean price normalized predicted by the models.

zero, used with value 10^-8. If the value of $V(t-1)$ was zero, $R(t)$ would also be masked to zero, signaling a abnormal ratio to the neural network and preventing large ratios.

$$R(t) = \frac{V(t)}{V(t-1)+\epsilon} \tag{7}$$

The sector were distributed in the following way with the company count: Farm 1, Beverages 5, Publishing 3, Communication 7, Chemicals 7, Transportation 5, Industrial 19, Forest 1, Employment 1, Steel 1, Oil 16, Biotechnology 1, Consulting 1, Metals 3, Brokers 2, Retail 17, Application 11, Aerospace 5, Tobacco 1, Medical 13, Health 4, Packaging 3, Computer 5, Drug 6, Utilities 11, Business 5, Airlines 4, Autos 3, Entertainment 2, Engineering 1, Restaurants 2, Semiconductors 8, Personal 1, Building 2, Travel 5, Consumer 18, Manufacturing 5, Asset 2, Advertising 2.

There were initially 96 quarters. After differentiating and taking one time step for forecasting, 94 quarters remained, from which the last 16 were separated to the test set, the last

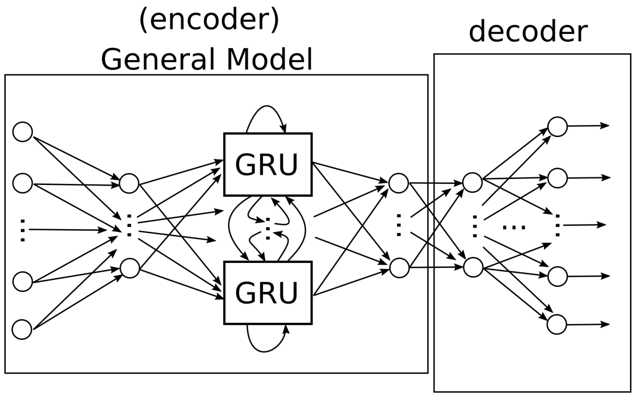

Fig. 6. An alternative training method for the General Model feature encoder.

remaining 16 to the validation set and the remaining 62 were taken to the training set, as shown in the Figure 3.

The results of the model are shown in Figure 4, where the actual outputs and targets to the neural network are shown. There were two models, the first which had fewer parameters and the second with more that generated, respectively, the predictions 1 and 2. It is important to note that model 2 has suffered from overfitting as it had a MSE of 0.718 in comparison with 0.361 from model 1. While model 1 had only 1 GRU, model 2 had 6.

Another way of training the General Model is trying to predict more than one characteristic at the same time, what is called multi-task learning, shown in the Figure 6. This alternative would be similar to an autoencoder with the difference that it is not trying to reproduce its input but rather predict the next sequence.

VI. Conclusions

This work's technique for forecasting stock prices from fundamental analysis using a shared, hierarchical neural network model, based on the state-of-the-art Gated Recurrent Unit (GRU) was not better than random choice. Even tough the model have used the quarterly reports to predict the mean stock price at the next report, a three month ahead forecast based on past performance, the effect of short term oscillations was still high.

There were 3 hierarchical models in the proposed architecture: the first was a general recurrent feature extractor, that can be interpreted as an encoder, the second was a specialized feedforward layer that captures the information from companies in the same sector, the third aws a highly specialized logistic regression performed on the activations from the second model, trained separately for each company. This structure provided a solid background for transfer learning in which multiple neural networks, one for each stock, had most of it weights shared between its instances.

In this way, the three-month time window for the prices estimate was very noisy and non-stationary due to the variety of factors that may affect the stock markets [19]. The historical corporate fundamental data series is not easily available, for instance, the Compustat database needs credentials to be accessed through Wharton Research Data Services, a database that was inaccessible to us.

References

[1] C. R. Harvey, Y. Liu, and H. Zhu, "… and the cross-section of expected returns," *The Review of Financial Studies*, vol. 29, no. 1, 2016.
[2] D. A. Hara, M. A. Botelho, A. Panariello, and C. H. C. Ribeiro, "Algorithmic trading using artificial intelligence tools," in *Workshop of Artificial Intelligence Applied to Finance (WAIAF)*, 2018.
[3] E. Jabbur, R. Oliveira, and A. Pereira, "Proposal and implementation of machine learning and deep learning models for stock markets," in *Workshop of Artificial Intelligence Applied to Finance (WAIAF)*, 2018.
[4] E. F. Fama, "Market efficiency, long-term returns, and behavioral finance," *Journal of Financial Economics*, vol. 49, pp. 283–306, 1998.
[5] J. Zheng, C. Xu, Z. Zhang, and X. Li, "Electric load forecasting in smart grids using long-short-term-memory based recurrent neural network," in *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, March 2017, pp. 1–6.
[6] Y. Song, "Stock trend prediction: Based on machine learning methods," Master's thesis, University of California Los Angeles, 2018.
[7] E. Chong, C. Han, and F. C. Park, "Deep learning networks for stock market analysis and prediction: Methodology, data representations, and case studies," *Expert Systems with Applications*, vol. 83, pp. 187–205, 2017.
[8] M. F. Dixon, N. G. Polson, and V. O. Sokolov, "Deep learning for spatio-temporal modeling: Dynamic traffic flows and high frequency trading," *Appl Stochastic Models Bus Ind.*, pp. 1–20, 2018.
[9] J. B. Heaton, N. G. Polson, and J. H. Witte, "Deep learning for finance: deep portfolios," *Applied Stochastic Models in Business and Industry*, vol. 33, pp. 3–12, 2016.
[10] Z. Xiong, X.-Y. Liu, S. Zhong, H. B. Yang, , and A. Walid, "Practical deep reinforcement learning approach for stock trading," in *NIPS 2018 Workshop on Challenges and Opportunities for AI in Financial Services: the Impact of Fairness, Explainability, Accuracy, and Privacy, Montréal, Canada.*, 2018.
[11] H. M, G. E. A., V. K. Menon, and S. K. P, "Nse stock market prediction using deep-learning models," *Procedia Computer Science*, vol. 132, pp. 1351–1362, 2018.
[12] F. Zhou, H. min Zhou, Z. Yang, and L. Yang, "Emd2fnn: A strategy combining empirical mode decomposition and factorization machine based neural network for stock market trend prediction," *Expert Systems With Applications*, vol. 115, pp. 136–151, 2019.
[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997. [Online]. Available: http://dx.doi.org/10.1162/neco.1997.9.8.1735
[14] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: http://arxiv.org/abs/1412.3555
[15] F. Chollet *et al.*, "Keras," https://keras.io, 2015.
[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, 2015.
[17] L. Tilden Group, "Corporate fundamental data," http://www.stockpup.com/data/, 2018.
[18] S. UG, "Simplifying finance: Data finder," https://simfin.com, 2018.
[19] J. Alberg and Z. C. Lipton, "Improving factor-based quantitative investing by forecasting company fundamentals," in *31st Conference on Neural Information Processing Systems (NIPS)*, 2017.