Conditional Time Series Forecasting with Convolutional Neural Networks

Anastasia Borovykh, Sander Bohte, and Cornelis W. Oosterlee

Dipartimento di Matematica, Università di Bologna, Bologna, Italy
Centrum Wiskunde & Informatica, Amsterdam, The Netherlands
Delft University of Technology, Delft, The Netherlands
borovykh_a@hotmail.com, s.m.bohte@cwi.nl, c.w.oosterlee@cwi.nl

Abstract. We develop a modern deep convolutional neural network for conditional time series forecasting based on the recent WaveNet architecture. The proposed network contains stacks of dilated convolutions that widen the receptive field of the forecast; multiple convolutional filters are applied in parallel to separate time series and allow for the fast processing of data and the exploitation of the correlation structure between the multivariate time series. The performance of the deep convolutional neural network is analyzed on various multivariate time series including commodities data and stock indices and compared to that of the well-known autoregressive model and a fully convolutional network. We show that our network is able to effectively learn dependencies between the series without the need of long historical time series and significantly outperforms the baseline neural forecasting models.

Keywords: Convolutional neural network; dilated convolutions; financial time series; residual connections

1 Introduction

Forecasting financial time series using past observations has been a significant topic of interest for obvious reasons. It is well known that while temporal relationships in the data exist, they are difficult to analyze and predict accurately due to the non-linear trends and noise present in the series. Feedforward neural networks have been a popular way of learning the dependencies in the data, by e.g. using multiple inputs from the past to make a prediction for the future time step [1]. One downside of classical feedforward neural networks is that a large sample size of data is required to obtain a stable forecasting result.

The main focus of this paper is on multivariate time series forecasting, specifically financial time series. In particular, we forecast time series conditional on other, related series. Financial time series are known to both have a high noise component as well as to be of limited duration – even when available, the use of long histories of stock prices can be difficult due the changing financial environment. At the same time, many different, but strongly correlated financial time series exist. Here, we aim to exploit multivariate forecasting using the notion of conditioning to reduce the noisiness in short duration series. Effectively, we use multiple financial time series as input in a neural network, thus conditioning the forecast of a time series $x(t)$ on both its own history as well as that of a second (or third) time series $y(t)$. Training a model on multiple stock series allows the network to exploit the correlation structure between these series so that the network can learn the market dynamics in shorter sequences of data. Furthermore, using multiple conditional time series as inputs can improve both the robustness and forecast quality of the model by learning long-term temporal dependencies in between series, even if these are misaligned.

A convolutional neural network (CNNs) is a type of network that has recently gained popularity due to its success in classification problems (e.g. image recognition [6] or time series classification [12]). Here, we employ this type of network and show that long-term temporal dependencies in and between financial time series can be learned by means of a deep convolutional neural network based on the WaveNet model [11]. The network makes use of dilated convolutions applied to multiple time series so that the receptive field of the network is wide enough to learn both short and long-term dependencies. We present an Augmented WaveNet architecture suitable for learning conditional time series by including batch normalization [5] and using a $1 \times k$ convolution with parametrized skip connections [3] from the input time series as well as the time series we condition on, in this way learning long-term interdependencies in an efficient manner. This improves the forecast, while at the same time limiting the requirement for a long historical price series and reducing the noise, since it allows one to exploit the correlations in between related time series. Knowing the strong performance of CNNs on classification problems we show that they can be applied successfully to forecasting financial time series, without the need of large samples of data. We compare the performance of the Augmented WaveNet model to a state-of-the-art fully convolutional network (FCN), as used successfully in [12] and [9] for time series classification, and an autoregressive model popular in econometrics and show that our model is much better able to learn important dependencies in between financial time series resulting in a more robust and accurate forecast.

2 The model

Consider a one-dimensional time series $x = \{x(0), ..., x(N-1)\}$. The task is for a predictor is to output the next value $x(t)$ conditional on the series' history, $x(0), ..., x(t-1)$ by maximizing the likelihood function

$$p(x) = \prod_{t=0}^{N-1} p(x(t)|x(0), ..., x(t-1)). \tag{1}$$

To learn this likelihood function, we present a convolutional neural networks in the form of the WaveNet architecture [11] augmented with a number of recent architectural improvements for neural networks such that the architecture can be applied successfully to time series prediction. In a convolutional neural networks, in each layer l the input feature map, $f_{l-1} \in \mathbb{R}^{1 \times N \times M_l}$ is convolved with a set of M_{l+1} filters $w_l^h \in \mathbb{R}^{1 \times k \times M_l}$, $h = 1, ..., M_{l+1}$.

Time series often display long-term correlations: to enable the network to learn these long-term dependencies we use stacked layers of dilated convolutions. As in [13], a dilated convolution outputs a stack of M_{l+1} feature maps given by

$$(w_l^h *_d f_{l-1})(i) = \sum_{j=\infty}^{\infty} \sum_{m=1}^{M^l} w_l^h(j,m) f_{l-1}(i - dj, m), \tag{2}$$

where d is the dilation factor and M_l the number of channels. We use an architecture similar to [13] and [11] with S stacks of dilated convolutions, where one stack consists of L layers of dilated convolutions, $l = 0, ..., L-1$ and with the dilations increasing with a factor of two: $d \in [2^0, 2^1, ..., 2^{L-1}]$. The filters w are chosen to be of size $1 \times k := 1 \times 2$. All elements in the feature map share the same weights, allowing features to be detected in a time-invariant manner.

To make sure that the receptive field of the network, i.e. the set of elements in the input that modifies the value of the output feature map, when predicting $x(t+1)$ contains only $x(0), ..., x(t)$, we use causal convolutions. In time series this is equivalent to padding the input with a vector of zeros of the size of the receptive field, so that the input is $[0, ...0, x(0), ..., x(N-1)]$. At training time the prediction of $x(1), ..., x(N)$ is computed by convolving the input $x(0), ..., x(N-1)$ with a kernel. At testing time the predictions are made sequentially so that each prediction is fed back into the network at the next time step.

The network weights, the filters w_l^h, are trained with backpropagation to minimize the mean absolute error (MAE); to avoid overfitting we use L2 regularization with regularization term γ, so that the cost function is given by

$$E(w) = \frac{1}{N} \sum_{t=0}^{N-1} |\hat{x}(t+1) - x(t+1)| + \frac{\gamma}{2} \sum_{l=0}^{L} \sum_{h=1}^{M_{l+1}} (w_l^h)^2, \tag{3}$$

where $\hat{x}(t+1)$ denotes the forecast of $x(t+1)$ using $x(0), ..., x(t)$.

Activation functions. In each layer we use a rectified linear unit (ReLU) as the activation function defined as $ReLU(x) := \max(x, 0)$, so that the output from layer l is:

$$f_l = \left[ReLU(w_l^1 *_d f_{l-1}), ..., ReLU(w_l^{M_{l+1}} *_d f_{l-1}) \right], \tag{4}$$

where $*_d$ denotes as usual the convolution with dilation d and $f_l \in \mathbb{R}^{1 \times N \times M_{l+1}}$ denotes the output of the convolution with filters w_l^h, $h = 1, ..., M_{l+1}$ in layer l. As opposed to the gated activation function used in [11] for audio generation, here we propose to use the ReLU as it was found to be most efficient when applied to the forecasting of the non-stationary, noisy time series. The final layer, $l = L$, has a linear activation function, which outputs the forecasted value of the time series $\hat{x} = \{\hat{x}(0), ..., \hat{x}(N)\}$.

Residual learning. When adding more layers to the network, standard backpropagation tends to become unable to find the optimal weights. This is known as the degradation problem [3]. The proposed way around this problem is to use residual connections [3] which force the network to approximate $\mathcal{H}(x) - x$, instead of $\mathcal{H}(x)$, the desired mapping, so that the identity mapping can be learned by driving all weights to zero. Similar to [11], in our network, we add a residual connection after each dilated convolution. In the case of $M_l > 1$ the residual connection follows a 1×1 convolution to reduce the number of filters back to one to match the size of the input vector. This allows us to stack multiple layers, while retaining the ability of the network to correctly map dependencies learned in the initial layers.

Batch normalization. In [8] it is noted that networks tend to converge faster when the mean of the input signal in each layer is close to zero and the variance is close to one. The problem of a changing distribution of the layer activations while training is known as the internal covariate shift, and can be reduced by fixing the distribution of the input to each layer, $f_l \in \mathbb{R}^{1\times N\times M_l}$ using normalization [5]. To make sure that each layer is able to represent the same as prior to the normalization, we scale and shift the normalized values in each layer using a pair of trainable parameters ρ and β, see [5]. The normalization layer is added prior to the dilated convolutions and is computed over the whole time series. While batch normalization makes the performance of the network less dependent on the initialization, we found the network to be even more robust when using a Glorot initialization [2] for the weights to the rectified linear and linear units:

$$w_l^h \sim \mathcal{U}\left(-\frac{\sqrt{6}}{\sqrt{2n}}, \frac{\sqrt{6}}{\sqrt{2n}}\right), \tag{5}$$

with $n = M_{l+1} \times 1 \times k$, the number of filters in layer l times the filter size $1 \times k$.

3 Conditioning

When forecasting one time series conditional on others, we model a conditional distribution given by

$$p(x|y) = \prod_{t=0}^{N-1} p\left(x(t)|x(0), ..., x(t-1), y(0), ..., y(t-1)\right). \tag{6}$$

The conditioning on the time series $y(t)$ is done by computing the activation function of the convolution with filters w^h and v^h in first layer as

$$ReLU(w_0^h *_d x + v_0^h *_d y). \tag{7}$$

In [11] the authors propose to take v_0^h as a 1×1 filter. Given the short input window, this type of conditioning is not always able to capture all dependencies between the time series. Therefore, we use a $1 \times k$ convolution, increasing the probability of the correct dependencies being learned with fewer layers. Note that this type of conditioning is similar to convolving the two-dimensional input with a two-dimensional filter.

Instead of the residual connection in the first layer, we add skip connections parametrized by 1×1 convolutions from both the input as well as the condition to

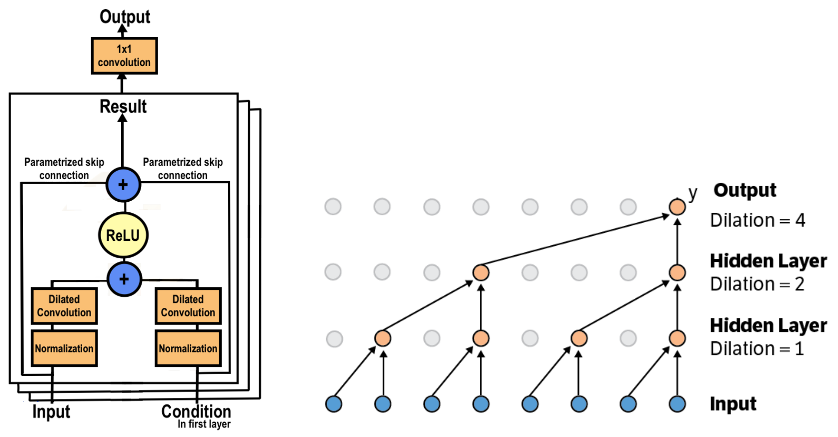

Fig. 1. The total network (L) and the dilated convolution structure (R)

the result of the dilated convolution. The parametrization of the skip connections makes sure that our model is able to correctly extract the necessary relations between the forecast and both the input and condition. The conditioning can easily be extended to a multivariate $N_C \times N$ time series by using an $N_C \times k$ filter. The network structure is shown in Figure 1.

4 Results

Here, we present the results of the CNN applied to forecasting financial time series. We compare the performance of the WaveNet architecture to a fully convolutional network (FCN) ([12], [9]) and a classic autoregressive model. The FCN was shown to be a strong state-of-the-art baseline for time series classification, performing better than a multilayer perceptron and a residual network [12] and can be applied to multiple time series in a way similar to WaveNet. It consists of three convolution blocks with the number of filters $M_l \in [128, 256, 128]$ and filter sizes $k \in [3, 5, 8]$. We replace the global average pooling layer, which generates one feature map for each corresponding category of a classification task, with a 1×1 convolution, since we require only one output class, namely the forecast of the time series and use a sigmoid activation function. The model structure was optimized on synthetic data, and then trained on 10 different initializations using different random seeds by minimizing the MAE on the training set with 10,000 iterations. This was the number of iterations that allowed the models to achieve a satisfactory convergence on all the time series for all initializations. The forecast for each initialization is computed over the test set and the average MAE is reported. We use a learning rate of $\lambda = 0.001$ and regularization $\gamma = 1$.

Financial Data The commodities data used consists of the daily prices of front monthly and seasonal UK gas, monthly German power, a CO2 December contract (for the period Jan. '09 to May '16) and monthly Brent oil (for Jan. '09 to May '15), taken from Reuters and Trayport and the data for the Nikkei and S&P500 indices (Jan. '12 to Jan '17) taken from Yahoo! Finance. The training and test split is done so that the last year of data falls into the test set (i.e. roughly 250 data points). In the WaveNet we use $L = 5$, $S = 1$ and $M_l = 1$.

Other Time Series The data is taken from the Time Series data library [4] and contains time series such as the temperature, rainfall and daily births. In the WaveNet we use $L = 9$, $S = 1$ and $M_l = 1$. Depending on the length of the time series we either take the length of the test set to be the final 200 points if the total length is bigger than 400, or 100 otherwise.

Unconditional forecasting Table 1 presents the results of the unconditional Augmented WaveNet applied to forecasting several time series. We see that our model is able to extract relations from the data efficiently and performs consistently better than the FCN independent of the initialization of the weights. We note that the FCN due to the large number of filters tends to overfit certain time series resulting in a low training error but a poor generalisation, as can also be seen from the high standard error in the MAE. Overall, the neural network in the form of a WaveNet, with its fairly simple architecture, achieves a competitive performance in terms of the MAE and tends to outperform the FCN with an improvement similar to the one observed in image recognition.

Conditional Forecasting The results for the conditional FCN and WaveNet applied to the financial time series are presented in Table 2. The conditioning allows the WaveNet to outperform the FCN, which often has trouble learning the correct dependencies. As we demonstrate, the conditional Augmented WaveNet leads to significant improvements in prediction accuracy as compared to the unconditional forecasts and thus works well for learning long-term dependencies in and between time series. In Table 2 we also estimated the time delay between two sequences using the arguments of the maximum (or minimum, in case of a negative cross-correlation) of the cross-correlation. Note that also in the neighborhood of these maxima/minima the cross-correlation is still significant. These results correspond to the results observed from the conditioning, e.g. the S&P500 is shown to precede the Nikkei index, which also makes sense from an economic standpoint, and therefore conditioning the Nikkei on the S&P500 gives a more significant improvement. The conditional CNN can therefore also give an idea of the relationship between the time delays in time series.

A final comparison We end this section with a comparison of the unconditional and conditional Augmented WaveNet model (uWN and cWN respectively) with both a naive forecast, $\hat{x}(t) = x(t-1)$, and the vector autoregressive model (VAR) that is commonly used in econometrics for multivariate forecasting for forecasting the daily return of the S&P500. The performance is evaluated using the mean absolute scaled error (MASE), which scales the measured error using the mean absolute error of a naive forecast and the hit rate (HITS), see also [10]. The forecast is conditioned on the volatility index (VIX) and the CBOE 10 year interest rate. We use data spanning the period Jan. '06 to Dec. '16. The model is trained on every 12 months, forecasted for one month, then shifted by one month and the process is repeated. The results are then averaged over all

Table 1. MAE for the forecast on the test set of the unconditional WaveNet model compared to the FCN and ARMA model.

Time series	WaveNet	FCN	ARMA(2,2)
Mean daily temperature Dallas	**0.62 ± 0.11**	0.88 ± 0.14	0.76
Minimum temperature Melbourne	**0.61 ± 0.13**	1.07 ± 0.64	0.64
Number of daily births Quebec	**0.58 ± 0.17**	**0.57 ± 0.15**	0.85
Daily total female births California	**0.81 ± 0.10**	1.09 ± 0.25	0.82
Mean daily rainfall Melbourne	**0.51 ± 0.05**	0.53 ± 0.17	0.59
Gas 1S	**0.87 ± 0.16**	1.08 ± 0.09	1.12
Power 1M	0.43 ± 0.05	**0.41 ± 0.03**	0.69
CO2 Dec.	0.32 ± 0.05	**0.27 ± 0.02**	0.31
Oil 1M	0.45 ± 0.01	**0.43 ± 0.00**	0.44
Nikkei Index	**0.23 ± 0.05**	0.43 ± 0.09	0.44
S&P500	**0.20 ± 0.09**	0.25 ± 0.05	0.46

Table 2. MAE for the forecast on the test set of the conditional WaveNet model compared to the FCN and the estimated time delays between the time series. The length of the commodities time series is $N = 1926$ and the indices have length $N = 1242$.

Time series	WaveNet	FCN	Time Delay
Gas 1S on Oil 1M	**0.74 ± 0.10**	0.89 ± 0.19	0
Oil 1M on Gas 1S	**0.27 ± 0.09**	0.47 ± 0.02	0
Oil 1M on Gas 1M	**0.44 ± 0.04**	0.75 ± 0.31	1670
Power 1M on CO2 Dec.	**0.37 ± 0.08**	0.61 ± 0.11	195
Gas 1S on Power 1M	**0.82 ± 0.14**	0.94 ± 0.25	-479
Gas 1S on Power 1M and Oil 1M	**0.78 ± 0.19**	0.96 ± 0.35	0
Power 1M on Gas 1S	**0.44 ± 0.13**	0.99 ± 0.29	479
Nikkei on S&P500	**0.13 ± 0.03**	0.20 ± 0.07	8
S&P500 on Nikkei	**0.16 ± 0.02**	0.27 ± 0.03	-8

Table 3. MASE and HITS for forecasting the one-day ahead return of the S&P500 with a naive forecast, the unconditional and conditional WaveNet (with $\gamma = 0.1$) and a VAR model.

	Naive		uWN		cWN		VAR	
Period	MASE	HITS	MASE	HITS	MASE	HITS	MASE	HITS
Jan. '06 - Dec. '08	1	0.32	**0.55**	0.57	0.60	**0.64**	0.65	0.51
Jan. '09 - Dec. '11	1	0.35	**0.60**	0.51	0.66	**0.62**	0.69	0.53
Jan. '12 - Dec. '14	1	0.32	**0.61**	**0.55**	0.69	**0.55**	0.69	0.53
Jan. '14 - Dec. '12	1	0.37	**0.60**	0.53	0.69	**0.56**	0.72	0.52

of these monthly testing periods. From Table 3 we see that the unconditional WaveNet performs best in terms of MASE. The conditional WaveNet exploits the correlation in between the three time series resulting in a high hit rate, but a slightly worse MASE compared to the unconditional one due to it being fitted on multiple noisy series. After the crisis the dependencies between the S&P500 and the interest rate and volatility index seem to have weakened (due to e.g. the lower interest rate or higher spreads) as the improvement of the conditional WaveNet over the unconditional WaveNet is smaller, so WaveNet can be used to recognize these switches in the underlying financial regimes. While not a one-on-one comparison, WaveNet scores significantly higher hit rates for the one-day ahead returns of the S&P500 compared to traditional neural networks [7]. We conclude that even though time series forecasting remains a complex task and finding one model that fits all is hard, we have shown that the WaveNet is a simple and efficient model that can act as a strong baseline for forecasting.

References

1. K. Chakraborty, K. Mehrotra, C. K. Mohan, and S. Ranka, *Forecasting the Behavior of Multivariate Time Series using Neural Networks*, Neural networks, 5 (1992), pp. 961–970.
2. X. Glorot and Y. Bengio, *Understanding the Difficulty of Training Deep Feed-forward Neural Networks*, Proceedings of the 13th International Conference on Artificial Intelligence and Statistics, (2010).
3. K. He, X. Zhang, S. Ren, and J. Sun, *Deep Residual Learning for Image Recognition*, Available at ArXiv, (2015).
4. R. Hyndman, *Time Series Data Library*, http://data.is/TSDLdemo, Accessed on 24/01/17.
5. S. Ioffe and C. Szegedy, *Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift*, coRR, (2015).
6. A. Krizhevsky, I. Sutskever, and G. E. Hinton, *ImageNet Classification with Deep Convolutional Neural Networks*, Advances in Neural Information Processing Systems 25, (2012), pp. 1097–1105.
7. M. Kumar et al., *Nonlinear Prediction of the Standard & Poors 500 and the Hang Seng Index Under a Dynamic Increasing Sample*, Asian Academy of Management Journal of Accounting & Finance, 5 (2009), pp. 101–118.
8. Y. LeCun, L. Bottou, G. B. Orr, and K. R. Müller, *Efficient BackProp*, Springer Berlin Heidelberg, 1998, pp. 9–50.
9. R. Mittelman, *Time-series modeling with undecimated fully convolutional neural networks*, arXiv preprint arXiv:1508.00317, (2015).
10. M. Peña, A. Arratia, and L. A. Belanche, *Multivariate dynamic kernels for financial time series forecasting*, in International Conference on Artificial Neural Networks, Springer, 2016, pp. 336–344.
11. A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, *WaveNet: A Generative Model for Raw Audio*, ArXiv e-prints, (2016).
12. Z. Wang, W. Yan, and T. Oates, *Time Series Classification from Scratch with Deep Neural Networks: A Strong Baseline*, CoRR, abs/1611.06455 (2016).
13. F. Yu and V. Koltun, *Multi-Scale Context Aggregation by Dilated Convolutions*, ArXiv e-prints, (2015).