Forex Forecasting

Combining forecasting methods with neural networks

Efstathios Kotsis [1] Michael Darmanis [1] Vasilios Venieris [1]

September 27, 2023

[1]National and Kapodistrian University of Athens

Time Series & Time Series Forecasting

- **Time series** are observations taken sequentially in time.
- **Time series forecasting** predicts future values based on past (historical data).

Date	Observation
2018-06-09	64
2018-06-10	66
2018-06-11	69
2018-06-12	72
2018-06-13	?
2018-06-14	?

Why Deep Learning Models?

- Deep learning models have shown to perform well in many forecasting scenarios
  - 2014 Global Energy Forecasting Competition (link)
  - 2016 CIF International Time Series Competition (link)
  - 2017 Web Traffic Time Series Forecasting (link)
  - 2018 Corporacion Favorita Grocery Sales Forecasting (link)
  - 2018 M4-Competition (link)
- Non-parametric
- Flexible and expressive
- Easily inject exogenous features into the model
- Learn from large time series datasets

Scenario: Forex Forecasting

- Forex is the global marketplace for trading national currencies against one another, operating 24/5.
- Here, we'll use the European Central Bank's (ECBs) dataset for various currencies against the euro

Table 1: ECB forex dataset

Date	USD	JPY	BGN	...	THB	ZAR
2023-09-15	1.0658	157.50	1.9558	...	38.145	20.2968
2023-09-14	1.0730	158.13	1.9558	...	38.387	20.3109
...	...	...	...	...	...	...
1999-01-04	1.1789	133.73	NaN	...	NaN	6.9358

Dataset considerations

- **Missing Values:** Require careful handling to avoid bias.
- **Data Frequency:** Daily data, resampling needed for other frequencies.
- **Currency Pairs:** Challenge in selecting relevant pairs.
- **Normalisation:** Essential for data comparability.
- **Institutional Bias:** ECB interventions can cause rate discrepancies.

All series were adjusted to a fixed length, ensuring consistency for effective vectorisation.

Approach and Assessment i

- Used forecasting methods to supplement a Convolutional Neural Network (S-CNN).
  - **Why?** To address the non-linear nature of Forex data.
- Adopted a traditional forecasting method as the foundation, and enhanced it with a Recurrent Neural Network (ES-RNN).
  - **Why?** To address the linear trend assumption of the forecasting method[1].

Benchmarking

Used a VAR (Vector Autoregressive) model for performance comparison and benchmarking.

Approach and Assessment ii

sMAPE (symmetric Mean Absolute Percentage Error) was used to assess the models.

$$\text{sMAPE} = \frac{200\%}{n} \sum_{t=1}^{n} \frac{|F_t - A_t|}{|F_t| + |A_t|},$$

where

- F_t is the forecasted value at time t,
- A_t is the actual value at time t,
- n is the total number of observations.

Approach and Assessment iii

Prediction rules as dictated by the M4 Competition[2].

Table 2: Forecasting Horizons in M4 Competition[2]

Frequency	Forecasting Horizon
Yearly	6
Quarterly	8
Monthly	18
Weekly	13
Daily	14
Hourly	48

Smoothed Convolutional Neural Network (S-CNN)

S-CNN: Origin and basic idea

- Implementation based on the paper “Time-series analysis with smoothed Convolutional Neural Network”[4] (no available code) .
- Model is univariate, multi-step.
- Simple exponential smoothing (remove outliers, average) + CNN.

S-CNN: Simple Exponential Smoothing i

$$s_t = \alpha x_t + (1 - \alpha) s_{t-1} = s_{t-1} + \alpha (x_t - s_{t-1})$$

- For α values close to 1 we get very little influence of the previous smoothed value.
- For α close to 1 we also see that the previous unsmoothed series observation influences the result more than the smoothed values.
- For the calculation of α, since there is no golden rule, we adopted the α value from the aforementioned publication.

S-CNN: Simple Exponential Smoothing ii

- Authors argued that α should be dependent on the dataset and must be between 0 and 1. Average value of the series should be less than the difference between max and min.

So we end up with the following formula

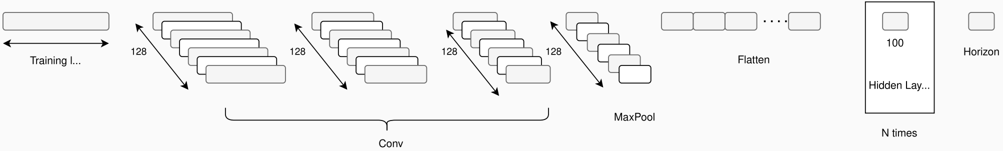
$$\alpha_{\text{opt}} = \frac{(X_{\text{max}} - X_{\text{min}}) - \frac{1}{n}\sum_{t=1}^{n} X_t}{X_{\text{max}} - X_{\text{min}}},$$

and the simple smoothing becomes

$$s_t = s_{t-1} + \frac{(x_{\text{max}} - x_{\text{min}}) - \frac{1}{n}\sum_{t=1}^{n} x_t}{x_{\text{max}} - x_{\text{min}}} \left(x_t - s_{t-1}\right).$$

S-CNN: Architecture

Basic architecture from Wibawa et al.[4] and experimented with different number of hidden layers.

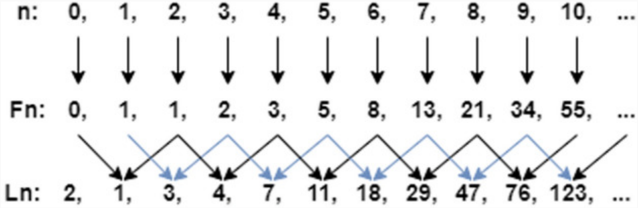


Kernel size: 1x2, Stride: 1x1

S-CNN: Training i

How to pick the number of hidden layers

We used the Lucas numbers suggested. They derive from the Fibonacci Sequence if instead of adding every 2 successive observations we skip the intermediate and add the K with the K+2 together.



S-CNN: Training ii

Model yielded the lower MSE, for the most stable currencies, for 76 layers of training.

Table 3: Training time for different numbers of hidden layers

Number of Hidden Layers	Training Time (hh:mm:ss)
3	0:27:43.15
11	0:34:27.77
47	1:05:21.78
76	1:28:47.70

S-CNN: Training iii

Training steps

- Split the dataset into series of different frequencies.
- For each time series we normalise + smooth.
- Feed the output into the convolutional model and train the model to predict a horizon of K steps for this frequency, where K is a model's designers choice.

S-CNN: Training iv

Training history

In order to balance training with only recent data, or feeding with irrelevant outdated history, we train with a list of possible training lengths.

- Daily: [14, 20, 240]
- Weekly: [52, 13, 26]
- Monthly: [24, 18]
- Quarterly: [12, 8]
- Yearly: [6]

S-CNN: Training v

Train/validation/test split

- We focused on data after 2010 since in 2008 – 2010 there was the European debt crisis. A plot from the data can show that the economy and the currency fell rapidly in most countries in the European continent and in many other countries.
- For the frequencies except Yearly, we trained with splits of 80%-20%, 70%-30%, and approximately 60%-40% for quarterly (training/validation).

S-CNN: Training vi

- We split the dataset accordingly and fed the created datasets to the generators.
- The models were trained for 200 epochs, using the MSE loss function and the Adam optimiser.

S-CNN: Results i

For stable currencies, longer training tends to yield more accurate predictions.

Table 4: sMAPE for Daily with 76 Hidden Layers (Biggest Lucas Number Tested)

	sMAPE	
	Training length 20 days	Training length 240 days
CAD	104.24	26.63
AUD	226.39	79.63
DKK	1.357	0.937

S-CNN: Results ii

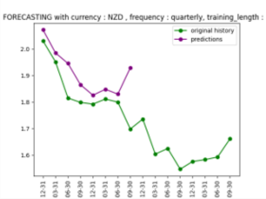
Unpredictable currencies benefit from short-term history and possibly higher α for forecasting. More hidden layers may overfit. Fewer hidden layers excel for daily predictions of unstable currencies.

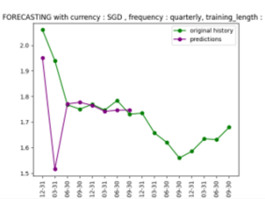
Table 5: sMAPE for 76 Hidden Layers

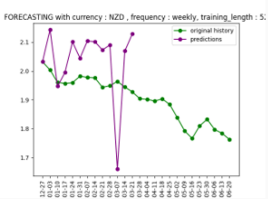
	sMAPE			
	T.L. 20 days	T.L. 240 days	T.L. 8 quarters	T.L. 12 quarters
USD	59.67	138.76	72.69	82.30

S-CNN: Results iii

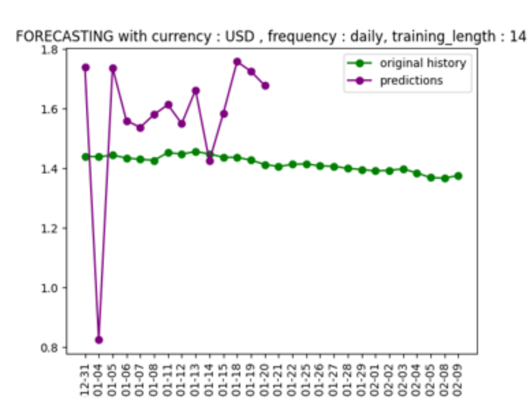
Quarterly and monthly predictions outperform daily ones for stable currencies. Daily and weekly forecasts show many fluctuations and are less accurate.

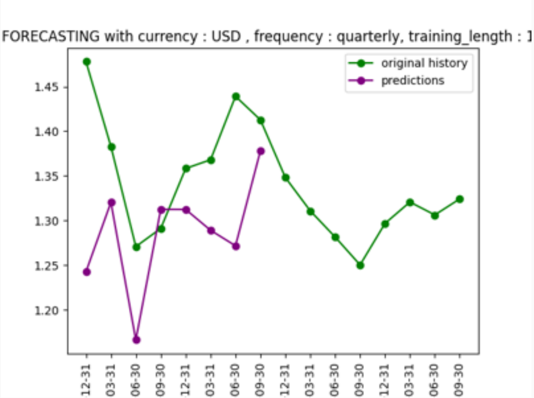






Model struggles with unpredictable currencies like *USD* and *GBP*.





Exponentially Smoothed Recurrent Neural Network

ES-RNN: Origin and basic idea

- Implementation based on Slawek Smyl[3], 2018 *https://eng.uber.com/m4-forecasting-competition/*, Winner of M4 competition[2] (code available in C++)
- Use the Holt-Winters equations[5], and replace the trend with an RNN.

ES-RNN: Holt-Winters model

Break down the problem of forecasting into three distinct equations:

- level, $l_t(\alpha)$
- trend, $b_t(\beta)$
- seasonality, $s_t(\gamma)$

Optimisation problem with respect to α, β, γ.

Issue: Formulation works for a linear trend

So plug in RNN for the trend component $b_t(\beta)$.

ES-RNN: Holt-Winters model (multiplicative)

$$l_t = \alpha(\frac{y_t}{s_{t-m}}) + (1-\alpha)l_{t-1}b_{t-1} \tag{1}$$

$$b_t = \beta(\frac{l_t}{l_{t-1}}) + (1-\beta)b_{t-1} \tag{2}$$

$$s_t = \gamma\frac{y_t}{l_{t-1}b_{t-1}} + (1-\gamma)s_{t-m} \tag{3}$$

$$\hat{y}_{t+h} = l_t b_t^h s_{t-m+h_m^+} \tag{4}$$

Equation 2 is replaced by an RNN from the model as follows:

$$\hat{y}_{t+1\dots t+h} = RNN(X_t) * l_t * s_{t+1\dots t+h} \tag{5}$$

$$x_i = \frac{y_i}{l_t s_i} \tag{6}$$

ES-RNN: Holt-Winters model (additive)

$$l_t = \alpha(y_t - s_{t-m}) + (1-\alpha)(l_{t-1} + b_{t-1}) \tag{7}$$

$$b_t = \beta(l_t - l_{t-1}) + (1-\beta)b_{t-1} \tag{8}$$

$$s_t = \gamma(y_t - l_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m} \tag{9}$$

$$\hat{y}_{t+h} = l_t + h \cdot b_t + s_{t-m+h \pmod m} \tag{10}$$

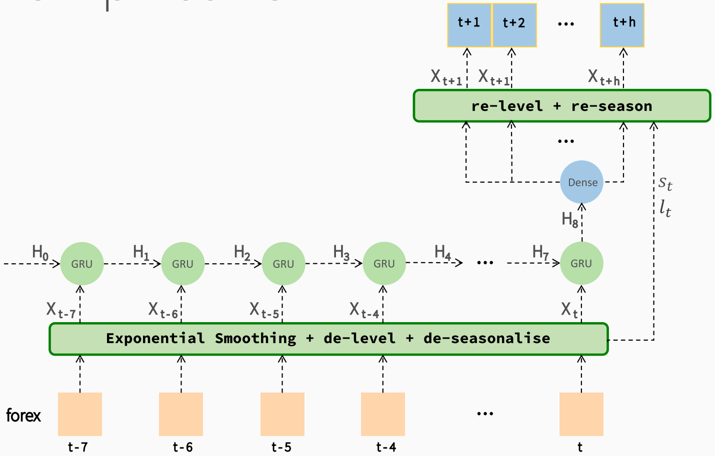
Equation 8 is replaced by an RNN in the model as follows:

$$\hat{y}_{t+1 \ldots t+h} = RNN(X_t) + l_t + s_{t+1 \ldots t+h} \tag{11}$$

$$x_i = y_i - l_t - s_i \tag{12}$$

ES-RNN: Architecture

Simplified ES-RNN



$$\hat{y}_{t+1|t} = l_t \cdot s_{t+1} \cdot RNN(x_t)$$

$$RNN(x_t)$$

Smoothing: $l_t = \alpha \frac{y_t}{s_t} + (1-\alpha) l_{t-1}$

Seasonal: $s_{t+m} = \gamma \frac{y_t}{l_t} + (1-\gamma) s_t$

De-seasonalised and de-leveled time series: $x_t = \frac{y_t}{l_t s_t}$

ES-RNN: Training

Used a sliding window. Input and output windows were of constant size. Output size matched the prediction horizon, while input size determined heuristically.

Loss Function

Considered the L1 difference as similar enough to sMAPE; both are mean absolute differences between forecasted and actual values.

Experimented with

- Both multiplicative and additive Holt-Winters formulations.
- A simple GRU with various hidden state features (8, 16, 32), i.e. input-window size.

ES-RNN: Results i

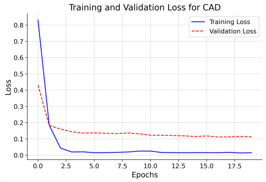
Multiplicative Holt-Winters is better suited; best results were achieved using an input window of 8.

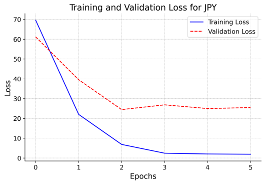
Table 6: Averaged sMAPE(%) values for daily and Weekly Frequencies

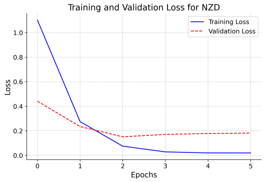
	Daily	Weekly
Additive Holt-Winters		
Input=8	23.45	17.89
Input=16	36.72	21.63
Input=32	28.83	19.42
Multiplicative Holt-Winters		
Input=8	13.72	7.96
Input=16	18.25	11.37
Input=32	16.92	9.84

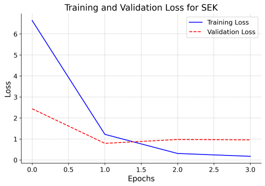
ES-RNN: Results ii

L1 as the training loss function created a positive bias.









Comparison with Benchmark

Average performance estimators

Table 7: Averaged performance estimators for daily and weekly frequencies

	Daily		Weekly	
	sMAPE(%)	*MSE*	*sMAPE(%)*	*MSE*
S-CNN	59.70	0.006	81.45	0.010
ES-RNN	13.72	560.580	7.96	323.021
V-AR	0.83	0.001	2.24	0.000

Conclusions

Conclusion and Key Findings i

- Explored the synergy between neural networks and traditional statistical models for forex forecasting.
- ES-RNN and S-CNN were both outperformed by Vector Autoregression (V-AR).
- Identified a positive bias issue in the ES-RNN due to an ineffective training loss function.
- S-CNN struggled with non-stationary and insufficient data.
- Emphasised the importance of ample data and the effectiveness of simpler, classic statistical methods.

Future Steps

- Explore alternative loss function, and input window, for ES-RNN to mitigate bias in predictions.
- Train the ES-RNN with all the series to exploit shared parameters and learn common local trends among the series.
- Address data non-stationarity and volume issues to enhance the performance of complex models.

References i

M. A. Junior, P. Appiahene, O. Appiah, et al.
Forex market forecasting using machine learning: Systematic literature review and meta-analysis.
J Big Data, 10:9, 2023.

S. Makridakis, E. Spiliotis, and V. Assimakopoulos.
The m4 competition: 100,000 time series and 61 forecasting methods.
Int. J. Forecasting, 36(1):54–74, 2020.

References ii

S. Smyl, J. Ranganathan, and A. Pasqua.
M4 forecasting competition: Introducing a new hybrid es-rnn model, 2023.
Accessed 15 Sep 2023.

A. Wibawa, A. Utama, H. Elmunsyah, et al.
Time-series analysis with smoothed convolutional neural network.
J Big Data, 9:44, 2022.

P. R. Winters.
Forecasting sales by exponentially weighted moving averages.
Management science, 6(3):324–342, 1960.