Evaluating uncertainty scores for deep regression networks in financial short time series forecasting

Mauricio Ciprián and **Leonardo Baldassini** and **Luis Peinado** and **Teresa Correas** and **Roberto Maestre** and **Jose A. Rodríguez-Serrano**
BBVA Data & Analytics, Spain

Oriol Pujol and **Jordi Vitrià**
Universitat de Barcelona, Spain

Deep regression networks can offer a valid alternative to classical time series forecasting methods, which will usually fail on a short-and-coarse time scale. However, regression networks output single-point estimates and not distributions over predictions or uncertainty scores. This is key information in a real-world financial application where we should only be interested in communicating those forecasts deemed reliable, and ignore the rest. We therefore propose and benchmark various methods to achieve this.

Problem setting Monthly aggregation is a natural choice for the analysis of personal financial records, as most financial transactions either only happen monthly or would introduce too much noise if considered on a finer scale. Furthermore, like in the case we faced, business constraints may limit the series to a year worth of data.

We are then left with a set of short-and-coarse time series $\mathbf{x} = (x_1, \ldots, x_T)$, one per each person and transaction category (e.g. “salary” or “utilities”), where $T = 12$ and entry x_t is the aggregate transaction value for month t. For example, an account holder with an electricity bill issued bimonthly may produce the series $(0, 100, 0, 70, 0, 50, 0, 100, 0, 70, 0, 80)$.

The problem of predicting $\hat{y} = \hat{x}_{T+1}$, the most likely value for the next month of each series, presents several challenges. First, the time series are very short, which makes classical time series methods impractical, as noted in (Hyndman et al., 2007). Second, consumers' spending behaviour is erratic, discontinuous and generally noisy. Third, in real applications, we need the system to scale to millions of time series. We believe such a scenario has been underexplored in time series forecasting.

LSTM regression Since extrapolation on only 12 data points per series is hardly achievable with classical methods, we chose to adopt a machine learning methodology: Learning a function $\hat{y} = f(\mathbf{x})$ from training examples $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$. To this end, we benchmarked several methods and found that the use of long short-term memory (LSTM) networks outperformed the rest in 4 out of 5 metrics, as shown in Table 1. Details on network architecture and the benchmarking are available in the appendix.

	Mean	Last	Zero	1-NN	100-NN	Act 1-NN	Act 100-NN	Random Forest	LSTM
MARE	0.570	0.707	0.952	1.095	0.806	0.935	0.823	0.844	**0.521**
MASE	0.748	0.867	2.247	1.057	0.748	0.925	0.785	0.783	**0.586**
RMSE	608.2	648.1	742.9	13337.1	871.5	6764.9	1069.9	828.4	**552.4**
%u.10	18.0	33.6	2.0	**36.4**	29.6	35.7	28.7	29.1	29.9
%u.20	29.2	40.2	5.2	46.1	42.1	45.2	41.1	40.2	**47.2**

Table 1: Benchmarking of prediction methods for short time series

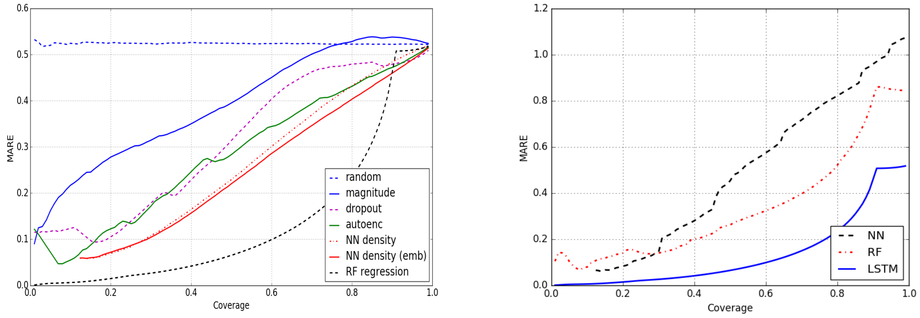

Figure 1: Benchmarking uncertainty scores for LSTM(left) and LSTM vs others (right).

Uncertainty scores In our real-world application, we plan to communicate forecasts as notifications of upcoming expenses. Here, producing forecasts for *all* the series risks overwhelming users, and producing meaningless estimates for unforeseeable expenses. Instead, we seek to filter out high-uncertainty predictions through the use of some *uncertainty score*. However, regression networks produce single-point estimates without uncertainty bounds. We identify and explore a few quantities that could be used as uncertainty proxies:

- *Distance to nth neighbor*: Euclidean Distance between the test series $\mathbf{x}$ and the nth nearest neighbor in the training series $\{\mathbf{x}^{(i)}\}$, i.e. a proxy of the density around $\mathbf{x}$. The intuition is to trust the prediction if the density is high, as the network will have learned similar patterns.
- *Distance to nth neighbor in the embedding space*: Likewise, but taking distances between $\phi(\mathbf{x})$ and $\{\phi(\mathbf{x}^{(i)})\}$, where $\phi(\mathbf{x})$ denotes the embedding of $\mathbf{x}$ learned by the network – i.e. the output of the last fully-connected layer of the network evaluated on $\mathbf{x}$.
- *Marginalized denoising autoencoder (MDA)* (Chen et al., 2012): The reconstruction error of a MDA (of 1 hidden layer) taking (normalized) $\mathbf{x}$ as input. Small reconstruction errors may indicate frequently seen patterns, which should be more reliable for prediction.[1]
- *Dropout simulation*: The size of the 95% confidence interval of $\hat{y}$, estimated as proposed by (Gal and Ghahramani, 2016). They argue that the application of dropout noise induces a distribution over $\hat{y}$, yielding a principled model of uncertainty in regression networks.
- *Random forest regression:* Supervised estimate of the prediction error, by training a random forest regressor on $(\mathbf{x}^{(i)}, \mathrm{MARE}(y^{(i)}, \hat{y}^{(i)}))$ pairs. The features $(\mathbf{x}^{(i)})$ were preprocessed using z-score normalization and PCA.

The MARE vs. yield (fraction of non-rejected samples) tradeoff for all these scores is depicted in Figure 1 [2] . All the rejection strategies perform significantly better than a random confidence and the simplistic magnitude score, with significant error reductions at acceptable yield levels of 20-50%. The best error-reject characteristic is obtained by the supervised estimate using the random forest regression, with a fivefold reduction of the error on as much as 60% of the sample. Among the non-supervised strategies, the best uncertainty score is obtained by taking the distance to the 1st neighbor in the space of learned embeddings. In Fig. 1 (right), we compare against the curves of 2 other prediction methods from Table 1 (RF and 1-NN) and confirm that LSTM is the best method.

Conclusions After verifying that LSTM networks are a practical solution to short financial time series, we benchmark several uncertainty scores for them. Our best score is the supervised prediction of the error using a random forest learned on pairs of series and observed errors. Among non-supervised uncertainty scores, the density in the space of network embeddings is an informative proxy of the uncertainty. We were not aware of such a result in the literature.

[1] We also experimented with autoencoders taking $\phi(X)$ as input but this led to worse results.

[2] We have also used the magnitude $|\hat{y}|$ as a control baseline (i.e. suspicious high-magnitude predictions).

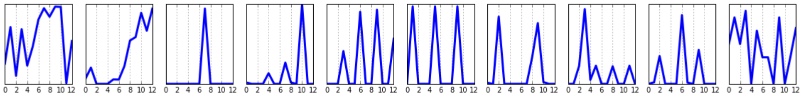

Figure 2: Examples of time series in our dataset

Appendix A.

Examples of time series Fig. 2 depicts some samples of **x** from our dataset. The great variability of patterns increases the difficulty of the problem and corroborates the need for a confidence score on predictions.

LSTM architecture LSTM networks are recurrent neural networks that can be trained using backpropagation through time and overcome the vanishing gradient problem, (Hochreiter and Schmidhuber, 1997; Gers et al., 2000). Furthermore, they can be vertically stacked to allow for greater model complexity. For this reasons they lend themselves well to sequence learning problems (Lipton et al., 2015), which is why we chose them as a viable candidate at the time of predicting the next value of a short time series, a task in which they outperformed alternative methods.

The network we used has a visible layer with 1 input, 2 stacked 128-neuron LSTM blocks (with dropout regularization), 2 fully connected layers and an output layer that makes a single value prediction.

Prediction benchmarking On a subset of 800K short time series (50% train/test split with 10% cross-validation during training), we benchmarked the predictive performance of LSTM networks against 3 control baselines (predicting using the mean, the last value or 0); k-NN with 1 and 100 neighbours using Euclidean distance, both between series and the feature vectors obtained from the network activation layer; and a random forest regressor. We evaluate against error and success metrics. The former are: Mean Absolute Relative Error (MARE), Root Mean Squared Error (RMSE), the Mean Absolute Scaled Error (MASE) of (Hyndman and Koehler, 2006); the latter are the percentage of series under 10% MASE and 20% MASE, indicated as %u10 and %u20, respectively.

References

Minmin Chen, Zhixiang Xu, Kilian Weinberger, and Fei Sha. Marginalized denoising autoencoders for domain adaptation. In *ICML*, 2012.

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *ICML*, 2016.

Felix A. Gers, J urgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with LSTM. *Neural Comput.*, 12(10), 2000.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, November 1997.

Rob J Hyndman and Anne B Koehler. Another look at measures of forecast accuracy. *International Journal of Forecasting*, pages 679–688, 2006.

Rob J Hyndman, Andrey V Kostenko, et al. Minimum sample size requirements for seasonal forecasting models. *Foresight*, 6(Spring):12–15, 2007.

Zachary C Lipton, John Berkowitz, and Charles Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.