ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

Zero-shot and few-shot time series forecasting with ordinal regression recurrent neural networks

Bernardo Pérez Orozco[1,2] and Stephen J. Roberts[1,2] *

1- Information Engineering, University of Oxford

2 - Mind Foundry Ltd.

Abstract. Recurrent neural networks (RNNs) are state-of-the-art in several sequential learning tasks, but they often require considerable amounts of data to generalise well. For many time series forecasting (TSF) tasks, only a few dozens of observations may be available at training time, which restricts use of this class of models. We propose a novel RNN-based model that directly addresses this problem by learning a shared feature embedding over the space of many quantised time series. We show how this enables our RNN framework to accurately and reliably forecast unseen time series, even when there is little to no training data available.

1 INTRODUCTION

Deep and recurrent neural networks (RNNs) have achieved great success across a wide range of sequential learning tasks, such as machine translation, speech recognition and time series forecasting (TSF) [1]. However, RNNs are ultra-parameterised models that overfit without sufficient data at hand, making them unfit to solve tasks where few data are available. We propose a novel RNN approach to infer full predictive distributions for TSF tasks where there is little to no training data at hand, respectively referred to as few-shot and zero-shot learning. We achieve this by learning a shared feature representation across multiple forecasting tasks, then transferring this knowledge to help predict unseen time series, even in observation poor environments. We show that our approach outperforms other (widely used) approaches across a range of metrics as well as providing a framework to visualise and interpret the embedding learnt by the neural network.

Related work. There is a rich literature associated with transfer learning and few-shot learning, especially in the Computer Vision community, where pre-trained models such as VGGNet, ImageNet and AlexNet are regularly used as either transferrable feature extractors, followed by a simpler classifiers, or as initial priors before fine-tuning a model.

However, to the best of the authors' knowledge, the transfer learning literature for time series forecasting is significantly more scarce and often limited to evaluations solely based on the residuals of point forecasts. Full probabilistic forecasts are, however, desirable as this enables practitioners to incorporate predictive uncertainty in their decision-making processes. This is often achieved using Gaussian Processes (GPs) [2], non-parametric probabilistic models that are known to generalise well even with little data.

*We acknowledge the support provided by CONACYT (CVU #598304) and the UK Royal Academy of Engineering.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

2 METHODOLOGY

Our strategy consists in inferring a joint Memory-endowed Ordinal Regression Deep Neural Network (MOrdReD) [3] over a collection of (auxiliary) time series. This results in a cross-task feature embedding that we use as a basis to forecast unseen time series even with scarce data. We briefly summarise this model below, but refer the reader to [3] for a more detailed explanation.

2.1 Memory-endowed Ordinal Regression Deep Neural Networks

MOrdReD is an ordinal regression approach to time series forecasting that has been shown to achieve long-term, reliable performance. In the ordinal setting, real-valued time series observations are firstly discretised into M (normally equal-sized) subintervals $\mathcal{C} = \{\mathcal{C}_k\}_{k=1}^{M}$ and encoded using a 1-of-M scheme. Such discrete symbol sequences have been most efficiently dealt with in NLP tasks by deep recurrent neural networks and in particular sequence-to-sequence extensions of the Long Short-Term Memory (LSTM) model [1].

Sequence-to-sequence models fit together two LSTM layers, an encoder $f^{(\text{enc})}$ and a decoder $f^{(\text{dec})}$. In the first stage of the model's *forward pass* the encoder sequentially scans P measurements of the time series $\mathbf{X}^{(\tau-1)} = (\mathbf{x}_{\tau-P}, \ldots, \mathbf{x}_{\tau-1})$ and summarises them as temporal characteristics $\mathbf{h}_0^{(\text{dec})}, \mathbf{C}_0^{(\text{dec})}$. These are then fed as initial states to the LSTM decoder, which iteratively produces an output sequence. The $(t+1)$-th element of the output sequence, $\hat{\mathbf{x}}_{t+1}$, is then given by:

$$(\mathbf{h}_0^{(\text{dec})}, \mathbf{C}_0^{(\text{dec})}) = f^{(\text{enc})}(\mathbf{X}^{(\tau-1)}),$$
$$\hat{\mathbf{x}}_{t+1} = \text{Softmax}\left(f^{(\text{dec})}(\mathbf{h}_{t-1}^{(\text{dec})}, \mathbf{C}_{t-1}^{(\text{dec})}, \mathbf{x}_t)\right).$$

The softmax output $\hat{\mathbf{x}}_{t+1}$ can be interpreted as both a categorical distribution over the set of ordinal bins, $\mathcal{C} = \{\mathcal{C}_k\}_{k=1}^{M}$, at time $t+1$, and as a piece-wise uniform over the original time series range. Output sequences are computed by iteratively feeding back $\hat{\mathbf{x}}_{t+1}$ into the model and predictive posterior distributions can readily be approximated through Monte Carlo Dropout [4].

2.2 A General Unified Model for time series forecasting

Our GUM approach is based on inferring a shared representation space of time series. This shared embedding is learnt by fitting a joint MOrdReD model over a single compilation dataset $\mathcal{X}_{\text{aux}}$ of 20 different auxiliary time series which are assumed to possess sufficient observations for training. The datasets were drawn from a variety of application domains, and further detail is given in `https://github.com/bperezorozco/ordinal_tsf`.

Crucially, our ordinal regression cross-task embedding is learnt over symbolic patterns that are invariant to the original scale of the time series, as all auxiliary time series are quantised independently, but with the same number of bins $M = 150$. A variety of shapes and motifs from the auxiliary datasets are thus encoded in a feature space of ordinal bin motifs. We analyse this cross-task embedding in Section 3.2.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

We note that the quantiser proposed in [3] requires prior knowledge of the time series range. With scarce data, the observed range could change within short predictive horizons. We handle this with a simple linear heuristic that estimates the range boundaries within a predictive horizon P_h. Given the largest first-order finite difference of the time series, Δ_{max}, our linear heuristic approximates the boundaries $\tilde{\mathbf{X}}_{\text{min}}, \tilde{\mathbf{X}}_{\text{max}} = \mathbf{X}_{\text{min}} - P_h\Delta_{\text{max}}, \mathbf{X}_{\text{max}} + P_h\Delta_{\text{max}}$. Future work for improving this estimation includes relating it to Lipschitz constant inference [5].

3 EXPERIMENTS

3.1 Task 1: Zero-shot forecasting

We first assess the model's performance at zero-shot time series forecasting.

Data. We used two dataset collections in our experiments. The first was used to fit our GUM ordinal regression neural network, and contains 20 varied auxiliary datasets drawn from [6] with outputs evenly quantised over $M = 150$ bins. A copy of our auxiliary datasets is provided in `https://github.com/bperezorozco/ordinal_tsf`. The second collection is our test data, which corresponds to the yearly-frequency segment of the M4 competition dataset [7]. We drew 150 excerpts that had no missing data (no interpolation needed) and that had at least 36 observations available.

Task. Every M4 time series was split into $T = 21$ timesteps, assumed to be observed, and $P_h = 15$ timesteps for prediction. For our baselines, each segment represents the training and test data, respectively; for GUM, the quantised 21-sample excerpt is fed as the encoder input sequence, before the decoder computes the forecast with horizon $P_h = 15$. Importantly, no neural network parameters are updated whilst scanning the 21-sample sequence.

Baselines. We compared our GUM neural network against two popular models: Gaussian Process Regression (GPR) [2] and state-space AR(p) models [8]. Detailed derivations of both methods can be found in the given references. In the case of GPR, we fitted two instances for every time series: one with the Mátern 5/2 kernel and one with the Rational Quadratic kernel, both commonly found in the literature to model a wide range of physical phenomena. In the case of State-space AR(p), we fitted two instances with lookback $p = 3, 4$.

Metrics. We evaluated models using three metrics: (1) the **Negative log-likelihood (NLL):** through this we assess how well predictive densities capture the ground truth prediction; (2) **The root mean squared error (RMSE):** to measure the mean predictive accuracy of the models; and (3) **Quantile-Quantile (QQ) distance:** QQ plots are graphical tools used to assess how well predictive densities match empirical probabilities. We summarise how well calibrated predictive distributions are through the QQ distance, defined as the deviation between the identity and the obtained QQ plot. We refer the reader to [3] for full details.

Results. We present our results in Table 1. The Table is divided into two sections. On the left side, we show the percentage of datasets in which our GUM neural network outperforms each of our benchmarks for every metric. We show that across all metrics, our model consistently performs better than all baselines for the majority of cases. On the right side of Table 1, we provide the average rank of our models. Ranks from 1

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

	(%) GUM outp.				Zero-shot rank			Few-shot rank			
	AR3	AR4	M5/2	RQ	GUM	GP	AR	GUM	Mor	GP	AR
NLL	65	70	59	62	**1.76**	2.04	2.20	**1.1**	3.1	3.2	2.6
RMSE	64	68	54	58	**1.83**	2.01	2.15	**1**	3.1	2.9	3
QQD	74	75	67	67	**1.65**	2.01	2.34	**1.8**	2.3	3	2.9

Table 1: **Left**: percentage of unseen evaluation datasets where GUM outperforms each of our baselines in the zero-shot setting. **Centre**: average rank achieved by each model in the zero-shot setting. Ranks 1 (best) to 3 were assigned to each model for each dataset and metric. **Right**: average rank achieved by each model in the few-shot setting. Ranks 1 (best) to 4 were assigned to each model for each dataset and metric.

(best) to 3 (worst) were assigned for every metric, dataset and model and averaged across all M4 evaluation datasets. In the case of GPR and AR(p), the instance with the best performance across kernels or model order p was chosen. This metric shows that, on average, we expect GUM to outperform all other models; and additionally, when it does not, it can still be expected to come as a second competitor.

3.2 Cross-task feature embedding analysis

We now provide a visual analysis of the cross-task feature embedding learnt by GUM. We first compute $\mathbf{h}_0^{(\text{dec})}$ (as described in Section 2) for a random sample of auxiliary time series excerpts, in addition to all M4 evaluation input sequences. Intra-group minimum variance clusters are then computed through Ward's agglomerative clustering. Results in Figures 1 and 2 are provided for $K = 36$ clusters, which yields the largest silhouette score for values between 5 and 50.

We show the resulting frame clusters in Figure 1, observing that those in red are only seen by our GUM network at testing time. We note that the excerpts mainly group towards the right-hand edge of each subplot, i.e. every cluster contains patterns with diverse features (such as degree of smoothness, number of optima and rate of change) that fire similar activations, which then produce similar forecasts at the output layer.

In Figure 2 we show the 2D UMAP [9] representation of each frame's feature vector $\mathbf{h}_0^{(\text{dec})}$, colour-coded by cluster id. Bluer and redder markers represent time series clusters that converge to lower- and higher-valued bins at the last timestep, respectively. The smooth colour degradation from left to right, with blue clusters grouping at the left-hand edge and red clusters towards the right, suggests that our model is learning an approximate ordering over ordinal bins.

3.3 Task 2: Few-shot forecasting

We now assess model performance in the few-shot setting.

Data. We use the same 20 auxiliary datasets from Section 3.1 to train GUM. For the testing phase, we now use 10 series of the monthly-frequency segment of the M4 dataset, those with no missing observations and with at least 1500 measurements available for model inference. All datasets are quantised using $M = 150$ bins.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

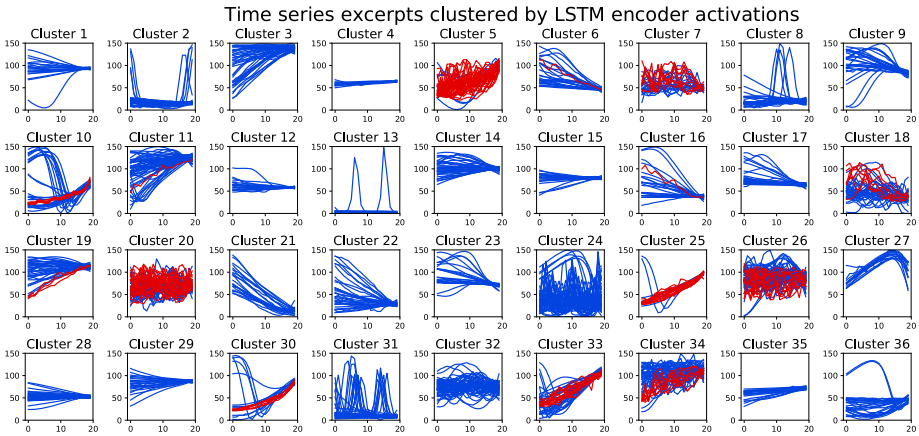

Fig. 1: Sample excerpt groups resulting from clustering the LSTM encoder activations over both auxiliary (in blue) and M4 unseen time series (in red). Frames in the same cluster trigger similar LSTM activations and thus produce similar forecasts.

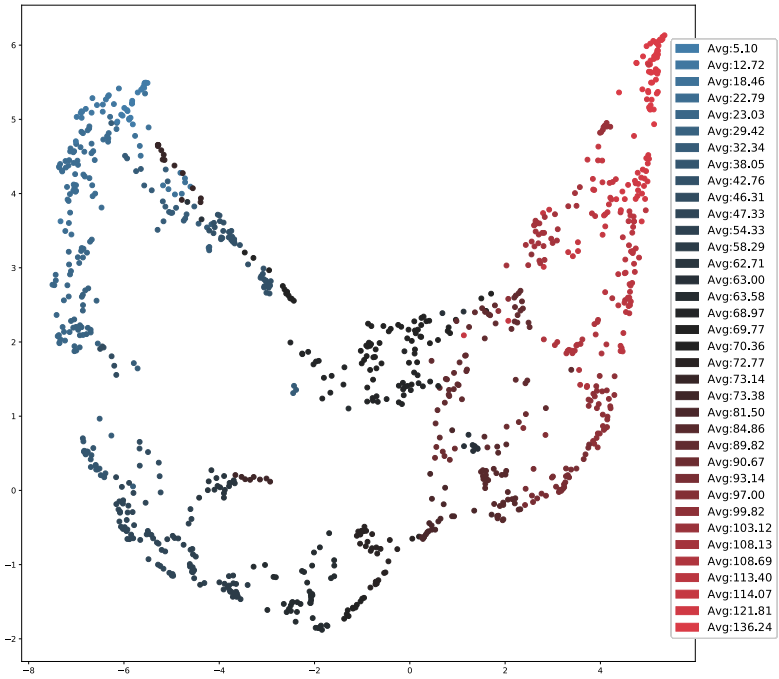

Fig. 2: 2D UMAP representation of LSTM encoder activations, where each cluster is colour-coded by the average value of the last timestep across its excerpts. Bluer and redder markers represent time series excerpts that converge to lower- and higher-valued bins at the last timestep, respectively. The colour degradation from left to right suggests that GUM is implicitly learning a quasi-ordering over ordinal bins.

ESANN 2020 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Online event, 2-4 October 2020, i6doc.com publ., ISBN 978-2-87587-074-2. Available from http://www.i6doc.com/en/.

GUM setup. In this setting, we pre-train a number of GUM models over the auxiliary datasets with different hyperparameters: hidden units $n_h \in \{64, 128, 256, 512\}$, dropout rate $\theta_{\text{drop}} \in \{0.25, 0.5\}$ and L_2 regularisation constant $\lambda \in \{10^{-5}, 10^{-6}, 10^{-7}, 10^{-8}\}$. These serve as initialisation parameters for each M4 time series. Models are trained for up to a further 50 epochs with early stopping and the best hyperparameters for each dataset are chosen by grid search over a validation segment.

Task, baselines and metrics. We forecast $P_h = 100$ measurements ahead from a lookback window of $P = 50$, using the same baselines and metrics from Section 3.1, in addition to a naive MOrdReD baseline that has not been pre-trained, i.e. is initialised from scratch and allowed twice as many epochs, also with early stopping. The AR(p) baseline's lookback is chosen from $p \in \{3, 6, 12, 18\}$.

Results. We note that our GUM neural network consistently outperforms our baselines across all metrics, as shown in Table 1. Crucially, GUM also outperforms the naive MOrdReD baseline across all metrics, suggesting that transferring knowledge from prior tasks makes a substantial contribution to predictive performance.

4 CONCLUDING REMARKS

We here introduce a novel neural network model, able to infer probabilistic time series forecasts in both zero- and few-shot learning settings. By introducing a quantisation scheme, we show how knowledge learnt from auxiliary time series can be transferred to previously unseen time series with scarce data. We provide empirical evidence that our approach performs well across a range of metrics, in comparison to other popular methods. By examining the neural activations of our model, we also show how our cross-task feature space encapsulates concepts such as ordinality and knowledge transfer.

References

[1] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. "Sequence to Sequence Learning with Neural Networks". In: *Advances in Neural Information Processing Systems 27*. 2014.

[2] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[3] Bernardo Pérez Orozco, Gabriele Abbati, and Stephen Roberts. "MOrdReD: Memory-based Ordinal Regression Deep Neural Networks for Time Series Forecasting". In: *arXiv:1803.09704* (2018).

[4] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning". In: *Proc. of The 33 International Conference on Machine Learning*. 2016.

[5] Jan-Peter Calliess. "Bayesian Lipschitz Constant Estimation and Quadrature". In: *Advances in Neural Information Processing Systems*. 2015.

[6] Ben Fulcher, Max Little, and Nick Jones. "Highly comparative time-series analysis: the empirical structure of time series and their methods". In: *Journal of The Royal Society Interface* 10.83 (2013).

[7] Spyros Makridakis, Evangelos Spiliotis, and Vassilios Assimakopoulos. "The M4 Competition: Results, findings, conclusion and way forward". In: *International Journal of Forecasting* (2018).

[8] J. Durbin and S.J. Koopman. *Time Series Analysis by State Space Methods: Second Edition*. Oxford Statistical Science Series. OUP Oxford, 2012. ISBN: 9780199641178.

[9] L. McInnes, J. Healy, and J. Melville. "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction". In: *ArXiv e-prints* (Feb. 2018). arXiv: `1802.03426 [stat.ML]`.