 doi:10.20944/preprints201907.0241.v1

TESTING RNN-LSTM FORECASTING WITH SIMULATED ASTRONOMICAL LIGHTCURVES

A PREPRINT

Nachiketa Chakraborty

July 19, 2019

ABSTRACT

With an explosion of data in the near future, from observatories spanning from radio to gamma-rays, we have entered the era of time domain astronomy. Historically, this field has been limited to modeling the temporal structure with time-series simulations limited to energy ranges blessed with excellent statistics as in X-rays. In addition to ever increasing volumes and variety of astronomical lightcurves, there's a plethora of different types of transients detected not only across the electromagnetic spectrum, but indeed across multiple messengers like counterparts for neutrino and gravitational wave sources. As a result, precise, fast forecasting and modeling the lightcurves or time-series will play a crucial role in both understanding the physical processes as well as coordinating multiwavelength and multimessenger campaigns. In this regard, deep learning algorithms such as recurrent neural networks (RNNs) should prove extremely powerful for forecasting as it has in several other domains. Here we test the performance of a very successful class of RNNs, the Long Short Term Memory (LSTM) algorithms with simulated lightcurves. We focus on univariate forecasting of types of lightcurves typically found in active galactic nuclei (AGN) observations. Specifically, we explore the sensitivity of training and test losses to key parameters of the LSTM network and data characteristics namely gaps and complexity measured in terms of number of Fourier components. We find that typically, the performances of LSTMs are better for pink or flicker noise type sources. The key parameters on which performance is dependent are batch size for LSTM and the gap percentage of the lightcurves. While a batch size of $10 - 30$ seems optimal, the most optimal test and train losses are under 10% of missing data for both periodic and random gaps in pink noise. The performance is far worse for red noise. This compromises detectability of transients. The performance gets monotonically worse for data complexity measured in terms of number of Fourier components which is especially relevant in the context of complicated quasi-periodic signals buried under noise. Thus, we show that time-series simulations are excellent guides for use of RNN-LSTMs in forecasting.

Keywords First keyword · Second keyword · More

1 Introduction

Time domain astronomy is a rapidly growing field with problems ranging from "physics-based" time-dependent modeling [Rieger(2019), Baring & Boettcher(2019)] of driving processes to the model independent forecasting of transients [Sadeh(2019), Vaughan et al.(2016), Ait Benkhali et al.(2019)]. Traditionally, astronomy has been a field where modeling of observational data is done with a theoretical or phenomenological, physics-based model. Now with ever increasing volumes of time domain data [Vaughan et al.(2016), Ivezić et al.(2019), Graham et al.(2019)], from contemporaneous observations, fast and precise forecasting is becoming increasingly important. Some of the most important forecast problems for astronomical time-series are for transient flares and quasi-periodic oscillations that are buried under stochastic signals. Note that the stochasticity or noisy component is actually a combination of background noise either due to instrumental or observational backgrounds and a genuine astrophysical component related to stochastic processes in the transient source. This constitutes an interesting forecasting challenge. Prediction of noisy time-series is certainly not limited to astronomy. In fact, a key application is in the field of finance. As explained in [Giles et al.(2001)], in general, forecasting from limited sample sizes in a low signal to noise, highly non-stationary

and non-linear setting is enormously challenging. And in this most general scenario, [Giles et al.(2001)] prescribes combination of symbolic representation with a self-organising map and grammatical inference with recurrent neural networks (RNNs). In our case of astrophysical signals, while we maybe in the high noise regime, the non-linearity and non-stationarity can be assumed to be weak in most typical cases. Therefore, we simply use RNNs. While we do not wish to draw conclusions and see patterns and trends by over-fitting a unwanted noise in the lightcurves, we do wish to study outbursts that emerge at short timescale of the noise spectrum. In fact, a key question in variability in AGNs is distinguishing between statistical fluctuations that in extreme cases lead to occasional outbursts and a special event isolated from these underlying fluctuations and thus drawn from a different physical process or distribution equivalently. An agnostic learning algorithm should be able to predict the former but not the latter. In order to do this, we must broadly characterise the response of RNN-LSTM architecture to noise characteristics. This is clearly relevant in predicting a well defined transient episode like an outburst or flaring episode or a potential quasi-periodic signal in advance in real astronomical observations.

One of the keys to such forecasting is understanding, as best as we can, the performance of the algorithm in presence of different types of temporal structures. And the quantification of the performance is done in terms of training and test losses.

There are several predictive frameworks for time-series, machine learning and otherwise. A number of time-series models are in use across domains, such as Holt-Winters [Andrysiak et al.(2018)], ARIMA [Namini et al., 2018], SARIMA, GARCH, etc. Typically the Autoregressive Integrated Moving Average (ARIMA) models are most popularly used for forecasting in several domains with the "Integrated" part dealing with the large scale trends. For detecting features and patterns, supervised machine learning algorithms such as Random Forest, Gradient Boosting Machine, etc. Viewing the problem as a regression problem one can use Lasso Regression, Neural Networks, etc. In this paper, we do not attempt to provide a comprehensive survey on the various time-series forecasting methods and their efficacy for forecasting in domains modest signal-to-noise domains like astronomy. Rather we focus on one of the most popular methods that has yielded success for long-term time-series data. In this respect, recurrent neural networks (RNNs) are an excellent candidate.

RNNs like other neural networks have the capabilities of handling complex, non-linear dependencies. But unlike other networks, its "directed cyclic" nature, which allows the input signal at any stage to be fed back to itself suitable for time-series work. In particular, we focus Long Short-Term Memory RNNs architectures (LSTM) [Hochreiter and Schmidhuber, 1997][Brownlee 2017] which are capable of keeping track of long-term dependencies. This is clearly important in astronomy with the rather large dynamic range of timescales of underlying physical processes as well as observations. In order to do this, we test with artificial datasets that do not have a very complex temporal structure. More significantly, we use artificial datasets with characteristics typically found in long term astronomy observations and that we have some apriori knowledge of.

In the spirit of this approach, we subdivide our so-called sensitivity tests in two sections. In section 3, we examine the effects of dataset characteristics, gaps and complexity in terms of fourier components on performance. In section 4, we examine dependencies on the neural network (hyper)parameters.

2 Recurrent Neural Network with Long Short Term Memory Architecture

Here we describe the structure of the forecasting machinery used. Astronomical time-series or lightcurves especially those that are obtained from long term monitoring campaigns tend to have correlations over a wide range of timescales ranging from minutes to years. Given the success of RNN-LSTMs in forecasting time-series with correlations over longer timescales, we wish to perform our sensitivity tests on this method. The network architecture is what is typically used in forecasting time-series or sequential data in general. We build the RNN with multiple LSTM layers with 3-dimensional input namely samples, time step and features. The samples represent the no. of data points in the time-series. The features are the no. of characteristics per time step. The layers are stacked. The simplest architecture will have a single input layer, one hidden layer which is an LSTM layer and an output layer (labelled "Dense"). Each LSTM layer has the tunable parameters, batch size, no. of units, neurons or cells, the input shape for our sensitivity tests. The LSTM layer also specifies an activation function which defines the way outputs are determined at the nodes. The default is a linear activation function in RNNs. This works nicely in feedforward networks as it is simpler and faster to compute and avoids the vanishing gradients problem. However, with LSTMs the latter is dealt with differently with the help of gates in each LSTM cell.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 July 2019 doi:10.20944/preprints201907.0241.v1

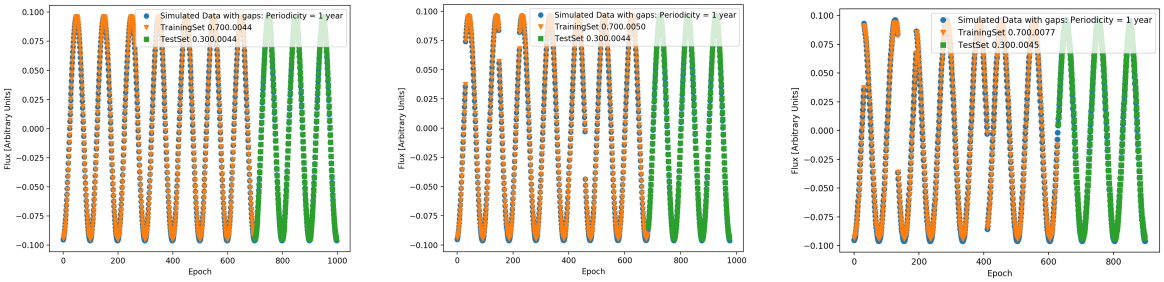

Figure 1: *Left:* The figure shows the prediction of a strictly periodic signal with a period of a year, with RNNs with LSTM. This is a trivial case of forecasting for a time-series exactly known.*Center:* The signal is once again the annual periodicity, but with missing data. As shown, this affects the prediction a small amount

3 Effect of data characteristics

We wish to examine the effects of two typical features of time-series found in astronomy. The first is missing data or gaps which are not exclusive to astronomy. Second is relevant for the (quasi-)periodic variations that very significant in astronomy, amongst other things in the context of gravitational wave sources.

3.1 Effect of missing data : Periodic signals with random gaps

There is extensive work on treating missing data in time-series analysis. Here we look at the effects of two types of gaps on the test and training losses in signals, random and periodic. We start with strictly periodic signals, which is exactly predictable as long as there is very little missing data. Therefore, this is a good testing ground for "calibrating" sensitivities to various algorithmic parameters and properties as well as those of the data as well.

The figure 1 shows a periodic signal with a period of 1 year. The leftmost figure shows simulated data for multiple cycles with an interval of 0.01 years with no missing data or gaps. With a train to test ratio of $70:30$, the prediction is very accurate. However, it is interesting to note that the score based on RMSE used here for the test set is not perfect. This is consistent with the finding in [Han et al.,2018] that more complex models do not necessarily lead to more accurate predictions.

Now as we increase the gaps by increasing the fraction of missing data (MDF), we find the difference between RMSE train and test losses increases. This shows that even a perfectly periodic signal cannot be perfectly predicted if there is missing data and the prediction accuracy generally becomes worse the greater the fraction missing.

3.2 Effect of missing data : Periodic signals with periodic gaps

In this case the gaps introduced were at random in a periodic signal with amplitude modulated by a stochastic component. In the case of data missing at regular or periodic intervals, from the lightcurves or time-series, the dependence of accuracy on the fraction of missing data is complex. As seen in 3, there seems to be a sweet spot for minimising the ratio of test loss to training loss near 1.0 or perfect prediction. And this occurs at around the missing data fraction of $MDF \approx 1/3$ beyond which the test to training loss ratio increases with MDF. For pink noise lightcurves, this effect is not very pronounced and is under $6-7\%$. However, for red noise lightcurves, this effect is significantly larger as seen in 3. Furthermore, for a "hard sigmoid" activation function for the final dense LSTM layer, this optimisation is far (< 3) worse than for a "tanh" activation.

3.3 Effect of data complexity ?

There are different ways of quantifying the complexity of a time-series. There's entropy which is inherently linked to the probability distribution function of the process. Here we look at complexity in the temporal structure of the time-series. Any continuous process represented by a time-series can be decomposed as a Fourier series. One could argue plausibly that complexity increases as one adds to the number of Fourier components. This notion would be highly relevant for astronomy signals which are a combination of multiple quasi-periodicities along with stochastic components. Here we test the effect of the no. of Fourier components on the prediction performance. We simulate signals with an annual periodicity $\Omega = 1/2\ \pi(1year)$, but add harmonics which are prime integer multiples of the

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 July 2019 doi:10.20944/preprints201907.0241.v1

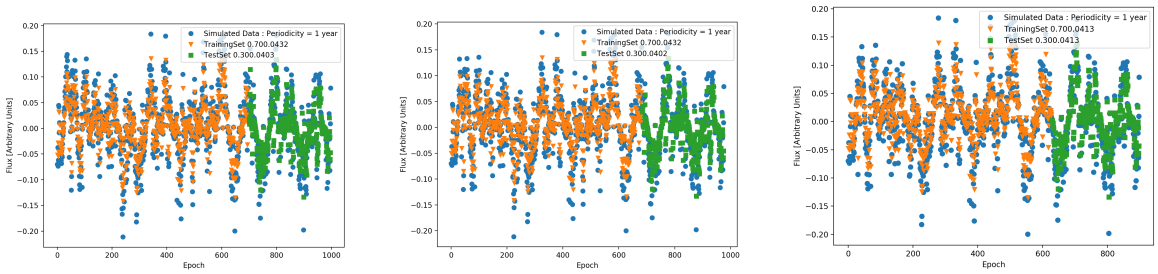

Figure 2: *Left:* The figure shows the prediction of the annual periodic signal modulating a stochastic, pink noise signal, with RNNs with LSTM. The prediction while not trivial as the strictly periodic case, is still strong. *Center:* The same signal with missing data (2%) clearly makes trickier to predict accurately. *Right:* With 10% of missing data seems to be exactly precise yet this is misleading.

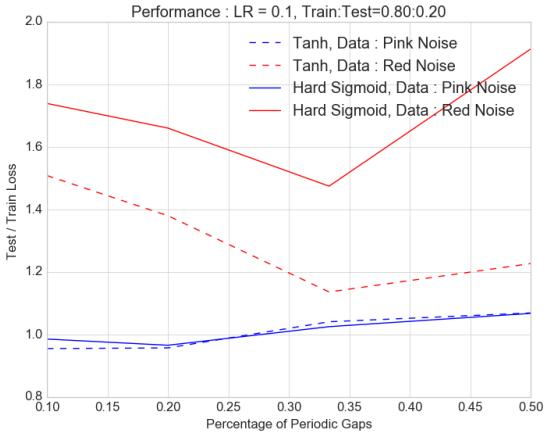

Figure 3: The ratio of the test to train loss is shown as a function of the percentage of periodic gaps in data for pink (*blue and red noise (*red). This is shown for hard sigmoid (*solid) and tanh (*dashed) activation with the latter as the most optimal scenario. For fraction of gaps exceeding $\approx 30\%$, the accuracy gets worse as proportion of missing data increases. This deterioration is less pronounced for pink noise.

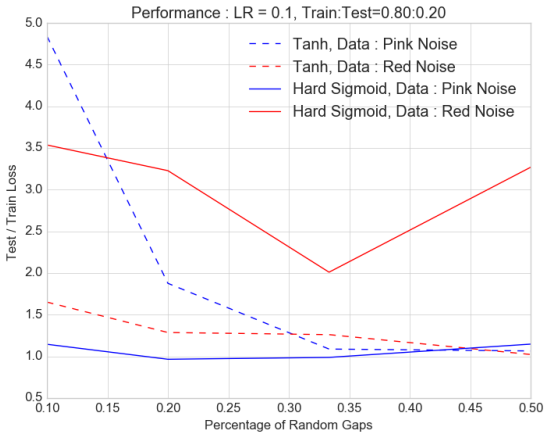

Figure 4: Like in figure 3, the ratio of the test to train loss is shown as a function of the percentage of random gaps in data for pink (*blue and red noise (*red) ; once again the activation functions are hard sigmoid (*solid) and tanh (*dashed). For fraction of gaps exceeding $\approx 30\%$, the accuracy gets worse as proportion of missing data increases. This deterioration is less pronounced for pink noise.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 July 2019 doi:10.20944/preprints201907.0241.v1

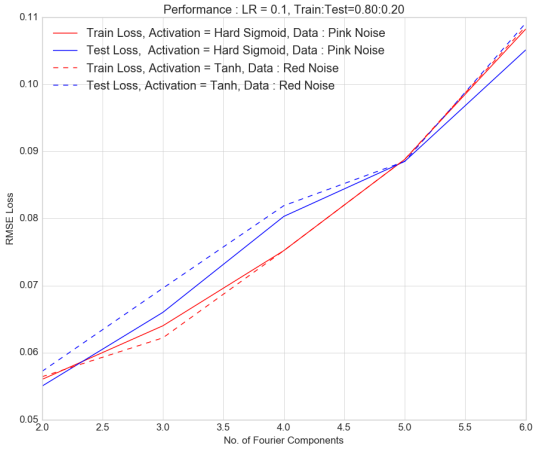

Figure 5: The test *(blue) and train (*red) losses are shown as a function of the no. of Fourier components in the pink noise signal ; the activation functions are hard sigmoid (*solid) and tanh (*dashed). As the complexity i.e. the number of Fourier components increase, both the train and test losses increase as predictions tend to get harder to make.

annual periodicity i.e. 2Ω, 3Ω and $5\ \Omega$ and so on. We then use one LSTM layer with a learning rate of 0.1 and add a dense layer with either tanh or hard sigmoid activation. In figure 5, we see clearly that broadly, as the complexity or the no. of Fourier components increases, both the test and training losses grow. This is consistent with the intuitive picture that the forecasting for any network is more difficult as the complexity increases and this necessitates increasing the complexity of the network itself.

4 Effect of RNN - LSTM structure

In case of more complex datasets, even if there is no missing data, the architecture of RNN or deep learning framework can affect the precision of prediction. In fact, the training of the framework on a subset of the data is how structure of the network optimised for precise predictions is determined. Therefore, there is an element of arbitrariness in prior choices of framework required for forecasting of specific datasets. By examining the sensitivity of precision of forecasting for more complex datasets, to changes in the architecture, one could build an intuition if not a precise calibration of the RNN-LSTM framework.

The different tunable parameters in a deep learning framework such as the RNN-LSTM could be related to the structure or architecture of the network or indeed those related to the cost function. The parameters related to the architecture are the number of layers, their activation functions, LSTM units, batch size, etc. These are likely to affect the forecast performance on datasets quantified in terms of the losses. The parameters related to the cost function like learning rate, momentum, etc. are likely to affect the rate and strength of convergence. These may not have a direct, interpretable effect on the losses.

We find this is indeed the case. For a range of train to test ratio of datasets from $0.60 : 0.40$ to $0.90 : 0.10$, we find that there is no effect of varying the learning rate from 0.01 to 0.1 on the train and test losses. On the other hand parameters like the batch size and number of layers do have significant effects as described in the sections below. For instance, we find that varying the lookback parameter we find that for a example of pink noise time-series, the ratio of test to train loss peaks at the value of 10 and is also closest to 1.0 at this value. This suggests this as an optimal choice atleast for this dataset. And we freeze the lookback at this value as our fiducial value in examining sensitivities of other important structural parameters.

4.1 Number of layers

The number of layers allows to model increased complexity in the time-series. In approximate terms, this implies an increased accuracy in the forecasts. This is what is seen in predicting red noise lightcurves. We use 1, 2 and 3 layers of RNN-LSTM networks with a learning rate of 0.1 and a train to test ratio of 0.8. And we find that the accuracy is increased substantially to achieve near perfect scores on addition of a third layer. It is important to note that there is no such thing as a "perfect score" in predicting stochastic time-series as there are statistical uncertainties with future

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 July 2019 doi:10.20944/preprints201907.0241.v1

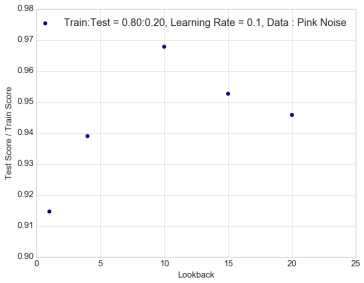

Figure 6: The ratio of test to train loss as function of lookback is shown for the pink noise case. It shows that the train and test loss values are the closest with the ratio being nearest to 1 for lookback of 10.

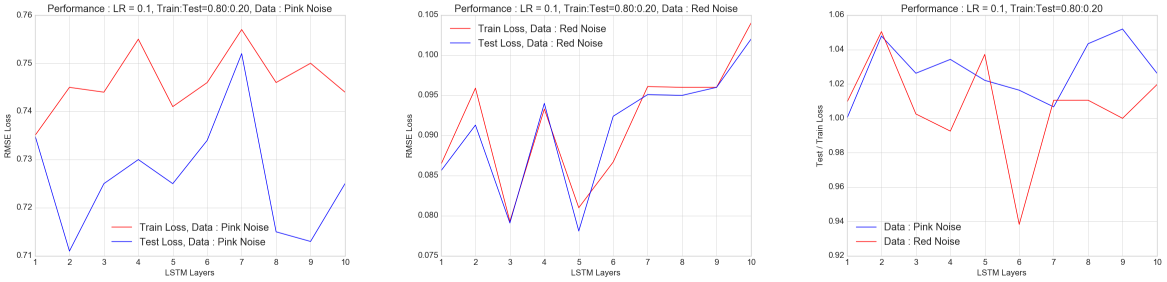

Figure 7: The dependence of RMSE loss of pink (*Left*) and red (*Right*) noise process on the layers is shown in the figure for a learning rate of 0.1 and a train to test ratio of $0.80 : 0.20$. While there is sensitivity to the number of identical RNN-LSTM layers, there is neither a trend nor a "sweet spot" for this number. This is suggestive that adding layers does not improve performance dramatically. In general, the performance is better or more consistent for pink noise.

data or additional realisations. Also, it is unreasonable to expect a precise "calibration" of the parameters to such a stochastic dataset. However, what these experiments establish are broad, guiding trends on how to use the LSTM structure effectively in the predictions of non-deterministic time-series.

In the figure 7, we see the RMSE losses as function of the number of layers for pink and red noise processes for a learning rate of 0.1 and train to test ratio of $0.80 : 0.20$. To the left, are the losses for the pink noise case, around the 70% mark. The losses oscillate above and below this level as the number of layers are increased. A similar effect is seen in the middle panel for the red noise scenario, around the 9% mark with an increase from 6 to 10 layers. In both cases, it is clear that there is neither a clear trend nor a "sweet spot" for no. of layers to provide the best performance for noisy time-series. The pink noise light-curves seem to be consistently overfit. Overfitting to noise that can lead to decreased losses can be compensated by random correlations between input and output, which can provide a complicated dependency as the one shown in the figure 7. This is true for both the cases, but is especially strong an effect for red noise. Furthermore, it is expected that the red noise process, due to weak non-stationarity can lead to poorer performance.

4.2 Learning rate

The learning rate is a crucial parameter in determining the convergence of any deep learning algorithm to the most accurate solution. We use the Stochastic Gradient Descent method and vary the learning rate from 0.05 to 0.5. The train and test losses are computed for a ratio of $0.80 : 0.20$ to with a batch size of 1 . From figure 8, it is evident that the learning rate does not impact the performance as measured by these losses. This is to be expected as the learning rate is more likely to influence the convergence rate to the precise solution, than the latter itself.

4.3 Batch Size

The batch size is a crucial, tunable hyperparameter in the learning process and network architecture. Typically smaller batch sizes are preferred on account of ease of storage and computational parallelisation. However, smaller batch sizes

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 July 2019 doi:10.20944/preprints201907.0241.v1

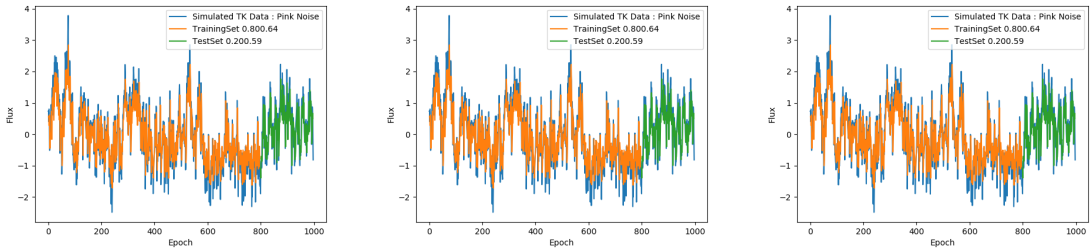

Figure 8: The dependence of prediction accuracy of the pink noise process on the learning rate is shown in the figure for a batch size of 1 and a train to test ratio of $0.80 : 0.20$. The learning rate does not impact the test and train scores significantly just the rate of convergence.

Figure 9: The dependence of training and test losses of the pink and red noise processes on the batch size is shown in the figure for a train to test ratio of $0.80 : 0.20$.

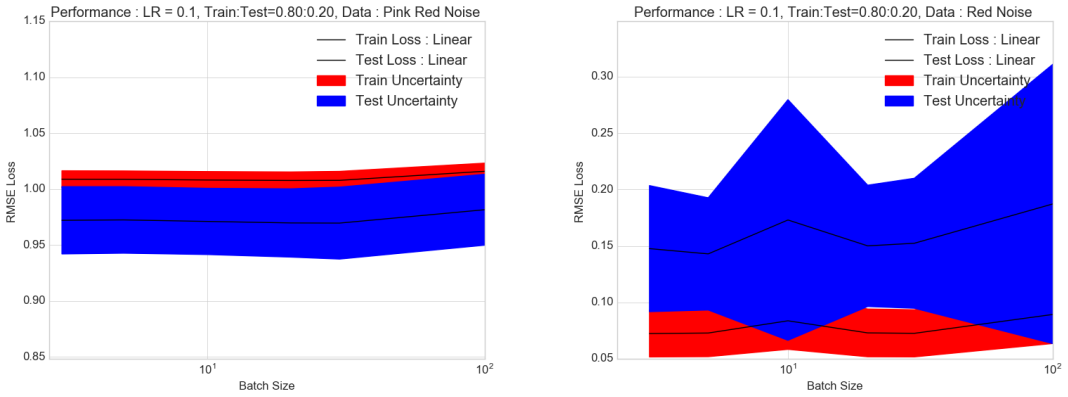

Figure 10: The uncertainty on the effect of batch size is calculated by repeating the exercise in fig 9 over 100 simulations of pink and red noise. Uncertainty on test loss is shown in blue, whereas that on the training loss is shown in red. This shows an optimal batch size of $\sim 10 - 30$ with a hint of overfitting with pink noise.

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 July 2019 doi:10.20944/preprints201907.0241.v1

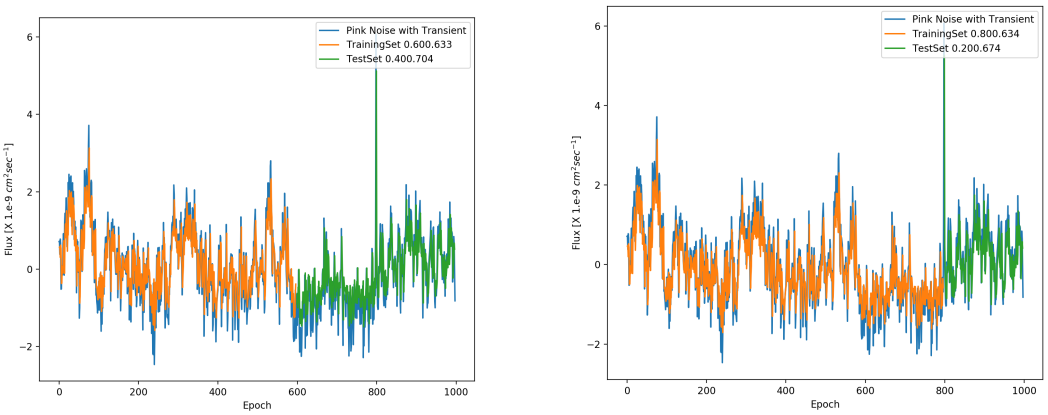

Figure 11: The figure shows detection of a transient in presence of pink noise. To the left, is a train to test split of $60:40$ and to the right is a $80:20$ split. The LSTM with batch size 30 performs well, improving as we go nearer to the exponentially shaped transient.

can adversely affect the generalisation capabilities. The detailed dependence of performance on batch size depends on the problem at hand and is not very well understood at the moment.

Here in the example in figure 9, we show how the RMSE losses vary with batch size for pink and red noise processes. In both cases, it is clear that this is not a monotonic dependence, and that there is a "sweet spot". For these time-series of 1000 steps or data points with a mean of 0.0 and variance of 1.0, this "sweet spot" seems to be approximately at $batchsize = 10$ for a learning rate of 0.1 for 5 LSTM layers and an additional dense layer with both tanh and hard sigmoid activations. For the pink noise case, to the left of the figure 9, the comparison of the train and test losses, show that there's overfitting, however, this RNN-LSTM architecture seems to perform a lot better for the red noise case.

Now these results are valid for single realisations of lightcurves. In order to robustly predict the sensitivity, we need an estimate of the uncertainty. In order to compute the statistical uncertainty we simulate 100 lightcurves and repeat the above procedure. For simplicity we do this for linear activation with fiducial values of other parameters as learning rate of 0.1 and train to test ratio of $0.80:0.20$ as before. In doing this we find, that the optimal batch size is approximately 30 as shown in the figure. The uncertainty in the case of pink or flicker noise in both the test (blue) and training (red) losses are under 10%. Thus, we conclude that a batch size between 10 to 30 is reasonable for predicting these lightcurves. However, the uncertainty can be as high as 30% for red noise, even though the absolute values of training and test losses are lower. This shows that the performance for red noise lightcurves is less stable compared to pink noise lightcurves.

4.4 Validation splitting

A key aspect of testing robustness and the "true" predictive accuracy and precision, involves subdividing the dataset further to include a validation subset. The validation subset allows us to evaluate the model while tuning the hyper-parameters. This evaluation is unbiased as the data used for training is not used. Validation leads to the final, fully trained model ready to be used on the test dataset. For the "ideal" model, both the validation and training loss should be low with the former just a bit higher than the latter. In our case of fitting the lightcurves with a significant stochastic component, there's always the danger of over-fitting the noise and this can hamper the predictive power of a QPO or a genuine transient flare. If the training loss is low and the validation loss is much higher, then we are over-fitting and ability to generalise to new data is poor. This requires us to chose a validation set as well as hyperparameters carefully.

5 Detection of Transient

Having characterised the performance of the LSTM network with pink and red noise with different parameters, we can now test detection of a transient. We once again simulate pink and red noise lightcurves, but just a single realisation this time. We add to it a sharp transient characterised by the exponential function as $\exp(-|t-t0|/\tau)$, where $\tau = 3 \times 10^5$ which is approximately a hundredth of the total length of the lightcurve or $Tobs \approx 100 * \tau \approx 1$ year. And we position

Preprints (www.preprints.org) | NOT PEER-REVIEWED | Posted: 22 July 2019 doi:10.20944/preprints201907.0241.v1

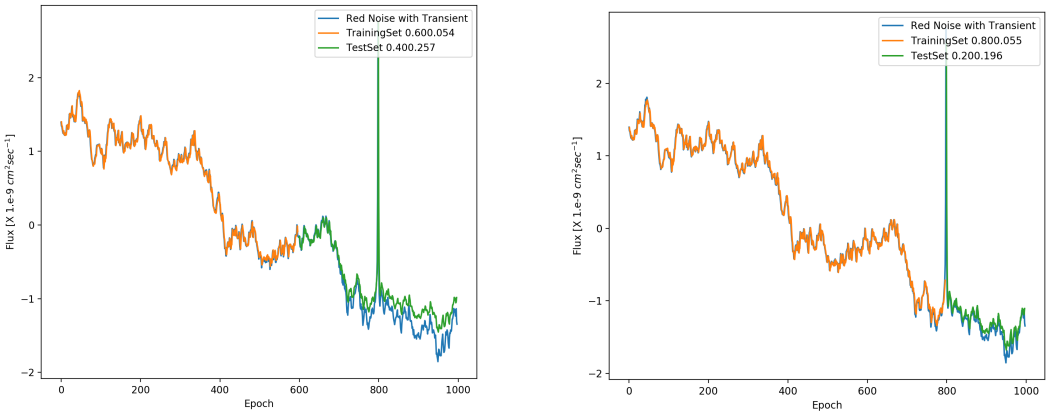

Figure 12: Same as figure 11 but with red noise. The LSTM with batch size 30 performs rather poorly showing the effect of noise.

t0 at 80% along the lightcurve, which is an choice made to be close to the typical ratio of train to test data. And the amplitude is multiplied by 2 times the maximum flux value of the noise to make it a significant outburst.

The results for the transient detection is shown in the figures 11 and 12 for pink and red noise respectively. We find that for pink noise, the prediction for the same transient is far superior to that for the red noise case. This is what we would expect from the simulation tests done in the previous sections. It is clear that the LSTMs perform rather poorly in presence of red noise and would give an imprecise prediction, not only for the transient, but in fact for the entire forecast. Now in absolute terms, one would expect the stateless LSTMs to fail at predicting a sharp, significant transient in either scenario, due to lack of long term correlation. However, given that there is a finite time, τ, and given the normalisation, we do find that the LSTMs have decent predictive power as a batch size of $10 - 30$ derives correlation on these timescales. And the closer we are to the outburst, the better the performance giving a very impressive value of loss (test to train) ratio of ≈ 1.063 for an $80 : 20$ split for pink noise. This deteoriorates to 1.112 for a $60 : 40$ split.

6 Discussions and Conclusions

Sensitivity tests are critical for complex algorithms to test the stability of predictions and tendencies for certain types of datasets to yield certain types of results. With ever increasing volumes of time-domain data in astronomy and the growing importance of the forecasting element, exhaustive efforts are needed to test forecasting algorithms that are bound to be used. In this context, the RNN-LSTM models are likely to be extremely popular. Astronomical lightcurves, time-series generated by an astrophysical system are apriori expected to be less complex than those in other domains. However, the combination of interesting quasi-periodic and outbursts with interesting stochastic astrophysical signals and background noise, makes astronomical forecasting a non-trivial challenge. In several multiwavelength and now increasingly multimessenger observations, an episode of increased flux measured in one waveband by one telescope or observatory triggers other facilities in different wavebands to observe the same source position. This essentially comes from the idea that physical processes leaves imprints on different parts of the electromagnetic spectrum with time-lags. And coordinated or synchronous observations amongst other things aim at identifying and quantifying the driving physical processes in terms of observed lags. To facilitate these coordinated observations, precise, fast, and reliable forecasting is key, and here machine learning algorithms like the RNN LSTMs can be very useful. Therefore, it is worthwhile to test them against types of time-series that maybe encountered with simulated lightcurves.

In testing with simulated lightcurves, we perform two types of tests. The first is sensitivity to properties or characteristics of data. The second is sensitivity to elements of the forecasting machinery for different datasets. Under characteristics of data, we look at incompleteness in data, or gaps in lightcurves that we routinely encounter in astronomy observations. We simulate purely periodic and random gaps and vary the percentage of missing data points to compute the training and test losses as measures of performance. We find that red noise lightcurves are generally more affected by presence of both periodic and random gaps, than the pink noise. Another general outcome seems to be presence of a sweet spot for the fraction of gaps at around 20 or 30%, where the test to train losses tend to be closest to unity and thereby optimal. One possible implication could be that the gaps could, to an extent reduce the complexity and thereby improve

performance of a particular forecasting architecture. Another alternate scenario could be that as we move from pink to red noise i.e. from a power spectral density index of 1.0 to 2.0, the longer timescales are more significant than the shorter ones on which gaps are introduced. And this leads to an initial improvement with increase in gaps until, the extent of the gaps is large enough that eventually starts to hamper predictability. These two effects would be degenerate. A stronger conclusion would require a comparison with other machine learning algorithms or indeed other forecasting methods, which we leave for a future paper. The complexity measured in terms of no. of Fourier components has a more monotonic effect. As the number of Fourier components increase, the performance tends to deteriorate for both pink and red noise.

From the elements of the forecasting machinery or architecture, the most important one appears to be the batch size. Once again, there seems to be a sweet spot at a batch size of around 10 for performance in terms of both absolute values of losses and the ratio of test to train losses. The performance for a learning rate of 0.1 and train to test ratio of $0.80 : 0.20$ at first glance seems to be slightly better for red noise, with a hint of overfitting for pink noise with the test to train loss ratio dropping below 1.0 in some cases. This is shown in figure 9. However this is not significant. What is more significant though is that this ratio and hence the performance is more stable for pink noise. This is confirmed by the uncertainty calculation in figure 10 which shows a larger uncertainty for red noise. A possible, even likely reason for this is that for stateless mode in which we are operating the LSTMs, it does not account for correlations over longer timescales than within a batch. With greater power at longer timescales in red noise compared to pink noise, the LSTMs should have a worse performance. This should also imply that if stateful mode is used then the performance on the red noise lightcurves should improve more than that for the pink noise. Once again, there is a need to compare with other forecasting models which is beyond the scope of this paper and will be addressed in a future one.

The results show that in general, that while RNN LSTMs are powerful for forecasting time-series for astronomy, there's enough complexity and attention needs to be paid to selection of architecture parameters for different types of noise. Time-series simulations can provide an excellent way of testing sensitivities to facilitate such selection and possibly pre-selection for forecasting algorithms to be deployed in actual campaigns. Clearly, the power spectral density of lightcurves from past long-term campaigns are an incredibly useful guide in construction of forecasting machinery. Multiwavelength lightcurves will need multivariate forecasting which will be the topic of a future paper. Finally, combining time-series simulations as generative models for feeding data to deep learning algorithms like RNN-LSTMs can provide a way of not only improving learning from specific datasets but also about the algorithms themselves.

References

[Vaughan et al.(2016)] Vaughan, S., Uttley, P., Markowitz, A. G., et al. 2016, MNRAS, 461, 3145

[Rieger(2019)] Rieger, F. 2019, Galaxies, 7, 28

[Baring & Boettcher(2019)] Baring, M., & Boettcher, M. 2019, AAS/High Energy Astrophysics Division, 17, 106.14

[Sadeh(2019)] Sadeh, I. 2019, arXiv:1902.03620

[Ait Benkhali et al.(2019)] Ait Benkhali, F., Hofmann, W., Rieger, F. M., & Chakraborty, N. 2019, arXiv:1901.10246

[Graham et al.(2019)] Graham, M. J., Kulkarni, S. R., Bellm, E. C., et al. 2019, PASP, 131, 078001

[Ivezić et al.(2019)] Ivezić, Ž., Kahn, S. M., Tyson, J. A., et al. 2019, ApJ, 873, 111

[Timmer & Koenig(1995)] Timmer, J., & Koenig, M. 1995, AAP, 300, 707

[Giles et al.(2001)] Lee Giles, C and Lawrence, Steve. (2001). Machine Learning, 44, 161-183

[Andrysiak et al.(2018)] Andrysiak, Tomasz, Saganowski, Łukasz, Maszewski, Miroslaw. (2018) Time Series Forecasting Using Holt-Winters Model Applied to Anomaly Detection in Network Traffic 567-576.

[Namini et al., 2018] Siami Namini, Sima, Tavakoli, Neda, Siami Namin, Akbar. (2018). A Comparison of ARIMA and LSTM in Forecasting Time Series. 1394-1401. 10.1109/ICMLA.2018.00227.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S.; Schmidhuber, J. (1997). "Long Short-Term Memory". Neural Computation. 9 (8): 1735–1780. doi:10.1162/neco.1997.9.8.1735. PMID 9377276.

[Brownlee 2017] Jason Brownlee, Deep Learning for Time Series Forecasting, 2017

[Han et al.,2018] Jae Hyuk Han, Comparing Models for Time Series Analysis, Ph.D. Thesis, University of Pennsylvania