УДК 004.032.26

ИССЛЕДОВАНИЕ АЛГОРИТМОВ ОБУЧЕНИЯ НЕЙРОННЫХ СЕТЕЙ ДЛЯ ПРОГНОЗИРОВАНИЯ ХАОТИЧЕСКИХ ВРЕМЕННЫХ РЯДОВ

М. Ю. Лазутов, Д. А. Савельев

В данной работе была проведена оценка качества работы двух моделей нейронных сетей (RNN и LSTM) в контексте задачи прогнозирования хаотических временных рядов с использованием следующих алгоритмов минимизации функции потерь: Adam, Nadam, SGD, Levenberg–Marquardt. В качестве источника генерации хаотических временных рядов была взята система Лоренца при параметрах σ = 10, ρ = 28, β = 8/3. Обучение и настройка сетей проводилась с использованием языка программирования Python, в котором применялись библиотеки для работы с нейронными сетями: keras, numpy, matplotlib, plotly. Для работы с хаотическими временными рядами была использована библиотека nolista. На этапе подготовки данных была использована теорема Такенса о реконструкции фазового пространства для построения матрицы из временных рядов с определенной задержкой. Было показано, что с помощью выбранных моделей нейронных сетей можно уменьшить среднюю абсолютную ошибку в задаче прогнозирования до 0,000896. Хорошо показали себя алгоритмы минимизации функции потерь Levenberg–Marquardt и Adam.

Ключевые слова: LSTM; RNN; Adam; Nadam; SGD; Levenberg–Marquardt.

Необходимо начать с разъяснения понятия хаотического временного ряда (ХВР). Известно, что скалярный временной ряд – массив из n чисел $\{x_i\}$ $(i = \overline{1,n})$, представляющих собой значения некоторой измеренной (наблюдаемой) динамической переменной x(t) с некоторым постоянным шагом τ по времени, $t_i = t_0 + (\mathrm{i} - 1)\tau$: $x_i = x(t_i)$, $i = \overline{1,n}$. Тогда можно определить ХВР как скалярный временной ряд, который был порожден хаотической динамической системой. Введем формальное определение хаотической системы, которое представлено, например, в [1]. То есть, подвергнем ось времени дискретизации, тогда динамическая система может быть записана в виде: $x_{n+1} = f(x_n)$, (1)
где f: M → M. M – метрическое пространство.

Отметим, что последовательность вида $x_1 = f(x_0)$, $x_2 = f\big(f(x_0)\big) = f^2(x_0), \ldots$, т.е. $\{x_k\}$, где $k = \overline{0,\infty}$, называется траекторией системы (отображения).

Тогда f будет являться хаотическим отображением, если соблюдены условия:

1) f неустойчиво к начальным условиям, иными словами:

$$\exists\delta > 0 : \forall x \in M,\ \varepsilon > 0\ \exists\ \mathrm{n} \in N, \mathrm{y} \in M: \mathrm{d}(\mathrm{x},\mathrm{y}) < \varepsilon\ \&\ d\big(f^n(x), f^n(y)\big) > \delta;$$

2) f топологически транзитивно, иными словами, $\exists x_0 : \{f^n(x_0) | \ n \in N\}$ – плотное множество в M;

3) периодические траектории системы плотны в M. То есть в любой окрестности любой точки из M найдется хотя бы одна периодическая траектория.

Из первого пункта приведённого выше определения можно сделать вывод, что предсказать поведение ХВР крайне трудно из-за неочевидного поведения траекторий. Ниже будут приведены графики некоторых ХВР для наглядного подтверждения этого факта.

Одна из самых известных хаотических систем – это система Лоренца:

$$\begin{cases} \dot{x} = \sigma(y - x) \\ \dot{y} = x(\rho - z) - y, \\ \dot{z} = xy - \beta z \end{cases} \quad (2)$$

где σ, ρ, β – наперед заданные параметры.

Лазутов Марк Юрьевич (lazutov.mark@mail.ru), студент III курса Института информатики и кибернетики;
Савельев Дмитрий Андреевич (dmitrey.savelyev@yandex.ru),
доцент кафедры технической кибернетики Самарского университета,
443086, Россия, г. Самара, Московское шоссе, 34.

В текущей работе эта система будет далее использоваться для генерации ХВР.

ХВР возникают во многих областях человеческой деятельности [2–7]. В частности, к таковым относятся: инженерное дело, финансовые рынки, управление энергией. Приведем конкретные примеры.

В инженерном деле хаотическое поведение можно наблюдать в системах управления, электронных схемах и механических системах. В [4] авторами наблюдалось хаотическое поведение на RL-diode схеме, где хаотическим поведением обладало напряжение на резисторе при определенных значениях входного напряжения и индуктивности катушки.

Далее, в области финансовых рынков накопилось достаточно эмпирических исследований, которые показали, что стохастические процессы не всегда описывают динамику финансовых рядов [5]. То есть, изменения цен не всегда являются случайным. Например, в работе [6] рассматривалась цена открытия фондового индекса S&P500 в период с 2014 по 2015 год. По методу вычисления максимальной экспоненты Ляпунова исследователями было установлено, что цена обладает слабо выраженным хаотическим поведением.

Затем, можно привести в пример работу [7] из области управления энергией, в которой авторы оценивали нагрузку на электросеть. Может показаться, что хаотическая динамика в подобных измерениях отсутствует из-за явно выраженной сезонности, однако это не так. Хаотическая динамика обеспечивается в присутствии других факторов (погода, экономическая ситуация и т.д.).

Резюмируя вышесказанное, можно отметить, что прогнозировать ХВР важно, во-первых, для отслеживания экстремальных событий, поскольку хаотическая динамика чаще всего нежелательна из-за риска возникновения сбоев в оборудовании. Соответственно, прогнозирование хаотических временных рядов может помочь определить, когда эти события могут произойти, что позволит лучше подготовиться и смягчить последствия. И, во-вторых, прогнозирование ХВР способствует повышению производительности и снижению затрат. Очевидно, что, зная дальнейшее поведение сложной нелинейной системы, можно корректировать использование ресурсов таким образом, чтобы получать больше пользы.

Стоит отметить, что подходы к прогнозированию ХВР достаточно разнообразны, однако в большинстве случаев разрабатываются и исследуются нейронные сети, так как они позволяют получить наиболее точные результаты прогноза [8]. В частности, для прогнозирования хаотических временных рядов известны такие методы как свёрточные нейронные сети (CNN) [9], вейвлет-нейронные сети (WNN) [10], нечёткие нейронные сети (FNN) [11] и сети долгой краткосрочной памяти (LSTM) [12].

Итак, в рамках этой работы задача прогнозирования ХВР была решена с использованием языка программирования Python и его главных библиотек для обучения нейронных сетей: tensorflow, keras, numpy. Для оценки топологических параметров нелинейных временных рядов была использована библиотека nolista. Построение графиков осуществлялось с помощью библиотек plotly и matplotlib.

Цель работы – обучить нейронные сети с помощью разных методов поиска минимума функции потерь, затем оценить получившиеся результаты.

Условия и методы исследования

В данной работе была поставлена следующая задача: имеется ХВР $\{x(1), x(2), \dots, x(n)\}$, где x(t) суть x-координата системы Лоренца при параметрах $\sigma = 10$, $\rho = 28$, $\beta = 8/3$, необходимо вычислить $\{x(n+1), x(n+2), \dots\}$, используя нейронные сети.

Приведем план решения задачи после генерации ХВР.

1) Проведем нормализацию данных. Заменим все x на $2\left(\frac{x-x_{min}}{x_{max}-x_{min}}\right)-1$, где x_{max} и x_{min} – максимальное и минимальное значения на исходной выборке соответственно. Очевидно, что таким образом все x окажутся в сегменте $[-1, 1]$. Такая нормализация способствует ускорению сходимости функции потерь.

2) Проведем трансформацию временного ряда в матрицу для обучения сетей. Отметим, что в поставленной задаче крайне плохо себя показывает трансформация по методу скользящего окна из-за хаотической ди-

намики. В связи с этим учтём теорему Такенса о реконструкции динамической системы [13], тогда временной ряд будет преобразован таким образом:

$$\begin{matrix} x(t) & x(t+d) & \cdots & x(t+(m-1)d) \\ x(t+1) & x(t+1+d) & \cdots & x(t+1+(m-1)d), \\ \vdots & \vdots & \cdots & \vdots \end{matrix} \quad (3)$$

где d – оптимальный временной лаг, а m – оптимальная размерность пространства вложения. Для вычисления временного лага можно построить график функции взаимной информации и определить, где расположен первый минимум. Выбранная таким образом точка минимума будет совпадать с искомым временным лагом. Оптимальную размерность пространства вложения можно вычислить, например, по методу ложных ближайших соседей. Очевидно, что в получившейся матрице последний столбец будет взят в качестве целевых значений при обучении сетей.

3) Зададим гиперпараметры, затем обучим RNN и LSTM, используя широко известные алгоритмы минимизации критерия качества: Adam, Nadam, стохастический градиентный спуск (SGD), Levenberg–Marquardt. RNN и LSTM выбраны по той причине, что они отлично себя показывают в предсказывании последовательностей в силу способности запоминать предыдущие результаты [14].

Прокомментируем настройку гиперпараметров нейронных сетей. Ниже, в таблице 1, представлены выбранные значения для некоторых из них. Отметим, что SGD используется с поправкой Нестерова и «моментом» (momentum), чтобы немного ускорить сходимость. В структуре обеих сетей во входном слое количество нейронов равно $m-1$, а в выходном слое расположен один нейрон. В структуре RNN сети задан лишь один скрытый слой из 16 нейронов. В структуре LSTM сети заданы два скрытых слоя из 64 нейронов. В обеих сетях в выходном слое задана линейная функция активации, а в скрытых слоях задан гиперболический тангенс. Отметим, что по методу ложных ближайших соседей определяется лишь левая граница значения размерности пространства вложения, которая получилась равной 3 (рисунок 1). Однако допустимо взять близкое значение, например, равное 5, так как это может поспособствовать улучшению качества прогноза. Также далее приведен график получившейся функции взаимной информации (рисунок 2).

Результаты и их обсуждение

Как было указано выше, минимизируемая функция во всех случаях – это среднеквадратическая ошибка. Однако для простоты понимания качества прогноза была введена метрика – средняя абсолютная ошибка. Рассмотрим полученные MAE и коэффициенты детерминации (R^2) после обучения выбранных нейронных сетей (таблица 2).

Таблица 1

Гиперапараметры нейронных сетей

Гиперпараметр	**Значение**
Максимальное количество эпох	300
Количество батчей	32
Обучающее множество	17640
Множество валидации	1960
Тестовое множество	1000
Настройка SGD	Исходная скорость обучения = 0,15; momentum = 0,7
Настройка Adam	Исходная скорость обучения = 0,001
Настройка Nadam	Исходная скорость обучения = 0,001
Настройка Levenberg-Marquardt	Исходная скорость обучения = 0,1
Функция потерь/Метрика	Среднеквадратическая ошибка (MSE)/Средняя абсолютная ошибка (MAE)
Временной лаг	16
Размерность пространства вложения	5

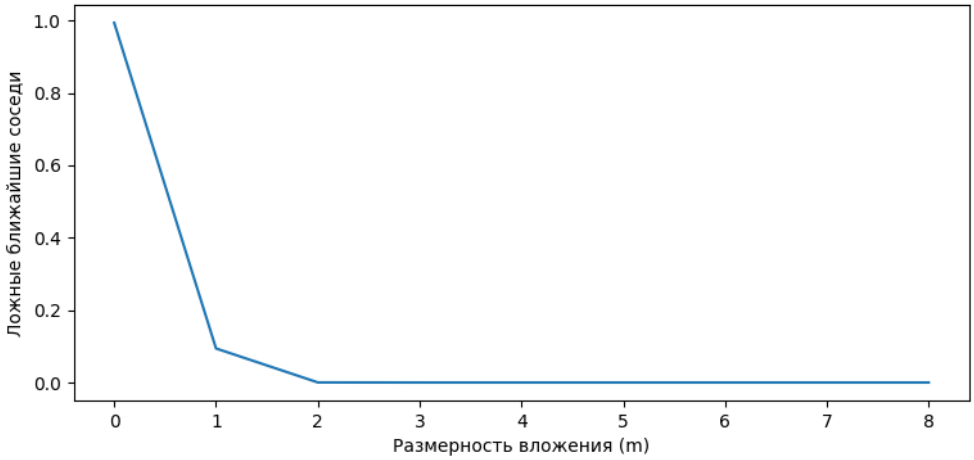

Рис. 1. Результат применения метода ложных ближайших соседей

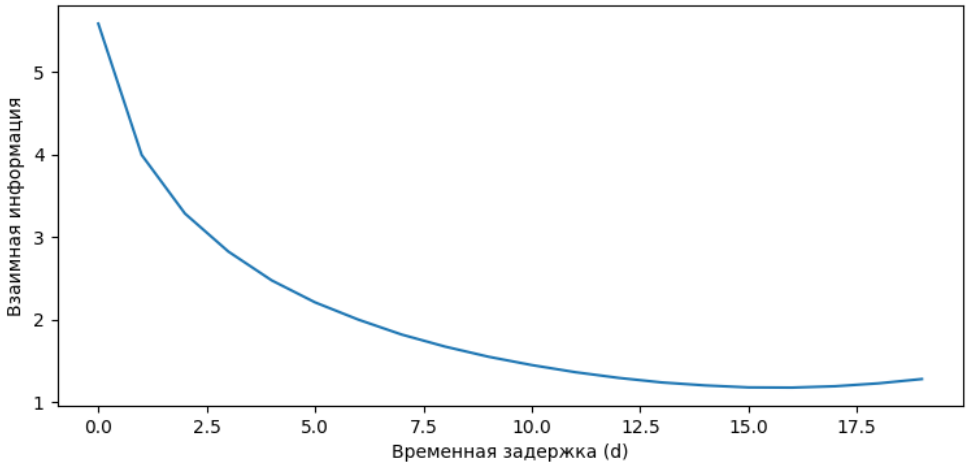

Рис. 2. График функции взаимной информации

Таблица 2

Результаты обучения

RNN			
Алгоритм	**MAE на тестовом множестве**	R^2	**Эпохи**
Adam	0,0035	0,99983	300
Nadam	0,0024	0,99992	300
SGD	0,0023	0,99992	215
Levenberg–Marquardt	0,0014	0,99996	20
LSTM			
Алгоритм	**MAE на тестовом множестве**	R^2	**Эпохи**
Adam	0,0008958	0,99999	300
Nadam	0,0016	0,99997	300
SGD	0,0053	0,99957	240
Levenberg–Marquardt	0,0015	0,99994	25

Не трудно обозначить лидирующие результаты. В случае RNN это Levenberg–Marquardt с $MAE = 0{,}0014$ и $R^2 = 0{,}99996$, а в случае LSTM это Adam с $MAE = 0{,}0008958$ и $R^2 = 0{,}99999$. В столбце «Эпохи» в случае SGD и Levenberg–Marquardt указаны такие номера эпох, начиная с которых среднеквадратическая ошибка на обучающем множестве стабилизировалась. В случае с Levenberg–Marquardt очевидно, что не требуется большого количества эпох для достижения ста-

бильного значения функции потерь, поскольку в этом методе используются производные второго порядка, которые значительно ускоряют поиск минимума. Однако из-за этого время минимизации по Levenberg–Marquardt гораздо дольше, чем у остальных алгоритмов. В случае с SGD, как было указано выше, ускорить сходимость помогает поправка Нестерова вместе с «моментом».

Оценим качество прогноза на рисунках 3–5. На рисунке 3 изображен временной ряд целиком, зеленым цветом отмечены действительные значения, красным – спрогнозированные значения с помощью LSTM сети, обученной по алгоритму Adam. На рисунке 4 отмечены все спрогнозированные значения в масштабированном виде. Можно заметить, что спрогнозированные значения практически точно накладываются на истинные, поскольку за «красной» частью на графике почти не видны «зеленые» значения. На рисунке 5 можно сравнить истинное значение со спрогнозированным в точке с номером 19268. Абсолютная ошибка в этой точке примерно равна 0,007. То есть, в третьем знаке наступает существенная разница между значениями.

Заключение

Таким образом, можно сделать вывод, что с помощью выбранных нейронных сетей (RNN и LSTM) в поставленной задаче прогнозирования можно достичь хороших результатов: $MAE \approx 0.000896$ на 1000 точках в тестовом множестве. Наиболее хорошо показали себя алгоритмы: Levenberg–Marquardt с $MAE = 0{,}0014$ на RNN сети и Adam с $MAE \approx 0{,}000896$ на LSTM сети. Заметим, что даже простая RNN сеть из 16 нейронов в скрытом слое способна демонстрировать приемлемое качество прогноза с большим горизонтом прогнозирования. Кроме того, использованная громоздкая LSTM сеть не сильно улучшает результат RNN сети. В связи с этим можно попробовать подобрать такие архитектуры нейронных сетей, с помощью которых получится улучшить результат на несколько порядков.

Литература

1. Лоскутов А. Ю. Очарование хаоса // УФН. 2010. Т. 180. № 12. С. 1305–1329.

2. Sangiorgio M., Dercole F. Robustness of LSTM neural networks for multi-step forecasting of chaotic time series // Chaos, Solitons & Fractals. 2020. Vol. 139. P. 110045.

3. Lara–Benítez P., Carranza–García M., Riquelme J. C. An experimental review on deep learning architectures for time series forecasting // International Journal of Neural Systems. 2021. Vol. 31. № 4. P. 2130001.

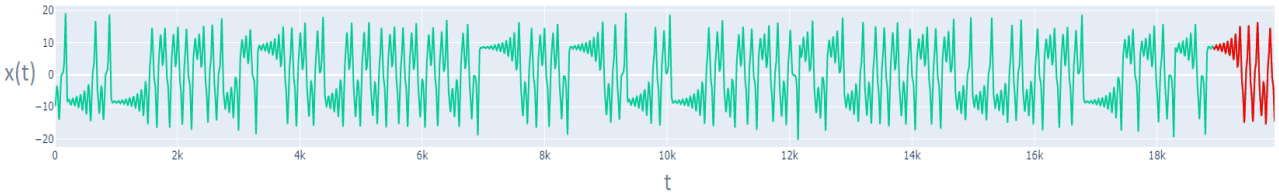

Рис. 3. X-координата системы Лоренца

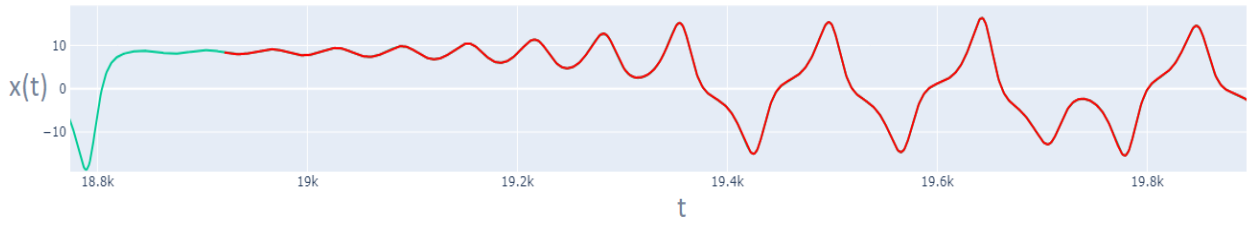

Рис. 4. Масштабированное изображение спрогнозированных значений

Рис. 5. Максимальное приближение в произвольно выбранной точке

4. Hanias M. P., Karras D. A. Improved Multistep Nonlinear Time Series Prediction by applying Deterministic Chaos and Neural Network Techniques in Diode Resonator Circuits // IEEE International Symposium on Intelligent Signal Processing, Alcala de Henares, Spain. 2007. P. 1–6.

5. Faggini M., Bruno B., Parziale A. Does Chaos Matter in Financial Time Series Analysis? // International Journal of Economics and Financial Issues. 2019. Vol. 9. № 4. P. 18–24.

6. Xiaoxiang G., Yutong S., Jingli R. Low dimensional mid-term chaotic time series prediction by delay parameterized method // Information Sciences. 2020. Vol. 516. P. 1–19.

7. Short-term power load forecasting using integrated methods based on long short-term memory / W. Zhang, J. Qin, F. Mei [et al.] // Science China Technological Sciences. 2020. Vol. 63. P. 614–624.

8. Ramadevi B. Chaotic Time Series Forecasting Approaches Using Machine Learning Techniques: A Review // Symmetry. 2022. Vol. 14. № 5. P. 955.

9. Flight delay prediction using deep convolutional neural network based on fusion of meteorological data / J. Qu, T. Zhao, M. Ye [et al.] // Neural Processing Letters. 2020. Vol. 52. P. 1461–1484.

10. Zhou B., Shi A. Application of wavelet neural network for chaos time series prediction // 5th International Conference on Intelligent Human-Machine Systems and Cybernetics. 2013. Vol. 1. P. 259–262.

11. Goudarzi S., Khodabakhshi M. B., Moradi M. H. Interactively recurrent fuzzy functions with multi objective learning and its application to chaotic time series prediction // Journal of Intelligent and Fuzzy Systems. 2016. Vol. 30. P. 1157–1168.

12. Pathan R. K., Biswas M., Khandaker M. U. Time series prediction of COVID-19 by mutation rate analysis using recurrent neural network-based LSTM model // Chaos Solitons Fractals. 2020. Vol. 138. P. 110018.

13. Takens F. Detecting strange attractors in turbulence // Lecture Notes in Mathematics, Springer–Verlag, Berlin. 1980. Vol. 898. P. 366–381.

14. Shahi S., Fenton H., Cherry M. Prediction of chaotic time series using recurrent neural networks and reservoir computing techniques: A comparative study // Machine Learning with Applications. 2022. Vol. 8. P. 100300.

A STUDY OF NEURAL NETWORK TRAINING ALGORITHMS FOR FORECASTING CHAOTIC TIME SERIES

M. Yu. Lazutov, D. A. Savelyev

In this work we have evaluated the performance of two neural network models (RNN and LSTM) in the context of chaotic time series prediction using various widely used loss function minimization algorithms, namely: Adam, Nadam, SGD, and Levenberg–Marquardt. The Lorenz system was taken as the source of chaotic time series generation with the parameters $\sigma = 10$, $\rho = 28$, $\beta = 8/3$. The networks were trained and configured using the Python programming language, which used the main libraries for working with neural networks: keras, numpy, matplotlib, plotly. To work with chaotic time series the library nolista was used. At the stage of data preparation the Takens theorem of phase space reconstruction was used to construct a matrix from time series with a certain delay. It was shown that with the using selected neural network models, it is possible to reduce the MAE in the forecasting problem to 0,000896. The Levenberg–Marquardt and Adam loss function minimization algorithms have shown themselves well.

Key words: LSTM; RNN; Adam; Nadam; SGD; Levenberg–Marquardt.

Статья поступила в редакцию 25.05.2023 г.

Lazutov Mark Yurievich (lazutov.mark@mail.ru), IIIrd year student of the Institute of Informatics and Cybernetics;
Savelyev Dmitry Andreevich (dmitrey.savelyev@yandex.ru),
associate professor of the Department of Technical Cybernetics of Samara University,
443086, Russia, Samara, Moskovskoye shosse, 34.