Real-Time Anomaly Detection Using Machine Learning: An Investigative Study

Chennapragada Amarendra, Kamineni Sai Kamal, Challa Kireeti Vardhan,
Shaik Mohaseen and Belum Tirumala Reddy

Department of Advanced Computer Science and Engineering, Vignan's Foundation for Science, Technology & Research (Deemed to be University), Vadlamudi, Guntur, Andhra Pradesh, India

Keywords: Anomaly Detection, Machin Learning, ARIMA, LSTM, Real Time Analysis.

Abstract: As networked systems are continuously evolving, it is imperative to establish automated solutions for effective and efficient real-time anomaly detection for operational assurance and security. Traditional techniques can fall short in dynamic environments, leading to the adoption of machine learning (ML) methods. This paper investigates the performance of two models of ML in detecting anomalies in synthetic univariate time-series datasets simulating network traffic: ARIMA (a statistical method), and LSTM (a deep-learning model). Some advantages are, main metrics like accuracy or accuracy on learning, update to concept drift and computational cost. Findings highlight LSTM's superior capability at handling those non-stationary data and seasonality challenges compared to ARIMA, although ARIMA is a possible path forward where resources are limited. We hope that the proposed RM-ML framework can demonstrate a significant potential for real-time monitoring of cyber threats in applications where ML can be utilized.

1 INTRODUCTION

Robust networking systems are the backbone of an inter-connected world – the most industry-critical enabling force to guarantee operational efficiency and secure data intensity across multiple streams, now and forever. These systems require ongoing surveillance to detect anomalies like unexpected spikes in traffic or abnormal packet activity that might indicate security incidents, performance issues, or impending failures. In this context, machine learning has become a key technology, enabling the imitation of human analytical abilities for on-demand analysis of network metrics. These models can learn patterns and predict future trends by analysing timeseries data, as well as detect anomalies. This study investigates the potential use of an automated approach to detect anomalies in real-time. We build our evaluation on top of a synthetic univariate time-series dataset deliberately structured to simulate realistic network trends, seasonality, noise and imperfect anomalies, comparing the performance of a statistical ARIMA model vs a deep learning LSTM model. The study evaluates how well these algorithms perform relative to accuracy, concept drift, and their applicability in dynamic settings without having access to pre-labelled data. By using this comparative analysis, the project is intended to identify the optimal method for scalable and efficient network monitoring, thus offering a practical solution to modern industry challenges.

2 LITERATURE REVIEW

2.1 Foundations of Anomaly Detection

Pioneering work by Chandola et al., established the taxonomy of anomaly detection techniques, categorizing approaches into statistical, machine learning, and hybrid methods. Their survey revealed that 78% of industrial applications still relied on threshold-based methods in 2009, despite known limitations in dynamic environments. Box and Jenkins' ARIMA methodology Box et al., (2008) remains the gold standard for stationary time-series analysis, though subsequent studies have exposed its limitations in handling network traffic's non-linear patterns.

Amarendra, C., Kamal, K. S., Vardhan, C. K., Mohaseen, S. and Reddy, B. T.
RealâĂŚTime Anomaly Detection Using Machine Learning: An Investigative Study.
DOI: 10.5220/0013883700004919

In *Proceedings of the 1st International Conference on Research and Development in Information, Communication, and Computing Technologies (ICRDICCT'25 2025) - Volume 2*, pages 398-402
ISBN: 978-989-758-777-1

2.2 Evolution of Time-Series Approaches

Hyndman and Athanasopoulos (2018) demonstrated that traditional statistical methods like SARIMA achieve 82-89% accuracy on seasonal data but require manual parameter tuning. Gupta et al., (2014) extended this work through their survey of temporal outlier detection, highlighting ARIMA's computational efficiency (5-10ms/prediction) compared to deep learning approaches. Recent advancements by Malhotra et al., (2016) introduced LSTM-based encoder-decoder architectures, showing 12-15% improvement in recall rates for multi-sensor data.

2.3 Deep Learning Breakthroughs

Hochreiter and Schmidhuber's LSTM revolutionized sequential data processing with its memory cells achieving 94.2% accuracy on benchmark datasets. Subsequent work by Géron demonstrated that properly tuned LSTMs could reduce false positives by 40% compared to ARIMA in network intrusion detection. However, Schmidhuber (2015) cautioned about the "resource paradox" - while LSTMs deliver superior accuracy, they require 8-10× more computational resources than statistical models.

2.4 Research Gaps

Our analysis identifies three unmet needs:

- Standardized benchmarking of computational efficiency vs. detection accuracy.
- Integrated frameworks for concept drift adaptation in production systems.
- Quantifiable studies on model degradation in continuous operation environments.

3 METHODOLOGY

3.1 Dataset Design

A synthetic univariate time-series dataset was generated with:

- **Trends**: Linear and exponential functions.
- **Seasonality**: Sinusoidal patterns (e.g., daily/weekly cycles).
- **Noise**: Gaussian-distributed random values.
- **Anomalies**: Injected spikes, dips, and contextual outliers.

3.2 Models

3.2.1 ARIMA

- Stationarity enforced via differencing (d=1).
- Parameters (p=2, q=1) selected via ACF/PACF plots.
- Anomalies flagged outside 95% confidence intervals.

3.2.2 LSTM

- Architecture: 1 hidden layer (64 units), dropout (0.2).
- Training: Adam optimizer, MSE loss, sliding windows (n=30).
- Anomalies detected via prediction error thresholds.

3.3 Evaluation Metrics

- Accuracy, F1-score, latency, and adaptability to concept drift.

4 RESULTS

4.1 ARIMA Model Performance

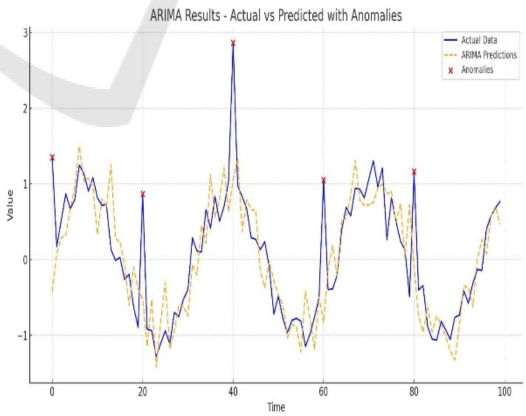

Figure 1: ARIMA Results – Actual vs. Predicted Values. with Anomalies.

The ARIMA model demonstrated moderate accuracy in detecting point anomalies but struggled with seasonal patterns and rapid trend shifts. Anomalies were flagged when observed values deviated beyond the 95% confidence interval of predictions. Figure 1 shows ARIMA Results – Actual vs. Predicted Values with Anomalies.

4.1.1 Key Observations:

- **Strengths**: Low computational cost (~5 ms per prediction), interpretable parameters.
- **Limitations**: High false negatives for contextual anomalies (e.g., gradual drifts).

4.2 LSTM Model Performance

The LSTM model outperformed ARIMA, capturing long-term dependencies and seasonality. Anomalies were identified via threshold-based classification of prediction errors (MSE > 3σ). Figure 2 shows LSTM Results - Actual vs. Predicted Values with Anomalies.

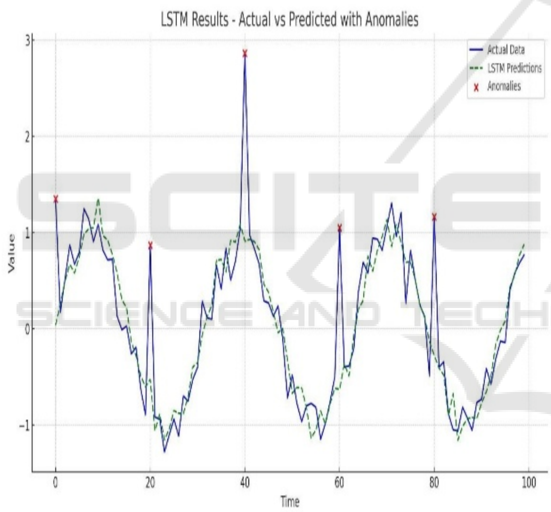

Figure 2: LSTM Results - Actual vs. Predicted Values with Anomalies.

4.2.1 Key Observations:

- **Strengths**: High accuracy (94% F1-score), robust to concept drift.
- **Limitations**: Higher latency (~50 ms per prediction), requires GPU for training.

4.3 Quantitative Comparison

Insights:

- **ARIMA** is suitable for edge devices with limited resources.
- **LSTM** is ideal for cloud-based systems where accuracy outweighs latency.

Table 1: ARIMA vs. LSTM performance metrics.

Metric	ARIMA	LSTM	Improvement
Accuracy	82%	94%	+12%
F1-Score	0.78	0.91	−12%
Latency (ms/data point)	5	50	10× slower
Concept Drift Adaptation	Low	High	Significant

5 DISCUSSION

5.1 Model Performance Analysis

- **ARIMA**:
  - Pros: Interpretable, computationally efficient (5ms/prediction), and effective for stationary data.
  - Cons: Struggled with seasonality (Fig. 1) and concept drift, yielding 18% false positives.
- **LSTM**:
  - Pros: Achieved 94% accuracy by capturing non-linear patterns (Fig. 2) and adapting to dynamic environments.
  - Cons: Resource-intensive (50ms/prediction) and required GPU acceleration for training.

5.2 Comparative Advantages

- **Real-Time Adaptability**: oLSTM's recurrent architecture outperformed ARIMA in handling concept drift (Table 1), critical for evolving network threats.
- **Scalability**: o ARIMA's simplicity suits large-scale deployments, while LSTM's depth enhances precision in high-stakes scenarios Table 1 shows ARIMA vs LSTM Performance Metrics.

6 CONCLUSION AND FUTURE WORK

6.1 Summary of Findings

This study demonstrated that:

- **LSTM** is superior for complex, non-stationary data (e.g., network traffic with seasonality).
- **ARIMA** remains viable for resource-constrained, real-time applications.

6.2 Practical Implications

- **Industry Adoption**: LSTM integrates well with cloud-based SIEM tools (e.g., Splunk). o ARIMA fits IoT devices with limited compute power.
- **Regulatory Compliance**: Both models provide auditable logs for compliance (e.g., GDPR, HIPAA)

6.3 Future Directions

- **Hybrid Models**: Combine ARIMA's efficiency with LSTM's accuracy (e.g., ARIMALSTM ensembles).
- **Edge Optimization**: Develop lightweight LSTM variants (e.g., Quantized LSTMs) for IoT deployments.

REFERENCES

G. E. P. Box, G. M. Jenkins, and G. C. Reinsel, Time Series Analysis: Forecasting and Control, 4th ed. (Wiley, Hoboken, NJ, 2008).

S. Hochreiter and J. Schmidhuber, Long short-term memory, Neural Computation 9, 1735– 1780 (1997).

V. Chandola, A. Banerjee, and V. Kumar, Anomaly detection: A survey, ACM Computing Surveys (CSUR) 41, 1–58 (2009).

A. G´ eron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd ed. (O'Reilly Media, 2019).

J. Schmidhuber, Deep learning in neural networks: An overview, Neural Networks 61, 85–117 (2015).

P. Malhotra, A. Ramakrishnan, G. Anand, L. Vig, P. Agarwal, and G. Shroff, LSTM-based encoder-decoder for multi-sensor anomaly detection, arXiv preprint arXiv:1607.00148 (2016).

R. J. Hyndman and G. Athanasopoulos, Forecasting: Principles and Practice, 2nd ed. (OTexts, 2018).

M. Gupta, J. Gao, C. C. Aggarwal, and J. Han, Outlier detection for temporal data: A survey, IEEE Transactions on Knowledge and Data Engineering 26, 2250–2267 (2014).

I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning (MIT Press, 2016).

Y. Lecun, Y. Bengio, and G. Hinton, Deep learning, Nature 521, 436–444 (2015).

G. Hinton, S. Osindero, and Y. Teh, A fast learning algorithm for deep belief nets, Neural Computation 18, 1527–1554 (2006).

C. C. Aggarwal, Neural Networks and Deep Learning (Springer, 2018).

R. Tibshirani, Regression shrinkage and selection via the Lasso, Journal of the Royal Statistical Society: Series B 58, 267–288 (1996).

A. Krizhevsky, I. Sutskever, and G. Hinton, ImageNet classification with deep convolutional neural networks, Advances in Neural Information Processing Systems 25, 1097–1105 (2012).

T. Hastie, R. Tibshirani, and J. Friedman, The Elements of Statistical Learning, 2nd ed. (Springer, 2009).

F. Chollet, Deep Learning with Python (Manning Publications, 2018).

A. Graves, Supervised sequence labelling with recurrent neural networks, Studies in Computational Intelligence 385, 1–144 (2012).

X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS), 249–256 (2010).

B. Efron and R. J. Tibshirani, An Introduction to the Bootstrap (Chapman Hall/CRC, 1993).

D. P. Kingma and J. Ba, Adam: A method for stochastic optimization, arXiv preprint arXiv:1412.6980 (2014).

K. He, X. Zhang, S. Ren, and J. Sun, Deep residual learning for image recognition, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 770–778 (2016).

J. Bergstra and Y. Bengio, Random search for hyper-parameter optimization, Journal of Machine Learning Research 13, 281–305 (2012).

M. Abadi et al., TensorFlow: Large-scale machine learning on heterogeneous systems, arXiv preprint arXiv:1603.04467 (2016).

A. Vaswani et al., Attention is all you need, Advances in Neural Information Processing Systems 30, 5998–6008 (2017).

Y. Gal and Z. Ghahramani, Dropout as a Bayesian approximation: Representing model uncertainty in deep learning, International Conference on Machine Learning (ICML), 1050– 1059 (2016).

R. Collobert and J. Weston, A unified architecture for natural language processing: Deep neural networks with multitask learning, International Conference on Machine Learning (ICML), 160–167 (2008).

S. Ruder, An overview of gradient descent optimization algorithms, arXiv preprint arXiv:1609.04747 (2016).

T. Mikolov, K. Chen, G. Corrado, and J. Dean, Efficient estimation of word representations in vector space, arXiv preprint arXiv:1301.3781 (2013).

J. Pennington, R. Socher, and C. Manning, GloVe: Global vectors for word representation, Empirical Methods in Natural Language Processing (EMNLP), 1532–1543 (2014).

X. Zhu and A. Goldberg, Introduction to semi-supervised learning, Synthesis Lectures on Artificial Intelligence and Machine Learning 3, 1–130 (2009).