Faculty of Mathematics
and Computer Science

Machine learning course (ML)

Long short-term memory networks (LSTMs) used for anomaly detection

Jose Luis Galiano Gómez

Department of Computer Science, Babeş-Bolyai University
1, M. Kogalniceanu Street, 400084, Cluj-Napoca, Romania
E-mail: joseluisggjolu@gmail.com

Abstract

Anomalies are data points within a dataset that deviate significantly from the expected behavior. They are often rare and unusual instances that do not conform to the typical patterns or distributions found in the majority of the data. Anomalies can be caused by errors, outliers, rare events, or instances of interest, and the goal of anomaly detection is to identify and flag these unusual observations for further investigation, as they may indicate important insights, potential problems, or security threats. Long Short-Term Memory Networks (LSTMs) are highly effective in modeling sequential data, capturing total dependencies, and extracting relevant features within time-dependent datasets, making them a suitable choice for the task of anomaly detection. In this paper, a brief overview of LSTM networks is provided, beginning with their origin as an extension of Recurrent Neural Networks (RNNs), designed to solve the vanishing gradient problem, followed by the mathematical model and theory behind how it works, as well as why they effectively manage to capture both long or persistent and short-term or fleeting features within datasets, much like the human memory works. The main bulk of the paper consists of a survey of the different approaches to anomaly detection using LSTMs in recent years, including pure LSTM, encoder-decoder-based, and hybrid approaches, along with the newer fields of graph-based LSTMs and Transfer Learning.

Keywords: Artificial intelligence; Recurrent neural networks; Long short-term memory; Anomaly detection; Autoencoder; Seq2Seq; Graph-based neural networks; Transfer learning

1. Introduction

There is no doubt that neural networks are one of the most relevant and well-researched topics in the field of machine learning in recent years. What was once a novel and cutting-edge field is now maturing, with a wealth of well-understood methods and algorithms, as well as different variations and improvements on the original neural network design, adapting it to better solve a specific task. Such is the case of Long short-term memory networks.

LSTMs are an extension of Recurrent Neural Networks, which can process sequential data, allowing for an input stream rather than a single input at the beginning of the training. LSTMs take this concept and solve a problem present in the original design. Thus, they are particularly well-suited for tasks that involve sequences and have dependencies over a long range. One such task is the one commonly referred to as anomaly detection.

Anomaly detection is a rapidly growing and increasingly relevant field nowadays. We live in the internet and information era, where huge amounts of data flow in and out of different systems, data that must be processed and checked, data that is likely to be greatly relevant and have a heavy impact on the correct functioning of many systems and organizations, in which many people rely on. Several works have shown that this problem can be overcome by using LSTMs.

In this paper, a survey will be conducted on the most recent and promising LSTM-based approaches for anomaly detection. These approaches include those that only consist of the use of the original LSTM design with slight variations or improvements, as well as those that combine LSTMs with other tools in the field of machine learning to achieve a better result.

2. LSTM networks overview

2.1. RNNs: brief overview. The vanishing gradient problem

Text, audio, and video are all forms of sequential data. This sequential data is ubiquitous in this day and age, and as such, many problems arise in the field of machine learning where processing this type of data is required. RNNs are designed to deal with the temporal information of input data; their defining feature is the cyclic connection between neurons in the network. This allows RNNs to take into account both current data and past states when outputting the present state [22].

Recurrent networks use their feedback connections to store representations of recent input events in the form of activations. This is what is commonly referred to as ”short-term memory” [7]. When training the network, it must learn which past inputs have to be stored to produce the desired current output. This is usually done with gradient-based learning methods such as ”backpropagation through time” (BPTT) and ”real-time recurrent learning” (RTRL), where the error signal flows ”back in time” over the feedback connections to past inputs [6]. However, a problem arises; error signals flowing backward in time tend to either ”blow up” or ”vanish”, depending on the size of the weights [7]. The first case happens when the weights’ values make the signal grow exponentially up to ridiculous values. The second case is the contrary, where the signal becomes progressively fainter as it is propagated, down to negligible values almost equal to 0.

This is what we call the ”vanishing gradient problem”, its consequence being that when the gap between the relevant input data is large, the above RNNs are unable to connect the relevant information[22]. These are called ”long-term dependencies”, and are what give LSTMs its name since these were designed to solve this problem specifically, as we will soon see.

2.2. LSTMs: description and utility. The modeling of human memory

LSTMs were conceived as an extension of RNNs to solve the vanishing gradient problem previously introduced. This is achieved by an efficient, gradient-based algorithm for an architecture enforcing constant error flow through internal states of special units [7]. A brief description of said architecture will now be provided.

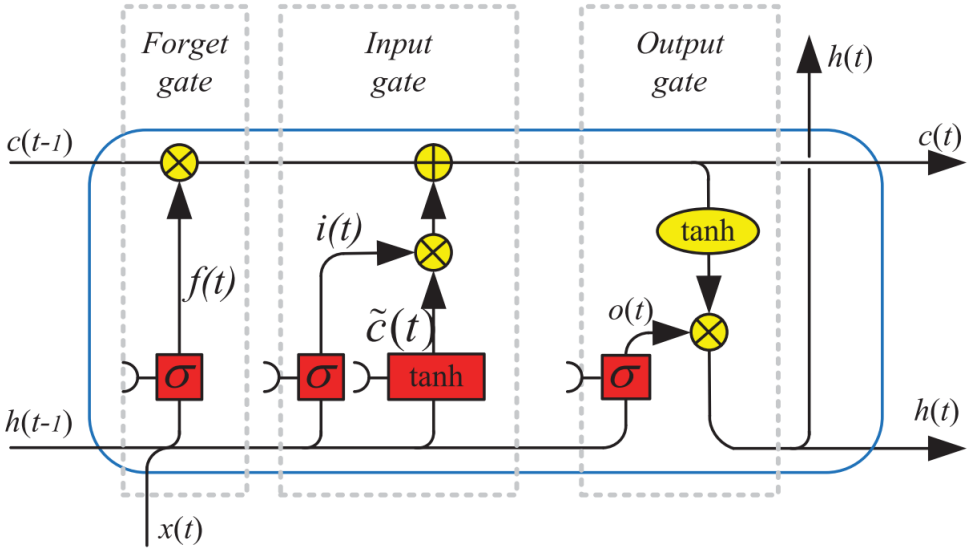

Fig. 1. LSTM architecture with forget, input and output gates [22]

LSTMs improve the remembering capacity of the standard recurrent cell by introducing ”gates” into the cell. Many variations of the LSTM cell exist, but the one usually denoted by the term is the one with three gates: input, output, and forget gate, as shown in Figure 1. That same architecture represented by the figure can be mathematically expressed as follows [22]:

$$
\begin{aligned}
f_t &= \sigma(W_{fh}h_{t-1} + W_{fx}x_t + b_f),\\
i_t &= \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i),\\
\bar{c}_t &= \tanh(W_{\bar{c}h}h_{t-1} + W_{\bar{c}x}x_t + b_{\bar{c}}),\\
c_t &= f_t \cdot c_{t-1} + i_t \cdot \bar{c}_t,\\
o_t &= \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o),\\
h_t &= o_t \cdot \tanh(c_t).
\end{aligned}
$$

where c_t is the current cell state, and W_i, $W_{\bar{c}}$, and W_o are the weights. We can see three flows of information. First, we have c, the cell state, which corresponds to the long-term memory. The lack of weights allows the long-term memory to flow through a series of units without causing the gradient to explode or vanish. Then we have h, the hidden state, corresponding to the short-term memory. Here, we do have weights connected to the short-term memories that can modify them, like in a standard recurrent network. Finally, we have x, the input.

Firstly, the forget gate receives the new input as well as the current states for both long and short-term memory and calculates a value between 0 and 1 using the input and short-term memory as shown in the mathematical expression for f_t. This number decides how much of the long-term memory is retained, thus the name ”forget gate”; if the value is 1, all the memory is retained, if it is 0, it is completely forgotten. In other words, it determines what percentage of the long-term memory is remembered. It has been found that the performance of the LSTM networks improves when increasing the bias of the forget gate, b_f [22].

The second stage is called the input gate. It calculates the values for both i_t, which combines the short-term memory and the input to generate a potential long-term memory, and $\bar{c}$, which determines what percentage of that potential memory is added to the cell state. Finally, we have the output gate, which updates the short-term memory. The new long-term memory generated by the input gate is used as input for the *tanh* activation function, generating a number

between 0 and 1 representing a potential short-term memory. This value is multiplied by o_t, which is also calculated based on the short-term memory and input.

To summarize, the forget gate decides which information to discard from the long-term memory, the input gate decides which information to add to the long-term memory, and the output gate controls the information that is output from the cell to the next layer in the network.

Although not explicitly stated in the original research papers introducing the architecture, it is often mentioned in educational materials and discussions that the LSTM network appears to be inspired by the way human memory works. While the analogy is not perfect, there are some conceptual similarities:

- Human memory is not an exhaustive recording of all experiences. Instead, attention and focus help determine which details are stored more permanently in memory. This selective attention is represented in the LSTM by the weights and biases in the forget and input gates, which selectively decide which information to remember and which to forget.
- Human memory is dynamic and subject to change based on new experiences. Learning involves modifying existing memories and forming new connections. In the same way, LSTMs adapt by updating their cells based on the patterns and dependencies observed in the data.
- Human memory is organized in a way that reflects the temporal order of events. Memory is indivisible from the concept of temporal dependency; we remember events in a temporal relation with other events. LSTMs are also inherently temporal since they are designed to process sequential data, where the order of the input information is essential to the dataset.

3. Anomaly Detection

3.1. Introduction and relevance

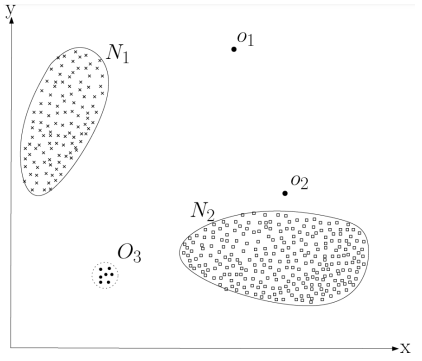

Fig. 2. Anomalies in a two-dimensional dataset[3]

”Anomaly detection” is the problem of detecting certain data points or finding patterns within a dataset that do not conform to expected behavior.

To provide a visual example, figure 2 shows a simple example of anomalies existing in a two-dimensional dataset. As we can observe, most data points fall inside one of two areas, N_1 or N_2. There are some data points, however, that do not, such as o_1 and o_2. These are commonly referred to as ”outliers” or ”anomalies” (in this context, both terms are interchangeable). O_3, however, is not a singular outlier but a region. This is what we meant by ”patterns not conforming to expected behavior”.

Anomalous dynamics almost always occur inadvertently, leading to instability and increased inefficiency and errors within a system [11]. As such, detecting them is of paramount importance when ensuring the correct functioning of said systems. Some applications of anomaly detection are fraud detection for credit cards, insurance, or health care, intrusion detection for cyber-security, fault detection in safety systems, and military surveillance for enemy activities [3].

3.2. Challenges present in anomaly detection. Types of anomalies

Anomaly detection is not an easy problem to solve. Some of the most prominent challenges of this task are the following:

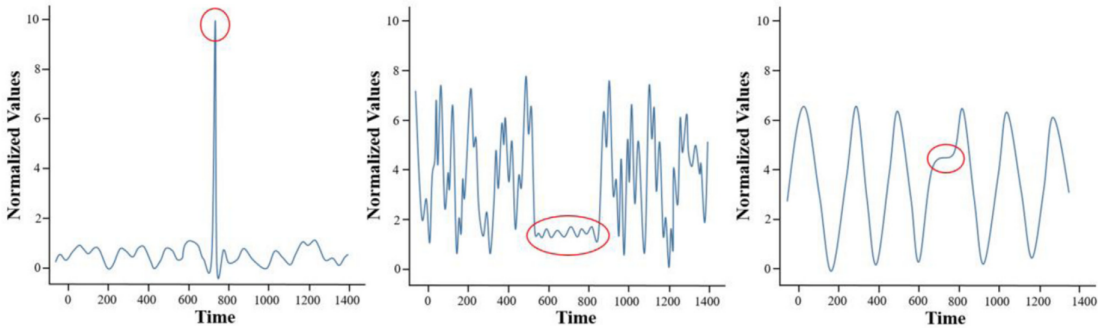

Fig. 3. Point anomaly, collective anomaly, and contextual anomaly, respectively [11]

- To detect anomalies the ideal scenario would be to define a region representing normal behavior, making everything outside of it an anomaly. Defining this region, however, is extremely challenging. The exact boundary between normal and abnormal is often not precise, and might even be dynamic [3].
- Anomalies are instances of abrupt new behaviors, structures, and distributions, remaining unknown until they occur, and are as such difficult to predict [16].
- The exact notion of an anomaly is subject to the specific application domain. As such, it is often difficult to generalize a solution for different applications [3].
- Anomalies appear in a minuscule proportion of the available data, and so finding sufficient amounts of labeled anomalous data instances for anomaly detection training is significantly more difficult than for normal training tasks. [16].

.

Furthermore, the specific challenges and complexities of the task will depend on the type(s) of anomaly present within the dataset. A distinction of three types of anomalies is presented in [8] (figure 3 presents a visualization of said anomaly types):

- ”Point anomalies” are individual data instances considered abnormal in relation to the surrounding data.
- ”Collective anomalies” are a group of data instances where each data point is considered normal, but the composition of the group indicates an irregularity [11].
- ”Contextual anomalies” refer to cases where a data point is unusual in a certain context, but not otherwise.

4. Approaches and examples

Now that we have a basic understanding of both the task at hand as well as the learning model we will use to approach it, we can start looking at different approaches and methods explored in the recent literature.

4.1. Basic LSTM approaches

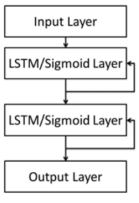

Fig. 4. Simple stacked LSTM architecture, used in [12].

In this section, we will take a look at some approaches that simply use the original LSTM structure, although in a novel way. Such is the case of the work presented in [12], where a stacked LSTM network is used for anomaly detection in time series. A network is trained on non-anomalous data and used as a predictor over a number of time steps. The resulting model is used to compute what is called an ”error vector” modeled to fit a multivariate Gaussian distribution. These are then used for assessing

the likelihood of anomalous behavior, evaluating the deviation of predicted output based on a variance analysis [11].

A very similar approach can be found in [20], a more recent paper in which a stacked LSTM architecture and error vectors are also used, but avoiding the need to have supervised anomaly information. This is achieved by building a multivariate normal error model for the data, which is then applied. Anomalies are detected through unusually high Mahalanobis distance values (MD takes into account the correlation in data since it is calculated using the inverse of the variance-covariance matrix of the dataset [4]).

Another interesting and recent work is that shown in [9], where a generic, real-time, ready-to-go, lightweight, and completely unsupervised approach is presented. The model employs two simple LSTMs to independently forecast each data point within the target time series by considering the data values recorded at the preceding *b* consecutive time points, this *b* value being called the "Look-Back parameter". Subsequently, it assesses whether the succeeding data point exhibits an anomaly or not. The two LSTMs are used as follows: one to detect single upcoming anomalous data points through the modeling of short-term features, and the other to detect collective anomalies based on long-term thresholds [11].

4.2. Encoder-decoder based approaches

In most cases acquired data is unlabeled, and as such unsupervised learning models are necessary. One such model is the Autoencoder neural network (AE), comprised of two sections; encoder and decoder. The encoder component aims to acquire a lower-dimensional representation of the input data, while the decoder component focuses on reconstructing these condensed features. Consequently, the AE undergoes training using data that mirrors typical system dynamics, enabling it to grasp the compression and reconstruction of such data. When the trained AE processes anomalous data, it leads to a reconstruction error. Leveraging the dynamics of this error can be employed to create an anomaly detection mechanism. The combination of LSTM and AE enables the acquisition of both short-term and long-term dependencies by learning lower-dimensional temporal features. This establishes a solid foundation for identifying intricate, time-varying anomalies. [11].

One example of the LSTM-AE architecture being used for anomaly detection is the one shown in [17]. The paper introduces a "Variational Autoencoder" (VAE) for multimodal anomaly detection, a variant of an AE able to model the underlying probability distribution of observations using variational inference [17]. The VAE is then combined with an LSTM by replacing the feed-forward network with an LSTM network. The method utilizes log-likelihood-based anomaly detection, determining a log-likelihood score for both real and reconstructed outputs [11].

A very recent LSTM-AE approach can be found in [21], where a new model is proposed; multiple LSTM networks collaborate to learn the long-term correlation of the data, and the autoencoder is used to establish the optimal threshold

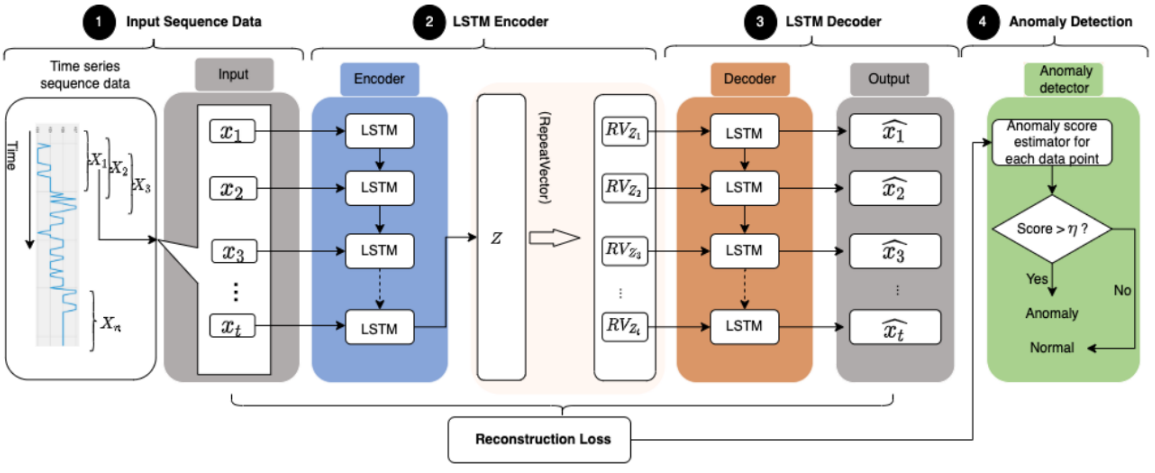

Fig. 5. Proposed LSTM-AE model in [21]

used for anomaly detection by assessing reconstruction error rates for each data point across all time-series sequences. A visual representation of this recent model is presented in Figure 5.

Another encoder-decoder-based model worth mentioning is the sequence-to-sequence (Seq2Seq) LSTM network. A first LSTM processes the input sequence timestep by timestep, generating a large fixed-dimensional vector representation. Subsequently, a second LSTM is employed to derive the output sequence from this vector. The second LSTM essentially functions as an RNN language model, with the distinction that it is conditioned on the input sequence [18]. This model can be used for anomaly detection as shown in [5], where anomalies are detected based on the cell states; unknown states and highly deviating copying vectors between the encoder and decoder layers are considered anomalies [11].

4.3. Hybrid approaches

The examples in this section demonstrate the combination of two distinct learning models, with one of them being an LSTM. This combination of models is employed to enhance performance beyond what can be achieved with LSTMs alone, benefitting from the advantages of both models while compensating for their shortcomings. Usually, one of the models is used to predict model dynamics and the second detects significant deviations from the actual process outcome [11].

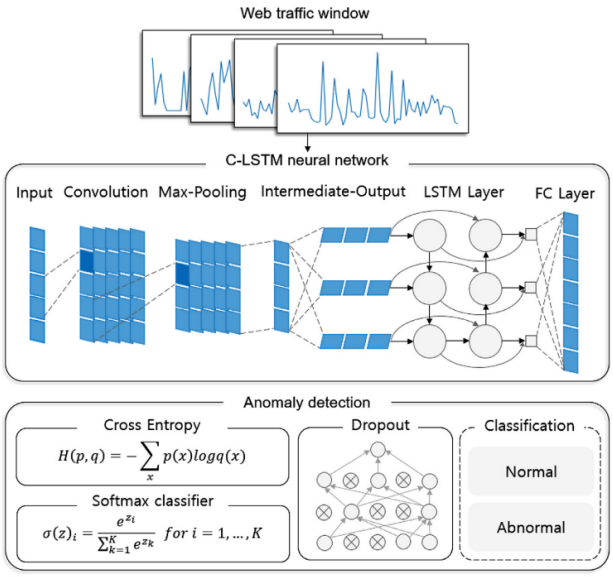

Fig. 6. C-LSTM architecture and softmax classifier for anomaly detection [8]

One such example is [8], where a combination of convoluted neural network (CNN) and LSTM layers is proposed (C-LSTM). The CNN layer, consisting of several convolution and pooling layers, extracts higher-level sequences of spatial features, and the output is then used as input for the LSTM layer, which extracts the temporal features. The trained model then performs anomaly detection using a softmax classifier, as shown in Figure 6. This compound use of CNN and LSTM allows for the detection of complex contextual anomaly structures by correlating dimensions (temporal, spatial, other application-specific dimensions) [11].

A very recent and interesting approach can be found in [1]. Here, a CNN-VAE network is used as well as an LSTM network that generates temporal-aware embeddings; the LSTM layer works between the encoder and decoder layers of the CNN-VAE network, taking the latent layer as input and creating embeddings of the same shape that are more aware of the long-term temporal features. Figure 7 details the steps behind this method.

4.4. Graph-based approaches

A recent and novel approach is that of graph-based anomaly detection. The motivation behind this approach is that data objects cannot always be treated as independent points in a multi-dimensional space, but are rather connected, establishing what we call ”inter-dependencies”. Graphs provide a natural expression of these inherent dependencies thanks to the links (or edges) between different data points [2]. Another advantage to graph-based anomaly detection is the improved ability to detect contextual and collective anomalies, thanks to the clustering of nodes based on contextual attributes, which allows for the detection of anomalies through the analysis of anomalous edges or nodes within the clusters. [11].

Anomaly detection has been studied for both static and dynamic graphs, but it is the second type we are interested in when talking about LSTMs since dynamic or ”time-evolving” graphs are used to represent time series, the type

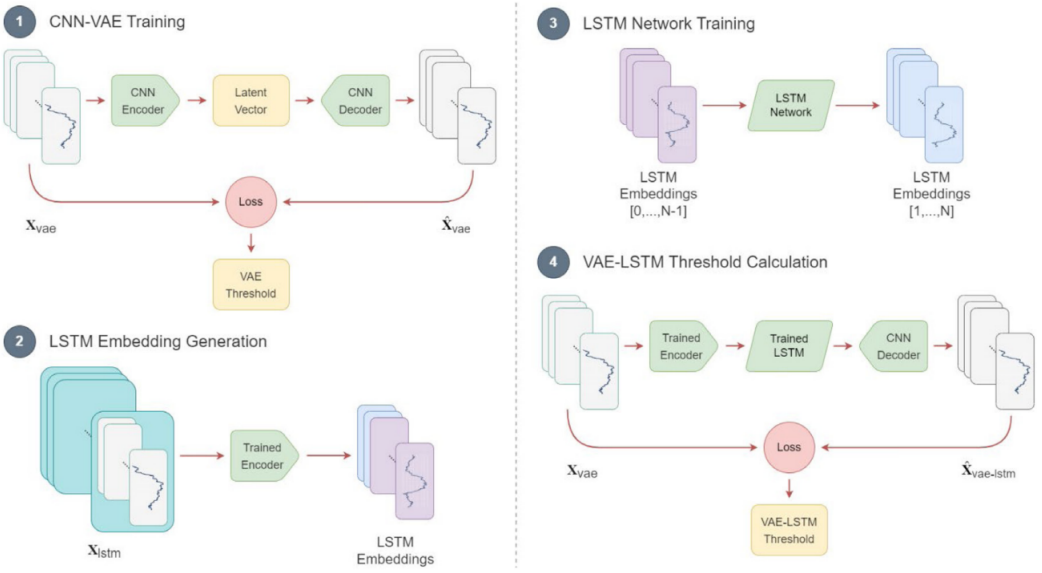

Fig. 7. Diagram detailing training steps in the CNN-VAE + LSTM embeddings method [1]

of input LSTMs are designed for. The main idea behind the usage of dynamic graphs is to leverage the relationships between multivariate time series to construct a graph for each time interval [2]. When treating these graphs, the method is usually the following [11]:

1. The input data might be already represented in a graph, if not, it must be modeled into one.
2. The graph models are stored in a database while being versioned based on different time instances of the analysis of time-based anomaly progression.
3. The graphs are then partitioned or clustered to sub-graphs based on structural and/or temporal features, allowing for the identification of the network structure. Context and content features can then be detected.
4. The final step consists of a time-based analysis of the clusters to detect anomalous nodes or edges. This is when LSTMs come in; the resulting features can be used as input for such networks in order to improve the efficiency of their use.

Furthermore, graph convolutional networks (GCNs), which extract structural and content features, can be extended to consider temporal features too, extending the original model using gated recurrent units (GRU). GRU is essentially a simpler variant of an LSTM network [23]. The original output of the GCN is then complemented with a GRU for long-term information capturing.

We find a very recent example in [14], where a general dynamic graph unsupervised anomaly detection framework is presented. This framework consists of two sections: graph embedding and anomaly detection. The first section obtains embedding vectors of all nodes in all graphs representing the structure and attribute features via a graph attention network (GAT), a type of network used for graphs that uses attention mechanisms to selectively weigh the influence of neighboring nodes, allowing for more nuanced capturing of relationships in the graph. However, these embeddings do not take into account the time dependency of the same node at different time steps. Thus, in the second section of the framework, an LSTM-AE is used to reconstruct the embedding vector to find anomalous nodes in the data. Alongside this reconstruction-based model, the detector also includes a forecasting-based model aimed at predicting the value in the next timestamp using a stacked LSTM-AE model. An overview of the proposed model is illustrated in Figure 8.

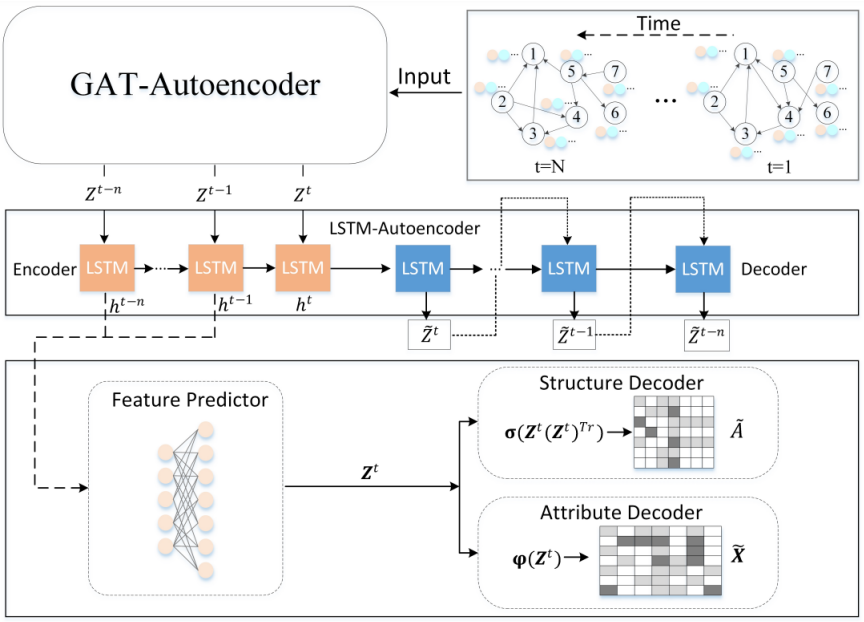

Fig. 8. Overview of the GAT-AE and LSTM-AE combination framework presented in [14]

4.5. Transfer learning

In section 2.2, we briefly mentioned one of the main challenges in anomaly detection; due to the small proportion of anomalous instances within any given dataset, the training of anomaly detection algorithms needs larger quantities of data than usual, which is often hard to acquire. Transfer learning approaches help mitigate this challenge.

The motivation behind transfer learning comes from the fact that most machine learning methods work under the assumption that the training and test data present the same distribution and feature space, needing to be rebuilt entirely if the distribution changes [15]. To reduce the need to re-collect training data, knowledge between several learning agents training on independent datasets is transferred.

Particularly, the type of transfer learning we will focus on is network-based, since it addresses the problems mentioned above the best. A network is pre-trained on the task and is then partially re-used, transferring its structure and connection parameters to be a part of the final network used in the target domain [19].

In [13] we can see an application of transfer learning used for multi-task machine learning (also known as continual learning) where the base algorithm is a stacked LSTM. A more recent example of combining transfer learning and LSTMs is [10]. This paper proposes an approach for chatter detection with an LSTM network, using sensor data and a rich data set from one machine, and then using transfer learning, similar sensors, and a smaller data set for the chatter detection algorithm on another machine. This allows for the transfer of knowledge from one machine to be applied to a similar machine, with some local optimization.

5. Discussion

Table 1 shows an overview of every surveyed approach, detailing the most relevant aspects of each of them. One such aspect is the metrics utilized for performance evaluation, which need a brief introduction to be interpretable in the table:

- The F_1 score is the harmonic mean of precision (the ratio of true positive predictions to the total number of positive predictions) and recall (the ratio of true positive predictions to the total number of actual positive instances in the dataset). A score of 1 indicates perfect precision and recall, and 0 means that either precision or recall (or both) is zero.

- The Receiver Operating Characteristic (ROC) curve plots the true positive rate against the false positive rate at various threshold settings, therefore presenting a graphical representation of the performance of the classification model. The Area Under the Curve (AUC) is the numerical representation of said performance since the goal of the ROC is to be as close to the top-left corner as possible, thus maximizing the AUC. An AUC of 1.0 indicates a perfect classification, and 0.5 means the classification is no better than random guessing.

- Cross entropy quantifies the difference between the predicted probability distribution and the true distribution (the true labels). As such the goal is to minimize its value as much as possible.

The table provides the values for these measures in each experiment, but it is important to note that these depend not only on the model itself, but also on the test dataset used, as well as other application-specific factors, and as such, the performance of the models should not be compared by just these numeric measures.

Source	Input data type	Labeled data	Anomaly type(s)	Architecture	Adp	Metrics	Performance
[12]	Multivariate time series	No	Contextual	Stacked LSTM	No	F_1 score: 0.87	Higher than RNN
[20]	Multivariate time series	No	Point	Stacked LSTM	No	F_1 score: 0.81	Higher than competitors in the specific application
[9]	Univariate time series	No	Point, collective	Dual LSTM	Yes	F_1 score: 0.69	Higher than competitors in the specific application
[17]	Multivariate time series	No	Point, contextual	Variational LSTM-AE	Yes	AUC: 0.8710	Higher than HMM, SVM, AE
[21]	Multivariate time series	No	Point, collective	Multi-LSTM AE	No	F_1 score: 0.9468, AUC: 92.82	Higher than other LSTM-AE approaches
[5]	Image sequence	No	Point, collective	LSTM-AE	No	Spatial error metrics	No comparisons
[8]	Multivariate time series	No	Point, collective	CNN + LSTM	No	F_1 score: 0.923, Cross entropy: 0.08 (approx.)	Higher than RF, MLP, KNN
[1]	Multivariate time series	No	Point, collective, contextual	CNN-VAE + VAE-LSTM	Yes	F_1 score: 0.693	Higher than competitors in the specific application
[23]	Time-stamped dynamic graph	No	Point	GCN with GRU	No	AUC: 0.8174	Higher compared against 3 graph-outlier algorithms
[14]	Time-stamped dynamic graph	No	Contextual	GAT + LSTM-AE	No	AUC: 0.9459	Higher compared against 4 graph-outlier algorithms and 2 AE approaches
[13]	Univariate time series	Yes	Point, collective, contextual	LSTM	No	Accuracy: 0.75	Higher than non-transfer approach
[10]	Multivariate time series	Yes	Point, collective, contextual	LSTM	Yes	Accuracy: 0.8243	Comparable to other transfer approaches, higher than non-transfer approach

Table 1. Comparison of all the surveyed approaches. NOTE: "Adp" in column 6 stands for "Adaptiveness"

Throughout section 4 we have gone over several approaches and methods for solving the task of anomaly detection using LSTMs in one way or another. Subsection 4.1 focuses on regular LSTMs, which can provide good results and allow for the detection of contextual and collective anomalies. Subsection 4.2 shows encoder-decoder-based architectures, which allow for optimized anomaly detection for dimensional data spaces higher than those present in regular LSTM approaches. Subsection 4.3 showcases how the combination of LSTMs with another entirely different ML technique can improve performance since the goal of these combinations is to benefit from the advantages of both models while compensating for their shortcomings, usually by dividing the task into a predictor and a detector component. Subsection 4.4 proposes the use of graphs as input data due to their higher ability to represent dependencies and heterogeneous data. Graphs allow for the contextual analysis of the cause and propagation of anomalies. Lastly, subsection 4.5 focuses on transfer learning, a method that addresses the problem of having insufficient training data, especially present in the field of anomaly detection. Transfer learning's ability to combine datasets during training

can lead to the possibility of training across different systems and/or applications, meaning a better generalization and applicability of the models.

6. Conclusions and future work

In this work, the use of LSTM networks in various ways for anomaly detection has been examined. Firstly, a brief overview of LSTMs, originally designed to solve the vanishing gradient problem, shows how they are especially well-suited for the task. Their architecture allows for the retaining of both long-term and short-term memories, which in turn makes possible the capturing of long-term time dependencies within sequential, time-dependent datasets. Anomaly detection, on the other hand, is an actual and relevant field that poses serious challenges and complexities, as well as different types of anomalies to consider.

After a brief introduction to both the task at hand and the machine learning model that will be used to solve it, we proceed to give an overview of the different approaches present in the literature in recent years. Examples include stacked LSTMs, encoder-decoder-based approaches, and hybrid approaches combining CNNs and LSTMs. All of these have proven to be more than capable of the task of anomaly detection and have offered satisfactory results. Two more recent approaches are also presented: graph-based and transfer learning approaches. The first proposes the use of dynamic graphs as input data to better represent dependencies and improve the detection of contextual and collective anomalies. The second proposes the transferring of knowledge between several learning agents, each trained in independent datasets. This addresses the problem of insufficient data for training these types of models.

Despite the good performance of the more direct LSTM approaches, this survey suggests that future research should focus on continuing to integrate LSTM networks into graph-based and transfer learning methods. Contextual anomalies remain the biggest challenge of anomaly detection, and graph-based approaches are currently the best at characterizing context via clustering and the ability to consider data about the process, the system as well and environmental attributes within a single graph.

On the other hand, transfer learning approaches can be further explored to detect anomalies not only within specific systems but also within networks of interconnected systems. Transfer learning also has the potential to further improve detection accuracy.

In the last year, a few works have combined both graph analysis and transfer learning while using the LSTM architecture, however, this combination has not yet been used for anomaly detection. It appears to be a suitable direction to continue research, given the promising results offered by the two approaches separately.

References

[1] Abir, F.F., Chowdhury, M.E., Tapotee, M.I., Mushtak, A., Khandakar, A., Mahmud, S., Hasan, A., 2023. Pcovnet+: A cnn-vae anomaly detection framework with lstm embeddings for smartwatch-based covid-19 detection. Engineering Applications of Artificial Intelligence 122, 106130. URL: https://www.sciencedirect.com/science/article/pii/S0952197623003147, doi:https://doi.org/10.1016/j.engappai.2023.106130.

[2] Akoglu, L., Tong, H., Koutra, D., 2015. Graph based anomaly detection and description: a survey. Data mining and knowledge discovery 29, 626–688.

[3] Chandola, V., Banerjee, A., Kumar, V., 2009. Anomaly detection: A survey. ACM Comput. Surv. 41. URL: https://doi.org/10.1145/1541880.1541882, doi:10.1145/1541880.1541882.

[4] De Maesschalck, R., Jouan-Rimbaud, D., Massart, D., 2000. The mahalanobis distance. Chemometrics and Intelligent Laboratory Systems 50, 1–18. URL: https://www.sciencedirect.com/science/article/pii/S0169743999000477, doi:https://doi.org/10.1016/S0169-7439(99)00047-7.

[5] Fernando, T., Denman, S., Sridharan, S., Fookes, C., 2018. Soft + hardwired attention: An lstm framework for human trajectory prediction and abnormal event detection. Neural Networks 108, 466–478. URL: https://www.sciencedirect.com/science/article/pii/S0893608018302648, doi:https://doi.org/10.1016/j.neunet.2018.09.002.

[6] Hochreiter, S., 1998. The vanishing gradient problem during learning recurrent neural nets and problem solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 6, 107–116.

[7] Hochreiter, S., Schmidhuber, J., 1997. Long Short-Term Memory. Neural Computation 9, 1735–1780.

[8] Kim, T.Y., Cho, S.B., 2018. Web traffic anomaly detection using c-lstm neural networks. Expert Systems with Applications 106, 66–76. URL: https://www.sciencedirect.com/science/article/pii/S0957417418302288, doi:https://doi.org/10.1016/j.eswa.2018.04.004.

[9] Lee, M.C., Lin, J.C., Gan, E.G., 2020. Rere: A lightweight real-time ready-to-go anomaly detection approach for time series, in: 2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC), pp. 322–327. doi:10.1109/COMPSAC48688.2020.0-226.

[10] Li, E., Bedi, S., Melek, W., 2023. Anomaly detection in three-axis cnc machines using lstm networks and transfer learning. The International

Journal of Advanced Manufacturing Technology 127, 5185–5198.

[11] Lindemann, B., Maschler, B., Sahlab, N., Weyrich, M., 2021. A survey on anomaly detection for technical systems using lstm networks. Computers in Industry 131, 103498. URL: https://www.sciencedirect.com/science/article/pii/S0166361521001056, doi:https://doi.org/10.1016/j.compind.2021.103498.

[12] Malhotra, P., Vig, L., Shroff, G., Agarwal, P., et al., 2015. Long short term memory networks for anomaly detection in time series., in: Esann, p. 89.

[13] Maschler, B., Pham, T.T.H., Weyrich, M., 2021. Regularization-based continual learning for anomaly detection in discrete manufacturing. Procedia CIRP 104, 452–457.

[14] Miao, G., Wu, G., Zhang, Z., Tong, Y., Lu, B., 2023. Addag-ae: Anomaly detection in dynamic attributed graph based on graph attention network and lstm autoencoder. Electronics 12, 2763.

[15] Pan, S.J., Yang, Q., 2009. A survey on transfer learning. IEEE Transactions on knowledge and data engineering 22, 1345–1359.

[16] Pang, G., Shen, C., Cao, L., Hengel, A.V.D., 2021. Deep learning for anomaly detection: A review. ACM Comput. Surv. 54. URL: https://doi.org/10.1145/3439950, doi:10.1145/3439950.

[17] Park, D., Hoshi, Y., Kemp, C.C., 2018. A multimodal anomaly detector for robot-assisted feeding using an lstm-based variational autoencoder. IEEE Robotics and Automation Letters 3, 1544–1551.

[18] Sutskever, I., Vinyals, O., Le, Q.V., 2014. Sequence to sequence learning with neural networks. Advances in neural information processing systems 27.

[19] Tan, C., Sun, F., Kong, T., Zhang, W., Yang, C., Liu, C., 2018. A survey on deep transfer learning, in: Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27, Springer. pp. 270–279.

[20] Thill, M., Däubener, S., Konen, W., Bäck, T., Barancikova, P., Holena, M., Horvat, T., Pleva, M., Rosa, R., 2019. Anomaly detection in electrocardiogram readings with stacked lstm networks., in: ITAT, pp. 17–25.

[21] Wei, Y., Jang-Jaccard, J., Xu, W., Sabrina, F., Camtepe, S., Boulic, M., 2023. Lstm-autoencoder-based anomaly detection for indoor air quality time-series data. IEEE Sensors Journal 23, 3787–3800.

[22] Yu, Y., Si, X., Hu, C., Zhang, J., 2019. A review of recurrent neural networks: Lstm cells and network architectures. Neural computation 31, 1235–1270.

[23] Zheng, L., Li, Z., Li, J., Li, Z., Gao, J., 2019. Addgraph: Anomaly detection in dynamic graph using attention-based temporal gcn., in: IJCAI, p. 7.