Unsupervised anomaly detection using bidirectional neural networks

Jovan Vještica*, Bratislav Predić*
* Faculty of Electronic Engineering, University of Niš, Serbia
jovan.vjestica@elfak.rs, bratislav.predic@elfak.ni.ac.rs

***Abstract*— Anomaly detection is a very important class of data processing techniques and as such has found wide applications in many different fields. One of the problems often faced when developing systems for anomaly detection is the absence of labeled datasets, which means that unsupervised learning techniques need to be developed. Anomaly detection is most often used on data that exhibits some type of locality. This, in effect, means that anomalies are context-dependent. This paper analyzes existing approaches for extracting context information and defines a novel approach based on a bidirectional recurrent neural network. This paper defines an end-to-end system that is capable of autonomously flagging anomalies with no human interaction needed.**

I. INTRODUCTION

Anomaly detection is a distinct branch of computer science and artificial intelligence that concerns itself with detecting data points that are in some way different from others. These points are called anomalies or outliers [1]. This process should be automated as much as possible and should take into account the structure of the data itself.

Anomalies can be considered the consequences of errors in the data generating process (i.e., white noise), but they can also be used as good indicators of the existence of new, latent variables of the data generation process. Thus, by properly understanding anomalies, new insights into the data generation process can be achieved which facilitates the creation of newer, more precise system models [2].

Anomaly detection has found widespread use in different domains. Some of the most often cited examples are video surveillance, picture analysis, healthcare, network security, the financial industry, etc.

Additionally, libraries such as TensorFlow Data Validation (TFDV) directly use anomaly detection as part of the data purification process [3]. This means that for machine learning systems, anomaly detection can also be applied as a step in the data pipeline in order to ensure high quality data for model training. In environments which have fast changing data and demand regular model retraining, this has proven crucial in practice. By optimizing existing approaches or by creating novel approaches, anomaly detection researchers can have a positive impact on the broader machine learning community.

This paper defines a novel technique for unsupervised anomaly detection, based on bidirectional recurrent neural networks. Specifically, the defined network architecture can be used to detect any contextual anomalies with one dimensional locality (i.e., time-series data). The network consists of two sub-networks – a context network and a predictive network. The network attempts to predict the value of each of the sequence elements by considering the element's neighborhood. For each element, the prediction error is calculated. This data is then fed into a statistical post-processing step, which flags the points as either regular data points or anomalies.

II. RELATED WORK

Because of the large impact anomaly detection continues to have on many different fields, it is no surprise that, throughout computing history, anomaly detection has been heavily researched. This trend continues today, with the rise of the Internet, being able to produce petabytes of data, and modern machine learning systems, which are able to effectively use that data.

Keeping in mind the broad nature of anomaly detection, it is expected to start a research with a systematization of modern machine learning systems. Paper [2] provides a high-level overview of both types of anomalies and deep anomaly detection (DAD) techniques. The discussion on different types of anomalies, and especially of contextual anomalies, is very useful. The paper defines contextual anomalies as being data points that can be considered anomalous, if viewed within a certain context and stresses the need to develop approaches to deal with the contextual information being presented by the data.

Paper [1] concerns itself mostly with deep anomaly detection models. A very interesting segment is devoted to learning feature representations of normality, which gave the inspiration for the structure of the current system. It abstractly defines what feature representation of normality is and gives a few practical examples.

Since the type of data analyzed in this paper is time-series data, the natural tool for processing this data are recurrent neural networks (RNNs) [4]. The main building block of an RNN is the recurrent cell, which unlike regular feedforward neural networks, contains a built-in feedback loop. This loop acts as a kind of iterator, allowing the recurrent cell to natively process sequential data [5].

In order to deal with the issue of the vanishing gradient, typically encountered in RNNs, multiple modifications to the RNN cell have been proposed. One of the most impactful is the long short-term memory cell [6] (LSTM cell), which features three additional control gates, giving the cell fine-grained control over gradient propagation.

An important aspect of contextual anomaly detection systems is the ability to create an automated method for getting context information out of raw data. Multiple

architectures for the context network were considered. One was the encoder-decoder, as described in [7]. The encoder-decoder uses a fixed size context vector to encode the entire sequence, which can lead to issues in the general case. Bahdanau attention [8] solves this issue by calculating a custom annotation vector for each element of the sequences. These annotation vectors are then sent to an alignment model, the output of which is used in the calculation of the final output.

III. CONTEXTUAL ANOMALIES

Anomaly detection is a highly diverse field of study, with many different domain use-cases requiring specialized techniques in order to achieve state of the art performance. One of the aspects in which different anomaly detection systems differ from each other is the types of anomalies they detect. All anomalies can be divided into three categories[2]:

1) Contextual anomalies
2) Point anomalies
3) Group anomalies

Point anomalies are considered irregularities in single points of data and may or may not have any useful interpretations. Since these types of anomalies are the simplest, the majority of research has been focused on them. Group anomalies, on the other hand, express anomalous characteristic when viewed as a group.

Contextual anomalies are also known as conditional anomalies. When considering contextual anomalies, a system needs to take into account not only the data point being analyzed, but other data points in the vicinity as well. This directly implies the concept of locality, where individual data points can be considered either “closer” or “farther” from each other. The dimensionality of the locality also plays an important role as well, and is typically fixed for any given application. For instance, in the case of image analysis, individual pixels exhibit two-dimensional locality, along the height and width axes of the image. On the other hand, when dealing with time-series data, individual datapoints are ordered only in one axis (time), with datapoints that occur closer in time to each other being considered local to a given data point.

Because context plays a key role in contextual anomaly detection, there is a need to express the context of a given data point as a vector. This is considered a very important part of an anomaly detection system, since properly extracting contextual information ensures that the system can generate high quality predictions down the timeline.

Since this paper deals with sequential data (time series), the natural way for extracting context information is from raw data. In this paper this is performed by using a specialized context network, based on recurrent neural networks. The context vector needs to be not only well defined, but also useful for the given application. Because of this, the context network is trained simultaneously with the prediction network, in order to ensure that the context vector is sufficiently specialized for the task at hand.

Multiple different configurations of the context network were considered, including an encoder-decoder based approach [7] as well as an attention-based approach [8]. Ultimately, a new approach was used, based on a bidirectional recurrent neural network.

IV. NETWORK ARCHITECTURE

The anomaly detection system that was developed consists of three main parts:

1) Context network
2) Prediction network
3) Statistical post-processing

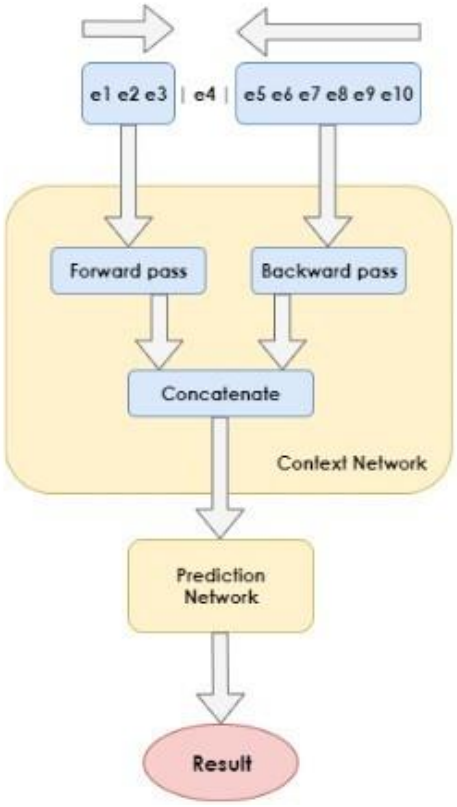

Figure 1. Network architecture

The context network is based on a bidirectional recurrent neural network. The architecture of each pass of the context network is that of a stacked recurrent neural network, where the outputs of one RNN layer are fed into the inputs of the next one. Since LSTM networks have been shown to be capable of learning long-term data dependencies, they were chosen as the main building block for the context network [9]. Deep-and-narrow stacked LSTM networks have been shown to have the best tradeoff of parameters number and overall performance, so this was the adopted approach for each of the passes.

One of the disadvantages of using standard RNN based approaches is the fact that they are causal filters, which means that they don’t consider the elements coming after the one being processed [4]. Since for the use case of detecting contextual anomalies the elements that are the successors of the element being processed are relevant because they can contain valuable context information. Because of this, two LSTM networks (named the forward and backward pass) were used when calculating the context. Since LSTMs (and in general all RNNs) are

biased towards closer data points, the backwards pass traverses the data in a backwards order. This gives a larger impact to data points that are closer to the point being analyzed, since it is expected that these are more important for the context.

The final result of the forward and backward passes of the network is concatenated into the context vector. This vector is calculated for each individual data point, which is similar in concept to the way Bahdanau attention calculates annotations for each sequence element. This context vector is then fed into a prediction network, which attempts to infer the value of the element being processed. The prediction network is a standard feedforward multi-layer perceptron network. Since the element is known during training, no labeled training set is needed, making training unsupervised.

In order to ensure that the context vector extracts useful information to be used by the prediction network, both are trained in a single pass. Since the architecture is a directed, acyclic graph (DAG), standard backpropagation can be applied [10]. Once the gradient gets propagated to the forward and backward passes, the two components are optimized by applying backpropagation through-time (BPTT). The gradient, defined as the mean square error between the element value and the actual prediction, gets propagated to all components of the network.

V. STATISTICAL POST-PROCESSING

The end result of the prediction network is a vector of predictions, with the same dimensionality as the input data. By subtracting the predictions from the input data (or vice versa), a vector of generated prediction errors is created. Optionally, if the sign of the error is considered irrelevant, the absolute value of the error can be generated, by taking the absolute value of the error vector. This error is expected to be non-zero for most elements, since data reconstruction is rarely perfect. These errors can be visualized as histograms.

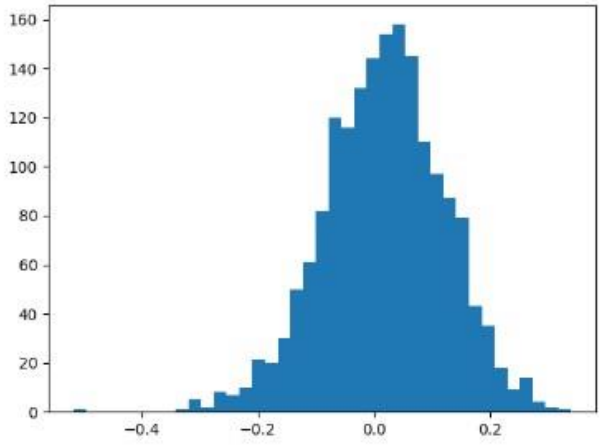

Figure 2. Error histogram

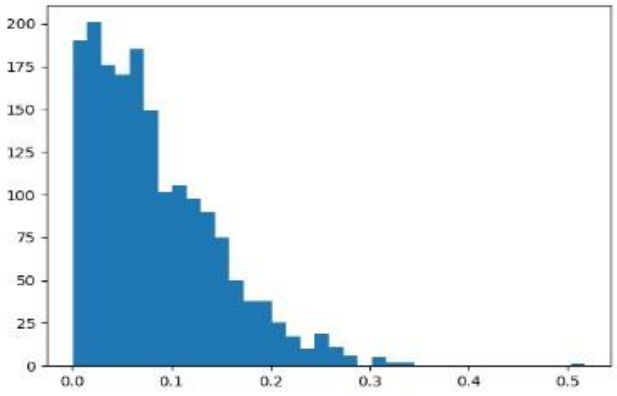

Figure 3. Absolute error histogram

If the context and prediction networks have learned meaningful representation of the data, a gaussian-like error distribution is assumed - if a random variable X follows a gaussian distribution, the absolute value of that random variable |X| will follow a folded gaussian distribution. In the case of the absolute error, a folded gaussian distribution is assumed, while in the other case, a gaussian distribution is assumed.

The parameters of the distribution can be inferred at runtime, for any specific dataset. The two parameters of the distribution are its mean value and its standard deviation. The mean value represents the value at which the distribution is centered at, while the standard deviation is a measure of how much the average element deviates from the mean value. The mean for the errors generated by the prediction network is expected to be in the range of zero, while smaller values of the standard deviation are also more preferable, since that indicates that the network generates accurate results overall. The estimation of these parameters is a well-known problem in statistics, and many statistics libraries provide out of the box implementations for this process. Since the parameters are estimated during training, they also act as learned parameters – there is no need to hand select these parameters and they shouldn't be changed at inference time.

Finally, after a distribution has been fit onto the errors, the filtering stage begins. By defining a confidence interval, some values of the errors and the datapoints associated with those errors are flagged as being either regular or anomalous. The confidence interval is defined by a parameter, the confidence level, that takes a value between 0 and 1. This value defines the width of the interval, with higher values being associated with wider intervals. By taking all the values that fall outside the defined confidence interval and flagging them as anomalies, the entire processing is finished. Since the end user can fine tune the confidence level, the system offers its users fine-grained control over which data points are flagged as anomalies and which aren't. This means that the confidence level of the confidence interval in practice acts as a kind of anomaly sensitivity parameter, where a setting of 100% indicates that no data points will be flagged as anomalies, while a setting of 0% flags every data point as an anomaly.

VI. RESULTS

In order to test out the functionality and performance of the proposed system, a time series data set was needed. Typically, Kaggle provides access to various kinds of datasets. For this paper, the Store Item Demand dataset was used to estimate the system performance [11]. The dataset represents sales data of 50 items at 10 stores over a five-year period. The data was sampled at regular intervals (daily), with no missing values.

The context and predictive networks were both implemented using the Keras [12] functional API with the TensorFlow backend. The statistical postprocessing step was implemented using the statistics module of the scipy library [13].

In figure 4, the blue line represents the actual data, the orange line represents the predicted values while the red dots show the detected anomalies. As shown in figure 4, the system is fully capable of recognizing and flagging extreme values of the data, as well as being able to flag smaller spikes as well. The smaller spikes, such as the third last anomaly in figure 4, are considered as contextual anomalies. Even though the value of the datapoint is relatively close to the mean value of the overall sequence, when viewed in its context, there is a reason to suspect the datapoint is anomalous. By decreasing the confidence level, new anomalies (in addition to the ones being shown) would be flagged as well.

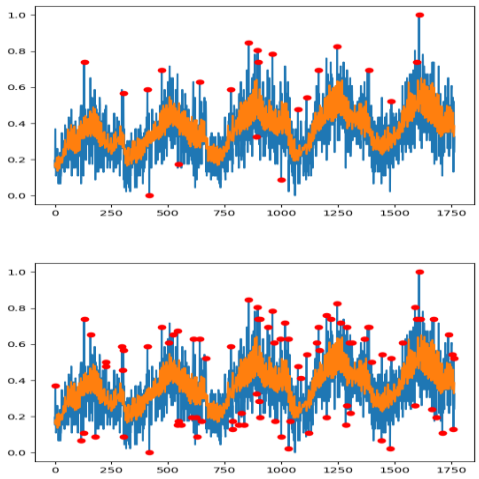

Figure 4. Signed error with 99% and 95% confidence level

VII. CONCLUSION

This paper discusses a novel technique for unsupervised anomaly detection, based on bidirectional recurrent neural networks. The paper outlines the architecture behind the entire anomaly detection system, as well as all of its individual components. By modifying existing attention-based approaches based on calculating annotation vectors on a per-element basis, the paper describes a new and flexible way of generating context vectors for sequential data sets. Through the specific training procedure, the system can process unlabeled datasets in an unsupervised manner. The system is shown to be able to learn information-rich context representations, allowing it to detect both point and contextual anomalies.

One of the ways to potentially further improve the system would be to add convolutional operators between the stacked LSTM layers. Additionally, anomaly detection systems may also analyze system logs (in the cases of network security and intruder detection, but also others), so adding native support to text data would be useful as well.

ACKNOWLEDGMENT

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

REFERENCES

[1] P. Guansong, S. Chunhua, C. Longbing, A. Hengel, *Deep Learning for Anomaly Detection: A Review*, arXiv preprint arXiv: 2007.02500, 2020.

[2] C. Raghavendra, C. Sanjay, *Deep Learning For Anomaly Detection: A survey*, arXiv preprint arXiv:1901.03407, 2019.

[3] Tensorflow Data Validation, https://www.tensorflow.org/tfx/data_validation/get_started, 30.05.2021.

[4] A. Sherstinsky , *Fundamentals of Recurrent Neural Network {(RNN)} and Long Short-Term Memory (LSTM) Network*, CoRR, abs/1808.03314, 2018.

[5] Z. Di, P. Qidong, L. Junqiang, W. Dongsheng, L. Xuefei, Z. Jiangbo. (2019). *Simulating Reservoir Operation Using a Recurrent Neural Network Algorithm*. Water 2019, 11. 865, 2019.

[6] Understanding LSTM Networks, *http://colah.github.io/posts/2015-08-Understanding-LSTMs/*, 12.08.2020.

[7] I. Sutskever, O. Vinyals, Q. V. Le, *Sequence to Sequence Learning with Neural Networks*, arXiv preprint arXiv: 1409.3215, 2014.

[8] D. Bahdanau, K. Cho, Y. Bengio , *Neural Machine Translation by Jointly Learning to Align and Translate*, arXiv preprint arXiv: 1409.0473, 2015.

[9] How LSTM networks solve the problem of vanishing gradients, *https://medium.com/datadriveninvestor/how-do-lstm-networks-solve-the-problem-of-vanishing-gradients-a6784971a577*, 14.08.2020.

[10] M. P. Deisenroth, A. A. Faisal, C. S. Ong, *Mathematics for machine learning*, Cambridge University Press, 2020.

[11] Dataset, https://www.kaggle.com/c/demand-forecasting-kernels-only/data, 18.10.2020

[12] Keras, *https://keras.io/*, 14.10.2020

[13] Scipy, https://www.scipy.org/, 15.10.2020.