TSXplain: Demystification of DNN Decisions for Time-Series using Natural Language and Statistical Features

Mohsin Munir$^{1,2[0000-0001-7818-0361]}$*, Shoaib Ahmed Siddiqui$^{1,2[0000-0003-4600-7331]}$, Ferdinand Küsters^{3}, Dominique Mercier^{1,2}, Andreas Dengel^{1,2}, and Sheraz Ahmed$^{2[0000-0002-4239-6520]}$*

[1] Technische Universität Kaiserslautern, Kaiserslautern 67663, Germany
[2] German Research Center for Artificial Intelligence (DFKI) GmbH, Kaiserslautern 67663, Germany
`mohsin.munir@dfki.de, sheraz.ahmed@dfki.de`
[3] IAV GmbH, Gifhorn 38518, Germany
`ferdinand.kuesters@iav.de`

Abstract. Neural networks (NN) are considered as black boxes due to the lack of explainability and transparency of their decisions. This significantly hampers their deployment in environments where explainability is essential along with the accuracy of the system. Recently, significant efforts have been made for the interpretability of these deep networks with the aim to open up the black box. However, most of these approaches are specifically developed for visual modalities. In addition, the interpretations provided by these systems require expert knowledge and understanding for intelligibility. This indicates a vital gap between the explainability provided by the systems and the novice user. To bridge this gap, we present a novel framework i.e. Time-Series eXplanation (TSXplain) system which produces a natural language based explanation of the decision taken by a NN. It uses the extracted statistical features to describe the decision of a NN, merging the deep learning world with that of statistics. The two-level explanation provides ample description of the decision made by the network to aid an expert as well as a novice user alike. Our survey and reliability assessment test confirm that the generated explanations are meaningful and correct. We believe that generating natural language based descriptions of the network's decisions is a big step towards opening up the black box.

Keywords: Demystification · Textual Explanation · Statistical Feature Extraction · Deep Learning · Time-series Analysis · Anomaly Detection.

1 Introduction

Deep neural networks (DNN) have become ubiquitous these days. They have been successfully applied in a wide range of sectors including automotive [12], government [27], wearable [21], dairy [16], home appliances [25], security and surveillance [15], health [6], and many more, mainly for regression, classification, and anomaly detection problems [5, 7, 19, 20, 32]. The neural network's capability of automatically discovering features to solve any task at hand makes them particularly easy to adapt to new problems and scenarios. However, this capability to automatically extract features comes at the cost of a lack of transparency/intelligibility of their decisions. Since there is no clear indication regarding why the network reached a particular prediction, these deep models are generally referred to as a black box [23].

It has been well established in the prior literature that an explanation of the decision made by a DNN is essential to fully exploit the potential of these networks [1, 22]. With the rise in demand for these deep models, there is an increasing need to have the ability to explain their decisions. For instance, big industrial machines cannot be powered down just because a DNN predicted a high anomaly score. It is important to understand the reason for reaching a particular decision, i.e. why the DNN computed such an anomaly score. Significant efforts have been made in the past in order to better understand these deep models [2, 23, 24, 30]. Since most of the prior literature is validated specifically for visual modalities, therefore, a very common strategy is to visualize the learned features [14, 24, 30, 31]. Visual representation of filters and learned features, which might provide important hints to the expert, is already a good step in this direction, but might not be intelligible to the end user. Adequate reasoning of the decision taken increases the user's confidence in the system.

The challenges pertaining to interpretability and visualization of deep models developed for time-series analysis are even more profound. Directly applying the interpretation techniques developed specifically for visual modalities are mostly uninterpretable in time-series data, requiring human expertise to extract useful information out of these visualizations [23]. Therefore, we aim to bridge this gap by generating explanations based on the concrete knowledge extracted out of the statistical features for the decision made by the network, assisting both novice and expert users alike. Statistical features are considered to be transparent which ameliorates the process of gaining user trust and confidence.

This paper presents a novel approach for generating natural language explanations of the classification decisions made by a DNN (system pipeline is shown in Fig. 1.). We take a pre-trained DNN and find the most salient regions of the input that were mainly leveraged for a particular prediction. Important data points that contributed towards the final network decision are tracked using the influence computation framework. These points are then combined with different statistical features, which are further used to generate natural language explanations. We specify confidence to the generated explanation by accessing its reliability through sanity check. Our experiments on the classification datasets and a survey show that the generated natural language explanations help in better understanding the classification decision. To the best of the authors' knowledge, it is the very first attempt to generate natural language explanations for classification task on time-series data.

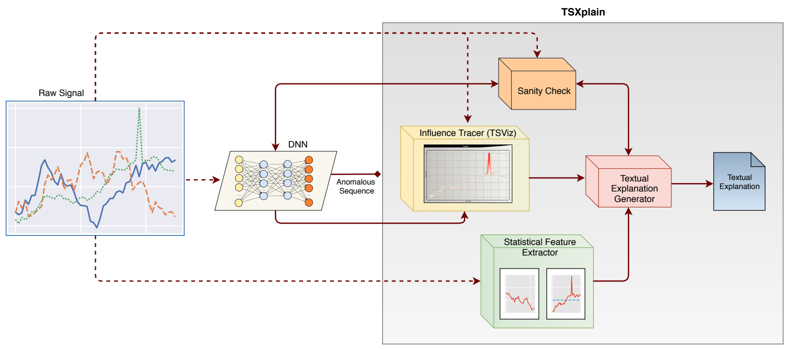


Fig. 1: TSXplain system diagram. Based on a time-series classifier, the time-series are passed on to the *influence tracer* module which highlights the most influential data points. Furthermore, these influential data points are used along with the point-wise and sequence-wise statistical features to generate textual explanations of the time-series.

2 Related Work

Over the past few years, there have been numerous advancements in the field of network interpretation and visualizations. For understanding a DNN on an image classification task, Zeiler and Fergus (2014) [31] proposed a technique for the visualization of deep convolutional neural networks. Their visualization reveals features in a fully trained network. Highlighting the input stimuli which excites individual feature maps at different layers, helps in understanding what the network has learned. Yosinski et al. (2015) [30] introduced DeepVizToolbox which helped in understanding how neural network models work and which computations they perform at the intermediate layers of the network at two different levels. This toolbox visualizes the top images for each unit, forward activation values, deconvolutional highlighting, and preferred stimuli. Simonyan et al. (2013) [24] also presented an approach to visualize the convolutional neural networks developed for image classification. Their visualization provides image-specific class saliency maps for the top predicted class.

Bau et al. (2017) [4] introduced a framework to interpret the deep visual representations and quantify the interpretability of the learned CNN model. First, they gather a broad set of human-labeled visual concepts and then gather the response of hidden variables to known concepts. To understand a neural network, Mahendran and Vedaldi (2015) [17] highlight the encoding learned by the network through inversion of the image representations. They also study the locality of the information stored in the representations. Melis et al. (2018) [18] designed self-explaining models where the explanations are intrinsic to the model for the robust interpretability of a network. Bach et al. (2015) [2] introduced an approach to achieve pixel-level decomposition of an image classification decision.

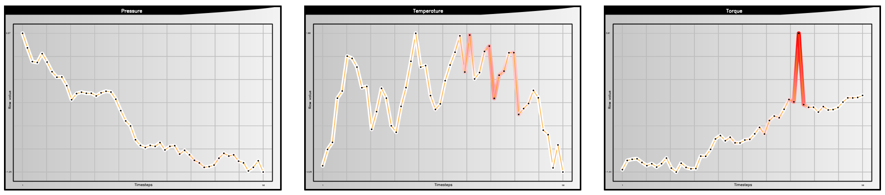
They generate a heatmap for a well-classified image that segments the pixels belonging to the predicted class.

Current DNN visualizations and interpretations only help an expert to understand and improve the overall process. However, they still lack the actual reasoning why a particular decision has been taken by the learned model. There has been some work in the domain of image captioning where visual attributes are leveraged to support the DNN decision. Guo et al. (2018) [8] proposed a textual summarization technique of image classification models. They train a model with the image attributes which are used to support the classification decision. In the same domain, Hendricks et al. (2016) [10] proposed a model that predicts a class label and explains the reason for the classification based on the discriminating properties of the visual objects. Kim et al. (2018) [13] introduced a textual explanation system for self-driving vehicles. They generate introspective explanations to represent the causal relationships between the system's input and its behavior which is also the target of our study. In most of the aforementioned techniques, a neural network black box is further used to generate descriptions and explanations of another neural network. Despite being a promising direction, this introduces another level of opaqueness into the system.

In the domain of relation extraction, Hancock at al. (2018) [9] proposed a supervised rule-based method to train classifiers with natural language explanations. In their framework, an annotator provides a natural language explanation for each labeling decision. Similar work has been presented by Srivastava et al. (2017) [26], but they jointly train a task-specific semantic parser and classifier instead of a rule-based parser. These systems, however, rely on a labeled set of training examples that are not available in most of the real-world applications.

3 Methodology

Different visualization and interpretation techniques developed specifically to understand deep models aid an expert in understanding the learning and decision-making process of the network. However, the provided interpretation/visualization cannot be readily understood by a novice user. It is up to the user to draw conclusions about the network's decision with the help of the available information. Many of the existing techniques use a separate deep network that is trained for the generation of explanations using the primary model [8, 10, 13]. These explanations still suffer from a lack of transparency, as they are also generated by a deep model. Therefore, we approach the problem of generating explanations in a way that significantly improves the intelligibility of the overall process. We leverage the statistical time-series features to provide a concrete natural language based explanation of an sequence. These features also help in gaining a user's trust because of their lucid nature. The proposed system is composed of different modules as highlighted in Fig. 1. The raw input is first passed on to DNN for classification. If the sequence is classified an , the whole TSXplain system is activated which is composed of four modules, namely *influence tracer (TSViz)* [23], *statistical feature extractor, sanity check,* and *textual explanation*



(a) Pressure signal (b) Temperature signal (c) Torque signal

Fig. 2: Visualization of the saliency information along with the raw signal values. Influential data points are highlighted in the shades of red, dark being more influential.

generator. The *influence tracer* is employed to discover the most salient regions of the input (Section 3.1). The *statistical feature extractor* module extracts different statistical features from the sequence (Section 3.2). The results from previous two modules are passed onto the *textual explanation generator* module in order to come up with a natural language description of the encountered anomaly (Section 3.3). Furthermore, we introduce a *sanity check* module to get a coarse estimate of the system's confidence regarding the generated explanation (Section 3.4).

3.1 Influence Tracer (TSViz)

The influence tracer module is based on the TSViz framework [23]. The proposed influence tracing algorithm can be used to trace the influence at several different levels. However, we only consider the main influence for our method i.e. the influence of the input on the output. These influences can be effectively computed as the gradients of the output y w.r.t. the input components x_j:

$$I^j_{\text{input}} = \left| \frac{\partial y}{\partial x_j} \right| \tag{1}$$

The absolute value of the gradient is used here, as only the magnitude of the influences is of relevance to us. Once the influences have been computed, we use the max-min scaling to scale the values for visualization and further processing as specified by:

$$I^j_{\text{input,scaled}} = \frac{I^j_{\text{input}} - \min\limits_j I^j_{\text{input}}}{\max\limits_j I^j_{\text{input}} - \min\limits_j I^j_{\text{input}}} \tag{2}$$

The computed influence values are visualized on top of the original signals to provide a hint regarding the encapsulated information. This visualization is presented in Fig. 2, where color-coded data points in each input channel are shown with respect to their influence on the final decision of the DNN. The dark shade represents the high influence of a data point. It can be observed in Fig. 2

that the network decision is mostly based on the small and big spikes in the observed time-series.

Siddiqui et al. (2018) [23] discussed the problem of extremely confident predictions for the influence tracing algorithm and suggested a remedy to overcome this issue by imposing regularization on top of the activations itself when training the network. The new objective can be written as:

$$\mathcal{W}^* = \arg\min_{\mathcal{W}} \frac{1}{|\mathcal{X}|} \sum_{(\mathbf{x},y)\in\mathcal{X}\times\mathcal{Y}} \mathcal{L}(\Phi(\mathbf{x};\mathcal{W}), y) + \lambda\|\mathcal{W}\|_2^2 + \beta\|z^L\|_2^2 \tag{3}$$

where Φ defines the mapping from the input to the output space, $\mathcal{W}$ encapsulates all the parameters of the network, z^L denotes the activation values of the last layer before application of the sigmoid layer, and λ and β denotes hyperparameters controlling the contribution of the regularization terms and the empirical risk. We use the same modified objective to train our network in order to avoid extremely confident predictions.

The *influence tracer* module consumes both the DNN model as well as the raw input. Then, it performs the backward pass through the network from output to input in order to obtain these influence values. The output from this module is consumed by the *textual explanation generator* module (Section 3.3).

3.2 Statistical Feature Extractor

This module extracts different statistical features from the input sequence. Since we are dealing with sequential data comprising of time-series, we calculate different point-wise as well as sequence-wise features. These features include, but not limited to, lumpiness, level shift, KL score, number of peaks, and ratio beyond r-sigma (explained below). These features have been previously used by Bandara et al. (2017) [3] and have been proposed in [11,29].

1. *Lumpiness:* Initially, daily seasonality from the sequence is removed by dividing it into blocks of n observations. Variance of the variances across all blocks is computed which represent the lumpiness of the sequence.
2. *Level shift:* The sequence is divided into n observations and the maximum difference in mean between consecutive blocks is considered as the level shift. It highlights the block which is different from the rest of the sequence.
3. *KL score:* To calculate this score, the sequence is divided into consecutive blocks of n observations. This score represents the maximum difference in Kullback-Leibler divergence among consecutive blocks. A high score represents high divergence.
4. *Number of peaks:* This feature identifies the number of peaks in a sequence. The sequence is smoothed by a Ricker wavelet for widths ranging from 1 to n. It detect peaks with sufficiently high signal-to-noise ratio.
5. *Ratio beyond r-sigma:* It gives the ratio of data points which are r standard deviations away from the mean of a sequence.
6. *Standard deviation:* This feature represents the standard deviation of a sequence.

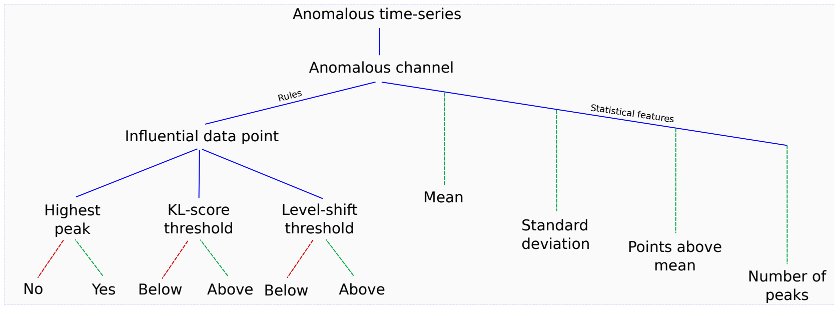


Fig. 3: An example of rules and statistical features used to generate textual explanations for a given anomalous time-series.

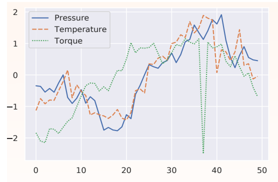
In addition to the above mentioned sequence-wise features, we also use point-wise features including peak, valley, maximum point, minimum point, highest spike, and lowest valley, etc. The fusion of sequence-wise and point-wise features provides vivid characteristics of the highly influential data points.

3.3 Textual Explanation Generator

The influential points determined by the *influence tracer* module along with the features computed by the *statistical feature extractor* module are passed onto this module for the generation of the textual explanations of a given anomalous sequence. This module also receives input from the *sanity check* module (explained in Section 3.4) which allows the system to specify its confidence over the generated explanation.

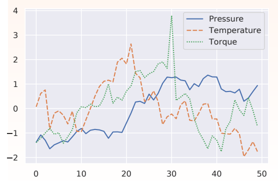
We designed a set of rules to incorporate a range of features. These rules are defined based on the statistical time-series features in a way that explains different characteristics of a given sequence. Based on the classification decision given by the network along with the time-series features, this module provides explanations of the data points which influence the network decision. The explanations are generated channel-wise so that the anomalous data points in each channel can be highlighted. An example of defining rules and an overall hierarchy of this process is shown in Fig. 3. The influential data points are provided by the *influence tracer* module and the rules defined under the 'Influential data point' parent node in Fig. 3 are applied on those data points. The statistical features (mean, standard deviation, points above mean, and number of peaks calculated on whole sequence mentioned on right branch of 'Anomalous channel' parent node in Fig. 3) provide supporting arguments to the defined rules. The values of child nodes (highest peak, KL-score, level-shift, mean, standard deviation, points above mean, and number of peaks) in green are incorporated in the textual explanations. For multi-channel input, the salient data points from an individual channel are passed onto this module.

We have defined two levels of abstraction for the textual explanations. The first level of explanation is defined for the users who are not interested in detailed



Most influential channel in the given time series is Torque signal because of the presence of anomalous data point(s) at index(s) 37. The value(s) at given index (s) is/are -2.4982. The data point(s) is/are anomalous because there is/are valley(s) around the mentioned index(s) and there is/are 1 valley(s) in this series. The lowest valley is the one detected. The anomalous point is also the minimum point of the series. The anomalous data point(s) is/are -2 standard deviation(s) away from the mean of the series, where the mean is -0.0728 while the standard deviation is 1.0457. Level shift of 1.1573 also shows that there is a consecutive block (with anomalous point) in this series for which the difference between means is relatively low. Kullback-leibler score of 3.5623 represents the presence of clearly separable density distributions in the series.

Sanity Check: The system is fairly confident regarding the provided explanation.



Most influential channel in the given time series is Temperature signal because of the presence of anomalous data point(s) at index(s) 21. The value(s) at given index(s) is/are 2.6410. The data point(s) is/are anomalous because there is/are peak(s) around the mentioned index(s) and there is/are 7 peak(s) in this series.

Sanity Check: The system is not confident regarding the provided explanation.

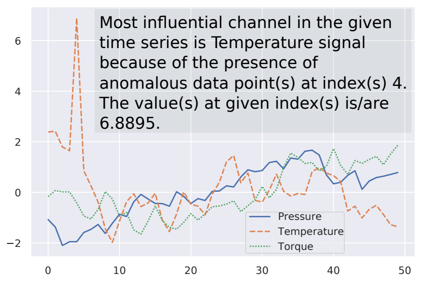
Fig. 4: Channel-wise natural language explanations generated for the expert users. The second time-series in this figure represents a failure-case where the explanations are not accurate, which is also detected in the sanity check.

explanations or don't have enough knowledge of time-series data (novice user) but would like to get information regarding the most salient regions and channels of the input. The second level of explanation, on the other hand, is defined for the expert users. Such users are generally interested in knowing the details such as, why a network took a particular decision? Sample explanations for the expert users are shown in Fig. 4. The explanations shown in this figure clearly point out the anomalous channel in a given anomalous time-series sequence. Moreover, characteristics of anomalous data points and anomalous channel are also explained in the form of feature values. Fig. 5a shows an example of a simple explanation which is generated for the novice users.

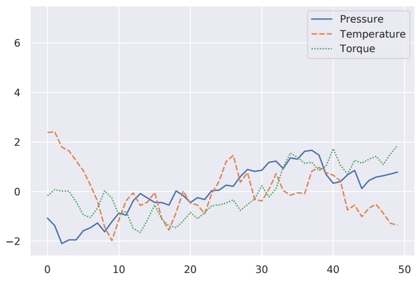
3.4 Sanity Check

In order to assess the reliability of a given explanation, a simple sanity check is performed. The output of the *textual explanation generator* module along with the original input is passed onto this module. The data point corresponding to the explanation is suppressed and this masked sequence is fed again to the network for inference. The masking is performed by linear interpolation between the last and the first retained points.

If the removed data points were indeed causal for the network prediction, we expect the prediction to flip. We use this sanity check to compute confidence over the provided explanation. If we observe a flip in the prediction, we assign high confidence to the provided explanation. On the other hand, if the prediction is retained, we assign low confidence to the provided explanation. This check confirms that the generated explanations are referring to the data points which are actually contributing towards the classification decision taken by the network.



(a) Anomalous sequence.



(b) Effect of sanity check.

Fig. 5: As a results of sanity check, the anomalous peak is suppressed and sequence in (a) is classified as normal in (b).

Finally, the sanity check output is passed back to the *textual explanation generator* module with the confidence information which is mentioned in Fig. 4 after the generated explanations. In Fig. 4, the second example didn't observe any flip in the prediction after the suppression of the deemed causal point (a failure case), resulting in low system confidence for the provided explanation. Fig. 5 visualizes the process of sanity check on an anomalous sequence. In Fig. 5a, the textual explanation highlights an anomalous data point in the temperature signal. The network prediction is flipped after suppressing the mentioned anomalous data point in the temperature signal as shown in Fig. 5b.

4 Experimental Setup and Dataset

To classify a time-series as normal or anomalous, we trained a CNN model with three convolutional layers comprising of 16, 32 and 64 filters respectively, with Leaky ReLU as the activation function, followed by a single dense layer. Since the focus of this paper is on a generation of textual explanations, we selected the hyperparameters (e.g. number of layers, number of filters) based on our experience and did not invest any significant effort into hyperparameter optimization or model selection. It is important to mention that we have chosen a convolutional neural network as our DNN, because CNNs are generally easier to optimize, achieved state-of-the-art results in anomaly detection [20, 32], and the base module of TSXplain (*influence tracer* module) is currently based on CNNs. To generate natural language explanations on time-series data, we used the datasets mentioned in this section.

4.1 Machine Anomaly Detection [23]

It is a synthetic time-series classification dataset curated by Siddiqui et al. (2018) [23]. This dataset comprises of 60000 time-series with 50 time-stamps each. Each sequence consists of three channels which represent values from pressure, torque, and temperature sensors. The dataset contains point anomalies in the torque and temperature signals, while the pressure signal is kept intact. A sequence is labeled as anomalous if it contains a point anomaly. The dataset is split into 45000 training sequences with 7505 anomalous sequences, 5000 validation sequences with 853 anomalous sequences, and 10000 test sequences with 1696 anomalous sequences.

4.2 Mammography [28]

This breast cancer screening dataset contains 11183 time-series and it is commonly used for classification purposes. Anomalies at certain points/features make the whole time-series anomalous. The dataset is split into 8000 training time-series with 186 anomalous time-series, 1000 validation time-series with 25 anomalous time-series, and 2183 test time-series with 49 anomalous time series.

4.3 NASA Shuttle [28]

This dataset describes radiator positions in a NASA shuttle with 9 attributes. There are 46464 time-series in this dataset. We split the dataset into 25000 training time-series with 483 anomalous time-series, 5000 validation time-series with 102 anomalous time-series, and 16464 test time-series with 293 anomalous time series.

5 Evaluation and Discussion

In order to completely assess the relevance and correctness of a given explanation, we conducted a survey in which novice and expert users were asked to evaluate the generated explanations. We provided 20 time-series from the *Machine Anomaly Detection* dataset along with the generated explanations to the participants and asked questions related to whether the generated explanation was i) relevant, ii) sufficient, iii) meaningful, iv) correct, and v) satisfactory from the experts. Whereas, the novice users were only questioned about i), ii), and

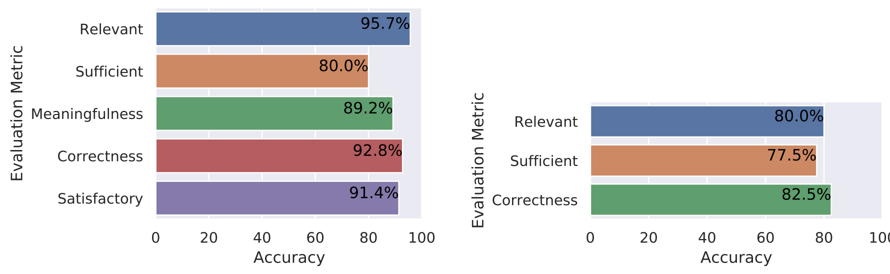


Fig. 6: Summary of evaluation done by expert (left) and novice (right) users on Machine Anomaly Detection dataset [23] explanations.

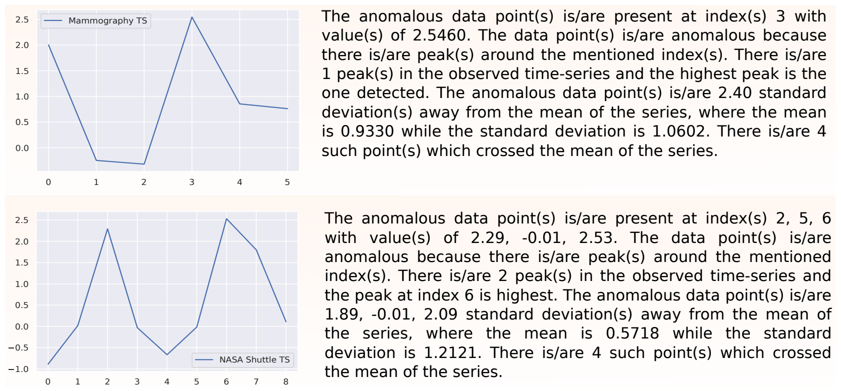
Table 1: Effect of masking the data points in *Machine Anomaly Detection* dataset which are relevant for the explanation.

Window size	Anomalous sequence	Flipped prediction after masking	Percentage flipped
1	1511	1104	73.0%
3	1511	1319	87.3%

iv). There were 7 expert and 6 novice participants, who provided their binary (agree/disagree) feedback to the aforementioned question category. By analyzing the accumulated feedback of the participants shown in Fig. 6, it is clear that most of the participants considered the provided explanations relevant, meaningful, and correct. The majority of the experts were satisfied with the reasoning of the NN decision provided in the explanation. Although, 20% experts and 22.5% novice participants thought that the provided explanation is not sufficient.

In this study, we are trying to infer the causality through the provided explanations, so it is also important to assess the reliability of the generated explanations. Therefore, we introduced the sanity check module in the system pipeline to obtain a measure of confidence over the provided explanations (as explained in Section 3.4). We also computed this confidence estimate over the entire test set of *Machine Anomaly Detection* dataset in order to get an impression regarding the overall reliability of the generated explanations. The cumulative results are presented in Table 1. Since it is important to compute these statistics only over examples where the classification from the network was correct, we were left with 1511 out of 1696 total anomalous sequences in the test set. In the first setup, a masked sequence is generated by suppressing the exact data points for which the explanations have been generated by the *textual explanation generator* module. Since we are suppressing the exact point, we represent this setup with a window size of one. In the second setup, a sequence is masked with a window size of three covering one preceding and one following value in order to cover up any minor misalignment of the most salient region highlighted by the *textual explanation generator* module. In this case, a total of three data points were suppressed. We represent this setup with a window size of three. The results shown in Table 1 indicate that for 73.0% of the anomalous sequences, the predictions were flipped by masking out the exact data points highlighted by the explanation module. When we relaxed the sanity check criteria to a window size of three, the percentage of flipped sequences rose up to 87.3%. This high success rate makes it evident that in most of the cases, plausible explanations for the predictions made by the network could be provided. However, it is important to note that this experiment does not strongly imply causality.

In the traditional interpretation settings where only visual explanations are available, it is difficult for a user to understand why a particular decision is taken by the network just by looking at the plots. However, it is relatively easy to understand the classification reason by reading the explanations provided by



The anomalous data point(s) is/are present at index(s) 3 with value(s) of 2.5460. The data point(s) is/are anomalous because there is/are peak(s) around the mentioned index(s). There is/are 1 peak(s) in the observed time-series and the highest peak is the one detected. The anomalous data point(s) is/are 2.40 standard deviation(s) away from the mean of the series, where the mean is 0.9330 while the standard deviation is 1.0602. There is/are 4 such point(s) which crossed the mean of the series.

The anomalous data point(s) is/are present at index(s) 2, 5, 6 with value(s) of 2.29, -0.01, 2.53. The data point(s) is/are anomalous because there is/are peak(s) around the mentioned index(s). There is/are 2 peak(s) in the observed time-series and the peak at index 6 is highest. The anomalous data point(s) is/are 1.89, -0.01, 2.09 standard deviation(s) away from the mean of the series, where the mean is 0.5718 while the standard deviation is 1.2121. There is/are 4 such point(s) which crossed the mean of the series.

Fig. 7: Sample explanations generated for Mammography (top) and NASA Shuttle (bottom) time-series.

our system for the corresponding plots in Fig. 4 and Fig. 7. We specifically opted for statistical features due to their strong theoretical foundations and transparency. The point-wise features of an anomalous data point help a user in understanding how that data point is different from the rest of the sequence. Whereas sequence-wise features help in highlighting the overall behavior of an anomalous sequence. An end-user can confirm part of the explanation by looking at the plot, which elevates his trust in the system.

One of the major limitations of the TSXplain is its specificity for the task at hand. The computed features and the rule-base are not easily transferable between different tasks. Another limitation is regarding the availability of a suitable set of statistical features for a particular task. We leave these open questions as future work. We would also like to test our system on more complex datasets along with the employment of more sophisticated influence tracing techniques [2] to further enhance the utility of the system.

6 Conclusion

This paper proposes a novel demystification framework that generates natural language based descriptions of the decision made by the deep learning models developed for time-series analysis. The *influence tracer* module identifies the most salient regions of the input. Statistical features from the sequence are simultaneously extracted from the *statistical feature extractor* module. These two results are passed onto the *textual explanation generator* for the generation of the final explanation by the system. We used statistical features over other alternates due to their strong theoretical foundations and transparency which significantly improved the intelligibility and reliability of the provided explanations. A confidence estimate is also provided by the system using the sanity check module which is based on assessment whether the estimated influential point is indeed causal for the prediction. The generated explanations are directly intelligible for both expert and novice users alike evading the requirement of domain expertise to understand the encapsulated information. We tested and generated the proposed framework on different synthetic and real anomaly detection datasets and demonstrated our results. The explanations can help the users in uplifting their confidence regarding the performance of the deep model. We strongly believe that natural language based descriptions are one of the most convincing ways in the long-term for the realization of explainable systems.

References

1. Andrews, R., Diederich, J., Tickle, A.B.: Survey and critique of techniques for extracting rules from trained artificial neural networks. Knowledge-based systems **8**(6), 373–389 (1995)
2. Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.R., Samek, W.: On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. PloS one **10**(7), e0130140 (2015)
3. Bandara, K., Bergmeir, C., Smyl, S.: Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. arXiv preprint (2017)
4. Bau, D., Zhou, B., Khosla, A., Oliva, A., Torralba, A.: Network dissection: Quantifying interpretability of deep visual representations. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6541–6549. IEEE (2017)
5. Cai, Y., Guan, K., Peng, J., Wang, S., Seifert, C., Wardlow, B., Li, Z.: A high-performance and in-season classification system of field-level crop types using time-series landsat data and a machine learning approach. Remote Sensing of Environment **210**, 35–47 (2018)
6. Crabtree, B.F., Ray, S.C., Schmidt, P.M., O'Connor, P.T., Schmidt, D.D.: The individual over time: time series applications in health care research. Journal of clinical epidemiology **43**(3), 241–260 (1990)
7. Goldstein, M.: Anomaly Detection in Large Datasets. Phd-thesis, University of Kaiserslautern, München, Germany (2 2014), http://www.goldiges.de/phd
8. Guo, P., Anderson, C., Pearson, K., Farrell, R.: Neural network interpretation via fine grained textual summarization. arXiv preprint arXiv:1805.08969 (2018)
9. Hancock, B., Bringmann, M., Varma, P., Liang, P., Wang, S., Ré, C.: Training classifiers with natural language explanations. In: Proceedings of the conference. Association for Computational Linguistics. Meeting. vol. 2018, p. 1884. NIH Public Access (2018)
10. Hendricks, L.A., Akata, Z., Rohrbach, M., Donahue, J., Schiele, B., Darrell, T.: Generating visual explanations. In: European Conference on Computer Vision. pp. 3–19. Springer (2016)
11. Hyndman, R.J., Wang, E., Laptev, N.: Large-scale unusual time series detection. In: 2015 IEEE international conference on data mining workshop (ICDMW). pp. 1616–1619. IEEE (2015)
12. Kang, M.J., Kang, J.W.: Intrusion detection system using deep neural network for in-vehicle network security. PloS one **11**(6), e0155781 (2016)

13. Kim, J., Rohrbach, A., Darrell, T., Canny, J., Akata, Z.: Textual explanations for self-driving vehicles. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 563–578 (2018)
14. Kuo, C.C.J., Zhang, M., Li, S., Duan, J., Chen, Y.: Interpretable convolutional neural networks via feedforward design. Journal of Visual Communication and Image Representation (2019)
15. Kushwaha, A.K., Dhillon, J.K., et al.: Deep learning trends for video based activity recognition: A survey. International Journal of Sensors Wireless Communications and Control **8**(3), 165–171 (2018)
16. Lark, R.M., Nielsen, B.L., Mottram, T.T.: A time series model of daily milk yields and its possible use for detection of a disease (ketosis). Animal Science **69**(3) (1999)
17. Mahendran, A., Vedaldi, A.: Understanding deep image representations by inverting them. In: CVPR. pp. 5188–5196 (2015)
18. Melis, D.A., Jaakkola, T.: Towards robust interpretability with self-explaining neural networks. In: Advances in Neural Information Processing Systems. pp. 7775–7784 (2018)
19. Munir, M., Siddiqui, S.A., Chattha, M.A., Dengel, A., Ahmed, S.: FuseAD: Unsupervised Anomaly Detection in Streaming Sensors Data by Fusing Statistical and Deep Learning Models. Sensors **19**(11) (2019). https://doi.org/10.3390/s19112451
20. Munir, M., Siddiqui, S.A., Dengel, A., Ahmed, S.: DeepAnT: A deep learning approach for unsupervised anomaly detection in time series. IEEE Access **7**, 1991–2005 (2019)
21. Ordóñez, F., Roggen, D.: Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition. Sensors **16**(1), 115 (2016)
22. Saad, E.W., Wunsch II, D.C.: Neural network explanation using inversion. Neural Networks **20**(1), 78–93 (2007)
23. Siddiqui, S.A., Mercier, D., Munir, M., Dengel, A., Ahmed, S.: TSViz: Demystification of Deep Learning Models for Time-Series Analysis. IEEE Access (2019)
24. Simonyan, K., Vedaldi, A., Zisserman, A.: Deep inside convolutional networks: Visualising image classification models and saliency maps. arXiv preprint (2013)
25. Singh, S., Majumdar, A.: Deep sparse coding for non–intrusive load monitoring. IEEE Transactions on Smart Grid **9**(5), 4669–4678 (2018)
26. Srivastava, S., Labutov, I., Mitchell, T.: Joint concept learning and semantic parsing from natural language explanations. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. pp. 1527–1536 (2017)
27. Trippi, R.R., Turban, E.: Neural networks in finance and investing: Using artificial intelligence to improve real world performance. McGraw-Hill, Inc. (1992)
28. Vanschoren, J., Van Rijn, J.N., Bischl, B., Torgo, L.: OpenML: Networked Science in Machine Learning. ACM SIGKDD Explorations Newsletter **15**(2), 49–60 (2014)
29. Wang, X., Smith, K., Hyndman, R.: Characteristic-based clustering for time series data. Data mining and knowledge Discovery **13**(3), 335–364 (2006)
30. Yosinski, J., Clune, J., Fuchs, T., Lipson, H.: Understanding neural networks through deep visualization. In: International Conference on Machine Learning Workshop on Deep Learning (2015)
31. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision (ECCV). pp. 818–833. Springer (2014)
32. Zheng, Y., Liu, Q., Chen, E., Ge, Y., Zhao, J.L.: Time series classification using multi-channels deep convolutional neural networks. In: International Conference on Web-Age Information Management. pp. 298–310. Springer (2014)