Time Series Anomaly Detection Using Convolutional Neural Networks and Transfer Learning

Tailai Wen[1] , **Roy Keyes**[2]
Arundo Analytics
{[1]tailai.wen, [2]roy.keyes}@arundo.com

Abstract

Time series anomaly detection plays a critical role in automated monitoring systems. Most previous deep learning efforts related to time series anomaly detection were based on recurrent neural networks (RNN). In this paper, we propose a time series segmentation approach based on convolutional neural networks (CNN) for anomaly detection. Moreover, we propose a transfer learning framework that pretrains a model on a large-scale synthetic univariate time series data set and then fine-tunes its weights on small-scale, univariate or multivariate data sets with previously unseen classes of anomalies. For the multivariate case we introduce a novel network architecture. The approach was tested on multiple synthetic and real data sets successfully.

1 Introduction

Time series anomaly detection plays a critical role in automated monitoring systems. It is an increasingly important topic today, because of its wider application in the context of the Internet of Things (IoT), especially in industrial environments [Da Xu *et al.*, 2014].

Before the boom of deep learning in the early 2010s, most time series anomaly detection efforts were based on traditional time series analysis (e.g. [Abraham and Chuang, 1989; Bianco *et al.*, 2001]), or on approaches to extract and represent time series properties (e.g. [Chan and Mahoney, 2005; Keogh *et al.*, 2005; Ringberg *et al.*, 2007; Ahmed *et al.*, 2007]). Some machine learning techniques for multivariate outlier detection were also widely applied to detect anomalies in multivariate time series (e.g. [Breunig *et al.*, 2000; Schölkopf *et al.*, 2001; Liu *et al.*, 2008]), although they treat data points independently and neglect temporal relationships.

In recent years, advances in deep learning have revolutionalized many areas of data-driven modeling. Recurrent neural networks (RNN) and convolutional neural networks (CNN) are two major types of network architectures that enabled these breakthroughs. RNN's are generally applied to temporal sequence tasks, while CNN's are typically the first choice for image related tasks. For this reason, most previous deep learning work on time series anomaly detection was based on RNN's (e.g. [Malhotra *et al.*, 2015; Kim *et al.*, 2016; Wang *et al.*, 2017; Yin *et al.*, 2017]). There has been some research using CNN's for time series tasks, primarily around sequence classification (e.g. [Zheng *et al.*, 2014; Yang *et al.*, 2015; Cui *et al.*, 2016; Rajpurkar *et al.*, 2017]).

We recognized that time series anomaly detection shares many common aspects with image segmentation. When a person visualizes a time series and selects the anomalous segment, if present, the perceptual process is very similar to a person looking at an image and marking a desired object. In this research, we created a CNN-based deep network for time series anomaly detection. In particular, we were inspired by a successful image segmentation network, U-Net, and applied a time series version of U-Net to detect anomalous segments in time series. As the limited occurrence of failures is a common blocker for anomaly detection in industrial IoT systems, we also propose a transfer learning framework to resolve the data sparsity issue, including a new architecture, MU-Net, for transferring a univariate base model to multivariate tasks.

The remainder of this paper is organized as follows: Section 2 includes a review of related work. We provide details of applying U-Net for time series anomaly detection in Section 3. Section 4 introduces the transfer learning framework and introduces the MU-Net architecture for multivariate time series anomaly detection. We provide experimental observations in Section 5, and conclude with some discussion in Section 6.

2 Related Work

Our approach was influenced by recent successes of deep learning for image segmentation. [Long *et al.*, 2015] proposed a fully convolutional network (FCN), where common convolutional architectures for image classification (e.g. AlexNet, the VGG net, and GoogLeNet) are used as encoders, and counterpart deconvolution layers are used for upsampling as decoders. U-Net [Ronneberger *et al.*, 2015] improved upon the FCN architecture by introducing so-called skip channels between encoding layers and decoding layers into the architecture, so that high-level features and low-level features are concatenated to prevent information loss along deep sequential layers. This architecture was proven successful when applied to segmentation of neuronal structures in electron microscopic images in the original paper. It was subsequently applied to several other biomedical image segmentation tasks (e.g. [Çiçek *et al.*, 2016; Dalmış *et al.*, 2017;

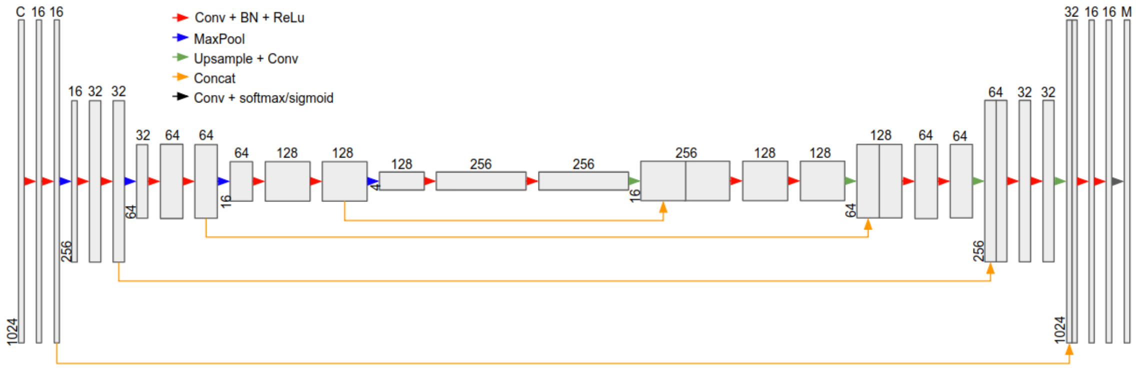

Figure 1: Architecture of U-Net for time series segmentation.

Dong *et al.*, 2017]) as well as image segmentation problems in earth science (e.g. [Karchevskiy *et al.*, 2018]), remote sensing (e.g. [Yao *et al.*, 2018]), and automated driving (e.g. [Siam *et al.*, 2017]).

Our transfer learning work follows the success of deep transfer learning in image and natural language processing. The design of synthetic pre-training data was in part inspired by [Mahajan *et al.*, 2018]. Our fine-tuning strategies incorporated some techniques presented in the Fast.AI courses[1].

Data augmentation proved to be an important part of the research. Our time series augmentation strategies extended the previous work of [Um *et al.*, 2017].

3 CNN-based Time Series Segmentation

We will introduce the details of how we build and train a time series version of U-Net, including the detailed architecture, how it is applied to streaming data in a production environment, and some important issues, including input normalization and augmentation.

3.1 U-Net for time series segmentation

A time series can be regarded as a one-dimensional image where the only dimension is temporal, whereas a typical image has two dimensions: width and height. A multivariate time series may have an arbitrary number of channels, which may have different properties and correlations with each other. In contrast, an image typically has only three channels, RGB, and their properties and correlations are not arbitrary. Following the design of U-Net, we propose the following architecture for time series segmentation as shown in Figure 1. For a time series with length 1024 and C channels, it is encoded by five sections of convolution layers. Each section includes two layer blocks, each of which includes a convolution layer, a batch normalization layer, and a ReLu layer. For all convolution layers, the kernel size is equal to 3. The number of filters in each convolution layer increases over the five sections, from 16 to 256 by a factor of 2. For every convolution layer, we apply zero padding and set the stride equal to 1. Between encoding sections, max pooling layers with a pool size equal to 4 are used to downsample the feature series. This is followed by four decoding sections, where each section includes an upsampling layer and two conv+BN+ReLu blocks. The upsampling rate is also equal to 4, and the convolution layers uses the same number of filters as their counterpart encoding layers, making the architecture symmetric. An important feature of U-Net is the skip channels between corresponding encoding and decoding sections. This is implemented by concatenating the output from every max pooling layer with the output from the corresponding upsampling layer, and performing convolution over the concatenated feature series.

The last section includes a convolution layer with a kernel size equal to 1 and an activation layer. Sometimes different classes of anomalies are not mutually exclusive, and therefore the problem is a multi-class, multi-label problem, i.e. a time point can be assigned to multiple classes. For example, a spike on a periodic series is both an additive anomaly and a seasonal anomaly. In such a case, the number of filters at the final convolution layer is equal to the number of anomaly classes M, and the final activation function is a sigmoid so that probabilities of classes are independent. If all classes of anomaly are mutually exclusive, the output shape has depth $M + 1$ and the additional column is for the nominal class (i.e. no anomaly present). In this case, softmax activation is applied so that the result is multi-class single-label. In both cases, soft Dice loss is used as the loss function, as is the case with many image segmentation networks, and the Adam optimizer is used.

3.2 Prediction on Streaming Data

When deploying a trained model for production in a streaming environment, we regularly take snapshots of the latest batch of data and run the model on it. The frequency of taking snapshots should be at least as high as the length of snapshot so that every time point is evaluated by the model at least once. We recommend using a frequency several times higher than this minimal limit, and thus every time point is evaluated by the model a few times and we may ensemble multiple results for the same time point to get more robust detection. Figure 2 shows an example where every data point is evaluated 3 times.

Although the model input shape is fixed (e.g. 1024), it is not necessary to always use it as the length of the snap-

[1] https://www.fast.ai

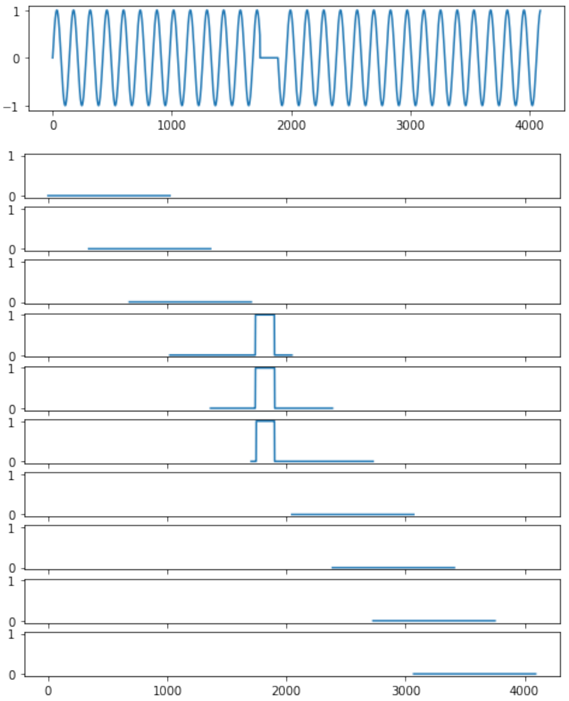

Figure 2: Anomaly detection on a data stream (top) by taking snapshots regularly and returning probability of anomaly (bottom).

shot. The length of the snapshot should be determined by the streaming frequency and the time scale of anomalous behavior. If the streaming frequency is too high with respect to anomalous behaviors, a long snapshot length (i.e. a large number of time points) should be used and every snapshot needs to be downsampled to the model input size. If the streaming frequency is too low, a short length should be used and upsampling is needed.

The strategy of sliding window could also work with a CNN-based classification model that classifies sequences by whether it includes anomalous subsequence. However, the choice of snapshot length would be problematic. If it is too long, the localization of anomaly segments tends weak, since a snapshot is only classified by not segmented. On the other hand, if it is too short, the snapshot may not contain sufficient context to distinguish itself from normal subsequence. The proposed segmentation-based method, however, may overcome this difficulty by segmenting anomalous period in high granularity.

3.3 Input Normalization

Different from usual image problems, where values in a RGB channel are always between 0 and 255, a time series may have arbitrarily large or small values, and the magnitude scale may vary over channels. Figure 3 shows the same subsequence from a stock price series in two different scales, where one looks essentially constant, while the other appears to have a big jump. This is a common misleading issue, even when a human views a time series. The proposed model requires a user to specify a magnitude scale first, and uses the scale to normalize the input series. Therefore, when applying the model in production over snapshots, the detection will always be based on the same scale. In some cases, the user does need to detect anomalies only with respect to values inside a snapshot. The user may then add a sample-wise input normalization layer at the start of the neural network, so that every input snapshot is normalized independently and the scale will be dynamic.

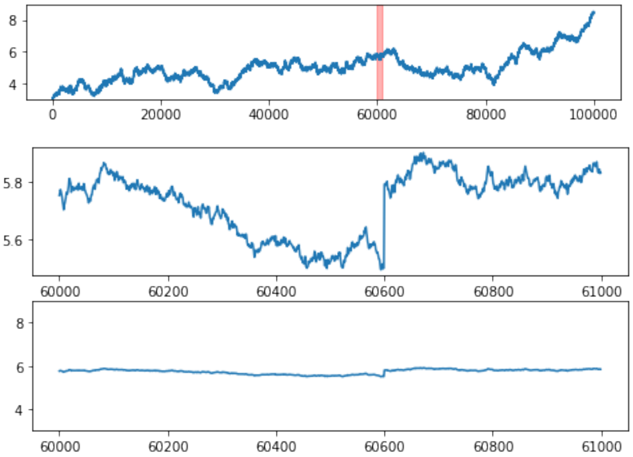

Figure 3: The same segment from a time series (top) in different scales (middle and bottom).

3.4 Augmentation

Data augmentation is a common approach to boost the size of training data for robust models. [Um *et al.*, 2017] introduced a list of augmentation methods for time series, including time warping, cropping, etc. We extended the list by adding a few methods, for example zooming, adding random trend, reversing series, applying a random linear operation, random mutation between multiple series, etc. These augmentation methods were used during testing over different data sets.

As in image problems, augmentation of training data must be label-invariant. In the context of anomaly detection, that means an augmentation method must not change the nominality of a time series. Some augmentation operations are not label-invariant to certain types of anomaly. For example, mutation operation may violate a periodic pattern when applied to a seasonal series, so it should not be used to augment training data if the expected anomaly is an aberration of a periodic pattern. Augmentation methods should always be selected appropriately for the case under consideration.

4 Transfer Learning

Sparsity of failure events in historical data often limits model training in practice. Transfer learning is a strategy to resolve the data sparsity issue. The transfer learning approach uses the weights from a base model pretrained on an available large-scale data set and then fine-tunes the model weights with a small-scale data set related to the task of interest. The hyper-parameter tuning process takes advantage of the model pretrained on the large-scale data set, which tends to extract useful features from the input, and therefore requires dramatically less training data to converge without overfitting.

[Mahajan *et al.*, 2018] explored factors that may impact performance of transfer learning in CNN-based image processing tasks. The similarity between the pretraining data set and target data set has proven to be an important factor. The correlation between pretraining task(s) and the target task also plays a significant role. In our work, we define three pretraining tasks, i.e. three types of anomalies to detect, including additive outliers, anomalous temporary changes of volatility, and violations of cyclic patterns. These are the most common types of anomalies found in univariate time series. Additive outliers can be interpreted on a short time scale, violations of cyclic patterns must be interpreted on a long time scale that covers at least a few cycles, and changes of volatility occurs on a medium time scale. We believe low-order features representing these anomaly types are necessary components to build higher-order features to represent more complex anomalous behaviors. In other words, those features are transferable to general the anomaly detection problem.

As features are not only extracted from anomalous segments, but also nominal segments, and thus, the nominal behavior of pretraining series should also be diverse. To prepare the pretraining data set, we generated a large number of various synthetic time series, including smooth curves, piece-wise linear curves, piece-wise constant curves, pulse-like signals, etc. For each type, we generated some cyclic series and some non-cyclic series. We augmented those time series by cropping, jittering, adding random trends, and time warping, with different levels of intensity, such that the training data covers a variety of nominal behaviors. We then adjusted random segments in these nominal series with the three previously mentioned types of anomalies to generate pretraining data with labelled anomalies.

Figure 4 shows some examples from the pretraining data set.

4.1 Transfer Learning for Univariate Tasks

For transfer learning to another univariate task, we keep the model architecture as in Figure 1, except that the output shape of the final convolution layer must change according to the number of classes in the target task. Weights of all layers except the output section are initialized with weights from the pretrained model. We found two fine-tuning strategies with good performance in our tests. The first one is to set up different learning rate multipliers in 10 sections (5 encoding sections, 4 decoding sections, and the output section) as 0.01, 0.04, 0.09, ..., 0.81, 1.0. The other one is to freeze the weights in the first two sections and only fine-tune the subsequent sections, and then to unfreeze the first sections and fine-tune all weights. Both strategies returned similar results in our tests.

4.2 MU-Net: A U-Net-based Network for Transfer Learning from Univariate to Multivariate Tasks

When the pretrained base model is transferred to a multivariate task, using the same U-Net architecture would be problematic. The kernel of the first convolution layer would have a different shape, as C is not equal to 1. If those kernels are initialized randomly, then transferring weights of the deeper layers is meaningless, because the lowest-order features are extracted differently. If we repeat the $3 \times 1 \times 16$ weight matrix in the first layer of pretrained model C times and create an initialization of the $3 \times C \times 16$ weight matrix for the transfer model, mathematically it is equivalent to extracting transferable features from the sum of all series channels. While the sum of RGB channels may still maintain important properties from original images, the sum of time series channels is generally meaningless, for example, if the channels represent different sensors in an IoT system such as pressure and temperature.

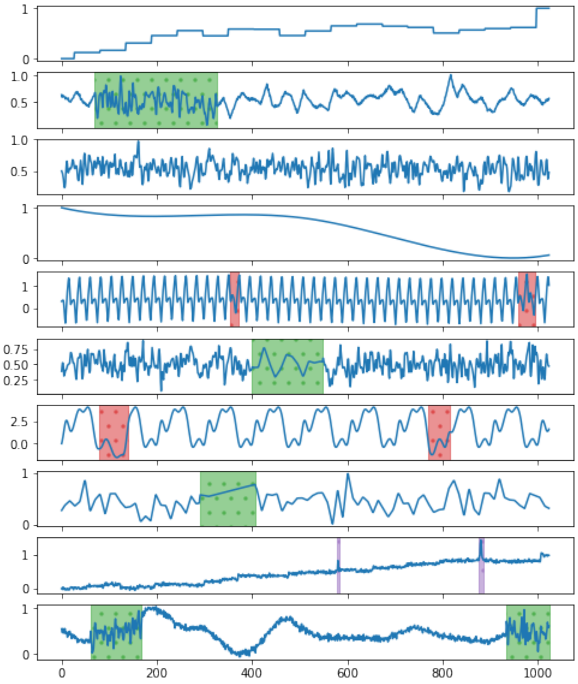

Figure 4: Some samples from the pretraining data set with labelled additive outliers (purple), changes of volatility (green), and violations of cyclic patterns (red).

We propose a new network architecture to transfer the weights from a pretrained model as shown in Figure 5. Similar to U-Net, this architecture also has an encoding-decoding structure. However, the C channels of input series are separated by a slicing layer first, and then every channel has its own univariate encoding sections, like the first four encoding sections in U-Net. The outputs from the fourth encoding section over all channels will be concatenated before an integrated fifth encoding section, followed by four decoding sections and a final output section, the same as U-Net. We have nicknamed this architecture MU-Net (multivariate U-Net).

When we transfer a pretrained U-Net model to a multivariate task, we initialize weights in the first four encoding sections for every channel in MU-Net with the corresponding weights in the pretrained model. During fine-tuning, we first freeze the first four encoding sections and tune the weights of the remaining layers. We then unfreeze the third and fourth

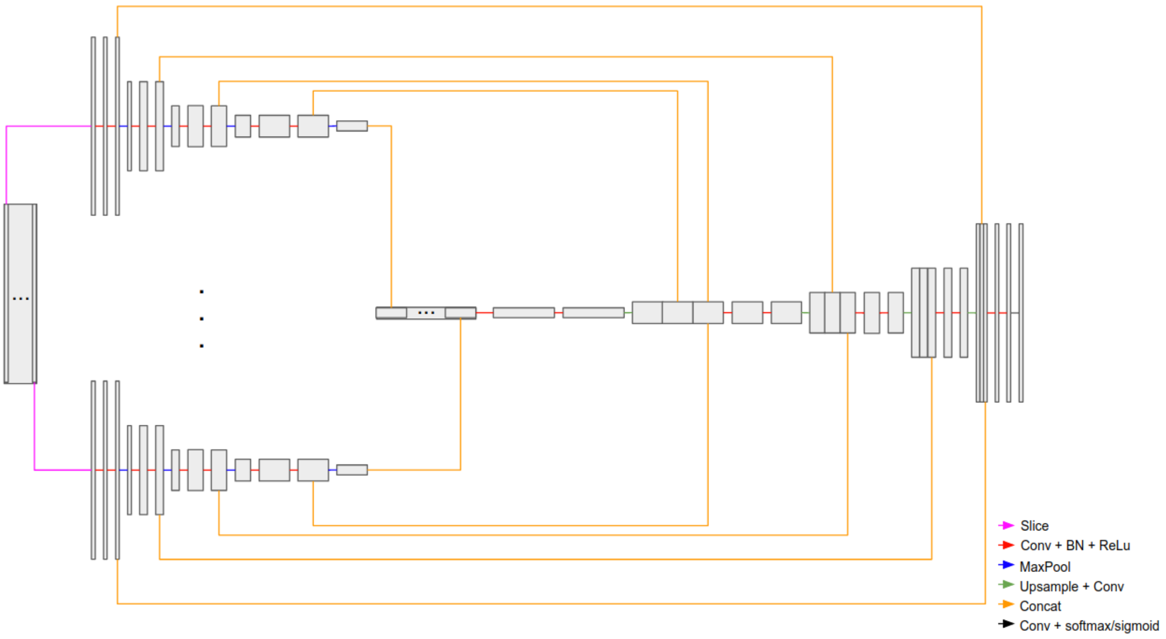

Figure 5: Architecture of MU-Net, a U-Net-based transfer network for multivariate tasks.

encoding sections and continue tuning. Finally we tune all weights including the first two sections.

5 Experimental Evaluation

We tested the proposed approach in four scenarios: a univariate task with sufficient data, a multivariate task with sufficient data, a univariate task with insufficient data and transfer learning, and a multivariate task with insufficient data and transfer learning.

All tests were conducted on an Nvidia GeForce GTX 1080Ti GPU. All implementation was done using Keras with the TensorFlow backend.

5.1 Dodgers Loop Sensor Data Set

This data set[2] was originally introduced by [Ihler *et al.*, 2006]. It includes a single time series of the traffic on a ramp close to Dodger Stadium over 28 weeks with 5-min frequency. The task is to detect anomalous traffic caused by sporting events. There are a total of 81 known events. We use the first half of the data (including 42 events) to train the model and the other half (including 39 events) for testing.

The training sequence was randomly cropped into 500 snapshots with lengths varying between 1,024 (about 3.5 days) and 4,096 (about 2 weeks), and then downsampled to 1,024. A univariate U-Net was created and trained. Only 3 known events out of 39 were not detected, and the missing detections were because these events were very close to periods with many missing values. A few false positives also occurred, mostly near missing values. Figure 6 shows detection results in some snapshots from the testing set.

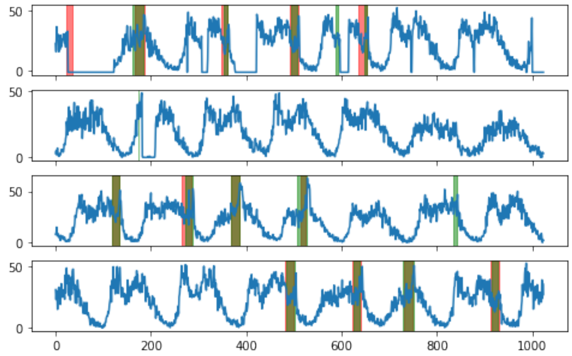

Figure 6: Some snapshots in the Dodgers test case. Red markers represent known events, green markers represent model detection.

5.2 Gasoil Plant Heating Loop Data Set

This data set[3] was originally introduced by [Filonov *et al.*, 2016]. It includes 48 simulated control sequences of a gasoil plant heating loop, which suffered cyber-attacks at some points. All time series have 19 variables and an average length of 204,615. We used the tag DANGER as the indicator of attack events.

We used 30 sequences for model training and the remaining 18 for testing. The training data was randomly cropped into 300 snapshots with length 50,000, and then downsampled to 1,024. A multivariate U-Net ($C = 19$) was created and trained. Among the 18 testing sequences with 22 cyber-attacks in total (14 sequences with 1 attack, 4 with 2 attacks), only 1 attack was missed by detection. There were 3 false alarms.

[2] https://archive.ics.uci.edu/ml/datasets/Dodgers+Loop+Sensor

[3] https://kas.pr/ics-research/dataset_ghl_1

5.3 Synthetic Curves with Unusual Shapes

We created a data set of synthetic curves where each series may have one or several segments with different curve shape from the rest of that series. For example, a series could be mostly smooth except for a segment that is piece-wise linear. We augmented the data set with several augmentation methods mentioned in Section 3.4, so that series behaviors are diverse. We generated 1,400 samples with length 1,024: 700 for training and 700 for testing. Figure 7 shows some examples. The task is to detect unusual segments, which is a more challenging task than the pretraining tasks. Although 700 is larger than the size of data sets in the previous two testing cases, it is still insufficient considering the diversity of time series behaviors over these samples.

We used intersection over union (IoU) score to evaluate the performance of this task. When training a U-Net from scratch, the testing IoU was 50.96%. When we used univariate transfer learning, the testing IoU reached 71.95%. Some results from the transferred model are shown in Figure 7.

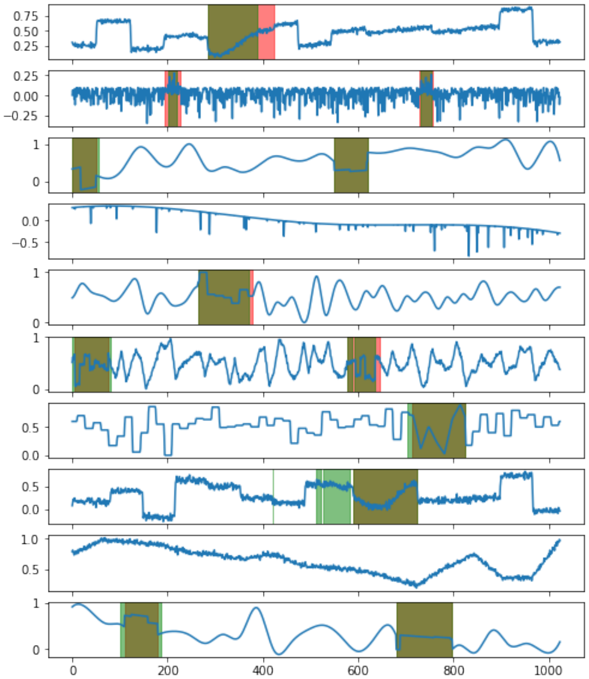

Figure 7: Some examples of synthetic curves with unusual shapes. Red markers represent true anomalies, green markers represent model detection.

5.4 Electromyography (EMG) Data Set

This data set[4] was originally introduced by [Lobov *et al.*, 2018]. It includes 8-channel myographic signals recorded by bracelets worn on 36 subjects' forearms. Each subject performed the same set of seven gestures twice sequentially. The task is to detect different gestures. Precisely speaking, this task is not an anomaly detection task but a segmentation one, because negative labels (i.e. segments with no gesture) are much less frequent than positive labels (i.e. segments with a gesture). However, since the approach we propose is fundamentally a time series segmentation method, we believe this is still a good case for testing. This case is the only testing case assuming a multi-class, single-label scenario, because we know there was at most one gesture performed at a time point.

We trained a U-Net from scratch, and the IoU score evaluated on the test set was 56.61%. We also trained an MU-Net from scratch, and the IoU score was 64.10%. We then transferred the pretrained U-Net to a MU-Net and fine-tuned the weights, the IoU score then reached 70.04%. Figure 8 shows some testing snapshots with results from the transferred model.

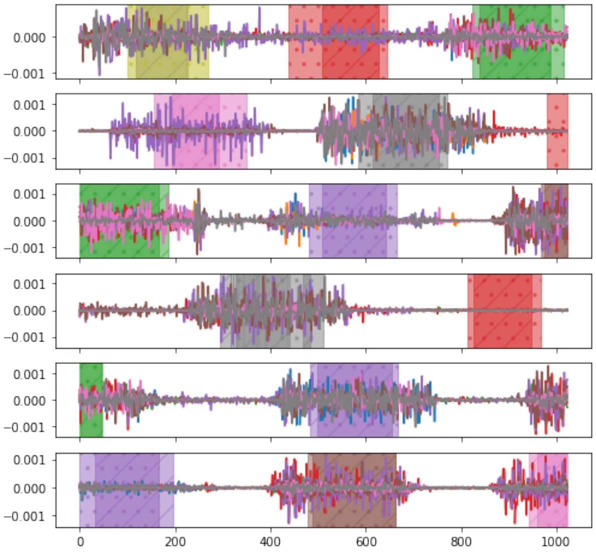

Figure 8: Some snapshots in the EMG test case. Dotted shadows represent true segments, and striped shadows represent model segments. Different colors represent different gestures.

6 Conclusion and Discussion

In this work, we proposed a time series version of the convolutional U-Net for time series anomaly detection. As far as we are aware of, this is the first work to use a CNN-based deep network for time series segmentation in the context of anomaly detection. The architecture was tested with both univariate and multivariate examples and showed satisfactory performance.

To address the challenge of data sparsity that often occurs in real-world anomaly detection tasks, we proposed a transfer learning framework to transfer a U-Net model pretrained on a large-scale univariate time series set to general anomaly detection tasks. In particular, to transfer to a multivariate task, we proposed a new architecture, MU-Net, that may take advantage of the pretrained univariate U-Net. The transfer learning framework was also tested with both univariate and multivariate examples and returned promising results.

[4] http://archive.ics.uci.edu/ml/datasets/EMG+data+for+gestures

One of the primary, inherent challenges in time series anomaly detection is defining ground truth. For time series, delineating exactly when anomalous behavior occurs and when it stops is a fundamental difficulty, as even human experts are likely to differ in their assessments. Additionally, when detecting anomalies in time series, there is the question of what counts as a useful detection. For a typical image segmentation task, the goal is to reproduce segmentation masks as created by human labellers. For image tasks the level of ambiguity is relatively low, i.e. edges are relatively well defined. For time series anomaly detection, the predicted mask and the ground truth mask may have less than perfect overlap, but, operationally, the predicted mask may still serve its purpose of alerting to the presence of a specific type of anomaly. Ultimately this means that the quantifying the goodness of time series anomaly detection via segmentation is difficult.

A transferable model is particularly well-suited for handling the large variety of possible behaviors in time series data. The pretraining dataset was populated with what we considered to be the most fundamental anomalous behaviors in time series data. Because the definition of anomalous behavior is context-dependent, the pretraining dataset was designed to train a model capable of extracting informative, underlying features for various types of time series and anomalies, such that a new anomaly type may be easily learned from a small-scale dataset. This methodology is fundamentally different from most previous work on anomaly detection, which were based on identifying outliers with an explicit or implicit definition. While this scenario is less suited to handling previously unseen anomalies than more traditional outlier detection based methods, its strength lies in its robustness and ability to handle complex signals.

Future work on this topic could include extensive experiments on factors that may impact transfer learning performance, such as [Mahajan *et al.*, 2018] performed for image classification. That paper explored, given specific target tasks, how a pretrained model may perform differently if it was pretrained on ImageNet, or a certain part of ImageNet, or an even larger and more comprehensive data set. The correlation between different types of time series (on both nominal and anomalous behaviors) is even more subtle than that between image classes. We believe that more research in this direction would help us improve the pretraining data set as well as the transfer learning framework, such that the pretrained model would be even more transferable.

Another area of possible future work is to create performance comparisons with benchmark algorithms, such as statistical time series analysis, RNN-based anomaly detection methods, and CNN-based classification methods with sliding windows.

Acknowledgments

We would like to thank Jason Hu and Gunny Liu for their help with synthetic data generation. We also thank Fausto Morales, Pushkar Kumar Jain, and Henry Lin for helpful discussions.

References

[Abraham and Chuang, 1989] Bovas Abraham and Alice Chuang. Outlier detection and time series modeling. *Technometrics*, 31(2):241–248, 1989.

[Ahmed *et al.*, 2007] Tarem Ahmed, Mark Coates, and Anukool Lakhina. Multivariate online anomaly detection using kernel recursive least squares. In *IEEE INFOCOM 2007-26th IEEE International Conference on Computer Communications*, pages 625–633. IEEE, 2007.

[Bianco *et al.*, 2001] Ana Maria Bianco, M Garcia Ben, EJ Martinez, and Víctor J Yohai. Outlier detection in regression models with arima errors using robust estimates. *Journal of Forecasting*, 20(8):565–579, 2001.

[Breunig *et al.*, 2000] Markus M Breunig, Hans-Peter Kriegel, Raymond T Ng, and Jörg Sander. Lof: identifying density-based local outliers. In *ACM sigmod record*, volume 29, pages 93–104. ACM, 2000.

[Chan and Mahoney, 2005] Philip K Chan and Matthew V Mahoney. Modeling multiple time series for anomaly detection. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. IEEE, 2005.

[Çiçek *et al.*, 2016] Özgün Çiçek, Ahmed Abdulkadir, Soeren S Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: learning dense volumetric segmentation from sparse annotation. In *International conference on medical image computing and computer-assisted intervention*, pages 424–432. Springer, 2016.

[Cui *et al.*, 2016] Zhicheng Cui, Wenlin Chen, and Yixin Chen. Multi-scale convolutional neural networks for time series classification. *arXiv preprint arXiv:1603.06995*, 2016.

[Da Xu *et al.*, 2014] Li Da Xu, Wu He, and Shancang Li. Internet of things in industries: A survey. *IEEE Transactions on industrial informatics*, 10(4):2233–2243, 2014.

[Dalmış *et al.*, 2017] Mehmet Ufuk Dalmış, Geert Litjens, Katharina Holland, Arnaud Setio, Ritse Mann, Nico Karssemeijer, and Albert Gubern-Mérida. Using deep learning to segment breast and fibroglandular tissue in mri volumes. *Medical physics*, 44(2):533–546, 2017.

[Dong *et al.*, 2017] Hao Dong, Guang Yang, Fangde Liu, Yuanhan Mo, and Yike Guo. Automatic brain tumor detection and segmentation using u-net based fully convolutional networks. In *annual conference on medical image understanding and analysis*, pages 506–517. Springer, 2017.

[Filonov *et al.*, 2016] Pavel Filonov, Andrey Lavrentyev, and Artem Vorontsov. Multivariate industrial time series with cyber-attack simulation: Fault detection using an lstm-based predictive data model. *arXiv preprint arXiv:1612.06676*, 2016.

[Ihler *et al.*, 2006] Alexander Ihler, Jon Hutchins, and Padhraic Smyth. Adaptive event detection with time-varying poisson processes. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 207–216. ACM, 2006.

[Karchevskiy *et al.*, 2018] Mikhail Karchevskiy, Insaf Ashrapov, and Leonid Kozinkin. Automatic salt deposits segmentation: A deep learning approach. *arXiv preprint arXiv:1812.01429*, 2018.

[Keogh *et al.*, 2005] Eamonn Keogh, Jessica Lin, and Ada Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*, pages 8–pp. Ieee, 2005.

[Kim *et al.*, 2016] Jihyun Kim, Jaehyun Kim, Huong Le Thi Thu, and Howon Kim. Long short term memory recurrent neural network classifier for intrusion detection. In *2016 International Conference on Platform Technology and Service (PlatCon)*, pages 1–5. IEEE, 2016.

[Liu *et al.*, 2008] Fei Tony Liu, Kai Ming Ting, and Zhi-Hua Zhou. Isolation forest. In *2008 Eighth IEEE International Conference on Data Mining*, pages 413–422. IEEE, 2008.

[Lobov *et al.*, 2018] Sergey Lobov, Nadia Krilova, Innokentiy Kastalskiy, Victor Kazantsev, and Valeri Makarov. Latent factors limiting the performance of semg-interfaces. *Sensors*, 18(4):1122, 2018.

[Long *et al.*, 2015] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[Mahajan *et al.*, 2018] Dhruv Mahajan, Ross Girshick, Vignesh Ramanathan, Kaiming He, Manohar Paluri, Yixuan Li, Ashwin Bharambe, and Laurens van der Maaten. Exploring the limits of weakly supervised pretraining. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 181–196, 2018.

[Malhotra *et al.*, 2015] Pankaj Malhotra, Lovekesh Vig, Gautam Shroff, and Puneet Agarwal. Long short term memory networks for anomaly detection in time series. In *Proceedings*, page 89. Presses universitaires de Louvain, 2015.

[Rajpurkar *et al.*, 2017] Pranav Rajpurkar, Awni Y Hannun, Masoumeh Haghpanahi, Codie Bourn, and Andrew Y Ng. Cardiologist-level arrhythmia detection with convolutional neural networks. *arXiv preprint arXiv:1707.01836*, 2017.

[Ringberg *et al.*, 2007] Haakon Ringberg, Augustin Soule, Jennifer Rexford, and Christophe Diot. Sensitivity of pca for traffic anomaly detection. In *ACM SIGMETRICS Performance Evaluation Review*, volume 35, pages 109–120. ACM, 2007.

[Ronneberger *et al.*, 2015] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[Schölkopf *et al.*, 2001] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[Siam *et al.*, 2017] Mennatullah Siam, Sara Elkerdawy, Martin Jagersand, and Senthil Yogamani. Deep semantic segmentation for automated driving: Taxonomy, roadmap and challenges. In *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–8. IEEE, 2017.

[Um *et al.*, 2017] Terry Taewoong Um, Franz Michael Josef Pfister, Daniel Pichler, Satoshi Endo, Muriel Lang, Sandra Hirche, Urban Fietzek, and Dana Kulić. Data augmentation of wearable sensor data for parkinson's disease monitoring using convolutional neural networks. *arXiv preprint arXiv:1706.00527*, 2017.

[Wang *et al.*, 2017] Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International joint conference on neural networks (IJCNN)*, pages 1578–1585. IEEE, 2017.

[Yang *et al.*, 2015] Jianbo Yang, Minh Nhut Nguyen, Phyo Phyo San, Xiao Li Li, and Shonali Krishnaswamy. Deep convolutional neural networks on multichannel time series for human activity recognition. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[Yao *et al.*, 2018] Wei Yao, Zhigang Zeng, Cheng Lian, and Huiming Tang. Pixel-wise regression using u-net and its application on pansharpening. *Neurocomputing*, 312:364–371, October 2018.

[Yin *et al.*, 2017] Chuanlong Yin, Yuefei Zhu, Jinlong Fei, and Xinzheng He. A deep learning approach for intrusion detection using recurrent neural networks. *Ieee Access*, 5:21954–21961, 2017.

[Zheng *et al.*, 2014] Yi Zheng, Qi Liu, Enhong Chen, Yong Ge, and J Leon Zhao. Time series classification using multi-channels deep convolutional neural networks. In *International Conference on Web-Age Information Management*, pages 298–310. Springer, 2014.