NEURON-NET: SIAMESE NETWORK FOR ANOMALY DETECTION

Technical Report

Karel Durkota, Michal Linda, Martin Ludvík, Jan Tožička

NeuronSW SE
{karel.durkota, michal.linda, martin.ludvik, jan.tozicka}@neuronsw.com

ABSTRACT

This paper describes our submission to the DCASE 2020 challenge Task 2 ”Unsupervised Detection of Anomalous Sounds for Machine Condition Monitoring.” Acoustic-based machine condition monitoring is a challenging task with a very unbalanced training dataset. In this submission, we combine the Siamese Network feature extractor with KNN anomaly detection algorithm. Experiment results prove it to be a viable approach with an average AUC 85 and 0.1-AUC of 77.1. This is a novel approach and have not been used by NeuronSW SE so far.

Index Terms— Predictive Maintenance, Anomaly Detection, Siamese Network

1. INTRODUCTION

Machine condition monitoring is an essential component of predictive maintenance. It allows to schedule maintenance work to fix machine problems in the earliest stages and thus reducing maintenance costs and preventing consequential damages. Acoustic emission monitoring can be used for machine condition analysis and prognosis. ISO 22096[1] suggests that the nature of acoustic emissions can be used even without an understanding of the operating mechanics of the monitored machine. The recent progress in AI allows us to create an automatic machine condition monitoring system. To allow a large scale, we need a system that does not require the knowledge of the monitored system's operation mechanics. Nevertheless, it is impossible to collect all possible failures for a newly monitored machine without such knowledge. In practice, it is exceptional to get even any example of a failed state. Unfortunately, most of the recently developed AI methods require a huge amount of well-labeled examples, which makes them unusable for the task of machine condition monitoring. Task of learning from a few or even just one sample is called a few or one-shot learning. On the other hand, anomaly detection methods seem suitable for this problem as it lacks the samples representing the failure modes of the monitored machines.

Our approach combines Siamese Network[1] and KNN anomaly detector. First, the Siamese Network is trained to extract needed features for sound classification from audio spectrograms. Then, KNN anomaly detector is trained on the extracted features of the audio samples. This is a novel approach that have not been used by NeuronSW SE so far.

[1]https://www.sis.se/api/document/preview/908883/

2. OUR APPROACH

Our approach can be divided into three phases: (1) converting audio to spectrograms, (2) training Siamese Network as a classifier, and (3) train KNN, using Siamese Network encoder, for anomaly detection. We will describe each part separately.

2.1. Audio Transformation

First, we transform all audio samples into spectrograms using STFT and taking the absolute value of the complex values. We further convert spectrograms to decibels and normalize each sample to have mean 0 and standard deviation 1. Finally, we resize each spectrogram to size 128x128x1 to speed up the training.

2.2. Siamese Network

A standard Siamese Network (see [1] for more details) consists of two parts: (1) two *encoders* and (2) an *aggregator*. *Encoders* transform the inputs into multi-dimensional latent space using the same weights. The *aggregator* then computes the distance of the two encoded samples and scores how similar or different they are. The Siamese Network is trained to tell whether the presented two samples come from the same class or not. During the training, random pairs of samples are input into the network. If both samples come from the same machine type and machine id (e.g., *slider_id_00* and *slider_id_00*), the Siamese Network minimizes their encoded distance. While in the case that the two samples come from different classes (e.g., *slider_id_00* and *fan_id_00*; or *slider_id_00* and *slider_id_01*), the Siamese Networks maximizes their encoded distance.

The *encoder's* architecture is as follows:

- Input (128x128)
- Conv (64 @ 9x9) + BN + MaxPooling2D + ReLU
- Conv (128 @ 7x7) + BN + MaxPooling2D + ReLU
- Conv (256 @ 5x5) + BN + MaxPooling2D + ReLU
- Conv (512 @ 3x3) + BN + MaxPooling2D + ReLU
- Dense (32) + ReLU
- Output (32)

The *aggregator* computes weighted L1 distance between extracted features h_1 and h_2 combined with sigmoid activation function Formally:

$$\sigma\left(\sum_{j}^{N=32} \alpha_i |h_1^j - h_2^j|\right)$$

Training After the accuracy reaches a value close to 1, we continued with fine-tune training of six individual Siamese Networks, one per each machine type (fan, slider, valve, pump, ToyCar, and ToyConveyor). Here, each classifier is trained to distinguish id's of a specific machine type.

To train Siamese Network, we use parameters as follows: 1000 epochs, batch size of 64, Stochastic Gradient Descent optimizer with initial learning rate 0.05 and for fine-tuning 0.02.

2.2.1. *Submission 1*

In our first submission, the neural network follows exactly the structure as described above.

2.2.2. *Submission 2*

In our second submission differ from submission 1 by omitting BN (Batch Normalization) in the encoder's architecture.

2.2.3. *Submission 3*

Finally, our third submission differs from submission 1 in the encoder's architecture. In submission 3 we used the same encoder architecture as described in [1]. Particularly:

- Input (128x128)
- Conv (64 @ 10x10) + MaxPooling2D + ReLU
- Conv (128 @ 7x7) + BN + MaxPooling2D + ReLU
- Conv (128 @ 4x4) + BN + MaxPooling2D + ReLU
- Conv (256 @ 4x4) + BN + MaxPooling2D + ReLU
- Dense (32) + ReLU
- Output (32)

2.3. KNN Anomaly Detection

Finally, once the Siamese network is trained to distinguish machine id's, we use its *encoder* as a feature extractor. We extract 32-dimensional features from all samples using Siamese Network's encoder, and train standard KNN anomaly detection [2] using PyOD [2] library.

To train KNN we use the default setting from PyOD with n_neighbors 5, method large, radius 1.0, and leaf_size 30.

3. RESULTS

To evaluate our approach, we used ToyADMOS [3] and MIMII [4] datasets, which are part of DCASE2020 Task-2 Challenge [5]. In Table 1 we summarize the AUCs and pAUCs for $p = 0.1$ of all three submissions. All three submissions outperform the baseline AutoEncoder-based approach [5] which has AUC 73% and pAUC 59%. In our results, submission 1 has highest AUC 85.9% and submission 3 has highest pAUC 77.6%; while submission 2 is somewhere in between.

4. CONCLUSION

Our approach combines two existing algorithms, the Siamese Network and KNN anomaly detector. It reached AUC around 85% and

	1. submis.		2. submis.		3. submis.	
problem	AUC	pAUC	AUC	pAUC	AUC	pAUC
fan0	58.0	51.4	54.2	50.6	53.8	49.1
fan2	96.6	85.8	93.1	72.6	91.0	77.9
fan4	64.1	59.3	58.1	51.9	64.8	58.9
fan6	99.8	99.2	100.0	99.9	100.0	100.0
pump0	79.2	68.1	75.8	58.6	74.1	61.3
pump2	65.2	62.3	70.6	57.1	61.0	51.3
pump4	99.8	98.7	99.5	97.5	98.9	94.2
pump6	85.8	70.7	64.9	61.1	67.9	62.3
slider0	99.7	98.6	99.0	96.2	99.7	98.7
slider2	90.6	71.7	96.5	83.6	94.7	82.9
slider4	96.2	83.0	100.0	100.0	95.5	85.1
slider6	99.4	96.9	96.9	87.0	98.6	92.7
valve0	97.4	90.2	93.5	88.4	99.5	97.7
valve2	96.0	86.6	99.0	96.4	99.7	98.5
valve4	93.7	75.1	98.6	94.9	96.9	91.4
valve6	95.7	79.3	90.9	72.9	95.7	82.5
ToyCar1	78.8	71.0	85.3	72.5	83.1	68.5
ToyCar2	87.9	78.2	91.5	84.0	92.1	84.0
ToyCar3	88.2	74.7	96.2	89.8	94.5	86.4
ToyCar4	98.6	94.4	100.0	100.0	100.0	100.0
ToyCon1	79.6	68.3	74.4	62.1	74.1	59.3
ToyCon2	58.7	53.7	55.2	51.3	59.9	50.2
ToyCon3	65.8	56.1	64.3	53.8	59.3	52.5
Average	85.9	77.1	85.1	77.5	85.0	77.6

Table 1: AUC and pAUC of development dataset.

[2] https://pyod.readthedocs.io/

pAUC 77% for $p = 0.1$, outperforming baseline AutoEncoder solution by more than 11% and 18%, respectively.

In future, we plan to enhance training using data augmentation. We also plan to evaluate Siamese Network universality, i.e. when we train it on all but one type of machine, how well it performs on the omitted type of machine.

5. ACKNOWLEDGMENT

We want to thank NeuronSW SE company, for providing computational resources to train the algorithms.

6. REFERENCES

[1] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, vol. 2. Lille, 2015.

[2] S. Ramaswamy, R. Rastogi, and K. Shim, "Efficient algorithms for mining outliers from large data sets," in *Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, 2000, pp. 427–438.

[3] Y. Koizumi, S. Saito, H. Uematsu, N. Harada, and K. Imoto, "ToyADMOS: A dataset of miniature-machine operating sounds for anomalous sound detection," in *Proceedings of IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA)*, November 2019, pp. 308–312. [Online]. Available: https://ieeexplore.ieee.org/document/8937164

[4] H. Purohit, R. Tanabe, T. Ichige, T. Endo, Y. Nikaido, K. Suefusa, and Y. Kawaguchi, "MIMII Dataset: Sound dataset for malfunctioning industrial machine investigation and inspection," in *Proceedings of the Detection and Classification of Acoustic Scenes and Events 2019 Workshop (DCASE2019)*, November 2019, pp. 209–213. [Online]. Available: http://dcase.community/documents/workshop2019/proceedings/DCASE2019Workshop_Purohit_21.pdf

[5] Y. Koizumi, Y. Kawaguchi, K. Imoto, T. Nakamura, Y. Nikaido, R. Tanabe, H. Purohit, K. Suefusa, T. Endo, M. Yasuda, and N. Harada, "Description and discussion on DCASE2020 challenge task2: Unsupervised anomalous sound detection for machine condition monitoring," in *arXiv e-prints: 2006.05822*, June 2020, pp. 1–4. [Online]. Available: https://arxiv.org/abs/2006.05822