ISBN 13 978-952-5183-62-7

Mikhail Zolotukhin*, Riku Immonen, Pyry Kotilainen, and Timo Hämäläinen

Tiny machine learning for fault detection

Abstract: Never before has machine learning been characterized by such innovative waves of technology. At the same time, the rise of low-budget single-board microcontrollers allows service providers to meet privacy, low latency and energy efficiency requirements by deploying artificial intelligence on the edge. In our research, we implement, train and deploy various supervised and unsupervised machine learning models on tiny boards for several real-life data analysis applications.

Keywords: TinyML, deep learning, anomaly detection

***Corresponding Author: Mikhail Zolotukhin:** University of Jyväskylä, E-mail: mizolotu@jyu.fi
Riku Immonen: University of Jyväskylä, E-mail: rijuimmo@jyu.fi
Pyry Kotilainen: University of Jyväskylä, E-mail: pyjopeko@jyu.fi
Timo Hämäläinen: University of Jyväskylä, E-mail: timoh@jyu.fi

1 Introduction

Artificial intelligence (AI) and machine learning (ML) are revolutionising almost every industry with a seemingly endless list of applications ranging from object recognition in autonomous vehicles to helping doctors detect and diagnose diseases. At the same time, the recent progress in development of low-budget sensors and single-board computers has made it possible to deploy tiny machine learning models not only for inference but also for training [1]. Furthermore, according to IoT Analytics, by the end of 2022, the market for the internet-of-things (IoT) is expected to grow by 18% up to 14.4 billion active connections. Thus, increasing computing and connectivity capabilities of smart devices allow for their usage on the edge in order to decrease latency and increase availability of ML-driven services and applications.

In our research, we focus on implementing state-of-the-art machine learning models on tiny IoT devices which can be then deployed on the edge. In particular, in this study, we concentrate on two following use cases: wind speed estimation and anomalous vibration detection. The rest of the document is organised as follows. The use case scenarios studied are overviewed in Section 2. Section 3 highlights the algorithms and devices used. Numerical evaluation results are presented in Section 4. Section 5 concludes the study and outlines future work.

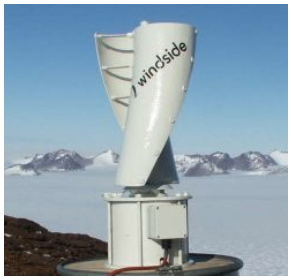
2 Aims

As mentioned in the introduction, we focus on tiny ML for wind speed estimation and anomalous vibration detection. In the former case, the estimations are carried out using electric current time-series observed during the recent time interval on a wind turbine similar to the one shown in Fig. 1a. The rationale behind this task is that wind speed estimations obtained can be monitored by the turbine operators in real time for more efficient power extraction and detection of anomalous patterns which may be indicative of a fault.

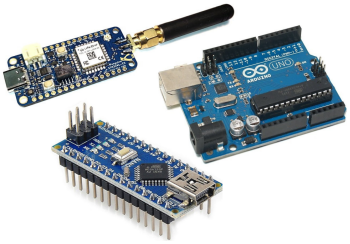
In the second use case, we study the problem of outlier detection in the data recorded by an accelerometer. The task is to train a model of normal behavior which can then be used to classify anomalous vibrations that significantly deviate from the norms described by the model. In distinction from the previous case, the training is expected to be carried out on the device itself which can be useful in many real-life scenarios when no training data is available in advance. The test environments for this use case are shown in Fig. 1d and 1e.

3 Methods and materials

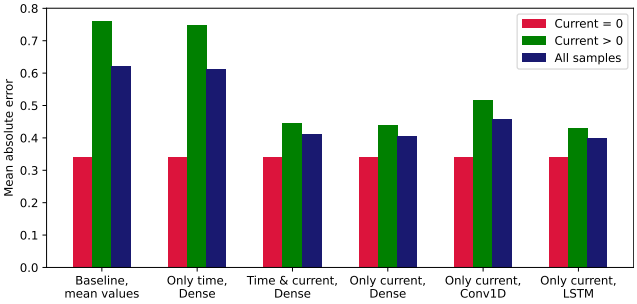
Speaking of the algorithms employed in our research, we mostly rely on supervised and unsupervised deep learning for wind speed estimation and anomalous vibration detection respectively. Deep learning models use the first layers to find compact low-dimensional representations of high-dimensional data whereas later layers are responsible for achievement of the task given. The most widely used deep learning architectures for time series data include convolutional and recurrent neural networks. Other methods used that are worth mentioning include one-class classification with deep support vector description (DeepSVDD) [2] and stream clustering techniques [3]. The aforementioned AI/ML models are implemented to run on various tiny single-board computers (see Fig. 1b). The application prototypes are first tested using Arduino Nano and then the code is ported to LoRa E5 which under ultra-low power consumption constraints is able to achieve a transmission range of up to 10 km using LoRaWAN protocol.



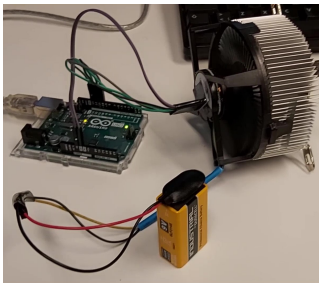
(a) Wind turbine example.



(b) Boards used in the study.

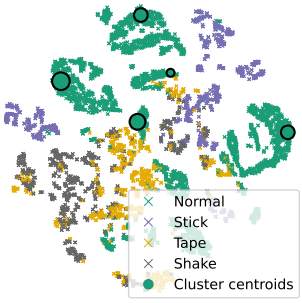


(c) Comparison of deep learning models for wind speed estimation.

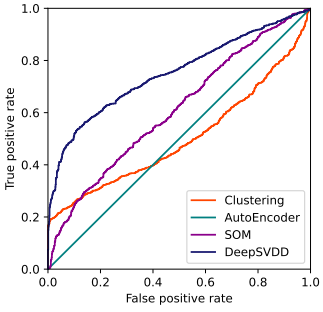


(d) Fan environment.

(e) Bearing environment.



(f) Fan data clustering results.



(g) Bearing model ROCs.

Fig. 1. Materials used in the research and some of the results obtained. Wind turbine similar to the one used in the study is shown in Fig. (a). Fig. (b) shows the single-board computers on which ML models have been deployed: Lora E5, Arduino Nano an Arduino Uno. Comparison of several tiny deep learning models tested in the wind speed estimation use case can be seen in Fig. (c). Fan and bearing environments for the anomalous vibration detection use case are shown respectively in Fig. (d) and (e). An example of stream clustering algorithm, namely ScalableKmeans++ with 9 clusters, applied to the data generated in the fan environment can be seen in Fig. (f). Fig. (g) shows ROC curves for several deep learning models tested for anomaly detection in the bearing environment.

4 Results

In the wind speed estimation use case, we first train and evaluate several deep neural network models using fully-connected (Dense), convolutional (Conv1D) and recurrent long short-term memory (LSTM) layers (see Fig. 1c). Using timestamps as additional input features has not provided any accuracy gains. The model with the lowest absolute error, which is just under 0.4 m/s, can then be deployed on the edge.

In the fan environment, anomalous vibrations are generated by shaking the table on which the fan is located, attaching a piece of tape on one of the fan blades, or poking a stick into the blades. The resulting anomalies can be distinguished from the normal samples with the help of a stream clustering algorithm, e.g. CluStream and ScalableKmeans++: the samples that are far from the clusters obtained for the normal data are classified as outliers (see Fig. 1f). This approach allows us to reach 77% accuracy with no false alarms.

Unfortunately, the clustering approach fails to accurately distinguish between samples generated using normal and faulty bearings. For this reason, the following neural network based models are implemented and tested: autoencoder, self-organising map (SOM) and DeepSVDD. The latter outperforms analogies in terms of true and false positive rates according to the preliminary results which can be found in Fig. 1g.

5 Future work

We are planning to improve accuracy of the methods tested as well as develop novel algorithms which can run on microcontrollers for real-time data analysis.

References

[1] Ren H, Anicic D, Runkler TA. Tinyol: Tinyml with online-learning on microcontrollers. In: *IJCNN*. IEEE. 2021; pp. 1–8.

[2] Ruff L, Vandermeulen R, Goernitz N, Deecke L, Siddiqui SA, Binder A, Müller E, Kloft M. Deep one-class classification. In: *ICML*. PMLR. 2018; pp. 4393–4402.

[3] Silva JA, Faria ER, Barros RC, Hruschka ER, Carvalho ACd, Gama J. Data stream clustering: A survey. *ACM Computing Surveys (CSUR)*. 2013;46(1):1–31.