DTU Compute
Department of Applied Mathematics and Computer Science

Thesis title

by Mohammad Amin Hasanpour

PhD Thesis

Thesis title
by Mohammad Amin Hasanpour

PhD Thesis
January, 2026

By
Mohammad Amin Hasanpour

Cover photo: Vibeke Hempler, 2012
Published by: DTU, Department of Applied Mathematics and Computer Science, Richard Petersens Plads, Building 322, 2800 Kgs. Lyngby Denmark
www.compute.dtu.dk
ISSN: [0000-0000] (electronic version)
ISBN: [000-00-0000-000-0] (electronic version)

ISSN: [0000-0000] (printed version)
ISBN: [000-00-0000-000-0] (printed version)

English Abstract

English abstract goes here.

Dansk Abstrakt

Danish abstract goes here.

Acknowledgements

Acknowledgements go here.

Tool Acknowledgements

In writing this thesis, I have been making use of Generative AI services such as Grammarly and ChatGPT to improve writing and make grammatical corrections.

Contents

English Abstract ii
Dansk Abstrakt iii
Acknowledgements iv

1 Introduction 1

2 Paper A: Pump Cavitation Detection with Machine Learning: A Comparative Study of SVM and Deep Learning 3
2.1 Paper Abstract 3
2.2 Introduction 3
2.3 Dataset and Problem Formulation 4
2.4 Methodology 5
2.5 Results 8
2.6 Conclusion 13

3 Paper B: EdgeMark: An automation and benchmarking system for embedded artificial intelligence tools 15

4 Paper C: A Survey of Quantization Techniques in Embedded AI Toolchains 17

5 Conclusions and Future Directions 19

Bibliography 21

1 Introduction

Introduction goes here.

2 Paper A: Pump Cavitation Detection with Machine Learning: A Comparative Study of SVM and Deep Learning

2.1 Paper Abstract

In the pursuit of enhancing industrial pump reliability and efficiency, this paper addresses the challenging issue of pump cavitation detection through the innovative application of Machine Learning (ML) techniques. Cavitation, a prevalent problem in pumps, significantly compromises their performance, causing damage and operational inefficiencies. Traditionally, cavitation detection has relied on numerical analysis and signal processing methods, which, despite their merit, often fall short in real-world applications due to their requirement for extensive domain knowledge and controlled operational conditions. This study diverges from conventional approaches by harnessing the power of ML to predict cavitation occurrences in pumps under varying real-world conditions with high accuracy.

We present an analysis of a cavitation dataset compiled by the Danish pump manufacturer Grundfos, which includes vibration data from 297 experiments on seven different pumps, using both traditional ML models, specifically Support Vector Machine (SVM), and advanced Deep Learning (DL) techniques. Our methodology includes a detailed examination of the dataset, feature engineering, target definition, problem formulation, model design, and rigorous model testing on target hardware. Remarkably, our study not only demonstrates that ML models, particularly DL models, can adaptively and accurately predict cavitation but also emphasizes the importance of testing these models on target hardware to ensure their practical applicability. This work is accompanied by an open-source implementation.

2.2 Introduction

Pumps are indispensable in various industrial processes, including water treatment, oil and gas production, and manufacturing. Their efficient operation is crucial for ensuring uninterrupted flow in these systems. However, pumps are susceptible to numerous issues that can compromise their performance and reliability. Among these, cavitation stands out as a particularly detrimental phenomenon that can lead to significant damage and operational inefficiencies [1].

Cavitation occurs when the pressure in the pump falls below the liquid's vapor pressure, leading to the formation of vapor bubbles [2]. These bubbles collapse violently when carried to regions of higher pressure, causing shock waves that can erode the pump's components and degrade its performance. The consequences of cavitation extend beyond repair costs, as it can also cause system downtime and energy loss.

While much effort has been devoted to detecting and predicting cavitation in pumps, most of the methods are based on numerical analysis and signal processing techniques [3, 4, 5, 6, 7, 8]. These methods often require extensive domain knowledge and typically work best in controlled environments, where the pump's operating conditions are well understood. However, in real-world scenarios, pumps operate under varying conditions, making it challenging to predict cavitation accurately. This is where ML models can help. By training on data collected from pumps under different conditions, these models can

learn the patterns associated with cavitation and predict its occurrence with high accuracy. ML models are more adaptable, can handle the complexity of real-world data, and can generalize well to unseen conditions.

With the spread of ML techniques, many researchers have started to explore data-driven approaches to predict cavitation in pumps [9, 10, 11, 12, 13]. These approaches have shown very promising results on the efficacy of ML, but their key limitation is that the cavitation detection is done offline.

Different to the related work, we opt for the approach of deploying these ML models directly on embedded systems integrated into pumps. This methodology offers several key advantages. First, it preserves privacy by processing data locally. Second, it eliminates reliance on internet connectivity, which can be inconsistent or unavailable in many industrial settings. Furthermore, processing data on the device leads to lower energy consumption compared to sending data to be processed elsewhere, conserving energy and reducing heat generation. This *Embedded AI* approach necessitates attention to the model's memory requirement and performance, which we address in our study.

In summary, this paper presents a study of a cavitation dataset compiled by the Danish pump manufacturer Grundfos (hereafter referred to as the Grundfos Cavitation dataset), which is unique in size and quality. Specifically, we focus on the problem of pump cavitation detection both as a binary classification problem (i.e. detection of the presence of cavitation) and as a regression problem (i.e. prediction of the amount of cavitation). Aiming to detect the cavitation on pumps, we design models based on SVMs and Deep Neural Networks, and we explore various optimization techniques, including feature optimization and quantization, that reduce the amount of required processing resources. Our experimental results demonstrate that both SVMs and neural networks can provide accurate predictions, with the choice between them depending on the application's specific needs; SVMs offer speed and efficiency, whereas neural networks provide broader functionalities.

2.3 Dataset and Problem Formulation

The Grundfos Cavitation dataset is the result of Grundfos' effort in testing the company's centrifugal pumps in laboratory conditions. To the best of our knowledge, this dataset is unique in size, number of pumps tested, and quality of data. It consists of vibration data from 297 recordings (or experiments) on seven pumps. In each experiment, the corresponding values of flow, pressure, and temperature were captured along with a 30 second long recording of the vibration data sampled at 48 kHz. The settings are kept fixed in each recording but changed between recordings. The pumps contain several stages and are installed vertically. The vibration sensor is located at the top of the pump, while the cavitation is happening at the bottom. This escalates the impact of the pump's stages.

The Net Positive Suction Head (`NPSH`) is an important concept in the operation of centrifugal pumps. `NPSH` is used to measure the absolute pressure available in the pump's suction line at the pump impeller's inlet relative to the vapor pressure of the pumped liquid at the operating temperature. It is a measure of how close the fluid at a given point is to boiling, and it is used to ensure that the pump can operate without causing cavitation. The `NPSH` curve depends on inlet pressure, flow rate, media temperature, inlet pipe diameter, and a few other minor parameters.

`NPSH` required (`NPSHr`) is the minimum pressure required at the suction port of the pump to keep the pump from cavitating. The `NPSHr` is determined by the pump design and operating speed, and it increases with the flow rate. At any point, the pump has an actual

pressure available at the pump suction, called NPSH available (NPSHa), which is a property of the system in which the pump is installed and varies with the system configuration and operating conditions [2]. NPSHa should be greater than NPSHr to prevent cavitation, but a safety margin is usually considered, called NPSHm, and the pump should operate above this margin. NPSHm is described in (2.1). As also seen in (2.1), we defined the Cavitation Factor (CF) as the ratio of NPSHa to NPSHm. Values of CF below one indicate that the pump is cavitating.

$$\mathtt{CF} = \frac{\mathtt{NPSHa}}{\mathtt{NPSHm}}, \quad \mathtt{NPSHm} = \mathsf{max}(\mathtt{NPSHr} + 0.5, 1.2 \cdot \mathtt{NPSHr}) \tag{2.1}$$

By knowing the properties and NPSHr curve of each pump and controlling the flow rate and the inlet pressure, the engineers can estimate the NPSHa and derive the CF. Since the flow and pressure sensors are very expensive, it is preferred to detect cavitation based on the vibration data. In each experiment, the flow and pressure are kept fixed, and the voltage of the vibration sensor is logged. The voltage is then converted to acceleration, and in this study, we further compute its Fast Fourier Transform (FFT). The FFT of the acceleration data is the main feature of the dataset and is used to predict the CF.

Fig. 2.1 illustrates the distribution of recordings and the CF values in the dataset. Fig. 2.2 shows some examples of the FFT of the acceleration data and the corresponding CF values. The figure suggests that, to a certain extent, it might be possible for us as humans to predict the CF value using the FFT information. However, this prediction is not straightforward or highly accurate, as the behavior appears similar across both large and small CF values. The reason for this is that the pumps stall due to the little inlet-pressure in the lab setting. Additionally, certain pumps exhibit unique behavior in specific instances, further complicating the prediction process. These observations suggest that a ML model can be beneficial in predicting the CF based on the FFT of the acceleration, and also might give us good insights if we intend to exploit useful features from the data.

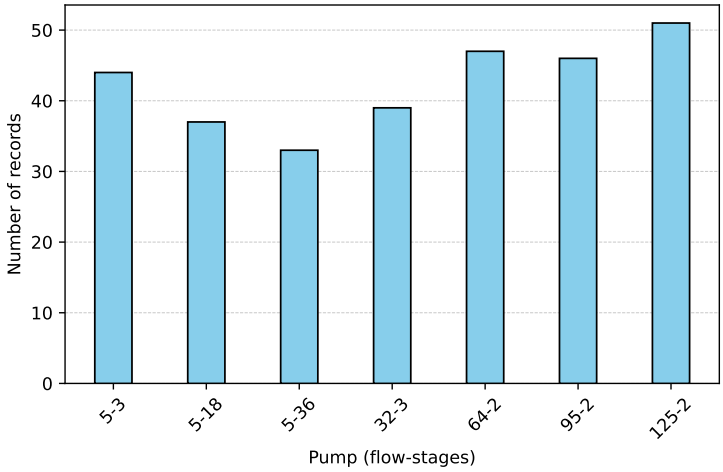
Various problems can be formulated based on this dataset. First, we characterize the problem as a binary classification that has its own use cases and might be relevant when the utilized algorithm only supports classification problems. Then, we change our approach and define the problem as regression. In addition to simple regression, we define a more competent regression problem to focus on the values close to one, and we'll show that this approach can obtain better results even if we put the trained model in a classification situation.

2.4 Methodology

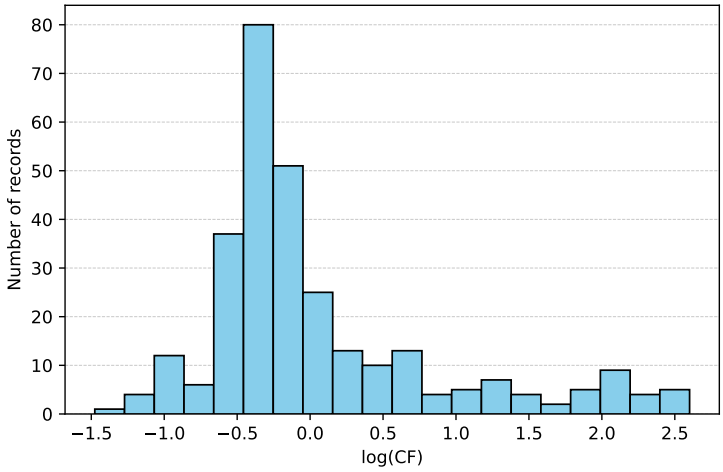
In this study, we used Support Vector Machine (SVM) and deep learning models to solve the Grundfos Cavitation dataset. We considered the support vector machine as a traditional machine learning algorithm that is computationally lightweight compared to deep learning models and still powerful enough to solve many real-world problems. Furthermore, several deep learning models were utilized in this study to compare with the performance of SVMs and benefit from their advantages. The codebase of this study is publicly available [14].

2.4.1 Support Vector Machine

A support vector machine is a powerful and versatile supervised machine learning algorithm used for both classification and regression tasks. It is particularly well-suited for the classification of complex but small- or medium-sized datasets [15]. The basic format of SVM can do linear classification by finding the hyperplane that best separates the classes. However, in many real-world problems, the data is not linearly separable. In such cases,



(a) Number of records for each pump



(b) Distribution of CFs

Figure 2.1: Distribution of dataset elements among different pumps and different values of CF.

SVM uses a kernel trick to transform the input space to a higher dimensional space where the data can be linearly separated. The most common kernels used in SVM are linear, polynomial, and radial basis function (RBF) kernels. In addition to linear kernel, we tested polynomial and RBF kernels, but the performance degraded significantly. Therefore, we only report the results of the linear kernel in this paper.

Each pump has its own attributes, and its vibration data can be completely different from that of other pumps, even after normalization. Therefore, employing a single SVM model for all pumps might not be the best approach. To address this issue, we trained a separate SVM model for each pump and tested the performance of each model. The reporting value will be the average performance of these models. To make it more concrete, we also trained a single SVM model for all pumps and have seen the test accuracy drop from 91.76% to 80.96%.

In the simplest format, the input features are the vibration data captured sequentially in

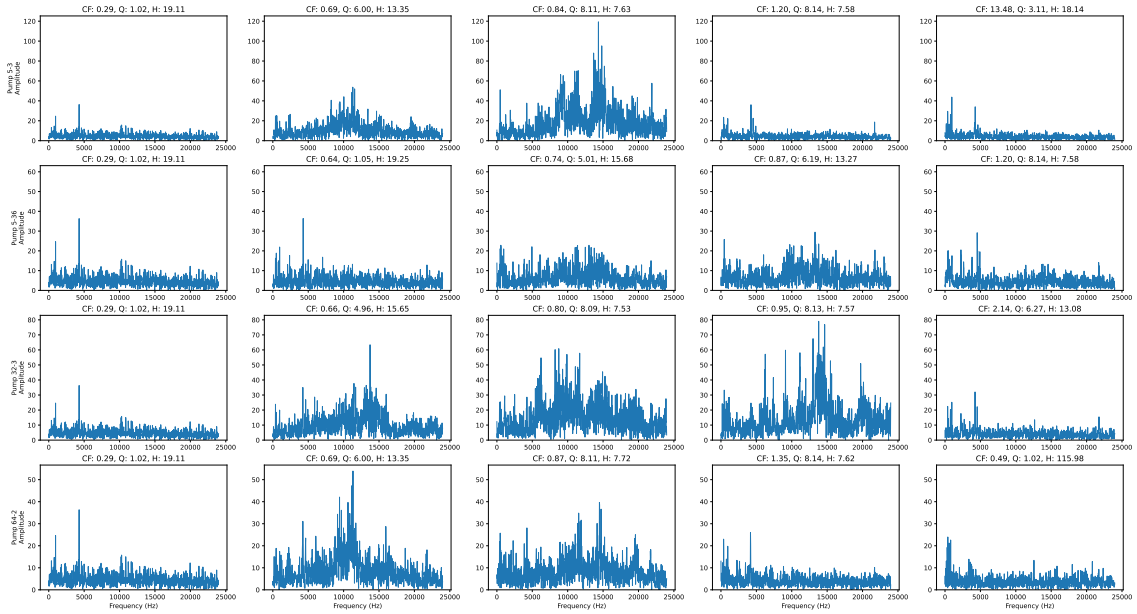


Figure 2.2: Examples of the data. Each row belongs to a specific pump. In the header of each plot, CF is the cavitation factor, Q is the flow (m^3/h), and H is the fluid-independent pressure (m). From left to right in each row, the CF is increasing, i.e., we are going from severe cavitation situations to cavitation free situations, $\texttt{CF} = 1$ being the border of cavitating.

time. Given the high sampling frequency of vibration sensors, the resulting input data can become extensive, leading to increased computational costs and energy consumption for the SVM model. This can be addressed by downsampling the data to the point that the model is still doing well [16]. Furthermore, the feature space constructed from the vibration data may not be well-suited for linear separation. These challenges suggest a need for a feature engineering step to extract the important features from the raw data. In this regard, based on the analysis discussed in Section 2.5, several properties (90th percentile, energy, and standard deviation) of the whole or parts of data were extracted and used to train the SVM models. This approach showed slightly better performance, indicating a better feature space for linear separation, and also can reduce the feature size significantly.

2.4.2 Deep Learning

Deep learning models are capable of automatically discovering the representations needed for feature detection or classification from raw data. This eliminates the need for manual feature extraction, which is one of the key advantages of deep learning over traditional machine learning techniques. It is also capable of solving much more complex problems. For example, in our case, instead of training a separate model on each pump, we were able to train a single model on all of them and reach the same level of accuracy on individual pumps. Still, in most cases, these advantages come with a cost of higher computational complexity, memory usage, and energy consumption compared to traditional machine learning.

Since the raw data we'll feed to the deep learning models is large, naively employing fully connected layers would lead to an excessively high number of parameters, which is undesirable. Therefore, we used Convolutional Neural Networks (CNN), which are well-suited for processing the spatial structure in the data and fit well with the sequential

nature of the vibration data. Recurrent Neural Networks (RNNs), including LSTMs and GRUs, were not deployed because of their complexity and the fact that the data does not exhibit long-term informational diversity. This makes the use of CNNs with proper windowing more beneficial. Initially, we defined the regression problem in its basic form, i.e., predicting the $\mathtt{CF}$ value. Respecting the properties of the dataset and keeping in mind the importance of the values close to one, we defined a better regression problem to focus on these values. First, the $\mathtt{CF}$ value ranges between zero and positive infinity with a cavitating threshold of one. We defined a new target value as $\log(\mathtt{CF})$ to make the problem more balanced and easier for the model to learn. Second, as we get away from the threshold, the change in the pattern of the vibration data becomes subtle and will very soon become negligible. For example, even the difference between $\mathtt{CF} = 4$ and $\mathtt{CF} = 5$ is less significant than the difference between $\mathtt{CF} = 1.1$ and $\mathtt{CF} = 1.2$. Considering the usual applications of this system (like monitoring the condition of the pump and taking preventive actions), it is much more critical to accurately predict the $\mathtt{CF}$ values close to one and the difference between $\mathtt{CF} = 4$ and $\mathtt{CF} = 5$ can even be ignored. Therefore, we defined our final target value as $\sigma(\alpha \times \log(\mathtt{CF}))$, in which σ represents the sigmoid function, and α is a hyperparameter affecting the range of values close to the border that the model should be sensitive to. In our experiments, we have found that $\alpha = 4$ is a good choice.

2.5 Results

We conducted many experiments to find the best combination of settings under which the models could perform well. The results of these experiments are reported in this section.

2.5.1 Support Vector Machine

Our default setting and problem formulation are as follows: Given the suitability of SVMs for classification tasks, we have framed our problem as a binary classification challenge. The data is split by a window size of 4096 samples; and 20% of the data is used for testing, and the rest is used for training. The train set and test set are kept distinct by ensuring that no part of the test data comes from the same recording as any of the training data. For each data, we computed the 90th percentile, energy, and standard deviation of the entire signal and subsequently for each of the 50 subdivisions of its length, yielding a total of 153 properties, which were then considered as input features for our model.

All experiments were performed three times, and our observations and conclusions remained consistent throughout.

Data Inclusion

We considered feeding the model with the FFT of the acceleration data, 90th percentile, energy, and standard deviation of the whole data with or without subdivisions. This is due to the fact that, unlike deep neural networks, SVMs are not capable of automatically discovering the representations needed for classification, and preparing the input data is crucial for the performance of the model. The selection of these feature spaces was based on their rich representation and, in most cases, their low cost of extraction. This decision was partially inspired by [17].

Table 2.1 shows the performance of the SVM model receiving various input features. It proves that the 90th percentile (90th), energy, and standard deviation (SD) can all be good features for the model, energy being the best. However, a combination of these features can improve the model's performance. Remarkably, this combination outperforms the FFT of the data and achieves equivalent performance when integrated with the FFT. This is crucial because the FFT generates a lengthy sequence, significantly increasing the model's computational burden, which can be avoided. This computational burden

is related to the number of multiplications and accumulations (MACs), as detailed in the table.

Table 2.1: Including combination of data types

	Accuracy (%)	MACs
FFT	91.21	4096
90th	89.89	306
Energy	91.54	306
SD	90.77	306
90th, Energy, SD	91.76	918
FFT, 90th, Energy, SD	91.75	5014

Another variable that can change the performance of the model is the number of subdivisions for which we compute statistics. Increasing these numbers should give more detailed information about the behavior of the signal and should improve the performance of the model in the cost of computational complexity. However, we expect this improvement in accuracy to saturate. Finding a good number of subdivisions is crucial to balance the performance and computational cost of the model.

In Table 2.2, we show the performance of the SVM model with different numbers of subdivisions. Based on it, the importance of using subdivisions for giving the model more detailed and spatial information about the signal becomes clear (*0 parts* versus *5 parts*). Further, up to *50 parts*, we see a good improvement in the performance, but there is not much improvement after that. Therefore, we chose *50 parts* as a good number of subdivisions for our model.

Table 2.2: Partitioning strategies on SVM model

Parts	**0**	**5**	**10**	**25**	**50**	**100**
Accuracy (%)	73.06	83.75	87.97	89.92	91.76	91.95
MACs	6	36	66	156	306	606

Window Size

Each data point in the original dataset belongs to a separate test (also called recording), consisting of 30 seconds of continuously logging the vibration data at 48 kHz. This obviously is a very long sequence and is mostly repetitive. Therefore, we windowed the data into smaller sequences and fed them to the model. Although not much relevant to the SVM model, in case of deep learning models, different windows of the same recording will be similar to each other and can be considered as augmented data, preventing the model from overfitting. We've found a window size of 4096 samples to be a consistent good choice through several experiments, Table 2.3 illustrating one of them. It can be concluded that a data frame of around 85 milliseconds with a high enough sampling rate should contain enough information for successfully detecting the cavitation.

Table 2.3: Division of data by different window sizes

Window Size	**256**	**1024**	**4096**	**16384**	**65536**	**262144**
Accuracy (%)	82.12	88.38	91.76	91.44	90.84	92.18

Other Experiments

As discussed before, training a separate SVM for each pump should get better results than having a single SVM for all pumps. To prove this, we compared the performance of

these two approaches. If a model is trained on each pump separately, an average test accuracy of 91.76% is achieved, while this value degrades to 80.96% if a single model is trained for all pumps.

We also tested the performance of the SVM model with different kernels. Using the linear kernel, 91.76% of the data will be classified correctly. However, using the polynomial and RBF kernels, this value goes down to 75.52% and 82.16%, respectively. This is a clear indication that the data is linearly separable, and among these, the linear kernel is the best choice for this problem.

2.5.2 Deep Learning

The same default setting as the SVM model was used for the deep learning model, except that we fed the model with the FFT of the acceleration data and a single model was trained on the whole dataset of pumps. The problem is also formulated as a regression problem, and the target value is defined as $\sigma(4 \times \log(\texttt{CF}))$. To make the results being more sensible, instead of reporting the loss value, mean squared error (MSE), or mean absolute error (MAE), we check whether the prediction of the model and the actual value both indicate the cavitation situation and report the accuracy of the model in the assumed classification problem.

We have studied various network architectures, the performance of the network in classification and regression setups, and the impact of meticulously choosing the target value. Moreover, a spectrum of different sizes of the model was benchmarked on the STM32L4 microcontroller using TensorFlow Lite for Microcontrollers (TFLM) [18], a necessary step to study the impact of models in real-world applications that is missing from most of the literature in this field.

Model Architecture

We tested several deep learning architectures. More specifically, a fully connected network, a convolutional neural network without global average pooling, and several sizes of convolutional neural networks with global average pooling were studied. All models used batch normalization layers and they used *ReLU* as the activation function. They were trained for ten epochs with a batch size of 32.

Each neuron in the first layer of the fully connected network will increase the static memory requirement of the model by more than 8 kB. This is significant, and therefore, we only used 16 neurons in the first layer. Thereafter, a layer of 10 neurons connected these features to the single output neuron. The model achieved an accuracy of 90.41% on the test set.

Convolutional layers can find the properties of the data by sliding a kernel over it, making them parameter-efficient and suitable for processing the spatial structure in the data. Based on the properties of the data, we chose relatively large kernels. In our default model architecture, we used a global average pooling layer at the end of convolutional layers to conclude the spatial information and significantly reduce the number of neurons in that feature map. This model achieved an accuracy of 90.60%, in par with the fully connected network, but having about three times less parameters. This is why we preferred this model over the fully connected network, although it's important to note that the convolutional model requires more computations. In certain scenarios, the fully connected model might be more suitable.

With the intuition that the global average pooling layer removes the spatial information, we tested the performance of the convolutional model without this layer. The model achieved

an accuracy of 91.03%, which, as expected, is slightly better than the base model. Meanwhile, this modification increased the number of parameters by 67%, which is undesirable.

An experiment for comparing the classification and regression setups was conducted in which the same base model as in regression was used in the classification problem. The model achieved an accuracy of 89.27% in the classification setup, which is worse than 90.60% achieved in the regression setup. This is a clear indication that the regression setup is more suitable for this problem.

Instead of using the value of CF for regression, we defined a new target as $\sigma(4 \times \log(\mathtt{CF}))$ to make the problem more balanced and easier for the model to learn. To prove the importance of this approach, we tested the model's performance with the original target value. The simple regression model achieved an accuracy of 88.07%, while the model with the new target achieved an accuracy of 90.60%. Although the superiority of the new target can be concluded from its accuracy, the difference is more significant when we look at their prediction graph in Fig. 2.3. The model trained by the new target can more accurately follow the sigmoid function shape that we expected it to learn. Another observation of this figure can be that some data points close to the border might be identified as misclassification, even though the model has a good prediction for them.

Based on the default CNN model that we've shown to be successful, we designed several smaller models. Table 2.4 contains information of these models and their performance. *CNN_4* is the default CNN model.

Table 2.4: CNN models' information

	Params	**MACs**	**Loss**	**Accuracy (%)**
CNN_1	1,089	557,592	0.0858	75.11
CNN_2	3,409	2,925,080	0.0580	80.68
CNN_3	8,561	2,114,072	0.0333	87.77
CNN_4	11,409	940,632	0.0238	90.60

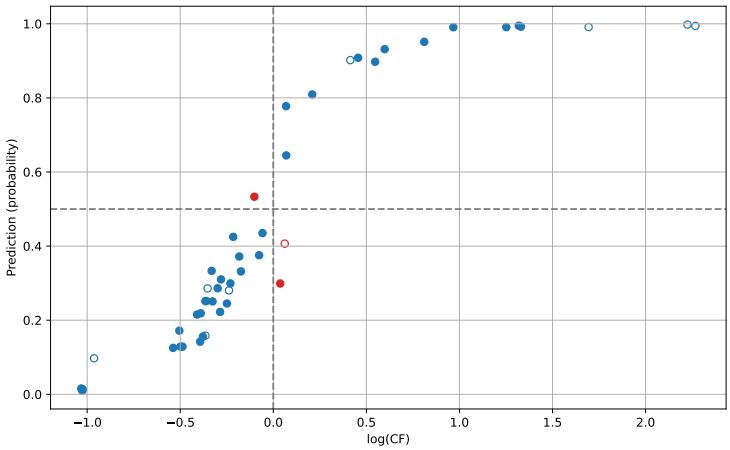
We have deployed the models to a NUCLEO-L4R5ZI board and benchmarked their performance. NUCLEO-L4R5ZI contains an STM32L4R5ZI microcontroller, which is based on an Arm Cortex-M4 core. TFLM is used to port the model to its C++ implementation. In addition to the float32 format, several quantized versions of the models have been tested, namely, dynamic range quantization, int8 quantization, 16x8 quantization, and float16 quantization. Since the most effective quantization method was the int8 quantization, we only report the results of this method and the default float32 format. Table 2.5 contains the benchmarking results of the models.

Table 2.5: CNN models' performance

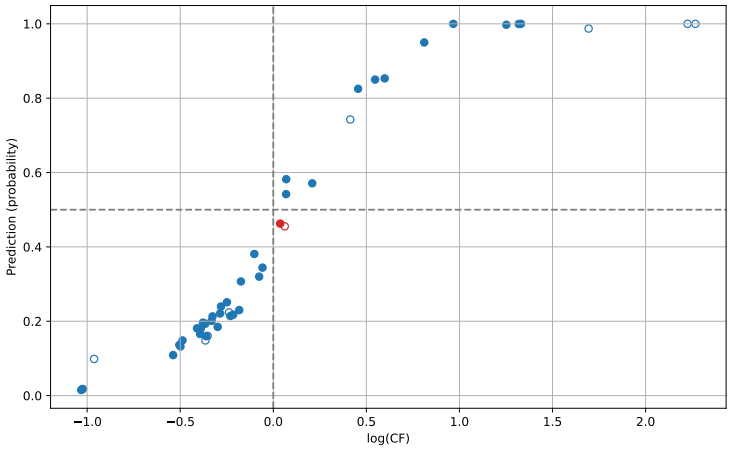
	float32			**int8**		
	Exe time	**Flash**	**RAM**	**Exe time**	**Flash**	**RAM**
CNN_1	3159.06	420.7	221.5	825.19	**358.8**	61.6
CNN_2	13362.05	430.3	279.8	2449.60	362.1	75.3
CNN_3	9442.18	450.9	150.9	1594.58	368.5	44.0
CNN_4	4354.99	463.1	127.5	**731.78**	373.3	**38.1**

Exe time (execution time) is in ms. $Flash$ and RAM are in kB.

This table reveals that the number of parameters is not the only factor constructing the memory requirement, but there are other requirements of TFLM that are more significant.



(a) CF as target



(b) $\sigma(4 \times \log(\mathtt{CF}))$ as target

Figure 2.3: Impact of defining various regression targets for the model. Hollow circles represent data present in the test set, while filled circles denote data in the training set. Red indicates misclassification, whereas blue signifies correct classification.

For example, the weights of *CNN_4* will take around 45 kB of the flash memory in the float32 form, but still around 418 kB more is needed for the model to be deployed (a big chunk of it is required for TFLM operations registration and depends on the model we are deploying).

In a separate experiment, we found that the floating point multiplication and integer multiplication in STM32L4R5ZI take almost the same time (this microcontroller has an FPU). The speedup of the int8 quantization is due to the optimized implementation of integer operational kernels in CMSIS-NN library [19] that is used in conjunction with TFLM. The main portion of the speedup is due to the utilization of the Arm Cortex-M4 core's SIMD instructions. When comparing the execution time of *CNN_1* and *CNN_4*, it can be observed that *CNN_1* performs faster in the float32 format, while *CNN_4* excels in the int8 format. This underscores the significance of such comparisons in practical applications, invalidating many of our presumptions.

Since the int8 model improves all resource benchmarks without compromising the ac-

curacy of the model, we can conclude that the int8 quantization of *CNN_4* is probably the best choice for this problem, showing advantages in all aspects of the benchmarking, except for the flash memory requirement which was in par with the other int8 models.

In comparing SVMs with neural networks, it is evident from Tables 2.1, 2.2, and 2.4 that SVMs require orders of magnitude less computation than neural networks. While this efficiency is a notable advantage, it comes at the cost of SVMs' limited functionality compared to neural networks. Consequently, the decision to opt for one over the other should be guided by the specific application requirements.

2.6 Conclusion

This study illustrated the potential of machine learning techniques, notably support vector machines and deep learning models, in detecting pump cavitation. By leveraging the dataset provided by Grundfos, the study offered compelling evidence that machine learning approaches can accurately predict cavitation occurrences in pumps under varying real-world conditions. The study's methodology, which included detailed data analysis, feature engineering, problem formulation, model design, and rigorous testing, provided valuable insights into the challenges and opportunities in this domain.

The results of the study demonstrated that deep learning models, particularly convolutional neural networks, can effectively learn the patterns associated with cavitation and predict its occurrence with high accuracy. The study further emphasized the advantages and constraints of SVM models in this context. Particularly, with effective feature engineering, SVM models have the potential to surpass the performance of more intricate deep learning models in specific situations. However, this comes at the cost of reduced generality and limitation to classification problems. The study's findings underscored the importance of testing machine learning models on target hardware to ensure their practical applicability, a missing point in most of the embedded artificial intelligence literature, especially in the context of industrial applications.

Although a very compact and powerful model was designed in this study, still it is interesting to see how the model can be further optimized by leveraging the Neural Architecture Search (NAS) and model compression techniques [20, 21, 22, 23, 24, 25], especially those designed for embedded systems [16, 26, 27]. This can be a promising direction for future research.

3 Paper B: EdgeMark: An automation and benchmarking system for embedded artificial intelligence tools

Paper B goes here.

4 Paper C: A Survey of Quantization Techniques in Embedded AI Toolchains

Paper C goes here.

5 Conclusions and Future Directions

Conclusion goes here.

Bibliography

[1] Maxime Binama, Alex Muhirwa, and Emmanuel Bisengimana. *Cavitation Effects in Centrifugal Pumps-A Review*. 2016.

[2] Christian Brix Jacobsen. *The Centrifugal Pump*. 1st ed. Grundfos Management A/S, Sept. 2008.

[3] Ahmed Ramadhan Al-Obaidi. "Detection of Cavitation Phenomenon within a Centrifugal Pump Based on Vibration Analysis Technique in both Time and Frequency Domains". In: *Experimental Techniques* 44 (3 2020), pp. 329–347. ISSN: 17471567. DOI: 10.1007/s40799-020-00362-z.

[4] Yi Li et al. "An experimental study on the cavitation vibration characteristics of a centrifugal pump at normal flow rate". In: *Journal of Mechanical Science and Technology* 32 (10 Oct. 2018), pp. 4711–4720. ISSN: 1738494X. DOI: 10.1007/s12206-018-0918-x.

[5] Ahmed Ramadhan AL-OBAIDI. "Experimental comparative investigations to evaluate cavitation conditions within a centrifugal pump based on vibration and acoustic analyses techniques". In: *Archives of Acoustics* 45 (3 2020), pp. 541–556. ISSN: 2300262X. DOI: 10.24425/aoa.2020.134070.

[6] A. M. Abdulaziz and Ashraf Kotb. "Detection of pump cavitation by vibration signature". In: *Australian Journal of Mechanical Engineering* 15 (2 May 2017), pp. 103–110. ISSN: 14484846. DOI: 10.1080/14484846.2015.1093261.

[7] Ruijia Cao et al. "Numerical method to predict vibration characteristics induced by cavitation in centrifugal pumps". In: *Measurement Science and Technology* 32 (11 Nov. 2021). ISSN: 13616501. DOI: 10.1088/1361-6501/ac1181.

[8] Georgios Mousmoulis et al. "Experimental analysis of cavitation in a centrifugal pump using acoustic emission, vibration measurements and flow visualization". In: *European Journal of Mechanics, B/Fluids* 75 (May 2019), pp. 300–311. ISSN: 09977546. DOI: 10.1016/j.euromechflu.2018.10.015.

[9] Mohammad Taghi Shervani-Tabar et al. "Cavitation intensity monitoring in an axial flow pump based on vibration signals using multi-class support vector machine". In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 232 (17 Sept. 2018), pp. 3013–3026. ISSN: 20412983. DOI: 10.1177/0954406217729416.

[10] Ali Hajnayeb. "Cavitation Analysis in Centrifugal Pumps Based on Vibration Bispectrum and Transfer Learning". In: *Shock and Vibration* 2021 (2021). ISSN: 10709622. DOI: 10.1155/2021/6988949.

[11] Nabanita Dutta et al. "Centrifugal Pump Cavitation Detection Using Machine Learning Algorithm Technique". In: *International Conference on Environment and Electrical Engineering (EEEIC)* (2018).

[12] Marios Karagiovanidis et al. "Early Detection of Cavitation in Centrifugal Pumps Using Low-Cost Vibration and Sound Sensors". In: *Agriculture (Switzerland)* 13 (8 Aug. 2023). ISSN: 20770472. DOI: 10.3390/agriculture13081544.

[13] Seyed M. Matloobi and Mohammad Riahi. *Identification of cavitation in centrifugal pump by artificial immune network*. Dec. 2021. DOI: 10.1177/09544089211028402.

[14] Mohammad Amin Hasanpour. *Cavitation*. 2024. URL: https://github.com/Black3rror/Cavitation.

[15] Jair Cervantes et al. “A comprehensive survey on support vector machine classification: Applications, challenges and trends”. In: *Neurocomputing* 408 (Sept. 2020), pp. 189–215. ISSN: 18728286. DOI: 10.1016/j.neucom.2019.10.118.

[16] Emil Njor, Jan Madsen, and Xenofon Fafoutis. “Data Aware Neural Architecture Search”. In: *TinyML Research Symposium* (Apr. 2023). URL: https://arxiv.org/abs/2304.01821.

[17] Atis Elsts et al. “On-board feature extraction from acceleration data for activity recognition”. In: *EWSN’18: Proceedings of the 2018 International Conference on Embedded Wireless Systems and Networks*. ACM. 2018, pp. 163–186.

[18] Robert David et al. “Tensorflow lite micro: Embedded machine learning for tinyml systems”. In: *Proc. Machine Learning and Systems* 3 (2021), pp. 800–811.

[19] Liangzhen Lai, Naveen Suda, and Vikas Chandra. “CMSIS-NN: Efficient Neural Network Kernels for Arm Cortex-M CPUs”. In: *arXiv preprint arXiv:1801.06601* (Jan. 2018). URL: http://arxiv.org/abs/1801.06601.

[20] Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. “Neural Architecture Search: A Survey”. In: *Journal of Machine Learning Research* 20 (2019), pp. 1–21.

[21] Hanxiao Liu, Karen Simonyan, and Yiming Yang. “DARTS: Differentiable Architecture Search”. In: *arXiv preprint arXiv:1806.09055* (June 2018). URL: http://arxiv.org/abs/1806.09055.

[22] Han Cai, Ligeng Zhu, and Song Han. “ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware”. In: *arXiv preprint arXiv:1812.00332* (Dec. 2018).

[23] Han Cai et al. “Once-for-All: Train One Network and Specialize it for Efficient Deployment”. In: *arXiv preprint arXiv:1908.09791* (Aug. 2019). URL: http://arxiv.org/abs/1908.09791.

[24] Zhuang Liu et al. “Learning Efficient Convolutional Networks through Network Slimming”. In: *IEEE international conference on computer vision* (2017), pp. 2736–2744.

[25] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. “Distilling the Knowledge in a Neural Network”. In: *arXiv preprint arXiv:1503.02531* (Mar. 2015). URL: http://arxiv.org/abs/1503.02531.

[26] Colby Banbury et al. “MicroNets: Neural Network Architectures for Deploying TinyML Applications on Commodity Microcontrollers”. In: *Proc machine learning and systems* 3 (Oct. 2021), pp. 517–532. URL: http://arxiv.org/abs/2010.11267.

[27] Nikhil P Ghanathe and Steve Wilton. “T-RECX: Tiny-Resource Efficient Convolutional neural networks with early-eXit”. In: *Proc. ACM International Conference on Computing Frontiers* (July 2023), pp. 123–133.

Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like “Huardest gefburn”? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

Technical
University of
Denmark

Richard Petersens Plads, Building 322
2800 Kgs. Lyngby
Tlf. 45 25 17 00

www.compute.dtu.dk