Reducing Energy Consumption of Pressure Sensor Calibration Using Polynomial HyperNetworks with Fourier Features

Muhammad Sarmad, [*1] **Mishal Fatima,** [2] **Jawad Tayyub** [*2]

[1] Department of Computer Science, Norwegian University of Science and Technology, Trondheim, Norway
[2] Endress + Hauser, Maulburg, Germany
muhammad.sarmad@ntnu.no, mishalfatima9966@gmail.com, jawad.tayyub@endress.com

Abstract

Our research aims to reduce the cost of pressure sensor calibration through machine learning. Pressure sensor calibration is a standard process whereby freshly manufactured pressure sensors are subjected to various controlled temperature and pressure setpoints to compute a mapping between the sensor's output and true pressure. Traditionally this mapping is calculated by fitting a polynomial with calibration data. Obtaining this data is costly since a large spectrum of temperature and pressure setpoints are required to model the sensor's behavior.

We present a machine learning approach to predict a pre-defined calibration polynomial's parameters while requiring only one-third of the calibration data. Our method learns a pattern from past calibration sessions to predict the calibration polynomial's parameters from partial calibration setpoints for any newly manufactured sensor. We design a novel polynomial hypernetwork coupled with Fourier features and a weighted loss to solve this problem. We perform extensive evaluations and show that the current industry-standard method fails under similar conditions. In contrast, our approach saves two-thirds of the calibration time and cost. Furthermore, we conduct comprehensive ablations to study the effect of Fourier mapping and weighted loss. Code and a novel calibration dataset validated by calibration engineers are also made public.

Introduction

The pressure sensor is the most prevalent sensor type in use. These sensors are installed in a wide range of industries, e.g., smartphones, medical, food, chemical, oil and gas, etc. However, freshly manufactured sensors cannot be deployed directly. The raw readings of these sensors do not correspond to true pressure values due to uncontrollable variation in the manufacturing process. Therefore, these sensors undergo a calibration process. This ensures that the sensor's output agrees with the actual pressure.

The calibration process is a costly part of a production pipeline. The is mainly because of the many pre-defined temperature and pressure points required to approximate the analytical relation between the sensor output and the actual pressure value. Energy-consuming industrial ovens are commonly used to subject uncalibrated sensors to these temperature and pressure points with high precision (see Fig. 1). Minimizing the time and effort needed for this process ensures significant energy savings and lower manufacturing costs. Heating and cooling is not only time-consuming but also creates a sizeable CO_2 footprint. It is therefore desirable to use the minimal number of temperature setpoints for compensation.

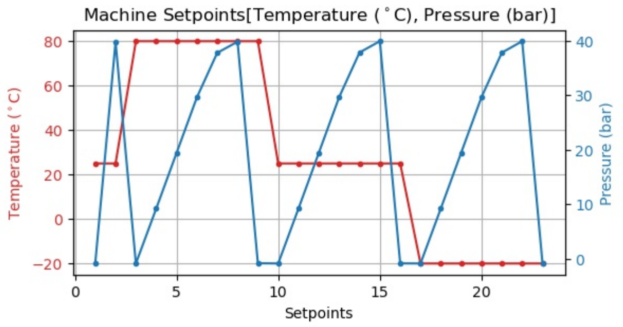

Figure 1: **Sensor Calibration Setpoints:** The plot shows pressure and temperature values that a sensor requires during the calibration process. In total, there are 23 points comprising of three temperature cycles i.e. 80°C , 25°C and -20°C having 7 pressure points within each cycle. We aim to reduce temperature cycles needed using machine learning.

Currently, the type of function approximator used for capturing the analytical relation between sensor outputs and true values is a polynomial. Polynomials are a good choice for their low memory footprint on the sensor's ASIC (Wikipedia contributors 2021). Usually, a least-square algorithm is used to calculate the coefficient of this polynomial. Ideally, this method requires the acquisition of multiple setpoints at extreme, mid, and low temperatures. These setpoints are selected to encompass the operational range of the sensor for calculating the coefficients of the polynomial. Skipping any of these setpoints lead to poor polynomial performance at those setpoints. This behavior is unacceptable for the industry as sensors should correctly operate in the stated ranges of temperature and pressure.

The process of manufacturing sensors and calibration

[*]These authors contributed equally.

over many years has accumulated data in the industry. This data presents an opportunity to apply data-driven methods such as deep learning to skip a subset of critical temperature setpoints from the operational range of the sensor. Neural networks can learn the dynamics of the calibration process from thousands of previously calibrated sensors. Therefore, in this work we address the question; can the coefficients of a calibration polynomial be predicted using only a subset of calibration setpoints for newly manufactured sensors?

We develop a hypernetwork architecture that predicts coefficients of the calibration polynomial by utilizing only one-third of the temperature setpoints. In other words, any costly temperature setpoint can be removed from the calibration process thereby reducing the time and energy needed for sensor calibration significantly. Calibration is also known as 'compensation' in the industry. We use these terms interchangeably. We experiment on two types of pressure sensors measuring different pressure ranges (0-40 bar and 0-10 bar). This encompasses a diverse range of pressure sensor types. Our contributions are as follows:

- We present a novel data-driven approach for pressure sensor compensation that reduces significant costs in the industry.
- We develop a polynomial hypernetwork architecture to predict the coefficients of the calibration polynomial.
- We propose a Fourier mapping function and a temperature-dependent weighted loss for improved training time and prediction accuracy.
- We demonstrate our approach using a real-world compensation dataset verified by calibration engineers. We show that our method highly conforms to an existing industrial standard while requiring only one-third of the compensation effort.

Related Works

Sensor Calibration

Conventional methods based on polynomial fitting constitute most previous work. (Crary et al. 1990) utilise automatic digital polynomial-based compensation of silicon pressure transducers using step-wise regression. Lyahou et al. (Lyahou, van der Horn, and Huijsing 1997) use calibration values in a series to gradually improve the sensor output. They use a simple correction coefficient for every calibration value. Horn et al.(Van Der Horn and Huijsing 1997) propose a greedy method that aims to correct the output at each step until the error is within the required bounds. Dickow et al. (Dickow and Feiertag 2015) propose a polynomial approach for micro-electromechanical systems (MEMS) sensor calibration using only a few setpoints at the cost of lower accuracy.

Optimal selection of setpoints is critical for efficient calibration. Many works propose selection criteria for optimal calibration (Pallàs-Areny, Jordana, and Casas 2004; Jordana and Pallas-Areny 2004; Rivera-Mejía, Carrillo-Romero, and Herrera-Ruíz 2010). These include minimising integrals of the square of errors or reducing errors at extremes of measurement ranges and so on. Rivera et al. (Rivera, Herrera, and Chacón 2009) cater to the effect of relative non-linearity in the sensor's output by using more setpoints at the cost of increased complexity. Whereas, Rahili et al. (Rahili, Ghaisari, and Golfar 2012) propose to incorporate additional setpoints progressively to minimise time and cost. Pieniazeket al. (Pieniazek and Ciecinski 2019) utilize similarities among sensors to reduce individual calibration. However, this requires an accurate sensor model which is time-consuming to construct and often exhibits simplifying assumptions. We do not rely on a physical sensor model and only utilise previous data for capturing sensor dynamics. Neural networks have been previously used to replace compensation polynomials (Rivera et al. 2007). Patra et al.(Patra and Van den Bos 2000; Patra et al. 2004, 2005) and Rath et al. (Rath, Patra, and Kot 2000) present individual MLPs instead of a simple polynomial function for calibration of sensors. This adds complexity as most sensor's ASICs cannot accommodate MLPs on-chip. None of the above-mentioned approaches completely skips entire temperature cycles from the operational range.

We use a neural network approach to predict the coefficients of a polynomial in a hypernetwork formulation (Ha, Dai, and Le 2016). Our method learns the behavior of sensors at different temperature cycles using previous calibration data allowing for reduction of setpoints at test time.

Advances in Deep Learning

The convolutional neural networks (CNNs) have performed well for multiple computer vision problems (Krizhevsky, Sutskever, and Hinton 2012; LeCun et al. 1990; Canziani, Paszke, and Culurciello 2016; He et al. 2015; Huang, Liu, and Weinberger 2016). Applications of deep learning include classification, semantic segmentation, image inpainting, shape completion etc. (Krizhevsky, Sutskever, and Hinton 2012; LeCun et al. 1990; Canziani, Paszke, and Culurciello 2016; Long, Shelhamer, and Darrell 2015; Noh, Hong, and Han 2015; Yeh et al. 2016; Dai, Qi, and Nießner 2016; Dong et al. 2015; Goodfellow et al. 2014).

Unlike images, 3D point clouds are an irregular datatype that requires a permutation invariant representation. Therefore, image-based CNNs do not perform well on the point cloud data. Accordingly, Qi et al. (Qi et al. 2016) proposes PointNet to process point cloud data in a neural network framework. Similarly, we utilise the PointNet backbone for our polynomial hypernetwork without the batch normalization layer (Ioffe and Szegedy 2015).

NeRF (Mildenhall et al. 2020) synthesizes novel views from sparse views. They introduce a positional encoding module inspired by Rahaman et al.(Rahaman et al. 2019). Similarly, Tancik et al. (Tancik et al. 2020) use Fourier features to map input coordinates to higher dimensions before passing it to an MLP. This mapping helps the neural network learn high-frequency variations easily. We also incorporate this module and demonstrate improved convergence.

Method

Background

The conventional compensation method comprises of two steps: prescaling and calibration. The prescaling step is a

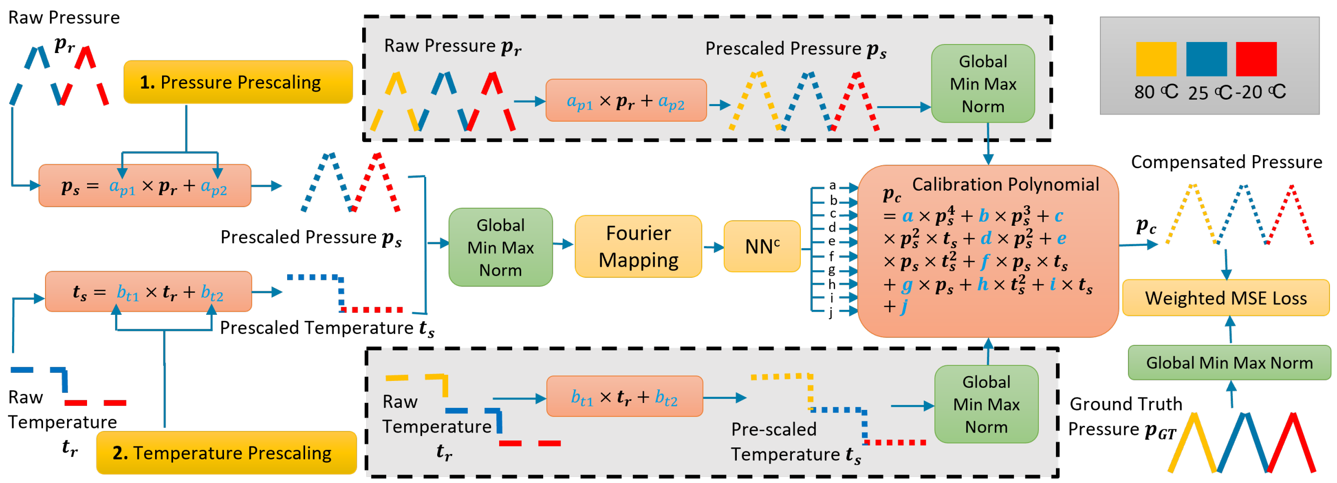
Raw Pressure p_r
1. Pressure Prescaling
$p_s = a_{p1} \times p_r + a_{p2}$
Prescaled Pressure p_s
$t_s = b_{t1} \times t_r + b_{t2}$
Prescaled Temperature t_s
Raw Temperature t_r
2. Temperature Prescaling
Raw Pressure p_r
$a_{p1} \times p_r + a_{p2}$
Prescaled Pressure p_s
Global Min Max Norm
80 ℃ 25 ℃ -20 ℃
Global Min Max Norm
Fourier Mapping
NN^c
a b c d e f g h i j
Calibration Polynomial
$p_c = a \times p_s^4 + b \times p_s^3 + c \times p_s^2 \times t_s + d \times p_s^2 + e \times p_s \times t_s^2 + f \times p_s \times t_s + g \times p_s + h \times t_s^2 + i \times t_s + j$
Compensated Pressure
p_c
Weighted MSE Loss
Global Min Max Norm
Ground Truth Pressure p_{GT}
Raw Temperature t_r
$b_{t1} \times t_r + b_{t2}$
Pre-scaled Temperature t_s
Global Min Max Norm

Figure 2: **Training**: The steps in training are pressure and temperature prescaling followed by the main calibration. Note that the input of NN^c are setpoints at two temperature cycles (blue, red) to predict coefficients of the calibration polynomial. This is used to compute compensated pressure $\boldsymbol{p_c}$. However, the weighted MSE loss also receives the third cycle (yellow) to ensure that NN^c indirectly learns the entire temperature spectrum from training data. We also use the min-max normalization and Fourier mapping to aid in training and convergence.

linear mapping of the input values. This mapping provides a prior for the main calibration to successfully discover a more complex and complete function. The main calibration step involves fitting a high-order polynomial function to the prescaled pressure and temperature values, such that the output values approximate predetermined ground-truth setpoints. Sensors are placed in industrial ovens, which are heated/cooled to these setpoints. Then a least-square algorithm is used to find the coefficients of the calibration polynomial. Each sensor's raw pressure reading is passed through its calibration polynomial during real-world operation to obtain compensated pressure outputs.

Approach Overview

The number of pressure and temperature setpoints used to fit the polynomial determine each sensor's calibration cost. One way to reduce this cost is by decreasing the number of pressure and temperature values used for fitting. Typical temperature and pressure compensation setpoints are shown in Fig. 1. Achieving temperature setpoints through heating/cooling is significantly more energy consuming than pressure setpoints. Therefore, in this work, we focus on removing complete temperature *cycles* to reduce the energy and time costs. A temperature cycle is defined as all setpoints corresponding to a single temperature. Next we detail our approach by removing 1 of 3 cycles for simplicity.

Consider a calibration process comprising of n setpoints. A sensor then produces n measurements corresponding to each setpoint. This results in a measurement set $N = \{(t_r^1, p_r^1), (t_r^2, p_r^2), ..., (t_r^n, p_r^n)\}$ where t_r and p_r are raw temperature and pressure respectively. $\boldsymbol{t_r}$ and $\boldsymbol{p_r}$ are vectors of the respective quantities in N. We aim to reduce setpoints to create a subset M by removing an entire temperature cycle α. Formally, M is described as $\{M \subseteq N | t_r \neq \alpha\}$. We then train our machine learning algorithm to only require M setpoints for compensation. We also utilise the ground-truth setpoints $\boldsymbol{p_{GT}}$ corresponding to $\boldsymbol{p_r}$. Our approach can be divided into training and inference phases. During training, a neural network is trained to predict coefficients of the compensation polynomial using only M points. However, our approach exposes the network to also learn the skipped temperature cycle/s by employing a loss function that uses all N points. In this way, during inference, the neural network determines the compensation polynomial coefficients of a new sensor with only M input setpoints.

Training

The training process is detailed in Fig. 2. The goal is to train the polynomial hypernetwork NN^c to predict coefficients of the compensation polynomial. The output of this polynomial is the compensated pressure $\boldsymbol{p_c}$. The polynomial's structure is determined as most suitable after years of manual engineering and empirical analysis. Therefore, it is considered as a prior to aid the learning process. A common pre-step in the compensation process is prescaling, see Fig. 3. The prescaling stage is used to provide a simple prior (linear mapping) before main compensation to discover higher accuracy polynomial coefficients. This step is detailed next.

Pressure Prescaling: The pressure prescaling step, as shown in Fig. 3, uses only M setpoints. The ground truth pressure $\boldsymbol{p_{GT}}$ is mapped linearly from analog to digital domain using the line equation $m_p \times \boldsymbol{p_{GT}} + c_p$, to obtain a normalised pressure value $\boldsymbol{p_g}$. Parameters m_p and c_p are determined using the two point line equation. We then use least-squares to fit raw pressure $\boldsymbol{p_r}$ and linearly mapped GT pressure $\boldsymbol{p_g}$ to determine the coefficients a_{p1} and a_{p2} of the pressure prescaling equation 1. This is then used to convert raw pressure $\boldsymbol{p_r}$ to prescaled pressure $\boldsymbol{p_s}$.

$$\boldsymbol{p_s} = a_{p1} \times \boldsymbol{p_r} + a_{p2} \quad (1)$$

The ranges of $\boldsymbol{p_{GT}}$ are set as [-1,40] or [-1,10] bar depending on the sensor family, whereas $\boldsymbol{p_g}$ is set as [-0.35,0.35]. This is chosen according to the conventional compensation process. The ranges of $\boldsymbol{p_{GT}}$ and $\boldsymbol{p_g}$ are used to find m_p and c_p in pressure prescaling normalization.

Temperature Prescaling: Temperature prescaling, as shown in Fig. 3, uses all N setpoints for training and M setpoints as input. We normalize raw temperature $\boldsymbol{t_r}$, using local min-max normalization to obtain normalized raw temperature $\boldsymbol{t_{rn}}$. This normalisation scheme uses raw temperature points $\boldsymbol{t_r}$ from each sensor in the training dataset to calculate the minimum and maximum value of the respective quantity. These values are then used to normalize $\boldsymbol{t_r}$ to obtain $\boldsymbol{t_{rn}}$ for each sensor.

Only M points from raw temperature $\boldsymbol{t_r}$ are processed in the forward pass by the prescaling hypernetwork NN^p to predict the prescaling polynomial coefficients b_{t1} and b_{t2}. The temperature prescaling polynomial in equation 2 is provided with all N points of raw temperature $\boldsymbol{t_r}$ to obtain prescaled temperature $\boldsymbol{t_s}$:

$$\boldsymbol{t_s} = b_{t1} \times \boldsymbol{t_r} + b_{t2} \quad (2)$$

This polynomial hypernetwork NN^p is trained using an l_2 loss, $L_p = \|\boldsymbol{t_s} - \boldsymbol{t_{rn}}\|_2^2$, using all N points from $\boldsymbol{t_s}$ and $\boldsymbol{t_{rn}}$. The range of $\boldsymbol{t_{rn}}$ is set as [-0.35, 0.35] according to industry convention.

Pressure Calibration: To train the polynomial hypernetwork NN^c, the raw pressure $\boldsymbol{p_r}$ and temperature $\boldsymbol{t_r}$ pairs in M are passed to the respective prescaling polynomials given in equation 1 and 2. The resulting prescaled pressure $\boldsymbol{p_s}$ and temperature $\boldsymbol{t_s}$ are normalized using a global min-max normalization scheme. In this scheme, min-max values across the entire training dataset are utilized for normalization. Next, a Fourier mapping is applied to learn high-frequency details, which allows for faster convergence. We use Gaussian mapping for this, as suggested by (Tancik et al. 2020), shown in equation 3.

$$\gamma(v) = [cos(2\pi\mathcal{B}v), sin(2\pi\mathcal{B}v)]^T \quad (3)$$

Where v is a concatenation of $\boldsymbol{p_s}$ and $\boldsymbol{t_s}$. $\mathcal{B}$ is sampled from a normal distribution $\mathcal{N}(0, \sigma^2)$. We assume an isotropic Gaussian distribution. The output of this Fourier mapping is passed through the NN^c hypernetwork to determines the final polynomial coefficient.

Next, we compute the loss L_c, as shown in equation 4, using all N points. The raw pressure $\boldsymbol{p_r}$ and temperature $\boldsymbol{t_r}$ are first prescaled using equation 1 and 2 to obtain $\boldsymbol{p_s}$ and $\boldsymbol{t_s}$ respectively (grey dotted block in Fig. 2). We normalize $\boldsymbol{p_s}$ and $\boldsymbol{t_s}$, using global min-max norm, and obtain N points of compensated pressure $\boldsymbol{p_c}$ using the compensation polynomial. For a sensor having three temperature cycles at 80°C, 25°C, -20°C, compensated pressure $\boldsymbol{p_c}$ expands to $(\boldsymbol{p_c^{80}}, \boldsymbol{p_c^{25}}, \boldsymbol{p_c^{-20}})$. $\boldsymbol{p_c}$ for each temperature cycle is then compared to the corresponding ground-truth $\boldsymbol{p_{GT}}$ using a weighted l_2 loss as shown in equation 4:

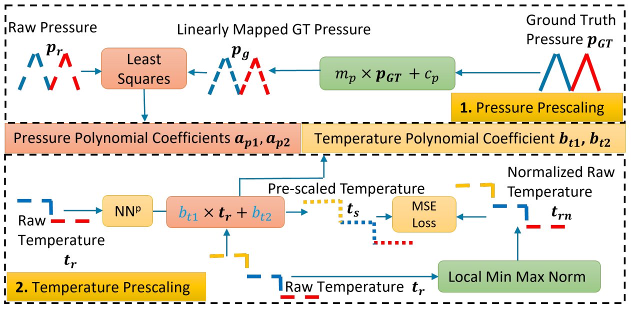
Raw Pressure
p_r
Least Squares
Linearly Mapped GT Pressure
p_g
$m_p \times \boldsymbol{p_{GT}} + c_p$
Ground Truth Pressure $\boldsymbol{p_{GT}}$
1. Pressure Prescaling
Pressure Polynomial Coefficients a_{p1}, a_{p2}
Temperature Polynomial Coefficient b_{t1}, b_{t2}
Raw Temperature t_r
NNp
$b_{t1} \times t_r + b_{t2}$
Pre-scaled Temperature
t_s
MSE Loss
Normalized Raw Temperature
t_{rn}
2. Temperature Prescaling
Raw Temperature t_r
Local Min Max Norm

Figure 3: **Prescaling**: The stage aims to find a prescaling polynomial for both pressure and temperature for a linear mapping. We use a least square approach for computing the pressure prescaling polynomial. Temperature prescaling polynomial is computed using the hypernetwork NN^p.

$$\begin{aligned} L_c = \lambda^{80}.\|\boldsymbol{p_c^{80}} - \boldsymbol{p_{GT}^{80}}\|_2^2 + \lambda^{25}.\|\boldsymbol{p_c^{25}} - \boldsymbol{p_{GT}^{25}}\|_2^2 \\ + \lambda^{-20}.\|\boldsymbol{p_c^{-20}} - \boldsymbol{p_{GT}^{-20}}\|_2^2 \end{aligned} \quad (4)$$

Note that due to this modified loss, we influence training by controlling the weights λ^{80}, λ^{25} and λ^{-20} of each temperature cycle. We use this to place more focus on skipped temperature cycles by assigning higher weights. This improves the performance of sensors for that particular temperature cycle. The phenomenon is explained further in the experiment section.

Inference The inference method is presented in Fig. 4. Pressure compensation is performed at the factory floor by inferring polynomial coefficients. During this stage, pressure prescaling, see Fig. 3, provides the coefficients of the polynomial in the equation 1. The prescaling polynomial hypernetwork NN^p provides temperature prescaling coefficients for equation 2. The output of these prescaling polynomials is then passed through global min-max norm and Fourier mapping. Our polynomial hypernetwork NN^c then predicts the coefficients of the compensation polynomial, which is deployed on the sensor along with the two prescaling polynomials. Note that the global min-max normalization uses the min and max statistics from the training set.

During real-world use, sensors readings of raw pressure $\hat{p}_r$ and temperature $\hat{t}_r$ are prescaled to form $\hat{p}_s$ and $\hat{t}_s$ using the deployed prescaling polynomials. Prescaled values are normalized and passed to the deployed calibration polynomial to obtain the final compensated pressure $\hat{p}_c$.

Network Architecture We use an architecture shown in Table 1 for both prescaling NN^c and compensation polynomial NN^p hypernetworks. This form was popularized by PointNet (Qi et al. 2016) for processing point clouds. A similarity to our data is that M points in $\boldsymbol{p_r}$ and $\boldsymbol{t_r}$ can be considered as a point set or nodes in a graph with no edges. Together they form a signature for each sensor. Therefore, unlike a 2D image where neighboring pixels are co-related, all points in our architecture are independent and form a global feature vector which is then max-pooled. The last layer out-

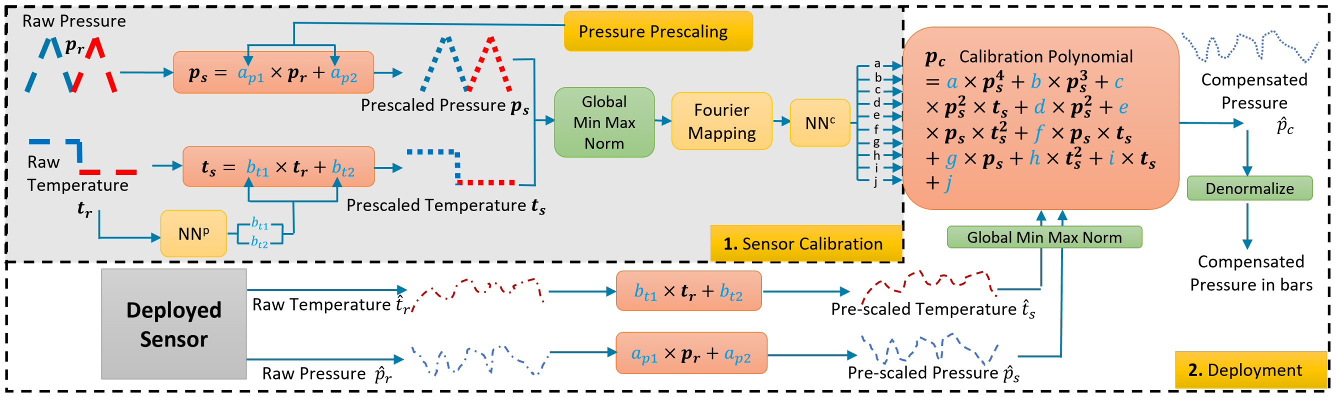
Raw Pressure
p_r
$p_s = a_{p1} \times p_r + a_{p2}$
Pressure Prescaling
Prescaled Pressure p_s
Raw Temperature
t_r
$t_s = b_{t1} \times t_r + b_{t2}$
Prescaled Temperature t_s
NN^p
b_{t1}
b_{t2}
Global Min Max Norm
Fourier Mapping
NN^c
a b c d e f g h i j
p_c Calibration Polynomial $= a \times p_s^4 + b \times p_s^3 + c \times p_s^2 \times t_s + d \times p_s^2 + e \times p_s \times t_s^2 + f \times p_s \times t_s + g \times p_s + h \times t_s^2 + i \times t_s + j$
Compensated Pressure
$\hat{p}_c$
Denormalize
Compensated Pressure in bars
1. Sensor Calibration
Global Min Max Norm
Deployed Sensor
Raw Temperature $\hat{t}_r$
$b_{t1} \times t_r + b_{t2}$
Pre-scaled Temperature $\hat{t}_s$
Raw Pressure $\hat{p}_r$
$a_{p1} \times p_r + a_{p2}$
Pre-scaled Pressure $\hat{p}_s$
2. Deployment

Figure 4: **Inference:** The first step during inference is sensor reading compensation, whereby NN^c predicts the coefficients of the calibration polynomial. This polynomial is then placed onto the sensor ASIC. During real-world operation, raw data is read by the sensor and compensated by the calibration polynomial to generate final pressure outputs. The denormalize block reverses the global min-max normalization.

Name	Kernel	Ch I/O	Input Res	Out Res	Input
conv1D1	1	$C_{in}/64$	I_s	I_s	Sensor values
conv1D2	1	$64/128$	I_s	I_s	conv1D1
conv1D3	1	$128/1024$	I_s	I_s	conv1D2
Maxpool	I_s	$1024/1$	–	–	conv1D3
Linear1	-	$1024/512$	–	–	Maxpool
Linear2	-	$1025/256$	–	–	Linear1
Linear3	-	$256/C_{out}$	–	–	Linear2

Table 1: **Neural Network (NN) architecture**: *conv1D1* to *conv1D3* are 1D convolution layers. The dimensions of the input to *conv1D1* i.e., C_{in} are either 2 with no mapping and 32 if Fourier features mapping is used. The output of *Linear3* i.e., C_{out} is either 10 for compensation neural network and or 2 for the prescaling neural net. The filter of the Maxpool layer is adaptable to input size I_s. I_s can be either be 7 for single-cycle removal or 16 for two-cycle removal.

puts the coefficient of a polynomial, thus making this configuration a hypernetwork.

Experiments

Dataset Our dataset is collected from real-world compensation processes and verified by calibration engineers. It contains compensation data for two sensor families measuring different pressure ranges (40 and 10 bar). The 40 bar and 10 bar sensor data contains 3094 and 3805 files respectively. Each sensor file in the dataset comprises of an entire compensation sequence as shown in Fig 1. This entails pressure measurement taken at three temperature cycles i.e. 80°C, 25°C, and -20°C. Each cycles has 7 setpoints and an additional 2 points at 25°C, making a total of 23 points of raw pressure $\boldsymbol{p_r}$ and temperature readings $\boldsymbol{t_r}$. The corresponding ground truth pressure value $\boldsymbol{p_{GT}}$ is also provided. Training and test splits are done such that the latest 100 sensors from both families are left out for testing and all older sensors are used as training data. This scheme reflects a freshly manufactured batch of sensors undergoing calibration.

Computational Resource We use Pytorch for all our experiments (Paszke et al. 2019). Training takes almost 48 hours and is done on a single Nvidia RTX 8000.

Evaluation Metrics As mentioned, our dataset comprises of sensors that have been calibrated on three temperature cycles. We remove one or two of these temperature cycles and compute an accurate compensation polynomial. In experiments, we evaluate the quality of this compensation polynomial on all temperature cycles. This is done using two metrics: mean square error (MSE) and the number of sensors out of bounds (#OB).

MSE: MSE is calculated between the final calibrated pressure and GT pressure for every sensor at every setpoint. Then the mean is calculated across the entire test set.

#OB: We also evaluate the proposed method according to current industry standards. This is done by using pre-defined error bounds specific to every setpoint. These error bounds define the tolerance range of the absolute error between calibrated pressure and ground truth. They are specified by calibration engineers having years of domain knowledge. The error bound is violated if the difference between a sensor's output and the ground truth pressure on any temperature and pressure setpoint is greater than the threshold. We track all test sensors that violate error bounds and report this number #OB as a measure of our algorithm's performance. For both 40 bar and 10 bar sensor, the allowed error bounds corresponding to each temperature cycles in Fig. 1 are {80 °C: 0.2%}, {25 °C: 0.15%}, {-20 °C: 0.2%}.

Industry Standard Baseline: For comparison, we employ a current industry standard algorithm for compensation (PolyReg) as a baseline. This method uses polynomial regression to find coefficients of the polynomial using available calibration data.

Hyperparameters We use a learning rate of 0.0085, batch size of 32, and Adam optimizer for all our experiments. We train our networks up to 260k epochs. We use Fourier mapping with a size of 32. The weights (λ^{80},λ^{25},λ^{-20}) for the

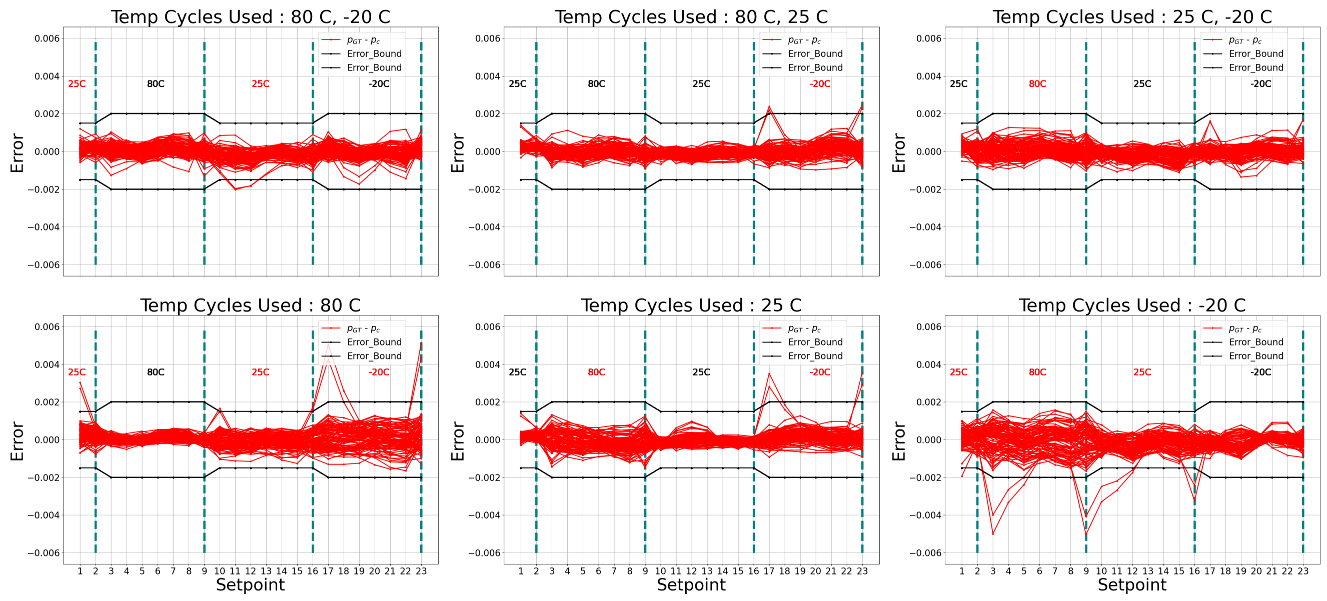

Figure 5: **Envelop Curve**: This diagram shows results of six experiments where each graph displays the error $p_{GT} - p_c$ at all setpoints (x-axis) for all 40 bar test sensors. Each subfigure is an experiment with either one or two out of three temp. cycles as input with the unused temp. cycle shown in red. The black lines indicate acceptable error margin as provided by industrial standards. Temperature cycles are segregated by dotted lines.

Method	**Metric**	**40 bar sensors**						**10 bar sensors**					
		25°C, -20°C	**80°C, -20°C**	**80°C, 25°C**	**80°C**	**25°C**	**-20°C**	**25°C, -20°C**	**80°C, -20°C**	**80°C, 25°C**	**80°C**	**25°C**	**-20°C**
PolyReg	MSE	0.0025	0.1973	0.0013	-	-	-	0.0033	0.0311	0.0016	-	-	-
	#OB	100	100	100	-	-	-	100	100	100	-	-	-
Ours w/o $\mathcal{F}$	MSE	3.34e-7	2.11e-7	2.63e-7	5.12e-7	2.67e-7	3.08e-7	1.78e-7	1.18e-7	1.25e-7	4.23e-7	2.28e-7	2.86e-7
	#OB	3	3	11	24	3	10	3	0	0	15	3	5
Ours w/ $\mathcal{F}$	MSE	1.34e-7	1.27e-7	1.03e-07	2.29e-7	1.44e-7	3.09e-07	1.61e-7	8.84e-8	8.27e-8	3.26e-7	1.55e-7	2.18e-7
	#OB	0	2	2	2	2	2	0	0	0	6	1	4

Table 2: This table shows the performance of our method for two sensor types, i.e., 40 bars and 10 bars. For each type, we indicate the temperature cycle/s needed for the calibration process. Note that PolyReg can not be evaluated for SCS experiments since the number of coefficients of the calibration polynomial (10) is higher than the calibration setpoints (7).

removed cycles are set to 10 while for the input cycles are kept at 1 in equation 4.

Results & Discussions

We perform an extensive evaluation of our approach for pressure sensor calibration using two sensor families (40 bar and 10 bar). Recall that these two types of sensors are calibrated using three temperature cycles i.e. 80°C, 25°C, and -20°C. We present an overview of our experiments below:

- Double Cycle Selection (DCS): In these experiments, two out of the three temperature cycles are required as input, while one temperature cycle is removed. This is done for all combinations of temperatures resulting in three settings i.e. {80°C, 25°C}, {25°C,-20°C} and {80°C, -20°C} as inputs.
- Single Cycle Selection (SCS): To further stress test the proposed approach, we experiment with only one of the three temperature cycles as input while removing two temperature cycles. This is also done for all combinations resulting in another three settings i.e. {80°C}, {25°C} and {-20°C} as inputs.

The summary of the results is presented in Table. 2. We report the #OB and MSE metrics for evaluation and compare them against polynomial regression (PolyReg), the current standard in the industry. We also report the effect of Fourier mapping for all experiments. It can be seen that our approach outperforms the PolyReg baseline by a large margin in all experiments for both sensor families. Significantly lower MSE and #OB are reported in each column, indicating that our approach successfully learns the dynamics of sensor behavior, especially in removed temperature cycles.

In the SCS experiments, PolyReg cannot be evaluated since this becomes an underdetermined system. On the other hand, our approach only consumes one temperature cycle, i.e., 7 setpoints, and provides excellent results. This is less number of datapoints than the coefficients of the polynomial that we predict. Moreover, Table. 2 further highlights the exceptional improvement in quality achieved with employing Fourier features within our framework.

Fig. 5 shows the envelope curves which represent a detailed error plot of each test sensor's performance at all setpoints. The red lines show the error in outputs of our hypernetwork compensated sensors. Six experiments are displayed here for brevity, 3 SCS, and 3 DCS. More examples are included in the supplementary. Note that the vast majority of sensors remain within the allowed error bounds of 0.15%-0.2%. This validates that the predictions of our network architecture remain within such a strict error margin. Moreover, note that one or two complete temperature cycles have been removed for the network in each experiment (specified in the plot title). Yet, the network generates valid predictions within those removed temperature cycles.

We further highlight that energy consumption is maximal when achieving extreme temperatures, i.e., 80 °C and -20 °C. Therefore, it is desirable to compensate sensors using only 25 °C, which is close to room temperature. However, using only 25 °C is challenging as the network lacks any information of the sensor behavior at extreme temperature setpoints and thus relies only on one temperature setpoint to approximate the complete operational range. Our approach overcomes these challenges and performs extremely well with an accuracy of 98% in such scenarios with only 25 °C as input, see (Fig. 5 bottom center sub-figure and Table. 2). Only two test sensors are out of bounds in this experiment; this could be due to variability in the manufacturing process or outliers in the data. In terms of energy consumption, this approach reduces energy usage by 72% compared to the existing method, which requires all three cycles. Our proposed system is deployment-ready, and since pressure sensors are commonly used in multiple industries, the benefits of the proposed method extend to all related sectors.

Temp°C	Train (#OB / Acc %)		Test (#OB / Acc %)	
	Fourier Features Evaluation ($\mathcal{F}$)			
	w/ $\mathcal{F}$	w/o $\mathcal{F}$	w/ $\mathcal{F}$	w/o $\mathcal{F}$
25,-20°C	**7 / 99.76**	71 / 97.6	**0 / 100**	3 / 97
80,-20°C	**8 / 99.73**	24 / 99.2	**2 / 98**	3 / 97
80,25°C	**7 / 99.76**	172 / 94.2	**2 / 98**	11 / 89
80°C	**94 / 96.86**	611 / 79.5	**2 / 98**	24 / 76
25°C	**29 / 99.03**	108 / 96.3	**2 / 98**	3 / 97
-20°C	**105 / 96.49**	214 / 92.8	**2 / 98**	10 / 90
	Weighted Loss Evaluation ($\mathcal{W}$)			
	w/ $\mathcal{W}$	w/o $\mathcal{W}$	w/ $\mathcal{W}$	w/o $\mathcal{W}$
25,-20°C	11 / 99.63	**7 / 99.76**	2 / 98	**0 / 100**
80,-20°C	**5 / 99.83**	8 / 99.73	**1 / 99**	2 / 98
80,25°C	**5 / 99.83**	7 / 99.76	2 / 98	2 / 98
80°C	**60 / 97.99**	94 / 96.86	2 / 98	2 / 98
25°C	35 / 98.83	**29 / 99.03**	2 / 98	2 / 98
-20°C	**68 / 97.72**	105 / 96.49	2 / 98	2 / 98

Table 3: **Ablation Study:** We compare the effect of Fourier mapping $\mathcal{F}$ and Weighted loss $\mathcal{W}$ for a fixed set of 260k iteration for 40 bar sensor. Acc % is found by dividing #OB by the total number of sensors in the respective dataset. Temp°C indicates the temperature cycles needed for compensation.

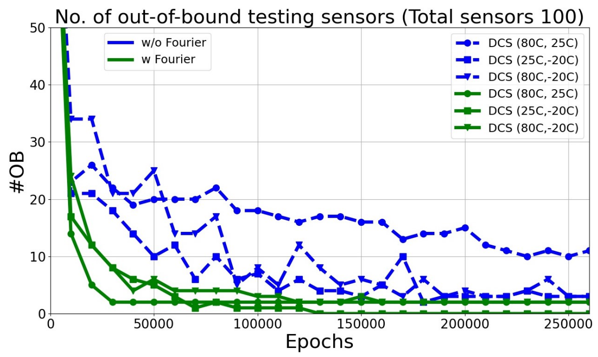

Figure 6: **Convergence Speed**: The error plots (#OB) for the 40 bar sensor with Fourier mapping as training progresses.

Effect of Fourier Mapping We investigate the effect of Fourier feature mapping within our architecture. Our input signals contain high-frequency information reminiscent of a step function and a sawtooth wave. Our assertion is similar to past work (Tancik et al. 2020) that, with Fourier mapping, the network well captures such high-frequency features of the input signals. Table. 3 demonstrates significant improvement in accuracy on all temperature cycles, while Fig. 6 displays that the use of Fourier mapping leads to faster convergence. This trend holds for all DCS and SCS experiments.

Effect of Weighted loss Recall that during training, our polynomial hypernetwork takes in one or two temperature cycles while the loss is computed on all cycles. Removed cycles are weighted higher than input cycles to emphasize the learning of sensor behavior across the entire temperature range. Table. 3 displays improved performance with weighted loss. We find that a higher weight for the loss term corresponding to a given temperature cycle improves performance on that temperature cycle. All weighted loss experiments are done with Fourier mapping. Utilizing a weighted loss within our architecture provides significant improvement. Note the improvement in SCS experiments with 80 °C and -20 °C compared to without weighted loss.

Conclusion

The use of machine learning for pressure sensor compensation has been scarcely explored. Current methods for calibrating pressure sensors are energy and time-consuming since they require expensive ovens to achieve temperatures and pressures across a broad spectrum. We have presented a polynomial hypernetwork that saves the time and energy needed for this process by two-thirds. Our method leverages past data to learn the characteristics of a sensor's behavior at different temperature setpoints. Using this method, most temperature setpoints can be skipped for compensating new sensors. In addition, we propose using Fourier mapping for fast convergence and a weighted loss function for higher performance. We demonstrate the integrity of our approach compared to strict industry standards and existing methods. We further present a real-world dataset of pressure sensor compensation which has been certified by professional calibration engineers.

References

Canziani, A.; Paszke, A.; and Culurciello, E. 2016. An Analysis of Deep Neural Network Models for Practical Applications. *CoRR*, abs/1605.07678.

Crary, S. B.; Baer, W. G.; Cowles, J. C.; and Wise, K. D. 1990. Digital compensation of high-performance silicon pressure transducers. *Sensors and Actuators A: Physical*, 21(1): 70–72. Proceedings of the 5th International Conference on Solid-State Sensors and Actuators and Eurosensors III.

Dai, A.; Qi, C. R.; and Nießner, M. 2016. Shape Completion using 3D-Encoder-Predictor CNNs and Shape Synthesis. *CoRR*, abs/1612.00101.

Dickow, A.; and Feiertag, G. 2015. A systematic MEMS sensor calibration framework. *Journal of Sensors and Sensor Systems*, 4(1): 97–102.

Dong, C.; Loy, C. C.; He, K.; and Tang, X. 2015. Image Super-Resolution Using Deep Convolutional Networks. *CoRR*, abs/1501.00092.

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y. 2014. Generative Adversarial Nets. In Ghahramani, Z.; Welling, M.; Cortes, C.; Lawrence, N. D.; and Weinberger, K. Q., eds., *Advances in Neural Information Processing Systems 27*, 2672–2680. Curran Associates, Inc.

Ha, D.; Dai, A. M.; and Le, Q. V. 2016. HyperNetworks. *CoRR*, abs/1609.09106.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2015. Deep Residual Learning for Image Recognition. *CoRR*, abs/1512.03385.

Huang, G.; Liu, Z.; and Weinberger, K. Q. 2016. Densely Connected Convolutional Networks. *CoRR*, abs/1608.06993.

Ioffe, S.; and Szegedy, C. 2015. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *CoRR*, abs/1502.03167.

Jordana, J.; and Pallas-Areny, R. 2004. Optimal two-point static calibration of measurement systems with quadratic response. In *Proceedings of the 21st IEEE Instrumentation and Measurement Technology Conference (IEEE Cat. No. 04CH37510)*, volume 1, 110–113. IEEE.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*, NIPS'12, 1097–1105. USA: Curran Associates Inc.

LeCun, Y.; Boser, B. E.; Denker, J. S.; Henderson, D.; Howard, R. E.; Hubbard, W. E.; and Jackel, L. D. 1990. Handwritten Digit Recognition with a Back-Propagation Network. In Touretzky, D. S., ed., *Advances in Neural Information Processing Systems 2*, 396–404. Morgan-Kaufmann.

Long, J.; Shelhamer, E.; and Darrell, T. 2015. Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 3431–3440.

Lyahou, K.; van der Horn, G.; and Huijsing, J. 1997. A noniterative polynomial 2-D calibration method implemented in a microcontroller. *IEEE Transactions on Instrumentation and Measurement*, 46(4): 752–757.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. *CoRR*, abs/2003.08934.

Noh, H.; Hong, S.; and Han, B. 2015. Learning Deconvolution Network for Semantic Segmentation. *CoRR*, abs/1505.04366.

Pallàs-Areny, R.; Jordana, J.; and Casas, Ó. 2004. Optimal two-point static calibration of measurement systems with quadratic response. *Review of scientific instruments*, 75(12): 5106–5111.

Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; and Chintala, S. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, 8024–8035. Curran Associates, Inc.

Patra, J. C.; Ang, E. L.; Das, A.; and Chaudhari, N. S. 2005. Auto-compensation of nonlinear influence of environmental parameters on the sensor characteristics using neural networks. *ISA Transactions*, 44(2): 165–176.

Patra, J. C.; Gopalkrishnan, V.; Ang, E. L.; and Das, A. 2004. Neural network-based self-calibration/compensation of sensors operating in harsh environments [smart pressure sensor example]. In *SENSORS, 2004 IEEE*, 425–428. IEEE.

Patra, J. C.; and Van den Bos, A. 2000. Auto-calibration and-compensation of a capacitive pressure sensor using multilayer perceptrons. *ISA transactions*, 39(2): 175–190.

Pieniazek, J.; and Ciecinski, P. 2019. Temperature and Nonlinearity Compensation of Pressure Sensor With Common Sensors Response. *IEEE Transactions on Instrumentation and Measurement*, 69(4): 1284–1293.

Qi, C. R.; Su, H.; Mo, K.; and Guibas, L. J. 2016. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *CoRR*, abs/1612.00593.

Rahaman, N.; Baratin, A.; Arpit, D.; Draxler, F.; Lin, M.; Hamprecht, F. A.; Bengio, Y.; and Courville, A. 2019. On the Spectral Bias of Neural Networks. arXiv:1806.08734.

Rahili, S.; Ghaisari, J.; and Golfar, A. 2012. Intelligent selection of calibration points using a modified progressive polynomial method. *IEEE Transactions on Instrumentation and Measurement*, 61(9): 2519–2523.

Rath, S. K.; Patra, J. C.; and Kot, A. C. 2000. An intelligent pressure sensor with self-calibration capability using artificial neural networks. In *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*, volume 4, 2563–2568. IEEE.

Rivera, J.; Carrillo, M.; Chacón, M.; Herrera, G.; and Bojorquez, G. 2007. Self-calibration and optimal response in intelligent sensors design based on artificial neural networks. *Sensors*, 7(8): 1509–1529.

Rivera, J.; Herrera, G.; and Chacón, M. 2009. Improved progressive polynomial algorithm for self-calibration and optimal response in smart sensors. *Measurement*, 42(9): 1395–1401.

Rivera-Mejía, J.; Carrillo-Romero, M.; and Herrera-Ruíz, G. 2010. Quantitative evaluation of self compensation algorithms applied in intelligent sensors. In *2010 IEEE Instrumentation & Measurement Technology Conference Proceedings*, 1116–1120. IEEE.

Tancik, M.; Srinivasan, P. P.; Mildenhall, B.; Fridovich-Keil, S.; Raghavan, N.; Singhal, U.; Ramamoorthi, R.; Barron, J. T.; and Ng, R. 2020. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *CoRR*, abs/2006.10739.

Van Der Horn, G.; and Huijsing, J. H. 1997. *Integrated Smart Sensor Calibration*, 45–60. Boston, MA: Springer US. ISBN 978-1-4615-6061-6.

Wikipedia contributors. 2021. Application-specific integrated circuit — Wikipedia, The Free Encyclopedia. [Online; accessed 26-August-2021].

Yeh, R. A.; Chen, C.; Lim, T.; Hasegawa-Johnson, M.; and Do, M. N. 2016. Semantic Image Inpainting with Perceptual and Contextual Losses. *CoRR*, abs/1607.07539.