micromachines

Article

Deep Learning Based Multiresponse Optimization Methodology for Dual-Axis MEMS Accelerometer

Fahad A. Mattoo [1,2], Tahir Nawaz [1,2], Muhammad Mubasher Saleem [1,2,*], Umar Shahbaz Khan [1,2] and Amir Hamza [1,2]

[1] Department of Mechatronics Engineering, National University of Sciences and Technology (NUST), Islamabad 44000, Pakistan
[2] National Centre of Robotics and Automation, Islamabad 44000, Pakistan
* Correspondence: mubasher.saleem@ceme.nust.edu.pk; Tel.: +92-51-54444470

Abstract: This paper presents a deep neural network (DNN) based design optimization methodology for dual-axis microelectromechanical systems (MEMS) capacitive accelerometer. The proposed methodology considers the geometric design parameters and operating conditions of the MEMS accelerometer as input parameters and allows to analyze the effect of the individual design parameters on the output responses of the sensor using a single model. Moreover, a DNN-based model allows to simultaneously optimize the multiple output responses of the MEMS accelerometers in an efficient manner. The efficiency of the proposed DNN-based optimization model is compared with the design of the computer experiments (DACE) based multiresponse optimization methodology presented in the Literature, which showed a better performance in terms of two output performance metrics, i.e., mean absolute error (MAE) and root mean squared error (RMSE).

Keywords: deep neural network; dual-axis MEMS accelerometer; microelectromechanical systems (MEMS); multiresponse optimization; deep learning (DL); neural network

Citation: Mattoo, F.A.; Nawaz, T.; Saleem, M.M.; Khan, U.S.; Hamza, A. Deep Learning Based Multiresponse Optimization Methodology for Dual-Axis MEMS Accelerometer. *Micromachines* **2023**, *14*, 817. https://doi.org/10.3390/mi14040817

Academic Editor: Ion Stiharu

Received: 6 March 2023
Revised: 27 March 2023
Accepted: 30 March 2023
Published: 4 April 2023

1. Introduction

MEMS inertial sensors, including accelerometers and gyroscopes, are commonly used in various applications such as motion sensing, navigation systems, vibration monitoring, and structural health monitoring [1–4]. The small size, low power consumption, and low cost of these micromachined sensors make these sensors an excellent alternative to the traditional macroscale inertial sensors. The function of MEMS accelerometers is generally based on different transduction principles such as electrostatic [5,6], piezoelectric [7], piezoresistive [8], and optical [9]. Among these, capacitive MEMS accelerometers are most widely used for different applications, owing to their relatively high dynamic range, small size, and low cost [10].

In the development cycle of MEMS in general and MEMS inertial sensors in particular, it is important to analyze and predict the effect of the geometric design parameters, microfabrication process constraints, and device operating conditions on the output performance characteristics of the sensor. The optimization of MEMS accelerometers is generally carried out by changing one design parameter at a time and estimating its effect on an output response using mathematical models, finite-element-method (FEM) simulations, or topology optimization approaches [11–15]. The capacitive MEMS accelerometers are multiphysics devices which involve a coupled field electro-structural-thermal interaction, and the output performance characteristics of these sensors generally have a contradicting dependence on geometric design parameters and operating conditions. This requires a comprehensive design optimization methodology that allows the MEMS designer to optimize the output performance characteristics of MEMS sensors simultaneously with respect to the device geometric parameters and operating conditions. The design of computer

experiments and machine-learning-based optimization techniques have been presented in the Literature for the multi-response optimization of MEMS accelerometers [16,17]. In [17], a machine-learning-based method was proposed for optimization of such parameters based on training a separate model for each output response. However, the solution was not generalizable and had a higher complexity due to the need for training as many models as the number of output responses. It would instead be desirable to have a single unified generic model that enables simultaneous prediction of all the output responses in an efficient manner.

Recently, the use of deep-learning-based approaches have shown highly encouraging results for such combinatorial optimization problems in other fields. However, their use and adoption in the MEMS field is still at its infancy. Deep learning is a subset of machine learning that uses neural networks with multiple layers to learn complex patterns in data [18,19]. It has gained popularity due to its ability to effectively and efficiently learn from large amounts of data and solve complex problems that were previously considered unsolvable [20,21].

To this end, this paper proposes an end-to-end deep-neural-network-based methodology that is aimed at optimization of design parameters by relying on a unified framework that does not require the learning of multiple separate models, and leads to an efficient simultaneous prediction of the accelerometer output characteristics. Indeed, the proposed method allows analysis of the effect of the geometric design parameters and operating conditions on the output performance characteristics of a capacitive MEMS accelerometer in an effective manner.

2. MEMS Accelerometer Design

The MEMS accelerometer design considered for the implementation of the proposed deep-neural-network-based optimization methodology is shown in Figure 1. The MEMS accelerometer design allows to measure input acceleration in two in-plane axes, thus making it a 2-DoF design. The design consists of a central proof mass with capacitive electrodes attached on the four sides. The T-shaped mechanical suspension beams attached on the four corners of the central proof mass allows to measure input acceleration in both x-axis and y-axis, while minimizing the cross-axis coupling. For an input acceleration in any axis, the proof mass displaces, and this displacement is measured by using stator and rotor capacitive combs attached on the sides of the proof mass. The stator and rotor combs are arranged in a gap-antigap configuration with a minimum gap value of 2.5 μm, which is defined as per the microfabrication process constraints of the multi-project-wafer-based SOIMUMPs process offered by MEMSCAP Inc., USA [22]. The displacement in the proof mass, corresponding to an input acceleration, results in an air gap change between the stator and rotor combs which leads to a net capacitance change. This net capacitance change is used as an output metric for the measurement of an input acceleration. The dynamic response of the MEMS accelerometer is strongly dependent on the mechanical compliance and stiffness of the suspension beams, which is defined by the geometric dimensions of the beams. Similarly, the net capacitance change for an input acceleration is strongly dependent on the initial air gap between the stator and rotor combs. In addition to the geometric design parameters, the performance of the MEMS accelerometer is affected by the operating temperature and air pressure conditions, which has been discussed in detail in [6,23]. In this work, we have considered both the MEMS accelerometer geometric design parameters and operating conditions as parameters for the optimization.

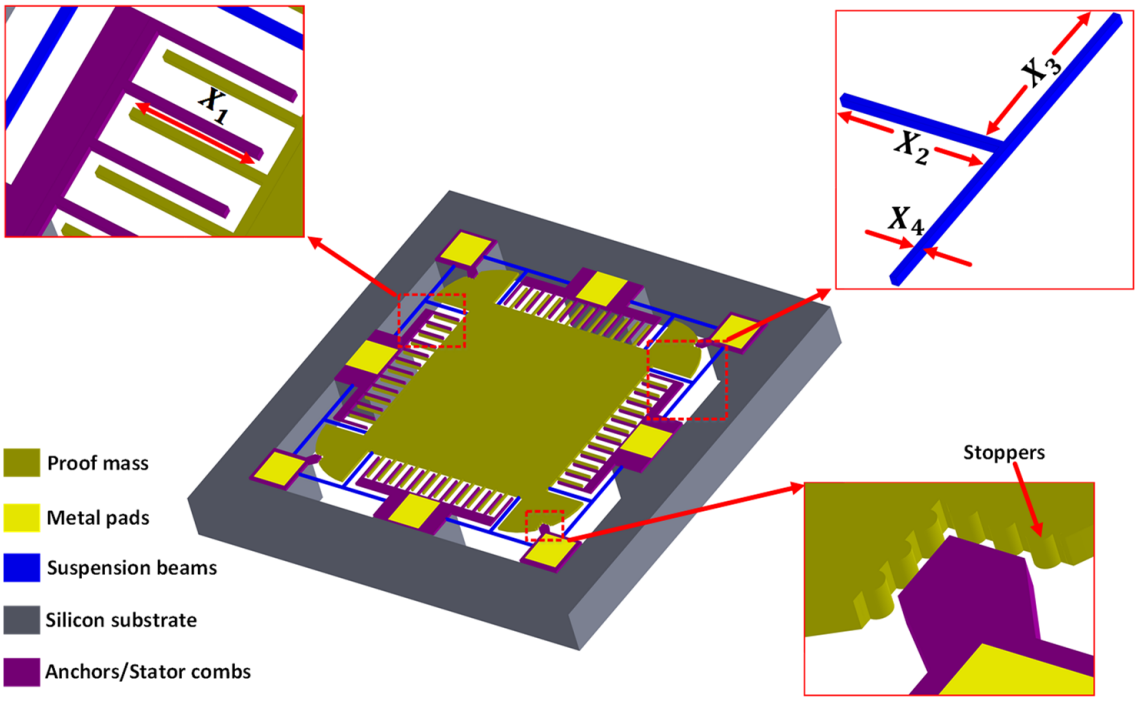

Figure 1. The 2-DoF MEMS capacitive MEMS accelerometer design [17].

The output characteristics considered for simultaneous optimization of the MEMS accelerometer are the central proof mass displacement for an input acceleration, the natural frequency, the pull-in voltage between the stator and rotor combs, the change in the capacitance between the stator and rotor combs for an input acceleration, and Brownian noise equivalent acceleration (BNEA).

3. Basics of Deep Learning Model

This section is aimed to develop some background of deep learning for a reader to facilitate an easier understanding of the proposed framework (Section 4) that is built on using these concepts.

The elementary unit of a deep learning network architecture is called a perceptron or an artificial neuron cell. When multiple perceptrons are combined, they form a complex logical system which is referred as a neural network. The simplest form of a perceptron is equivalent to an equation of a line. For the equation for a single perceptron unit, the slope of line (m) is replaced with weights (W), input (x) is replaced with input (X), y-intercept (c) with bias (B), and output (y) is replaced with function of input (f(X)), as shown in Equation (1) [24].

$$y = mx + c \Rightarrow f(X) = WX + B \tag{1}$$

$$y = g(f(X)) \tag{2}$$

The output response $f(X)$ is passed through an activation function to add non-linearity in the perceptron unit to make it able to separate data that is not separable by straight lines. In Equation (2), g represents the activation function that is applied to $f(X)$ to obtain the final output response y. Figure 2a shows the schematic representation of a perceptron with single value input. Multi variable input is represented as $x_1, x_2, \ldots, x_n$, where each x corresponds to an input variable, and each has a corresponding weight ($w_1, w_2, \ldots, w_n$), as shown in Figure 2b. For a simpler notation, the perceptron and activation function can be presented as a combined unit, as illustrated in Figure 2c.

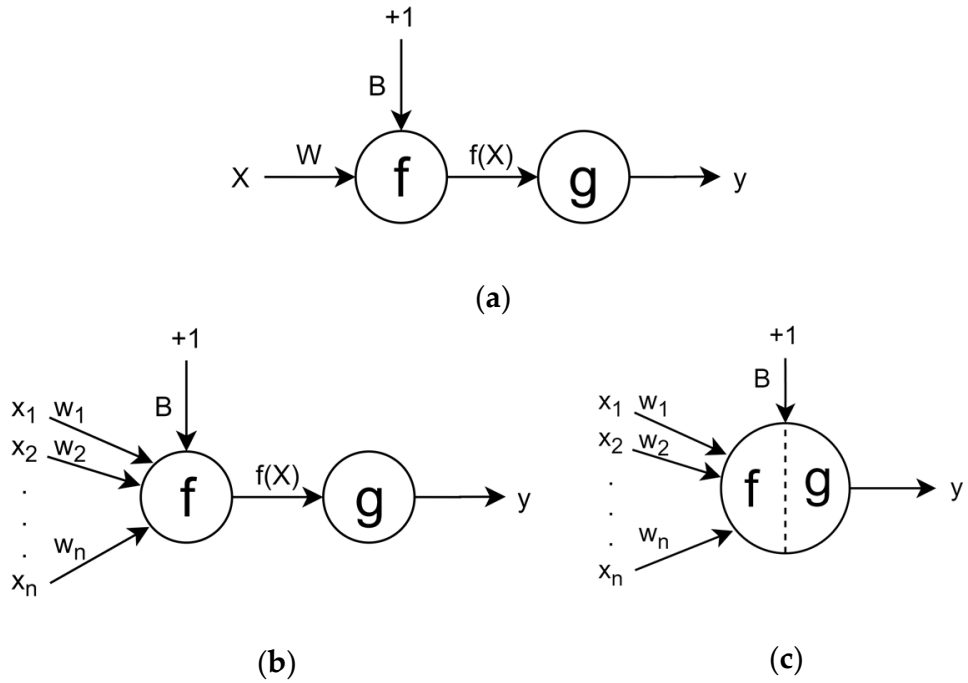

Figure 2. (**a**) Perceptron with single value input, (**b**) perceptron with multi-value input, and (**c**) merging perceptron and activation function.

Multiple perceptrons can be stacked in the vertical direction to form a layer (e.g., see Layer 1 in Figure 3). Each connection in Figure 3 has a corresponding weight, and the weights are stored in a 2D matrix and represented as W. The final output of this neural network is calculated using Equation (3). The left part of this equation represents the calculation occurring between Layer 0 (input) and Layer 1; first the dot product between input (X) and weight (W_1) is taken and bias (B_1) is added to the product. This submission is passed through an activation function (g) to obtain the output for each perceptron in Layer 1 which is represented as Y_1. The output of Layer 1 acts as the input to the next layer, Layer 2 (output). The calculation between Layer 1 and Layer 2 (output) is the right part of Equation (3), W_2 and B_2 are the weights and bias for Layer 2, Y_1 act as the input for this layer, and Y_2 is the final output of the neural network.

$$Y_1 = g(W_1 X + B_1), \;\; Y_2 = g(W_2 Y_1 + B_2) \tag{3}$$

Such a combination of perceptrons is collectively called a neural network (NN) [25]. Furthermore, a NN can be divided into three parts, an input layer (Layer 0), hidden layer (Layer 1), and an output layer (Layer 2), as shown in Figure 3. When there are 2 or more hidden layers, the NN is called a deep neural network. Since, on its own, a perceptron is simply an equation of a straight line (linear solution), an activation function is therefore needed to introduce non-linearity into the perceptron. Examples of some available activation functions include Sigmoid, Tanh, Rectified Linear Units (ReLU), Exponential Linear Unit (ELU), Swish, and Mish [26]. The learning process of a NN is based on a backpropagation algorithm [27] which uses gradient methods for decreasing the output error. After each training step, the output error is calculated using the prediction made by that current state of the NN. Back propagation is used to calculate the error for each neuron by going through the layers of NN in the reverse direction. Based on this calculated error for each neuron, the weights of the NN are updated. In summary, back propagation updates the weights of the NN to minimize the output error. To improve the generalization ability of NN, hyperparameter tuning [28] is required. A gradient-based algorithm [29] may get stuck at a local minima instead of reaching the global minima, or might even diverge instead of converging to a minima. Hence there is a requirement to select an appropriate

combination of activation function, learning rate, number of epochs, batch size, and weight initializer, along with the number of hidden layers and number of perceptrons per layer.

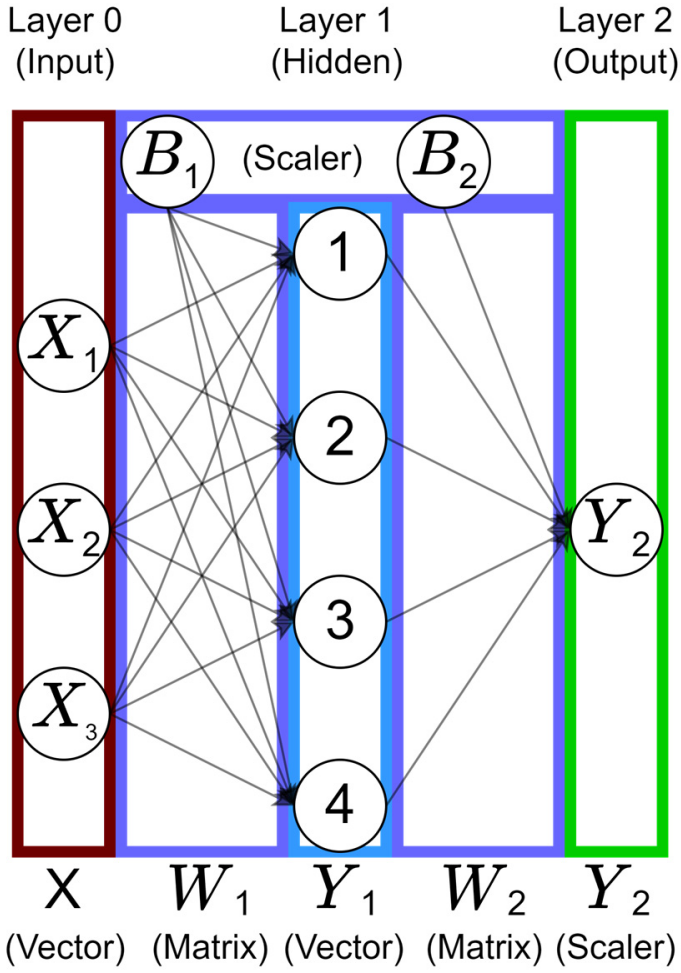

Figure 3. Representation of a neural network containing three inputs and a single output.

For the multiphysics design optimization of the MEMS accelerometer, we have created a deep neural network that is composed of 4 hidden layers such that each layer has a certain number of perceptrons along with their corresponding activation function for that layer. The number of layers and the number of perceptrons are set empirically. The selection of an activation function depends on the problem at hand. For the hidden layers, we experimented with different activation functions and obtained the best results with ELU in the first hidden layer, and ReLU in the preceding three hidden layers, Equations (5) and (6), respectively. Both ELU and ReLU affect the negative values. ELU uses an exponent operator for its function while ReLU sets the negative value to zero. In Equations (4)–(6), x is the value obtained after the dot product between the input and the weights and the addition of bias to this product.

$$g(x) = x \tag{4}$$

$$g(x) = \begin{cases} x, & x > 0 \\ 0, & x \leq 0 \end{cases} \tag{5}$$

$$g(x) = \begin{cases} x, & x > 0 \\ \exp(x) - 1, & x \leq 0 \end{cases} \tag{6}$$

The linear activation function is used in the last layer because the optimization of the MEMS accelerometer can be considered a regression problem in which the prediction of continuous values is desired.

4. Proposed Deep-Neural-Network-Based Framework

4.1. Design, Response, and Desirability Value Details

Table 1 shows the design variables (x_1, x_2, . . . , x_8) considered for the multiphysics design optimization of the MEMS accelerometer. These design parameters are the geometric parameters and the MEMS accelerometer operating conditions. The significance of the low and high levels for the design parameters has been discussed in [17]. The output responses considered for the optimization are natural frequency (y_1), proof mass displacement (y_2), pull-in voltage value (y_3), capacitance change corresponding to the input acceleration (y_4), and Brownian noise equivalent acceleration (BNEA) (y_5).

Table 1. Design parameters for the MEMS device.

Notation	Design Parameters	Low Level	High Level
x_1	Overlap length of comb	150 µm	250 µm
x_2	Length of suspension beam 1	400 µm	500 µm
x_3	Length of suspension beam 2	500 µm	500 µm
x_4	Width of suspension beam	6 µm	8 µm
x_5	Input acceleration	1 g	25 g
x_6	Operating temperature	233.15 K	373.15 K
x_7	Operating pressure	100 Torr	760 Torr
x_8	Frequency ratio	0.1	0.5

4.2. General Working of the Proposed Optimization Framework

The proposed optimization framework for the MEMS accelerometer is based on using a cascade of two separate neural network models, each relying on the architecture as discussed in the previous section. The first model, referred to as the Y model, is designed to predict the output response characteristics of the MEMS accelerometer (y_1, y_2, . . . , y_5) using the input design parameters (x_1, x_2, . . . , x_8). The second model is implemented for the simultaneous optimization of the output characteristics of the MEMS accelerometer with respect to the input design parameters, and is referred to as the D model. While the Y model enables a simultaneous prediction of the five output characteristics of the MEMS accelerometer, it does not allow to simultaneously optimize these five output characteristics with respect to the design parameters. The simultaneous optimization of the output characteristics is achieved through the D model, which is based on maximizing the desirability function corresponding to the optimization objective function [25,30]. Based on the output of the D model, the values of the eight input design parameters are ranked and the combination which gives the maximum desirability values is presented as the optimized solution. Figure 4 provides a high-level pictorial overview of the working of the proposed framework.

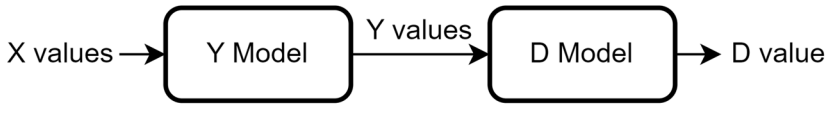

Figure 4. General working of the proposed parameter optimization methodology for the MEMS accelerometer. It is based on calculating the desirability (D) value for optimization using a cascade of two Deep Neural Networks (i.e., Y model and D model). X values correspond to (x_1, x_2, . . . , x_8); Y values correspond to (y_1, y_2, . . . , y_5); and D value refers to the desirability value.

4.3. Output Response Prediction Model

Figure 5 represents the deep neural network that is used to train the Y model for the output responses prediction. The input layer contains input features (x_1, x_2, . . . , x_8) corresponding to design variables and the output layer contains the corresponding output responses (y_1, y_2, . . . , y_5) that are to be predicted.

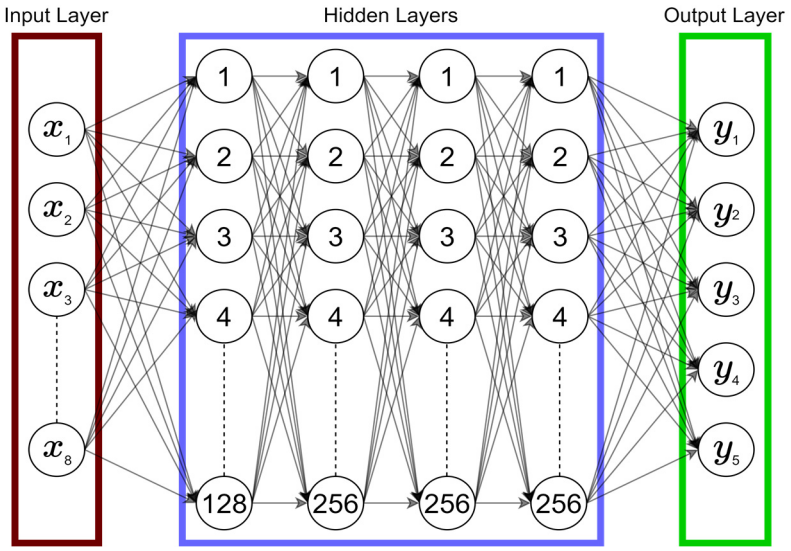

Figure 5. Representation of the architecture of the Y model.

To train a model for predicting Y_P values for a set of X values, we used the data provided by [17]. The data has 80 rows of values, each row has a set of X values generated using Latin hypercube sampling and will be represented as X_S; Ref. [17] obtained the Y values after performing simulations and these values will are represented as Y_{TS}. For our work we have normalized the values between 0 and 1 to standardize the scale of each input and output value. A split of 80/20 was made for hyper-parameter tuning and training of the model. Here, the assumption is that the simulated data (as provided by [17]) used for training the Y model was generated taking into account the realistic design conditions of the MEMS accelerometer. Figure 6 shows the steps involved in the training process as well as the evaluation of the Y model.

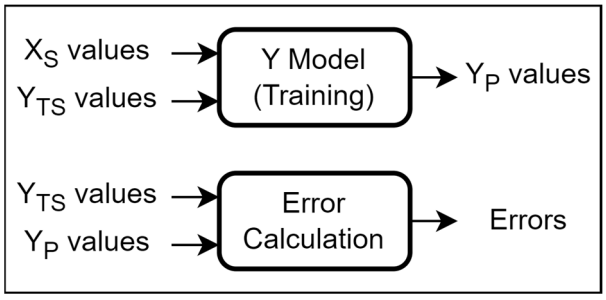

Figure 6. Steps for the training process and the evaluation of the Y model.

In comparison to [17], where five separate individual models were trained to obtain each output response value, the proposed method is based on training a single model to obtain all the output response values. The Y model is evaluated according to two error metrics, which are mean absolute error (*MAE*) and root mean squared error (*RMSE*), calculated using Equations (7) and (8), respectively.

$$MAE = \frac{1}{k}\sum_{i=1}^{k}|y_{oi} - y_{pi}| \quad (7)$$

$$RMSE = \sqrt{\frac{1}{k}\sum_{i=1}^{k}(y_{oi} - y_{pi})^2} \quad (8)$$

where y_{oi} is the true output value at index i and y_{pi} is the corresponding prediction value, and k is the total number of samples. The errors obtained are compared with the error values of [17]. It is observed that the proposed Y model has consistently outperformed [17] as shown in Table 2.

Table 2. Comparison of the predicted output responses (y_1, y_2, . . . , y_5) obtained using the proposed Y model with those obtained using the method in [17], in terms of MAE and RMSE.

Output Response	MAE		RMSE	
	Proposed	**[17]**	**Proposed**	**[17]**
Natural frequency (y_1)	12.67 Hz	29.64 Hz	15.41 Hz	41.19 Hz
Proof mass displacement (y_2)	0.004 µm	0.024 µm	0.004 µm	0.034 µm
Pull-in voltage (y_3)	0.065 V	0.085 V	0.072 V	0.134 V
Capacitance change (y_4)	5.292 fF	10.179 fF	6.28 fF	14.05 fF
BNEA (y_5)	0.004 µg/$\sqrt{\text{Hz}}$	0.019 µg/$\sqrt{\text{Hz}}$	0.005 µg/$\sqrt{\text{Hz}}$	0.029 µg/$\sqrt{\text{Hz}}$

4.4. Effect of Design Parameters on the Output Responses

The effect of variation of each design parameter (x_1, x_2, . . . , x_8) on the output responses (y_1, y_2, . . . , y_5) is also observed to obtain a deeper insight into their respective behaviors. In this regard, each input parameter is varied across its range while keeping all of the remaining input parameters fixed at the average of the low and high levels, as defined in Table 1. Since each output response has a different range and unit, they are normalized between 0 and 1 for comparisons.

Figure 7 shows the effect of variation of the overlap length of comb (x_1) on y_1, y_2, . . . , y_5. The graphs show that there is a much stronger impact of the variation of x_1 on the pull-in voltage (y_3) and BNEA (y_5) than on the natural frequency (y_1), proof mass displacement (y_2), and capacitance change (y_3). The results show that with an increase in the x_1, the pull-in voltage value decreases and BNEA increases for the MEMS accelerometer.

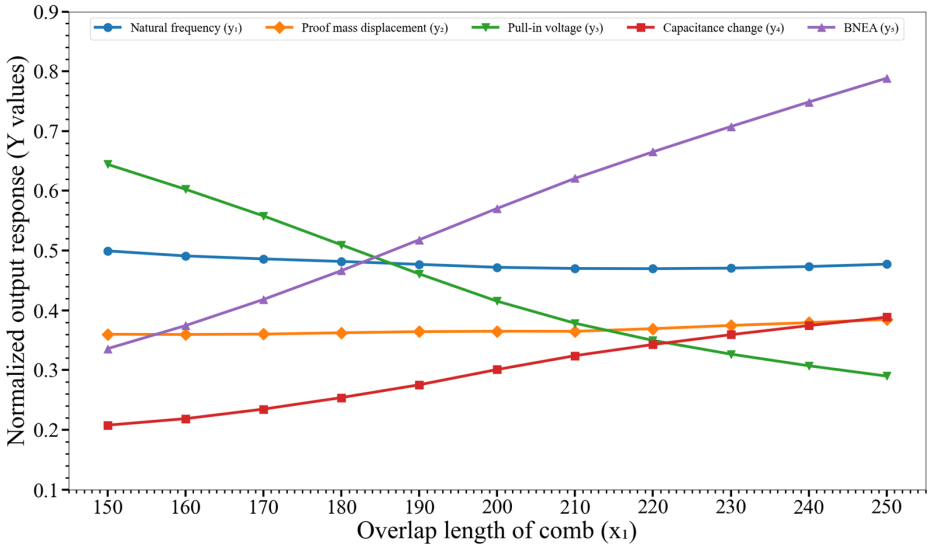

Figure 7. Effect of variation of the overlap length of comb (x_1) on the output responses y_1, y_2, . . . , y_5. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

Figure 8 shows the effect of variation of the length of the suspension beam 1 (x_2) on y_1, y_2, . . . , y_5. The results show that with an increase in the x_2 value, the natural frequency and pull-in voltage value for the MEMS accelerometer decreases while the proof mass displacement and capacitance change for an input acceleration increases. Moreover, the effect of change in the x_2 on the MEMS accelerometer BNEA value is negligible.

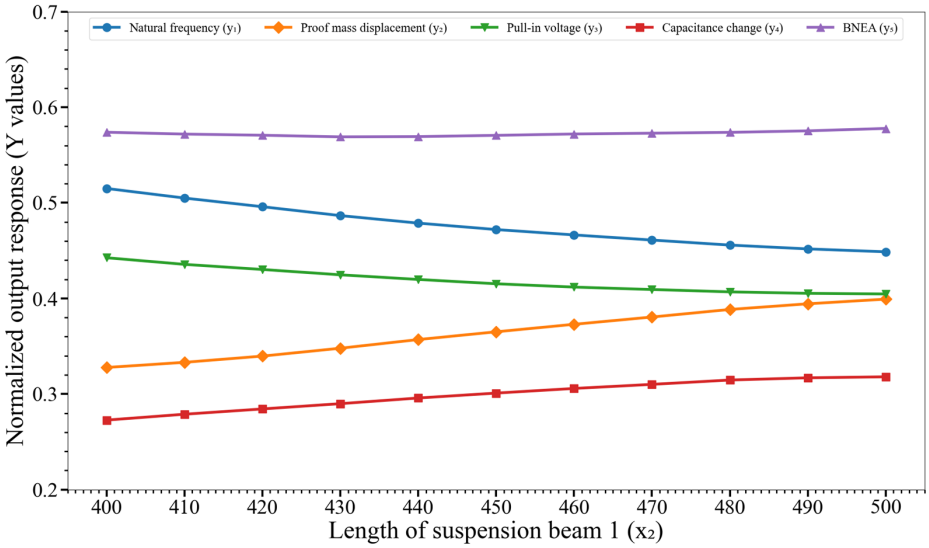

Figure 8. Effect of variation of the length of the suspension beam 1 (x_2) on the output responses y_1, y_2, . . . , y_5. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

Figure 9 shows the effect of the variation of the length of suspension beam 2 (x_3) on y_1, y_2, . . . , y_5. The strongest effect of the variation of x_3 is clearly visible on the natural frequency (y_1) that matches with the findings of [17]. Additionally, the graph also shows that x_3 also contributes at varying levels towards the proof mass displacement (y_2), pull-in voltage (y_3), and capacitance change (y_4).

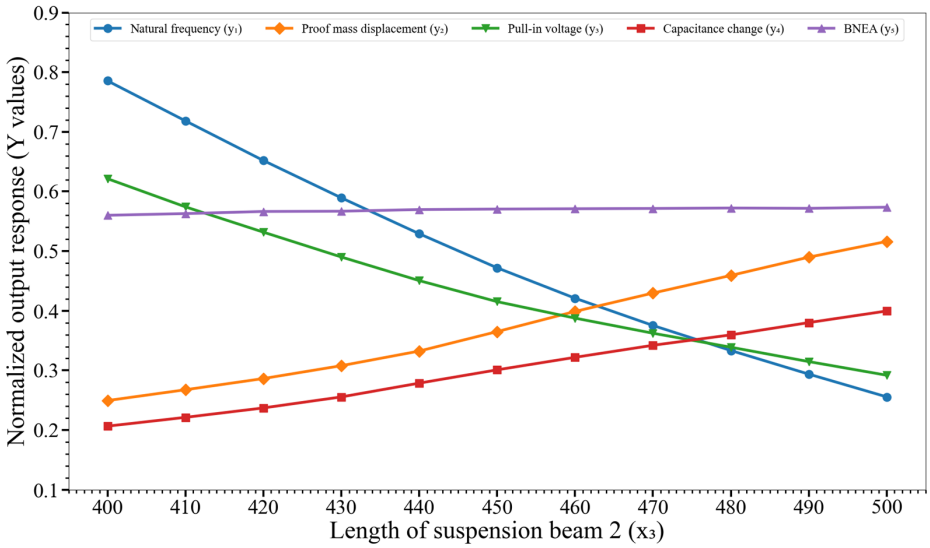

Figure 9. Effect of variation of the length of suspension beam 2 (x_3) on the output responses y_1, y_2, . . . , y_5. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

Figure 10 shows the effect of variation of the width of the suspension beam (x_4) on $y_1, y_2, \ldots, y_5$. The graph shows that there is a significant impact of variation of x_4 on the natural frequency (y_1), proof mass displacement (y_2), pull-in voltage (y_3), and capacitance change (y_4).

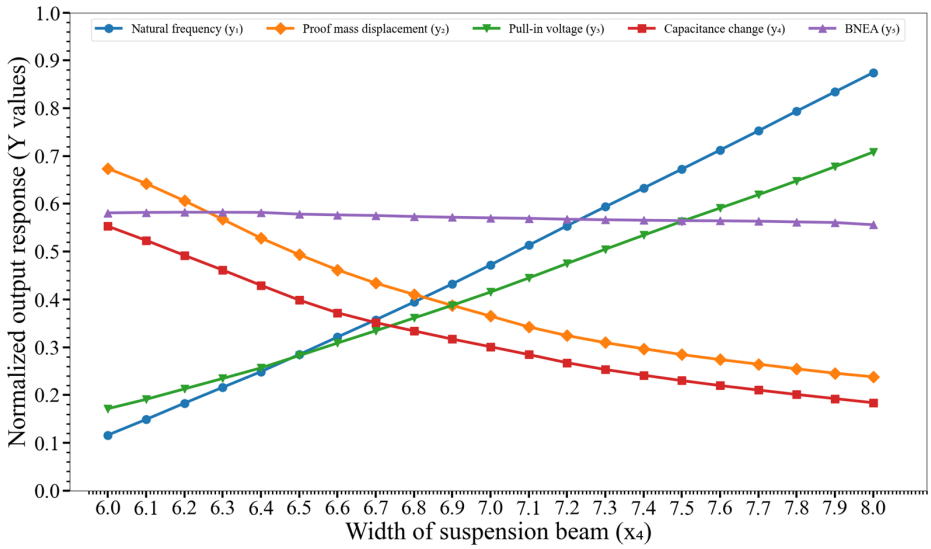

Figure 10. Effect of variation of the width of the suspension beam (x_4) on the output responses $y_1, y_2, \ldots, y_5$. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

Figure 11 shows the effect of variation of the input acceleration (x_5) on $y_1, y_2, \ldots, y_5$. It is evident from the graphs that x_5 strongly impacts the proof mass displacement (y_2) and capacitance change (y_4).

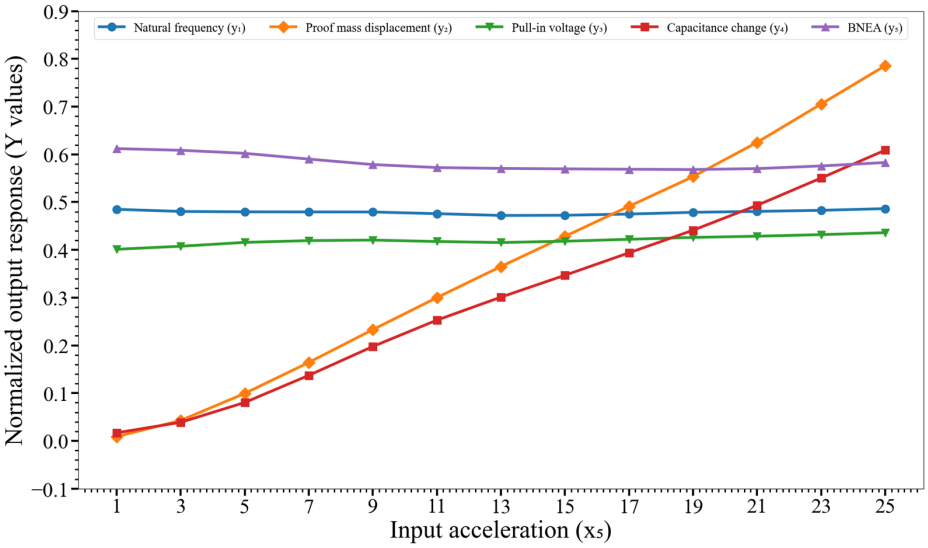

Figure 11. Effect of variation of the input acceleration (x_5) on the output responses $y_1, y_2, \ldots, y_5$. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

Figure 12 shows the effect of variation of the operating temperature (x_6) on $y_1, y_2, \ldots, y_5$. The graphs that only BNEA (y_5) is impacted by the variation of x_6, whereas the remaining output responses are not perturbed.

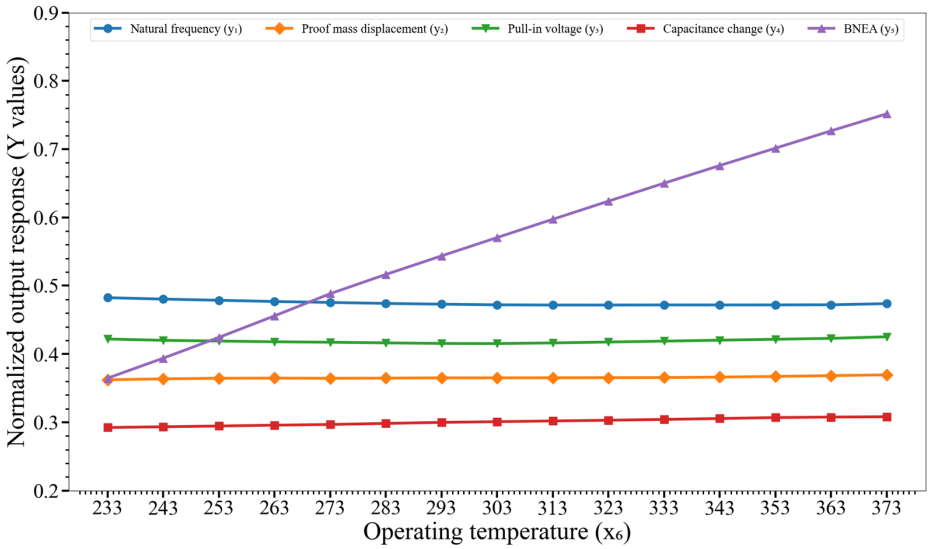

Figure 12. Effect of variation of the operating temperature (x_6) on the output responses $y_1, y_2, \ldots, y_5$. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

Figure 13 shows the effect of variation of the operating pressure (x_7) on $y_1, y_2, \ldots, y_5$. The behavior is similar to that observed for the case of x_6, i.e., x_7 also impacts only the BNEA (y_5), without really perturbing the remaining output responses.

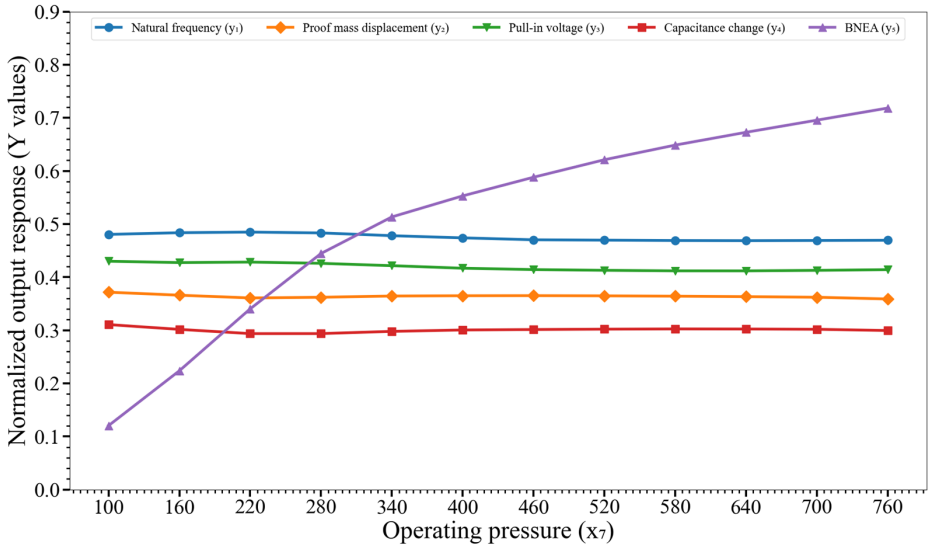

Figure 13. Effect of variation of the operating pressure (x_7) on the output responses $y_1, y_2, \ldots, y_5$. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

Figure 14 shows the effect of variation of the frequency ratio (x_8) on $y_1, y_2, \ldots, y_5$. There is not a strong impact of variation of x_8 on any output response that is in line with observations of [17]; though y_2 and y_4 appear slightly perturbed.

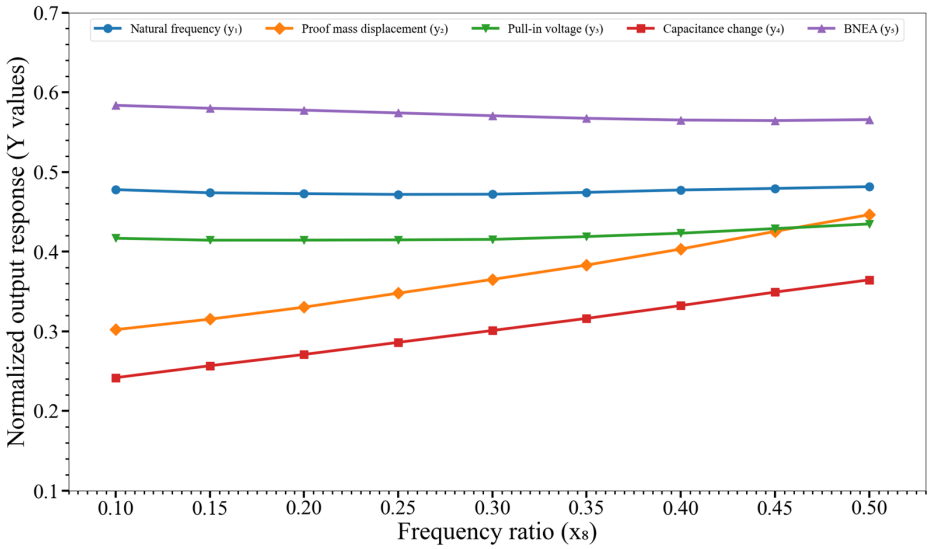

Figure 14. Effect of variation of the frequency ratio (x_8) on the output responses y_1, y_2, . . . , y_5. Key. y_1: natural frequency; y_2: proof mass displacement; y_3: pull-in voltage; y_4: capacitance change; and y_5: BNEA.

The results presented in Figures 6–13 allow to analyze the effect of the design parameters of the MEMS accelerometer on the five output responses simultaneously. The sensitivity analysis for each design parameter has been performed in terms of the effect of the variation of the design parameters on the output responses, and then comparing them with the results presented in [17], which showed a consistent behavior for each design parameter. Thus, the proposed deep-learning-based Y model allows to efficiently explore the MEMS accelerometer design space. Additionally, the effectiveness of the model has already been quantitatively demonstrated in the form of the comparison of the predicted output responses (y_1, y_2, . . . , y_5) obtained using the proposed Y model with those obtained using the method in [17], based on MAE and RMSE scores (Table 2). Moreover, unlike the procedure adopted in [17] for generating the simulated data that was extremely time consuming (taking extended periods of time to complete), the proposed Y model (once trained) offers an alternative for generating more data (where needed) in the design space in an accurate and time-efficient manner without the need to perform simulations over longer periods of time. In fact, in the next section, the trained Y model is used to generate a larger dataset as required for training the D model.

5. Multiresponse Optimization Using D Prediction Model

5.1. Training of the D Prediction Model

For the training of the D model, we use a larger dataset generated using the Y model. The deep neural network used for training the D model is exactly the same (except, of course, the input and output layers) as the one for the Y model (Figure 5). As for the dataset, it has been generated by first creating a list of different combinations of X inputs. This was performed by incrementing from the low and high level of X values. The increment was set between 15 and 25 percent of the low level, which generated a set of X values, referred to as X_G values (Figure 15). The X_G values are passed to the Y model to obtain the corresponding Y_G values. For each Y_G value, the D_G value was estimated using the same approach used in [17], and this obtained D value is represented as the D_{TG} value, or true value of desirability for the generated set of X_G values. A total of 3125 rows of values were obtained. All the values were normalized between 0 and 1 to standardize the scale of each input and output value. A split of 80/20 was made for hyperparameter tuning and training of the model. For training of the D model, the input layer therefore contains y_1, y_2, . . . , y_5 and the output layer contains only a desirability value. The manual formula-based calculation of D (as in [17]) is thus replaced with a robust deep-learning-based D model.

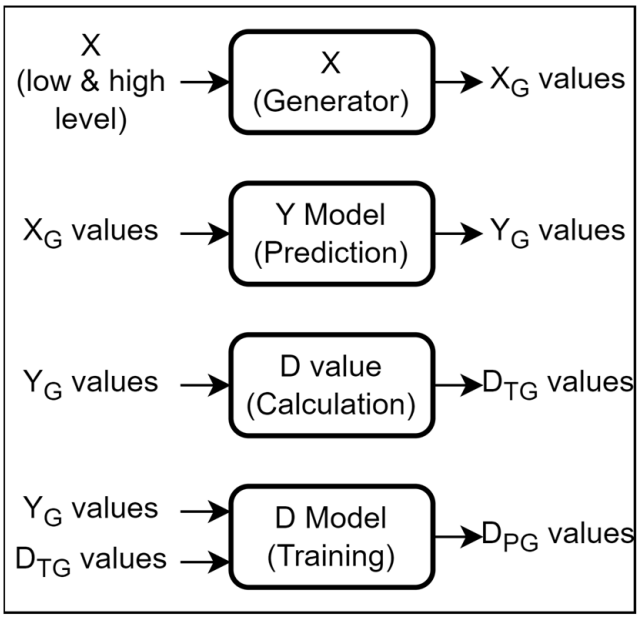

Figure 15. Steps involved in the training process of the D Model.

5.2. Multi-Response Optimization

To find the optimized values for X with respect to maximum D, we have proposed a method as illustrated in Figure 16. For the optimization process a dataset of about 100 K values was generated for the X values with an increment value below 10 percent of the lower bound; this set of X values is represented as X_R. The X_R is fed through the Y model to obtain the Y_R. The obtained Y_R values are fed to the D model to obtain the D_R, which are the D values for the corresponding design parameters. Then, the index of the maximum D value is searched, and the corresponding Y and X to this maximum index are considered as the optimized x_1, x_2, . . . , x_8 values. Table 3 presents the values obtained from the proposed method and the values reported in [17].

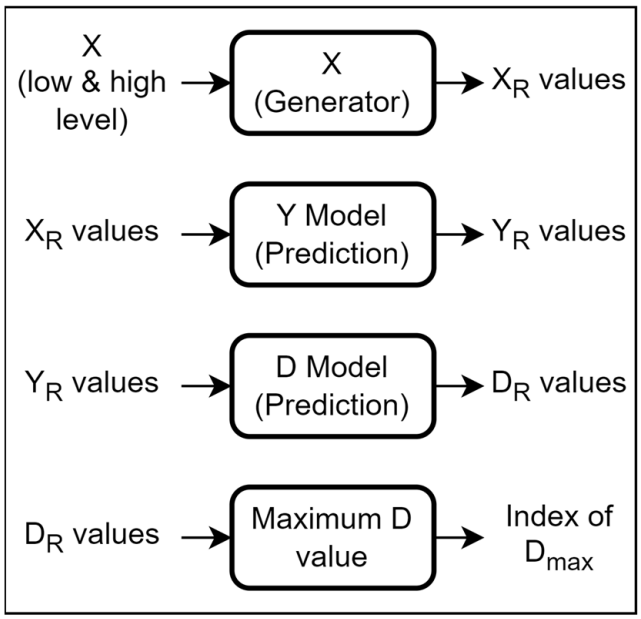

Figure 16. Final Optimization pipeline.

Table 3. Comparison of the obtained optimized X values (x_1, x_2, ... , x_8) using the proposed method with those reported in [17]. The corresponding Y values (y_1, y_2, ... , y_5) are also listed.

Optimized Values Reported in [17]								
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
Design Parameters (X values)	153.6 µm	403.6 µm	500 µm	6.26 µm	25 g	300 K	760 Torr	0.50
	y_1	y_2	y_3	y_4	y_5			
Output Responses (Y values)	3036.4 Hz	0.903 µm	6.76 V	676.2 fF	0.81 µg/$\sqrt{\text{Hz}}$			
Optimized Values Obtained Using the Proposed Method.								
	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8
Design Parameters (X values)	150.0 µm	430.0 µm	500 µm	6.40 µm	25 g	300 K	760 Torr	0.45
	y_1	y_2	y_3	y_4	y_5			
Output Responses (Y values)	3160.0 Hz	0.723 µm	7.22 V	571.0 fF	0.83 µg/$\sqrt{\text{Hz}}$			

To further validate the obtained results, we performed the statistical significance testing at the standard 5% significance level using the two-sample *t*-test. The Y values (y_1, y_2, ... , y_5) predicted using the optimized X values (x_1, x_2, ... , x_8) based on the proposed method are listed in Table 3. For comparison, we computed the Y values (referred to as observed Y values) by performing FEM simulations in CoventorWare® software (Coventor, Raleigh, NC, USA) using the same X values as obtained based on the proposed method. The computed observed Y values are as follows: y_1 = 3096.43 Hz, y_2 = 0.676 µm, y_3 = 7.0303 V, y_4 = 521 fF, and y_5 = 0.7959 µg/$\sqrt{\text{Hz}}$. Here, the *t*-test is performed to test the null hypothesis that the data in the two samples (predicted Y values and observed Y values) is derived from independent random samples having normal distributions of equal means and equal but unknown variances. The results of the *t*-test show that the null hypothesis is not rejected with *p*-value = 0.98, thus confirming that the data in the two samples is statistically highly similar.

6. Conclusions

This paper proposed a design optimization methodology for a dual-axis microelectromechanical systems (MEMS) capacitive accelerometer based on the use of a cascade of two deep neural network (DNN) models. Each model is made up of 4 hidden layers. The first hidden layer is composed of 128 perceptrons and ELU as activation function. The other three layers have 256 perceptrons and ReLU as activation function. A linear activation function was used in the output layer, as a regression system is required. The first instance is named as Y model and is used to predict the output response values. Y Model was trained on the original 80 values, as made available by [17]. Using the trained Y model and the ranges of design parameters, a larger dataset of 3125 values was generated. This generated dataset was used to train the second instance that is named as the D model. The output of the D model is the desirability value on which the design parameters are ranked and accordingly optimized.

The proposed method enabled an analysis of the effect of the individual design parameters on the output responses of the sensor using Y model. Additionally, the D model allowed a simultaneous optimization of the multiple output responses of the MEMS accelerometers in an efficient manner. Compared to the work [17] in which five separate models based on the Gaussian process were trained (one for each output response), plus the use of a desirability function, the proposed method is computationally less complex and more efficient as it offers a unified solution using a DNN model (replacing five separate models of [17]), which has been demonstrated to be more accurate and effective as compared to [17]. The results of the proposed method are also validated by means of statistical significance testing.

Author Contributions: Conceptualization, M.M.S., T.N. and F.A.M.; methodology, F.A.M. and T.N.; software, F.A.M.; validation, F.A.M., T.N. and M.M.S.; formal analysis, F.A.M. and T.N.; investigation, F.A.M., T.N., M.M.S., U.S.K. and A.H.; resources, M.M.S. and T.N.; data curation, M.M.S.; writing—original draft preparation, F.A.M., T.N. and M.M.S.; writing—review and editing, T.N., M.M.S., U.S.K. and A.H.; visualization, F.A.M. and T.N.; supervision, T.N. and M.M.S.; project administration, M.M.S. and T.N.; funding acquisition, M.M.S., T.N. and U.S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Higher Education Commission of Pakistan and the National Centre of Robotics and Automation under Grant DF 1009-0031.

Data Availability Statement: No new data were created or analyzed in this study. Data sharing is not applicable to this article.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. Zhang, M.; Wang, Q.; Liu, D.; Zhao, B.; Tang, J.; Sun, J. Real-time gait phase recognition based on time domain features of multi-MEMS inertial sensors. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 7504012. [CrossRef]
2. Zheng, J.; Li, S.; Liu, S.; Guan, B.; Wei, D.; Fu, Q. Research on the Shearer Positioning Method Based on the MEMS Inertial Sensors/Odometer Integrated Navigation System and RTS Smoother. *Micromachines* **2021**, *12*, 1527. [CrossRef] [PubMed]
3. Tsai, J.M.; Sun, I.C.; Chen, K.S. Realization and performance evaluation of a machine tool vibration monitoring module by multiple MEMS accelerometer integrations. *Int. J. Adv. Manuf. Technol.* **2021**, *114*, 465–479. [CrossRef]
4. Di Nuzzo, F.; Brunelli, D.; Polonelli, T.; Benini, L. Structural health monitoring system with narrowband IoT and MEMS sensors. *IEEE Sens. J.* **2021**, *21*, 16371–16380. [CrossRef]
5. Wang, C.; Hao, Y.; Sun, Z.; Zu, L.; Yuan, W.; Chang, H. Design of a Capacitive MEMS Accelerometer with Softened Beams. *Micromachines* **2022**, *13*, 459. [CrossRef] [PubMed]
6. Tahir, M.A.; Saleem, M.M.; Bukhari, S.A.; Hamza, A.; Shakoor, R.I. An efficient design of dual-axis MEMS accelerometer considering microfabrication process limitations and operating environment variations. *Microelectron. Int.* **2021**, *38*, 144–156. [CrossRef]
7. Liu, Y.; Hu, B.; Cai, Y.; Liu, W.; Tovstopyat, A.; Sun, C. A novel tri-axial piezoelectric MEMS accelerometer with folded beams. *Sensors* **2021**, *21*, 453. [CrossRef]
8. Kavitha, S.; Daniel, R.J.; Sumangala, K. High performance MEMS accelerometers for concrete SHM applications and comparison with COTS accelerometers. *Mech. Syst. Signal Process.* **2016**, *66*, 410–424. [CrossRef]
9. Abozyd, S.; Toraya, A.; Gaber, N. Design and Modeling of Fiber-Free Optical MEMS Accelerometer Enabling 3D Measurements. *Micromachines* **2022**, *13*, 343. [CrossRef]
10. D'Alessandro, A.; Scudero, S.; Vitale, G. A review of the capacitive MEMS for seismology. *Sensors* **2019**, *19*, 3093. [CrossRef]
11. Keshavarzi, M.; Yavand Hasani, J. Design and optimization of fully differential capacitive MEMS accelerometer based on surface micromachining. *Microsyst. Technol.* **2019**, *25*, 1369–1377. [CrossRef]
12. Benmessaoud, M.; Nasreddine, M.M. Optimization of MEMS capacitive accelerometer. *Microsyst. Technol.* **2013**, *19*, 713–720. [CrossRef]
13. Mohammed, Z.; Dushaq, G.; Chatterjee, A.; Rasras, M. An optimization technique for performance improvement of gap-changeable MEMS accelerometers. *Mechatronics* **2018**, *54*, 203–216. [CrossRef]
14. Xu, X.; Wu, S.; Fang, W.; Yu, Z.; Jia, Z.; Wang, X.; Bai, J.; Lu, Q. Bandwidth Optimization of MEMS Accelerometers in Fluid Medium Environment. *Sensors* **2022**, *22*, 9855. [CrossRef]
15. Pedersen, C.B.; Seshia, A.A. On the optimization of compliant force amplifier mechanisms for surface micromachined resonant accelerometers. *J. Micromech. Microeng.* **2004**, *14*, 1281. [CrossRef]
16. Ramakrishnan, J.; Gaurav, P.R.; Chandar, N.S.; Sudharsan, N.M. Structural design, analysis and DOE of MEMS-based capacitive accelerometer for automotive airbag application. *Microsyst. Technol.* **2021**, *27*, 763–777. [CrossRef]
17. Saghir, S.; Saleem, M.M.; Hamza, A.; Riaz, K.; Iqbal, S.; Shakoor, R.I. A Systematic Design Optimization Approach for Multiphysics MEMS Devices Based on Combined Computer Experiments and Gaussian Process Modelling. *Sensors* **2021**, *21*, 7242. [CrossRef]
18. Shinde, P.P.; Shah, S. A Review of Machine Learning and Deep Learning Applications. In Proceedings of the 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 16–18 August 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–6.
19. Nguyen, G.; Dlugolinsky, S.; Bobák, M.; Tran, V.; López García, Á.; Heredia, I.; Malík, P.; Hluchý, L. Machine learning and deep learning frameworks and libraries for large-scale data mining: A survey. *Artif. Intell. Rev.* **2019**, *52*, 77–124. [CrossRef]
20. Mathew, A.; Amudha, P.; Sivakumari, S. Deep Learning Techniques: An Overview. Advanced Machine Learning Technologies and Applications. *Proc. AMLTA* **2020**, *2021*, 599–608.
21. Coşkun, M.; Yildirim, Ö.; Ayşegül, U.Ç.; Demir, Y. An overview of popular deep learning methods. *Eur. J. Tech.* **2017**, *7*, 165–176. [CrossRef]

22. Cowen, A.; Hames, G.; Monk, D.; Wilcenski, S.; Hardy, B. *SOIMUMPs Design Handbook*; MEMSCAP Inc.: Durham, NC, USA, 2011; pp. 2002–2011.
23. Le, X.L.; Kim, K.; Choa, S.H. Analysis of Temperature Stability and Change of Resonant Frequency of a Capacitive MEMS Accelerometer. *Int. J. Precis. Eng. Manuf.* **2022**, *23*, 347–359. [CrossRef]
24. Derringer, G.; Suich, R. Simultaneous Optimization of Several Response Variables. *J. Qual. Technol.* **1980**, *12*, 214–219. [CrossRef]
25. Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychol. Rev.* **1958**, *65*, 386–408. [CrossRef] [PubMed]
26. Dubey, S.R.; Singh, S.K.; Chaudhuri, B.B. Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing* **2022**, *503*, 92–108. [CrossRef]
27. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning representations by back-propagating errors. *Nature* **1986**, *323*, 533–536. [CrossRef]
28. Bergstra, J.; Bengio, Y. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* **2012**, *13*, 2.
29. Ruder, S. An overview of gradient descent optimization algorithms. *arXiv* **2016**, arXiv:1609.04747.
30. Del Castillo, E.; Montgomery, D.C.; McCarville, D.R. Modified Desirability Functions for Multiple Response Optimization. *J. Qual. Technol.* **1996**, *28*, 337–345. [CrossRef]