Data – Driven Modelling of the Interaction Force between Permanent Magnets

Van Tai Nguyen[1]*, Michael Bermingham[1] and Matthew S. Dargusch[1],

[1]School of Mechanical and Mining Engineering, Faculty of Engineering, Architecture and Information Technology, the University of Queensland, Brisbane St Lucia, QLD 4072, Australia.

[*]Corresponding author: Van Tai Nguyen (Email: vantai.nguyen@uq.edu.au or 88.vantai.nguyen@gmail.com)

Abstract

Understanding the magnetic interaction force between permanent magnets is important for the design and optimization of the system where they are implemented. However, the methods that are utilized in the literature to compute this force are either time-consuming or approximated with a low degree of generalization. This article presents a surrogate model developed based on a data-driven approach using a deep learning method which addresses this problem. Firstly, a charge model is applied to derive a semi-analytical model (SAM) of the interaction forces between permanent magnets. Using this SAM, the features of the deep learning model (DLM) have been selected, and the training, validation and test datasets that are used to train the DLM have been generated. The DLM training process took 2 hours and 30 minutes to complete. The difference between the SAM and deep learning model is less than 4.2 %, and there are 99.2 % and 96.05 % of the cases over 885 random tested samples where the errors are less than 2 % and 1 %, respectively; this indicates that the selected deep learning model is feasible and can provide accurate results compared to the original SAM. Moreover, the permutation feature importance (PFI) analysis shows that the most predictive feature is the separation distance between the magnets, and the heights of the magnets have less predictive power than their radii; the generality of the deep learning model is also demonstrated based on the PFI criteria. Furthermore, compared with Finite Element Analysis (FEA) and the SAM, the surrogate model yields a high accuracy of prediction (the minimum, average and maximum differences between the surrogate and FEA models are 0.06 %, 0.42 % and 1.74 %, respectively) while it required a computational time less than 10^{-4} s, which is multiple orders of magnitude lower than its FEA and SAM counterparts. The developed data-driven surrogate model can facilitate the design, optimization processes of permanent magnet systems and online computation of the magnetic force through a dynamic study. In addition, using the superposition principle, the magnetic forces between cross-shaped permanent magnets can be computed using the surrogate model. The authors have further designed a user-friendly software interface to compute the magnetic force using the recently developed surrogate model; the software is publicly available under the CC BY 4.0 license, and can be found at: https://github.com/vantainguyen/Force-between-magnets-machine-learning.

Keywords: Data-driven modelling; Interaction force; Magnetic force; Permanent magnets.

I. Introduction

Computation of the interaction force between permanent magnets is essential for the design, optimization, and the dynamic study of magnetic devices [1, 2] such as magnetic springs [3], energy-harvesting applications [4, 5], permanent magnet motors, couplings and gears [1], medical robots [6] and soft-robots [7]. For example, in the work of Abdullah et al. [4], three permanent magnets are arranged coaxially in a novel energy harvester; in order to perform the dynamic analysis and characterization of this harvester, the interaction force between these permanent magnets is computed. In a soft robotic application studied by Kwok et al. [7], Neodymium-iron-boron (NdFeB) ring magnets are embedded in silicone elastomers and acrylonitrile-butadiene-styrene (ABS) to make and maintain the connections between these components; the magnetic force between these magnets as a function of distance of separation is calculated and analysed to assist the robot design process. In general, however, the magnetic force computation task is challenging and time-consuming due to the nature of Electromagnetism [8]. Even though this force can be computed using traditional Finite Element Analysis (FEA), this conventional method has a high computational cost and requires large computer memory resources [6, 9] which is not convenient for the purpose of optimization and real-time dynamic computational studies. In order to deal with this issue, the interaction force between permanent magnets is normally approximated with polynomial expressions that are applied to study the dynamic behaviour of the system [4]. Nonetheless, these approximations can be performed when the exact material and geometrical parameters of the magnets are known; in other words, the polynomial models need to be redetermined if a change in the parameters of the permanent magnets is required which is inconvenient and time-consuming for a parametric optimization study. Other semi-analytical expressions [6, 10] have been derived to compute the magnetic force between permanent magnets with more generalized geometrical shapes. However, they involve complicated terms [11], and the solutions involve numerical integration of multiple-integral expressions which again can be time-consuming [2, 12] for optimization and real-time computational purposes.

Currently, machine learning has evolved as a powerful tool which can help solve complex problems ranging from physics to industrial manufacturing [9, 13, 14]. Roy and Wodo [9] implemented neural networks to model the thermal history in additive manufacturing; as a result, the surrogate model that was developed can compute the thermal history much faster (1000 times) than traditional Finite Element Analysis which is beneficial for real-time prediction of the thermal history. Khan et al. [13] utilized the deep learning technique to formulate a model to predict the magnetic field distribution of electric motors with a wide range of the components' dimensions at low computational cost. Moreover, a Poison's equation has been solved using the deep learning technique with low computational time by Shan et al. [14]. The machine leaning method has therefore been shown to be useful in many applications, and it can potentially be applied to formulate a fast-computed model to calculate the magnetic interaction force between permanent magnets. Therefore, this study focuses on a rapidly executable and data-driven based model of the magnetic interaction force between permanent magnets using the deep learning technique. More details of the theory of deep learning can be found in the work of Goodfellow et al. [15].

The fast-computed model can facilitate the design and optimization processes of permanent magnet systems such as parametric design and optimization of magnetic springs, magnetic levitation systems and soft robots with embedded permanent magnets etc. In this study, the levitation force between two coaxial permanent magnets with a cylindrical shape is the focus;

however, this research can also provide a general guide to develop the magnetic force interaction between permanent magnets of any shape with an arbitrary orientation using the machine learning method. In the development of a machine learning model, feature selection is one of the important factors as it can affect the dataset volume required for the training and validation process. It is noted that the optimal machine learning model should have the minimum required input features which can save training time and memory resources of the computer used for the training process. One of the advantages of the data-driven approach presented in this article is that it optimizes the required input features of the machine learning model.

The data-driven approach includes the following steps. Firstly, a simplified semi-analytical model (SAM) to compute the magnetic force is derived based on the charge model [1]. Using this SAM, the architecture of the SM is constructed. Furthermore, the optimized features of the deep learning model are selected, and the datasets implemented for the training process have been generated using the SAM. The results and efficiency of the SM is verified with those of the FEA and the SAM. The permutation feature importance of the magnets' geometrical parameters and a test on the generality of the deep learning model w.r.t these parameters are carried out. Finally, a user-friendly software designed to compute the magnetic force based on the surrogate model is described.

The authors hope that the presented frameworks and results in this article can be an avenue to inspire further research in the application of machine learning method to solve other complex physics-based problems such as dipole interactions for micromagnetic analyses [16 - 19], and particularly the magnetic forces interaction between magnets of complex shapes different from those presented in this paper.

The remainder of this article is organized in the following sections. Section II explains in detail the underlying physics behind the interaction force between permanent magnets, and a semi-analytical model for the magnetic force calculation is developed. Section III formulates the surrogate model based on the deep learning method. Section IV describes the data generation and parameter selection for the deep learning model. Section V presents the training and verification results of the developed model. Feature importance analysis and a test on the generality of deep learning model is described in section VI. Section VII discusses the use of the superposition principle for permanent magnet addition and subtraction. A user-friendly software interface is presented in section VIII. Finally, the conclusions are provided in Section IX.

II. Underlying physics and the semi-analytical model for magnetic force calculations

In order to facilitate the development of the novel surrogate model based on the deep learning methods to compute the magnetic interaction force between two permanent magnets, a semi-analytical model of this force has been formulated based on the charge model [1]. Figure 1 depicts the schematic of two cylindrical permanent magnets with a co-axial arrangement in the Cartesian coordinate system OXYZ. The lower magnet has the radius of R (m) and thickness of h (m) with the axial magnetization vector $\vec{J}$; in addition, the upper magnet has the radius of R_1 (m) and thickness of h_1 (m) with the axial magnetization vector $\overrightarrow{J_1}$ in an opposite direction to $\vec{J}$; the separation distance between these magnets is ξ (m). It is noted that the unit of magnetic

remanences J and J_1 (the magnitudes of the magnetization vectors $\vec{J}$ and $\overrightarrow{J_1}$, respectively) used in this study is Tesla (1 T = 1 H·A/m^2).

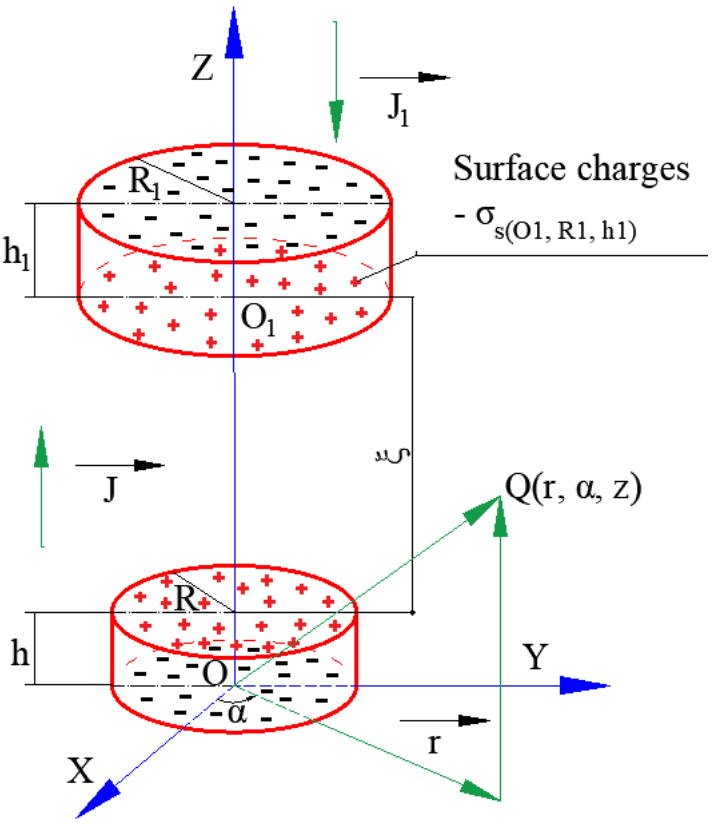

Fig. 1 – Schematic of two co-axial permanent magnets.

Due to the charge model [1], the magnetic force between these two permanent magnets can be calculated using Eq. (1). The surface integral in Eq. (1) reveals the contribution of the upper magnet's surface charge density $\sigma_{s(O_1,R_1,h_1)}$ to the force $\vec{F}$ (N); which is analogous, to the volume integral in Eq. (1) that represents the contribution of the upper magnet's volume charge density $\sigma_{v(O_1,R_1,h_1)}$ to the force $\vec{F}$ (N). In addition, $\vec{B}_{(O,R,h)}$ (T) is the magnetic field generated by the lower magnet.

$$\vec{F} = \iint_s \vec{B}_{(O,R,h)}\sigma_{s(O_1,R_1,h_1)}ds + \iiint_v \vec{B}_{(O,R,h)}\sigma_{v(O_1,R_1,h_1)}dv, \quad (1)$$

The surface charge density $\sigma_{s(O_1,R_1,h_1)}$ (A/m) can be determined as follows Eq. (2):

$$\sigma_{s(O_1,R_1,h_1)} = \vec{n}.\overrightarrow{J_1}/\mu_0 = \begin{cases} J_1/\mu_0 \text{ for the lower surface} \\ 0 \text{ for the cylindrical surface} \\ -J_1/\mu_0 \text{ for the upper surface} \end{cases}, \quad (2)$$

where $\mu_0 = 4\pi \times 10^{-7}$ (H/m) is the permeability of free space, and $\vec{n}$ is the unit vector normal to the corresponding surface.

The volume charge density is $\sigma_{v(O_1, R_1, h_1)} = -\vec{\nabla} \cdot \vec{J_1}/\mu_0 = 0$ (A/m^2), (3) as the magnetization is uniformly axially oriented.

Therefore, only the component with the surface charge density contributes to the interaction force between the given magnets Eq. (4).

$$\vec{F} = \iint_S \vec{B}_{(O,R,h)} \sigma_{s(O_1, R_1, h_1)} ds, \quad (4)$$

Due to the geometrical symmetry of the magnets and the current arrangement, only the axial component of the magnetic field $\vec{B}_{(O,R,h)}$ contributes to the interaction force (calculated using Eq. (4)) between these permanent magnets.

In the author's previous article [2], a semi-analytical model using fast computation time was derived in order to calculate the magnetic field generated by an elliptical cylinder with axial magnetization. These expressions can be applied to compute the magnetic field distribution of a circular cylinder provided the semi-major and semi-minor axes are equal. Including this condition, the axial component of the magnetic field $B_Z(r, \alpha, z)$ generated by the lower permanent magnet at point Q(r, α, z) (Fig. 1) in the air space can be expressed as Eq. (5) (It is worth noting that the expression is derived using the Cylindrical coordinate system $(r\ (m), \alpha\ (rad), z\ (m))$ whose origin is the same as the one of the Cartesian coordinate system O and the azimuth starts from the OX axis (Fig. 1)):

$$B_Z(r, \alpha, z) = B_Z^+(r, \alpha, z) + B_Z^-(r, \alpha, z), \quad (5)$$

where $B_Z^+(r, \alpha, z)$ (T) and $B_Z^-(r, \alpha, z)$ (T) are the magnetic field components from the upper and lower surfaces of the lower magnet (O, R, h), respectively; these values can be calculated using Eqs. (6) and (7).

$$B_Z^+(r, \alpha, z) = \frac{J}{4\pi} \int_{\theta=-\pi}^{\theta=\pi} \left(\frac{2(\delta R - 2\xi_+)}{(4\xi_+ - \delta^2)\sqrt{R(R-\delta) + \xi_+}} + \frac{4\sqrt{\xi_+}}{4\xi_+ - \delta^2} \right) (z - h) d\theta, \quad (6)$$

where $\delta = 2r\cos(\alpha - \theta)$ and $\xi_+ = r^2 + (z - h)^2$.

$$B_Z^-(r, \alpha, z) = \frac{-J}{4\pi} \int_{\theta=-\pi}^{\theta=\pi} \left(\frac{2(\delta R - 2\xi_-)}{(4\xi_- - \delta^2)\sqrt{R(R-\delta) + \xi_-}} + \frac{4\sqrt{\xi_-}}{4\xi_- - \delta^2} \right) (z) d\theta, \quad (7)$$

where, $\xi_- = r^2 + z^2$.

Inserting the geometrical parameters of the magnets into Eq. (4) and including Eqs. (2), (5), allows the semi-analytical model to be derived to compute the magnetic force F (N) as presented in Eq. (8).

$$F = \frac{J}{4\pi} \frac{J_1}{\mu_0} \int_{-\pi}^{\pi} \int_0^{R_1} (B^{*+} - B^{*-}) r \, d\alpha dr, \quad (8)$$

where $\frac{J}{4\pi}B^{*+}$ (T) and $\frac{J}{4\pi}B^{*-}$ (T) are the magnetic field at those points located on the lower and upper surfaces of the upper magnet (O_1, R_1, h_1), respectively.

III. Surrogate model formulation based on deep learning

Solving Eq. (8) to obtain the magnetic force involves a triple numerical integration which is time-consuming and difficult to optimise within a reasonable computational time. As mentioned earlier, the goal of this study is to develop a substitute model which can provide an accurate prediction of the magnetic force while requiring a low computational cost. A substitute model which can replace the original model to accomplish the goal is called a surrogate model [9, 20]. Mathematically speaking, if the original physics-based function is represented by Eq. (9), the surrogate model can be expressed as Eq. (10).

$$Y = F(X), \qquad (9)$$

where X and Y are the input and output of the function F, respectively.

$$\widehat{Y} = \widehat{F}(\widehat{X}), \qquad (10)$$

where $\widehat{X} \in X$, and $\widehat{Y}$ are the input and output of the surrogate model $\widehat{F}$.

From Eq. (8), it is noted that the interaction force F can be reformulated as a simple multiplication operation between two remanences J and J_1 representing the material properties of the magnets and the normalized magnetic force F_n (Eq. (11)). Due to this simple multiplication operation, J and J_1 are not chosen as the input features of the deep learning model; this reduces the volume of data required to train the model efficiently as well as avoiding the errors caused by these parameters, in other words, faster learning and more accurate predictions are the result of this feature reduction.

$$F = JJ_1F_n, \quad (11)$$

where the normalized magnetic force F_n is computed as follows Eq. (12) (please refer to Eq. (8) for more information on the parameters utilized in this equation):

$$F_n = \frac{1}{4\pi}\frac{1}{\mu_0}\int_{-\pi}^{\pi}\int_{0}^{R_1}(B^{*+} - B^{*-})r\,d\alpha dr, \quad (12)$$

It is clear from Eqs. (6), (7) and (12) that F_n is the function of geometrical parameters including radii and thicknesses of the lower magnet (O, R, h) and upper magnet (O_1, R_1, h_1) and the separation distance ξ. Therefore, these parameters can be implemented as the inputs of the deep learning model which seeks to replace Eq. (12) in order to compute the normalized magnetic force at a lower computational cost.

With a deep learning method, a surrogate model of the magnetic force F can be expressed mathematically as follows:

For the geometrical parameters $\Xi \in \mathbf{R}^{L \times 1}$ (L is the number of the input features; it is noted that L = 5 in this study) and the

function f: $\mathbf{R}^{(L+2)\,x\,1} \rightarrow \mathbf{R}$, the magnetic force can be formulated as Eq. (13).

$$F = f(J, J_1, \Xi) = JJ_1(W_n+b_n)\sigma\big(W_{n-1}\sigma\big(\ldots W_k\sigma\big(\ldots.\sigma(W_2\sigma(W_1\Xi+ b_1)+b_2)\big)+b_k\big)+b_{n-1}\big), \quad (13)$$

where $W_{i=1\ldots n} \in \mathbf{R}^{d_{i+1}\,x\,d_{i-1}}$, $b_{i=1\ldots n} \in \mathbf{R}^{d_{i+1}\,x\,1}$ are the matrices of the weights and biases (n = m+1; m - the number of the hidden layers of the deep learning model; d_{i+1} and d_{i-1} are the number of the neurons in the i+1 and i – 1 layers, respectively); σ denotes an applied activation function.

The computational graph of the surrogate magnetic force model is depicted in Fig. 2. The initial geometrical parameters of the two magnets (R, h, R_1, h_1 and ξ) are passed onto the five input neurons of the deep leaning model which include several hidden layers (the exact number of the hidden layers and their neurons are discussed in detail in Section IV) and one neuron of the output layer outputting the normalized magnetic force F_n. The remanences of the magnets (J and J_1) are passed onto two multiplication operators M_1 and M_2. The final output is the required magnetic force F as a result of the normalized force F_n multiplied by J and J_1.

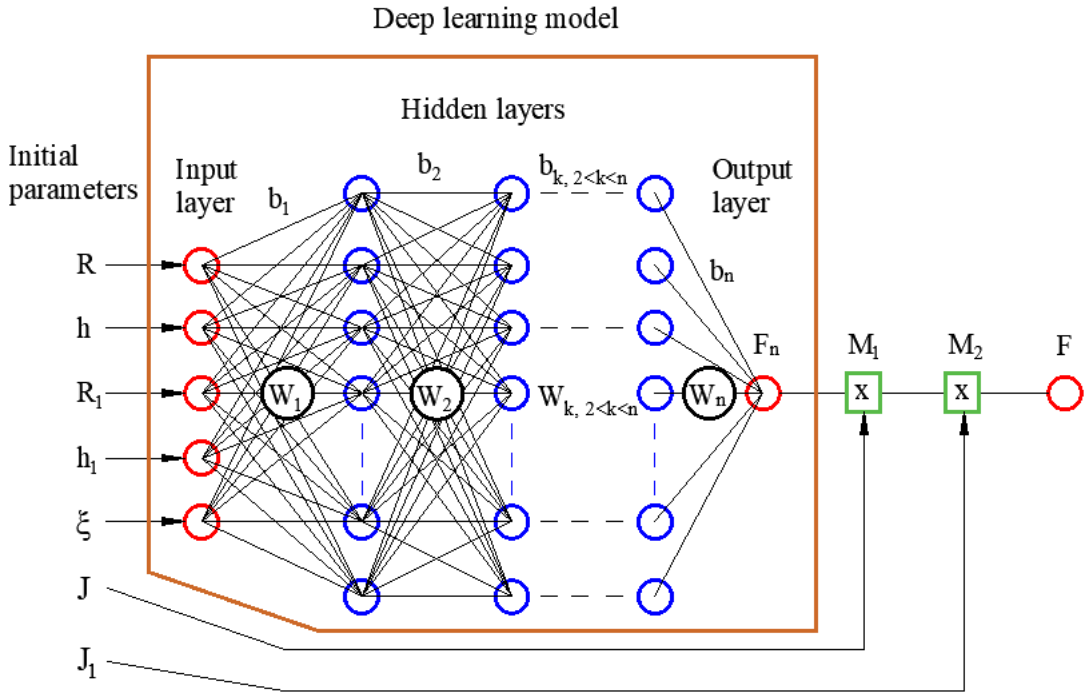

Fig. 2 – Architecture of the surrogate magnetic force model.

IV. Data generation and parameters of deep learning model

A dataset of 116 361 non-repeated random samples has been generated by solving Eq. (12) with the numerical integration of the triple integrals. The samples have been converted into float32 number types to reduce the required memory volume for the training process. The geometrical parameters of the simulated samples distributed within the range (Min, Max) as listed in Table I. This dataset is then divided into three subsets of training, validation and test data on the ratio of 92/4/4. In the other words, there are 107 052 samples of the training dataset, 4655 samples of the validation and 4654 samples of the test.

Before fitting the datasets into the developed machine learning model, they are normalized following the Min - Max normalization (Eq. (14)) approach. This normalization has some advantages such as improving the numerical stability of the model and speeding up the training process [21].

$$\hat{x}_i = \frac{x_i - \min(X)}{\max(X) - \min(X)}, \qquad (14)$$

where $x_i \in X$ is the i^{th} component of the dataset X; max(X) and min(X) are the maximum and minimum values of X, respectively; $\hat{x}_i$ is the normalized value of x_i.

Table I – Geometrical range of simulated samples

Parameters	Min (mm)	Max (mm)
R	3	30
h	5	50
R_1	3	30
h_1	5	50
ξ	2	50

The selection of the parameters used in the deep learning model is important in order to decrease the training time while increasing the prediction accuracy. Small numbers of neurons and hidden layers can result in low accuracy and high training time; on the other hand, excessive numbers of neurons and hidden layers can lead to overfitting and high computational complexity [22] such as memory resources. Unfortunately, there are no ideal methods to determine the hyperparameters since this depends on numerous reasons including the volume of used datasets etc. However, the trial and error method can be implemented to select these parameters. After training the deep learning model with different numbers of neurons and hidden layers (the number of hidden layer was randomly selected between 1 and 5 layers, and similarly, the number of neurons was randomly selected between 20 and 500 neurons for each layer), four hidden layers with 300 neurons for each layer were chosen as they provided a minimal training loss within a reasonably affordable training time (2 hours and 30 mins). Moreover, no overfitting was observed when using these hyperparameters. Therefore, the deep learning model with this configuration has been applied in this study. Adam algorithm (Appendix A) which combines the advantages of the well-known methods AdaGrad and RMSProp [23] was selected as the optimizer in this experiment.

The purpose of training the deep learning model is to determine the optimal parameters (the weights (W) and biases (b)) of this model to minimize the distances between the ground-truth (F) and predicted ($\hat{F}$) magnetic force values. The problem of finding these parameters can be represented using Eq. (15).

$$\min_{W,\ b} dist(F, f(J, J_1, X)) \stackrel{\text{def}}{=} \min_{W,\ b} \mathcal{L}(F, \hat{F}), \qquad (15)$$

where $\mathcal{L}(F, \hat{F})$ is a loss function (typically a convex function) of the two forces.

For a regression problem, root mean squared error (RMSE) (Eq. (16)) is widely chosen as the loss function to evaluate the prediction results. Thus, this function is implemented in this study.

$$\mathcal{L}(F, \hat{F}) = \text{RMSE}_F = \sqrt{\frac{1}{K}\sum_{i=1}^{K}(F_i - \hat{F}_i)^2}\ , \qquad (16)$$

where F_i and $\hat{F}_i$ are the ground-truth and predicted magnetic force of i^{th} sample, respectively; K is the number of magnetic force samples.

In order to solve the optimization problem Eq. (15), the backpropagation algorithm [15] is utilized in the training process.

Another important parameter of the deep learning model is the activation function. Rectified linear unit (ReLU) has proved to be an effective function in many studies [15, 24, 25]. Thus, ReLU has also been chosen as the activation function in this study.

$$y_i(x_i) = \begin{cases} 0, & \& \ x_i < 0 \\ x_i, & \& \ x_i \geq 0 \end{cases}, \qquad (17)$$

This function yields an output y_i equal to the input x_i if this input is greater or equal to zero; on the other hand, the output of this function is zero if its input is less than zero (Eq. (17)) (Fig. 3 visually describes the ReLU activation function).

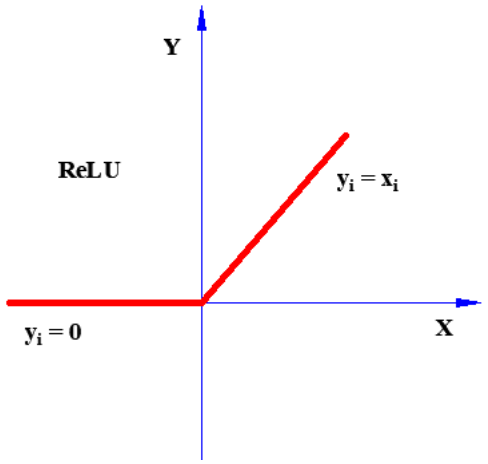

Fig. 3 – Rectified linear unit (ReLU) activation function: x_i and y_i are the input and output of the activation function y(x).

When the training dataset is large, minibatch could be implemented to speed up the convergence of the training process [15, 26]. In this study, a minibatch of 1024 samples was selected.

V. Training the deep learning model and Finite Element Analysis validation

The training process was carried out on a personal computer with the Intel® Core™ i7-9700 CPU @ 3.00GHz 3.00 GHz and 16.0 GB RAM processor. The code of the machine learning model was written based on the TensorFlow developed by Google Brain using the Python language (version 3.7.4). It was observed that the losses of the validation and test follow the trend of the training process, in the other words, there is no overfitting found during the

training. Therefore, the criterion to stop the training process is that the RMSE of the training process is less than 0.065. This criterion was accomplished after 2 hours and 30 minutes of training. It is worth noting that the purpose of the training process is to update the weights and biases of the deep learning model to reach the expected accuracy (RMSE = 0.065). After training, the model with the updated weights and biases is ready to use without spending time on retraining.

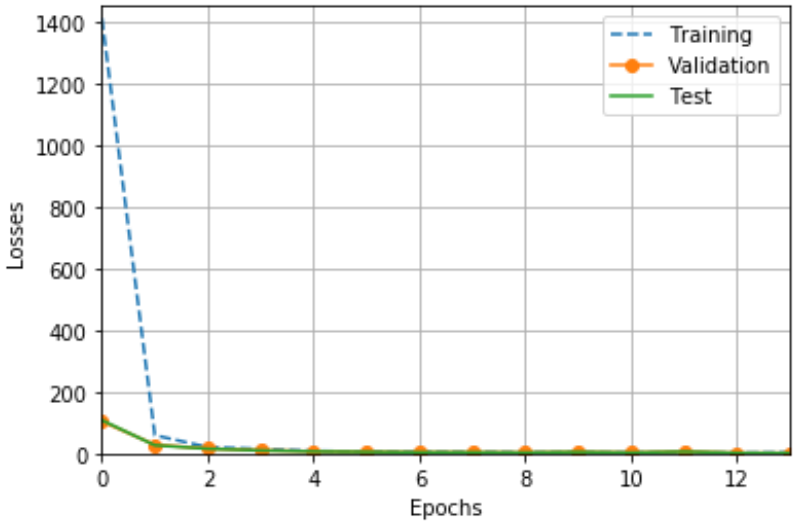

Fig. 4 – Losses of the training, validation and test processes.

To develop more insight into the accuracy of the trained deep learning model, 885 non-repeated random samples were applied to compare the results of this model and those of the semi-analytical model. The excellent agreement between the computed results can be visually demonstrated (colour -based) using Fig. 5 which represents their heatmap plots. It is worth noting that the correlation factor between the predicted and semi-analytical results is R_{corr} = 99.999%.

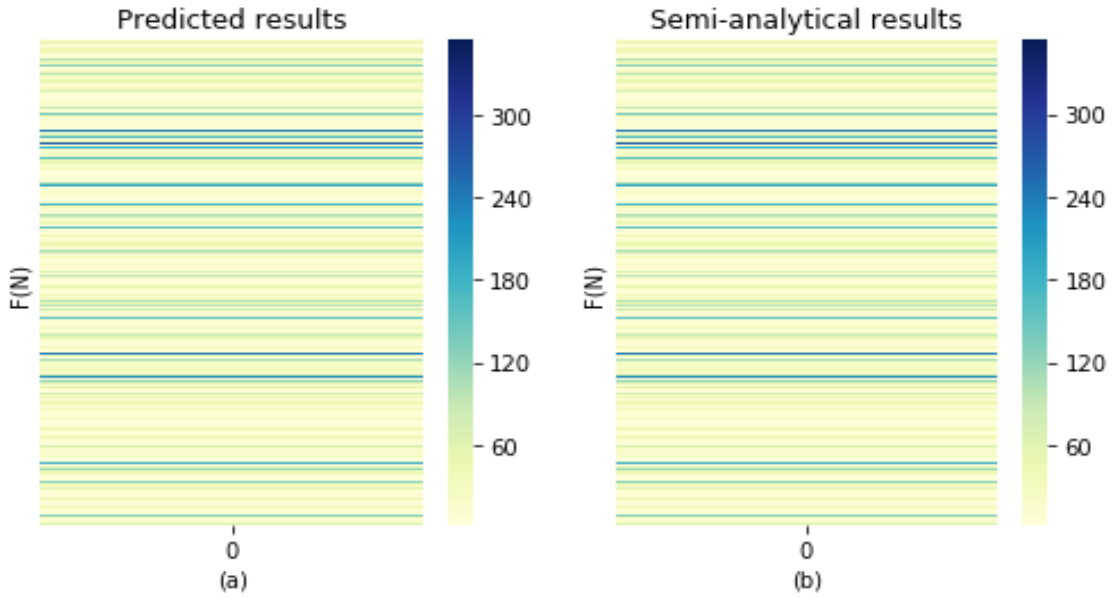

Fig. 5 – Heatmap plots of the predicted and semi-analytical results: (a) the predicted results of the surrogate model; (b) the results of the semi-analytical model.

Moreover, Fig. 6 shows the distribution of the error between the two models. The difference between the two models is less than 4.2 %, and there are 99.2 % and 96.05 % of the cases where the errors are less than 2 % and 1 %, respectively. This means that results from the deep learning model are in excellent agreement with the original semi-analytical model.

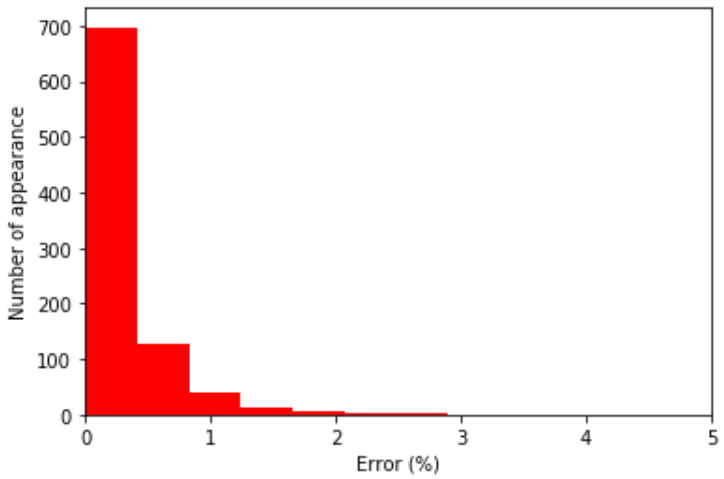

Fig. 6 – Distribution of the errors between the deep learning and semi-analytical models.

In order to validate the accuracy of the surrogate model and its efficiency in terms of the computational time, the magnetic force results computed using this model were compared with those calculated using the developed semi-analytical model and Finite Element Analysis (FEA). The FEA was performed using the Electromagnetic Simulation Software®(EMS) (EMWORKS, Inc, Quebec, Canada) [27, 28, 29] integrated in SolidWorks® (Dassault Systèmes SE, Vélizy-Villacoublay, France). Two subsets of data were selected for comparison. Subset 1 consists of samples of two permanent magnets with random material and geometrical parameters (J (T), R (mm), h (mm), J_1 (T), R_1 (mm), h_1 (mm), ξ (mm)) (Fig. 1). Subset 2 includes two permanent magnets with different separation distances. The setting magnetic materials' parameters [1, 8, 28] in this FEA demonstration are listed in Table II which includes the remanence and its corresponding coercivity; the relative permeability (μ_r) is set to 1.05 for all the cases (it is worth noting that the FEA results are invariant to the relative permeability as the remanence and coercivity are used in this study [7]). There are two air regions in cylindrical shapes are applied in the simulation. The two permanent magnets lie inside the smaller one; this smaller air region lie inside the bigger one. Finer mesh is assigned to the magnets and the small air region; coarser mesh is assigned to the remained air region. The sizes of the smaller and bigger air regions as well as the mesh sizes are tuned until the convergence of the magnetic forces achieved. The virtual work [29] is used as the computation method in this study. The details of the sample points and the results of the magnetic forces computed using the semi-analytical, surrogate and FEA models are listed in Table III.

Table II – Material properties using in setting up the FEA [1, 8, 28]

Remanence (J or J_1) (T)	Coercivity (H_c) (A/m)	Relative permeability	Computation method

0.5	393000	1.05	Virtual work
0.7	538000		
0.8	637000		
0.9	710000		
1	790000		
1.1	871000		
1.2	925000		

Table III – Comparison between the results of the semi-analytical, surrogate and FEA models

Material and Geometrical parameters (J (T), R (mm), h (mm), J_1 (T), R_1 (mm), h_1 (mm), ξ (mm))	Semi-analytical results (N)	Surrogate model results (N)	FEA results (N)
Subset 1 – With Random parameters			
(0.5, 20, 15, 0.5, 16, 18, 20)	7.0217	7.0289	7.0229
(0.9,17,11, 0.8,10, 8, 14)	7.5095	7.4741	7.5006
(1, 12, 12, 1, 24, 17, 37)	5.5064	5.5225	5.5309
(0.7, 24, 19, 1, 6, 36, 10)	10.4319	10.4672	10.4590
(1, 14, 42, 1.2, 23, 5, 15)	26.6690	26.6741	26.6530
(1.1, 11, 11, 1, 24, 36, 2)	59.7973	59.7948	59.8500
Subset 2 – With different separation distances			
(1, 8, 9, 1, 12, 19, 2)	36.3253	36.3335	36.4390
(1, 8, 9, 1, 12, 19, 4)	27.6379	27.8493	27.6200
(----------,6)	20.8733	20.8972	20.8680
(----------,8)	15.8355	15.7654	15.8850
(----------,12)	9.3697	9.3828	9.3773
(----------,16)	5.7812	5.7943	5.7753
(----------,20)	3.7149	3.7128	3.7046
(----------,24)	2.4769	2.4402	2.4674
(----------,28)	1.7065	1.6709	1.7005

From Table III and Fig. 7, it can be noted that the computed results of the three models are in excellent agreement with each other. Moreover, the minimum, average and maximum differences between the semi-analytical and FEA models are 0.017 %, 0.19 % and 0.44 %, respectively. Furthermore, the minimum, average and maximum differences between the surrogate and FEA models are 0.06 %, 0.42 % and 1.74 %, respectively. However, it took an average of 11.0 s to execute a single sample (1000 random samples were selected for this demonstration) using the semi-analytical model, and less than 10^{-4} s using the surrogate model which is suitable for the real-time computation and is best for the optimization process. On the other hand, the FEA model took an average of more than 10 minutes to complete (Table IV). It is worth noting that for this time comparison purpose, all three models were executed on a personal computer with Intel® Core™ i5-8250U CPU @1.60GHz with 8.0 GB RAM. This

means that the surrogate model is multiple orders of magnitude faster than its semi-analytical and FEA counterparts.

Table IV – Execution time comparison

Models	Semi-analytical (seconds)	Surrogate (seconds)	FEA (seconds)
Execution time	11.0	< 0.0001	>600

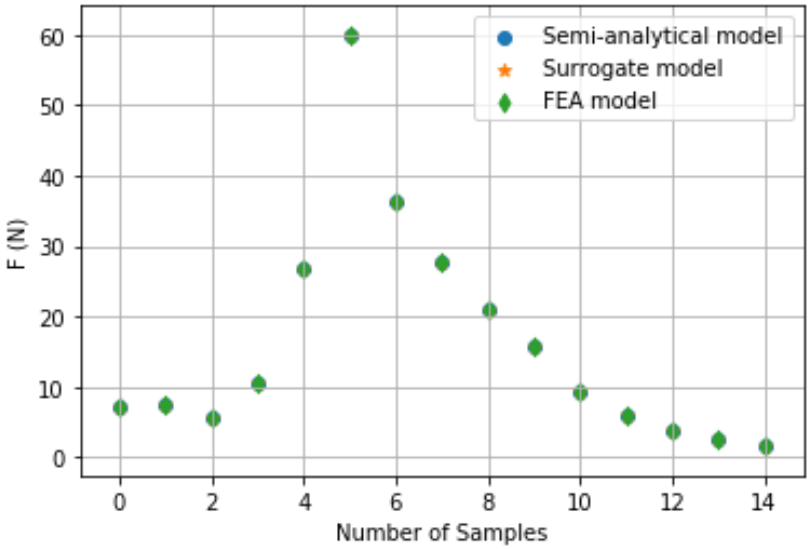

Fig. 7 – Comparison between the semi-analytical, surrogate and FEA models (It is noted that on this graph, the markers denoting the semi-analytical and surrogate models lie behind the green marker of the FEA model, that indicates the excellent agreement between these models).

VI. Feature importance analysis and a test on the generality of deep learning model

As described in section III, our deep learning model requires an input vector of five features which are the geometrical parameters of two magnets including the radius and height of the lower magnet R, h, respectively; the radius and height of the upper magnet R_1, h_1, respectively and the separation distance between these magnets ξ. However, the predictive power, or in the other words the importance of each feature may not be the same. This means that the variation of one feature may decrease or increase the predictive force more than the variation of the others. To quantitatively measure the predictive power of the input features, the so-called permutation feature importance [30, 31, 32] is implemented in this study. The algorithm steps of this method are as follows.

Inputs: The inputs for this algorithm includes the trained deep learning model, the validation dataset of 885 non-repeated samples including the input features (five geometrical parameters) and labels (associated magnetic forces) and metric functions $M(\mathrm{F}, \hat{\mathrm{F}})$ (F is the ground truth of the magnetic force and $\hat{\mathrm{F}}$ is the predictive one using the machine learning model). In this

investigation, the so-called R-squared (R^2) score function (the coefficient of determination) is utilized as the metric function. The R^2 function is determined as Eq. (18).

$$R^2(\mathrm{F}, \hat{\mathrm{F}}) = 1 - \frac{SS_r}{SS_t}, \qquad (18)$$

where $SS_r = \sum_{i=1}^{i=N}(F_i - \hat{F}_i)$ – the residual sum of squares due to error; $SS_t = \sum_{i=1}^{i=N}(F_i - \bar{F}_i)$ – the total sum of squares due to error; F, $\hat{\mathrm{F}}$ and $\bar{\mathrm{F}}$ denote the ground truth, predicted value and mean of the ground truth of the magnetic force, respectively; i denotes the computed i^{th} sample of the total N samples (N = 885).

Step 1. Compute the baseline R^2 (e_{br}): $e_{br} = R^2(\mathrm{F}, \hat{\mathrm{F}})$.

Step 2. For each feature q = 1…5, performing following three operations:

- Generate new dataset by permuting the feature q in the original dataset (the rest of the features remains unchanged);
- Estimate the new R^2 (e_{qr}) based on the new permuted dataset: $e_{qr} = R^2(\mathrm{F}, \hat{\mathrm{F}}_q)$; where $\hat{\mathrm{F}}_q$ – the predicted magnetic force as a result of the permutation of feature q;
- Compute the permutation feature importance (PFI): $PFI_R = e_{br} - e_{qr}$ for the R^2. It is worth mentioning that the PFI can also be evaluated using the ratios: $PFI_R = e_{qr}/e_{br}$. In this study, the first computation technique is utilized.

Step 3. Repeat Step 2 until a stop condition is met such as the convergence of the mean PFI or the number of iterations reached.

Step 4. Compute the mean and standard deviation values of PFIs.

Step 5. Sort the achieved mean PFIs by descending. The greater mean PFI of a feature means the more important it is.

The above-described algorithm can be implemented with the help of the open access machine learning Scikit-learn [33].

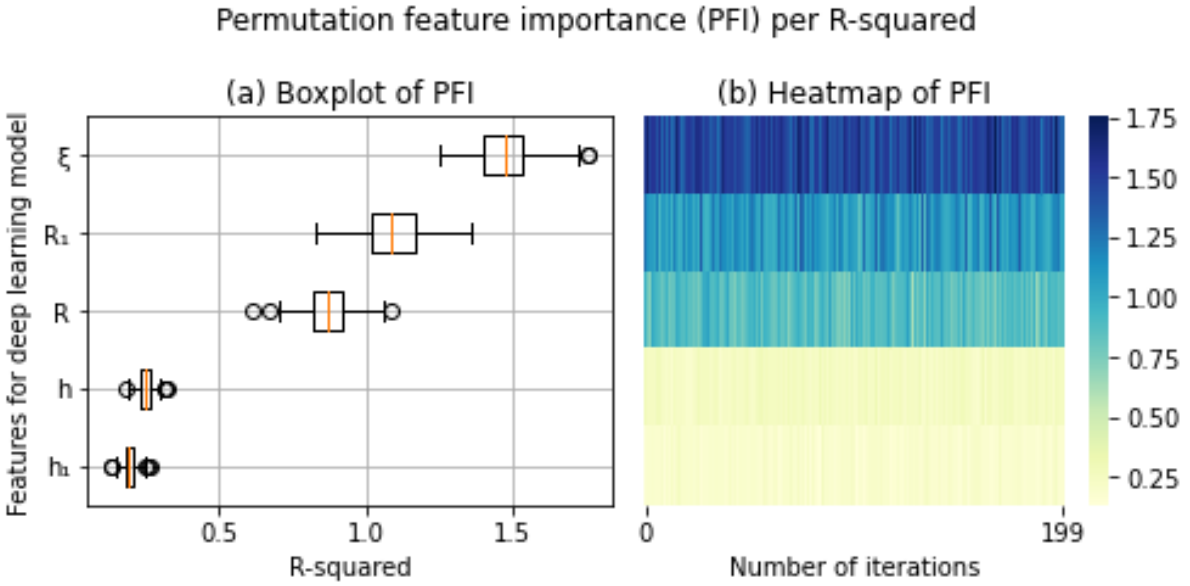

Fig. 8 – Feature importance per R^2

Figure 8 and Table V shows that the most predictive feature is the separation distance (ξ) (mean R^2 = 1.47), and the least ones are the heights of the permanent magnets (mean R^2 = 0.25 and 0.19) (it is noted that **Error! Reference source not found.** (a) presents the boxplot of the feature importance and **Error! Reference source not found.** (b) describes its heatmap; the number of iterations is 200 indexing with zero start). Moreover, all the features are non-trivial to the deep learning model as their feature importance is greater than zero. From Table V, the summation of the feature importance for the radius and height of the lower magnet is PFI_L = 1.28, and this summation for the radius and height of the upper magnet is PFI_U = 1.12 which is slightly smaller than PFI_L. This means that the order of the input features of these parameters can have some influence on the predictive results. However, in real-world, there should not be different results when the input order of these features between the lower and upper magnets is exchanged. It is worth mentioning that the permutation feature importance is also computed using the root mean squared error (Eq. 16) as a metric function, and similar results (the most and least predictive features and the differences between the summation of the importance of the radius and height of two magnets) to the R^2 criterion are observed; this demonstrates the adequacy of using one of the two metric functions in analysing the PFI. To verify the effectiveness of the trained deep learning model towards its generality, in the dataset of the non-repeated 885 samples with the input features' order of (R_1, h_1, R, h, ξ) for the deep learning model, the positions of the upper magnet's radius and height are swapped with those of the lower magnet to form a new dataset with the input features' order of (R, h, R_1, h_1, ξ). This dataset is then fitted to the deep learning model to predict the results, which are compared with the ground truth using the R-squared (R^2) criterion. As a result, the achieved R^2 is 0.99999523 which is very close to 1. This demonstrates that the trained model is accurate to predict the magnetic force regardless of the input order of the lower and upper magnets' geometrical parameters; in the other words, the deep learning model has good generality.

Table V – Permutation feature importance

Geometrical features (by descending order of importance)	Mean ± standard deviation
Separation distance (ξ)	1.47 ± 0.10
Radius of the upper magnet (R_1)	1.09 ± 0.11
Radius of the lower magnet (R)	0.87 ± 0.08
Height of the lower magnet (h)	0.25 ± 0.03
Height of the upper magnet (h_1)	0.19 ± 0.02

VII. Discussion on using the superposition principle for permanent magnet addition and subtraction

The fast-computed surrogate model recently derived for the computation of the magnetic forces between two solid cylinders can be applied to calculate the magnetic forces between cross-shaped (solid, annular) cylinders and between multiple cylinders using the superposition principle [1, 28]. The rule of thumb (subtraction rule) to construct an annular permanent magnet (upper magnet in Fig. a) with an outer radius R_1, inner radius R_2, a height h_1 and a magnetization $\overrightarrow{J_1}$ is that it is composed of two solid magnets; the first magnet has radius R_1, height h_1 and a magnetization $\overrightarrow{J_1}$, on the other hand, the second magnet (shaded part with yellow in Fig. 9) has radius R_2, height h_1 and a magnetization $\overrightarrow{J_2}$ which has the same magnitude as of

$\overrightarrow{J_1}$, but in opposite direction. In this case, the magnetic force between two magnets $\vec{F}[(O,R),(O_1,R_1,R_2)]$ can be calculated using the surrogate model (Eq. 13) as a summation of the magnetic forces between three cylinders (O,R), (O_1,R_1) and (O_2,R_2) as shown in Eq. (19). In the case of more than two magnets (for example, there are three magnets as depicted in Fig. 10) the resultant magnetic force is the summation of the magnetic forces between a base magnet (e.g. the lower magnet in Fig. 10) and the rest magnets (e.g. the two upper magnets in Fig. (10); in the case of three magnets, this summation can be represented as shown in Eq. (19)).

$$\vec{F}[(O,R),(O_1,R_1,R_2)] = \vec{F}[(O,R),(O_1,R_1)] + \vec{F}[(O,R),(O_2,R_2)], \quad (19)$$

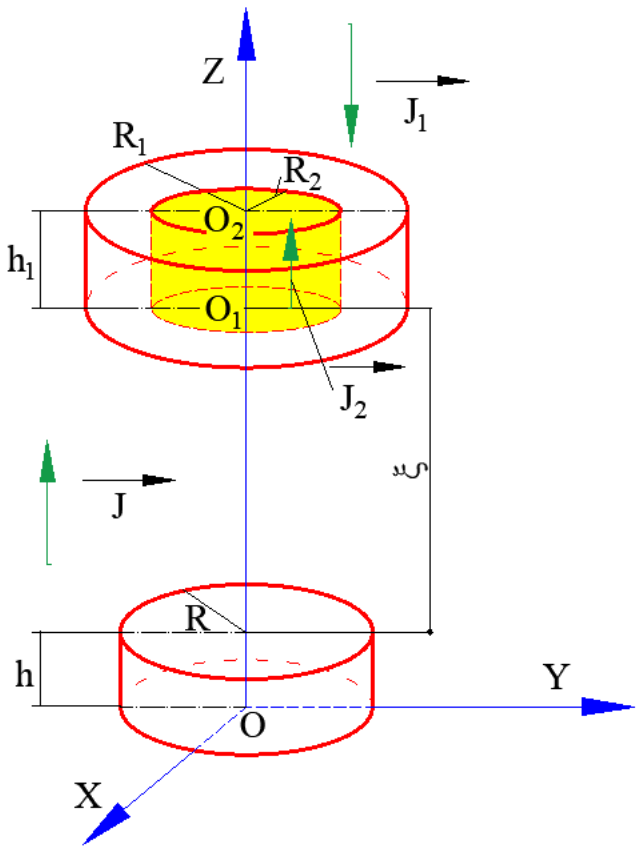

Fig. 9 - Computation of magnetic force between solid and annular cylinders

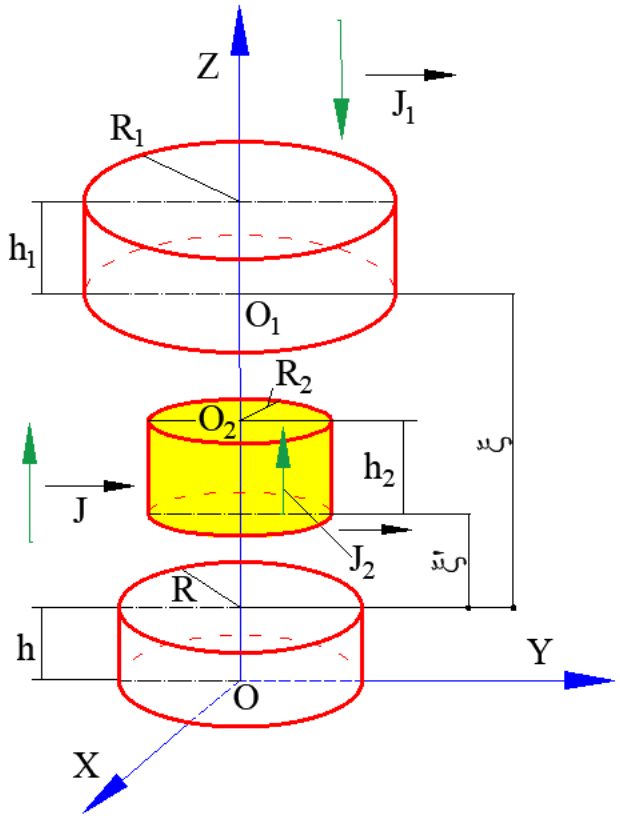

Fig. 10 – Computation of magnetic forces between multiple magnets

The surrogate model can be applied to calculate the magnetic forces between magnets of cross-shaped revolution (such as cylinders, cones, spheres etc.) using the segmentation and superposition principles. Firstly, a non-cylinder magnet is divided into N sections (here N is a hyperparameter which needs to be tuned until convergence reached) along the axis of revolution (e.g. a cone in Fig. 11); each section has the same height and magnetization. Assuming that each section has a cylindrical shape, the resultant magnetic forces between the cross-shaped magnets are the summation of the magnetic forces between their segmented sections based on the superposition principle. For instance, Fig. 11 depicts schematic of a magnetized cylinder (with radius R, height h and magnetization J) and a magnetized right circular cone with a circular cross-sectional area (with radii R_1, R_n, height h' and magnetization J_1). Dividing the cone into N sections S_1, S_2 … S_N; assuming that each section is a cylinder with radius R_i (i = 1…N), height h'', magnetization J_1 and separation distance ξ_i to the lower cylinder (Fig. 11); these geometrical parameters can be derived as follows (Eqs. (20, 21 and 22)).

$h'' = h'/N$, (20); $\xi_i = \xi_1 + (i-1)h'/N$, (21); $R_i = (i-1)(R_n - R_1)/N + R_1$, (22)

The resultant force between the cylinder and cone is represented in Eq. (23).

$$\vec{F}[(O,R),(O_1,R_1,O_n,R_n)] = \sum_{i=1}^{N} \vec{F}[(O,R),S_i(R_i,h'',\xi_i)], \quad (23)$$

where N is a hyperparameter (the number of divided sections) which needs to be tuned until convergence reached.

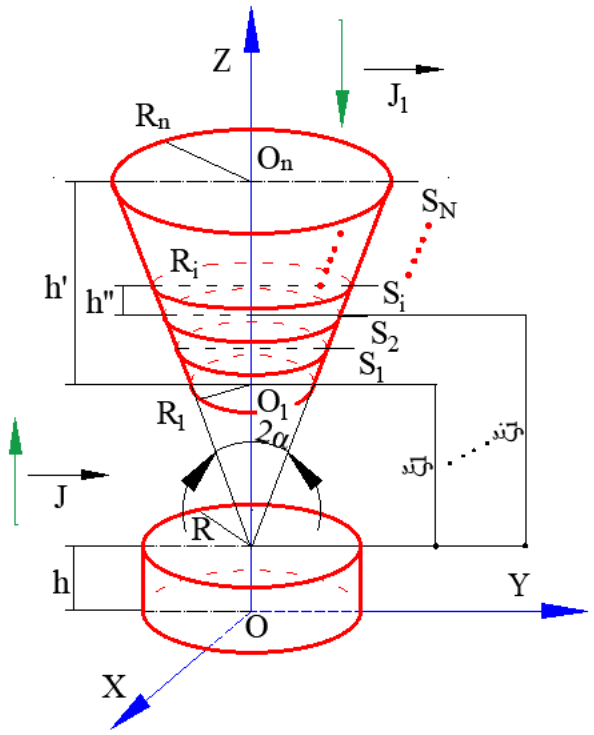

Fig. 11 – Computation of magnetic forces between cross-shaped magnets

VIII. A user-friendly software interface

The developed surrogate model with the deep learning model has been demonstrated to be efficient to compute the magnetic force between two permanent magnets. However, the application of this ready-to-use model can be challenging for readers who have limited access or knowledge on Python and its computation and machine learning libraries such as Numpy, Scikit-learn, TensorFlow and Keras. Therefore, the authors have developed a user-friendly software interface (USI). The architecture of this USI includes the user input interface, a server to process computation and the output result interface (Fig. 12).

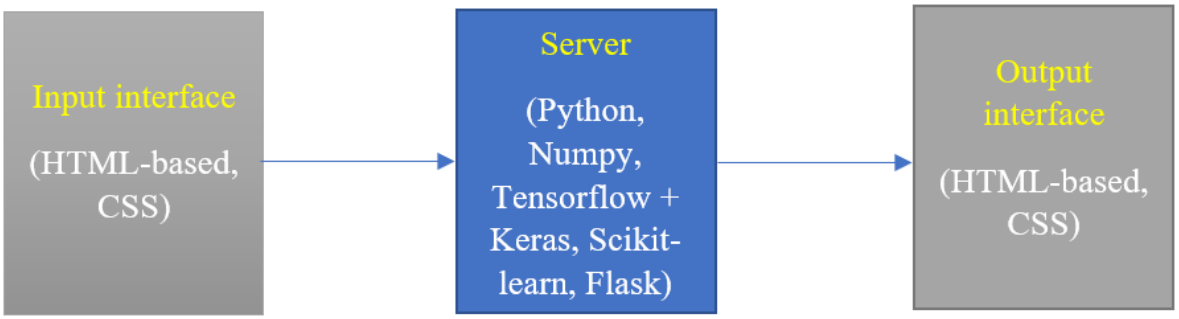

Fig. 12 – Information flow of the USI

The input and output interfaces (Figs. 13, 14) are built as HTML based web browsers (cascading style sheets or CSS is used to improve presentation style) where one can input the material and geometrical parameters of the permanent magnets, and get the results (magnetic force) back. They include a header, parameter input fields, a button to submit the inputs, a figure field, a predicted result field and the license declaration. It is noted that this is an open-source software under the CC BY 4.0 license, so anyone can adapt, share for non-commercial purposes.

After submitting the inputs (from the input interface), these parameters will go to the server written in Python, Numpy, Tensorflow + Keras, Scikit-learn and Flask [34] languages. This server will use the input parameters to fit into the surrogate model and return the predicted magnetic force which will appear in the predicted result field on the output interface. The output figure will depict the vectors showing the orientations of the remanences and magnetic force (Fig. 14)

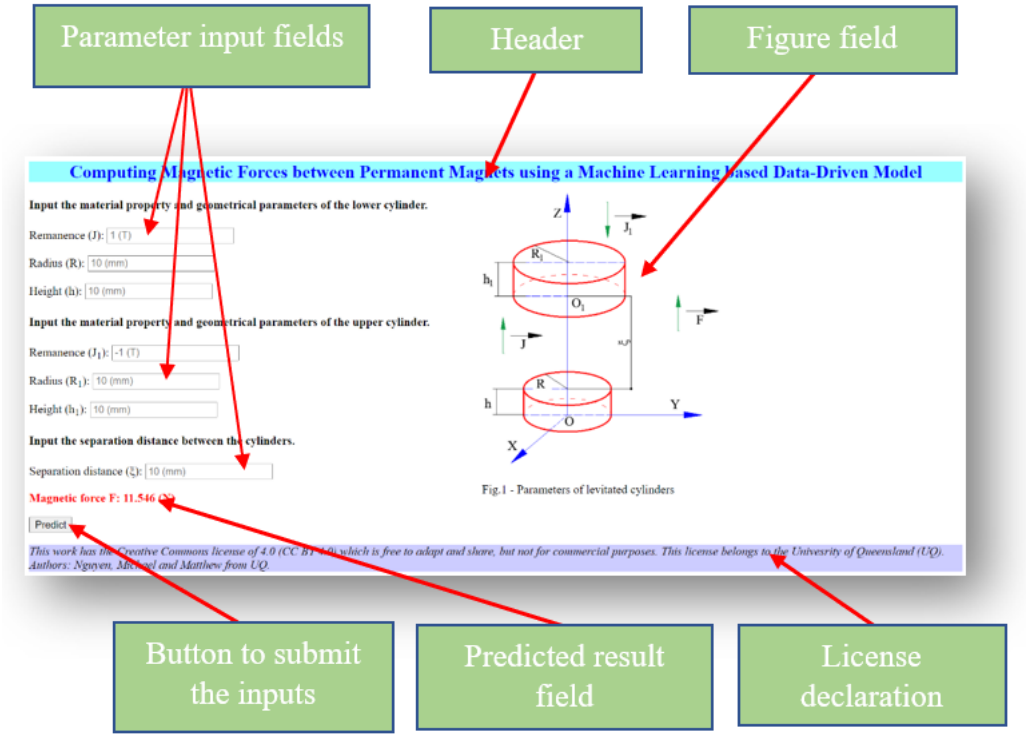

Fig. 13 – Input interface

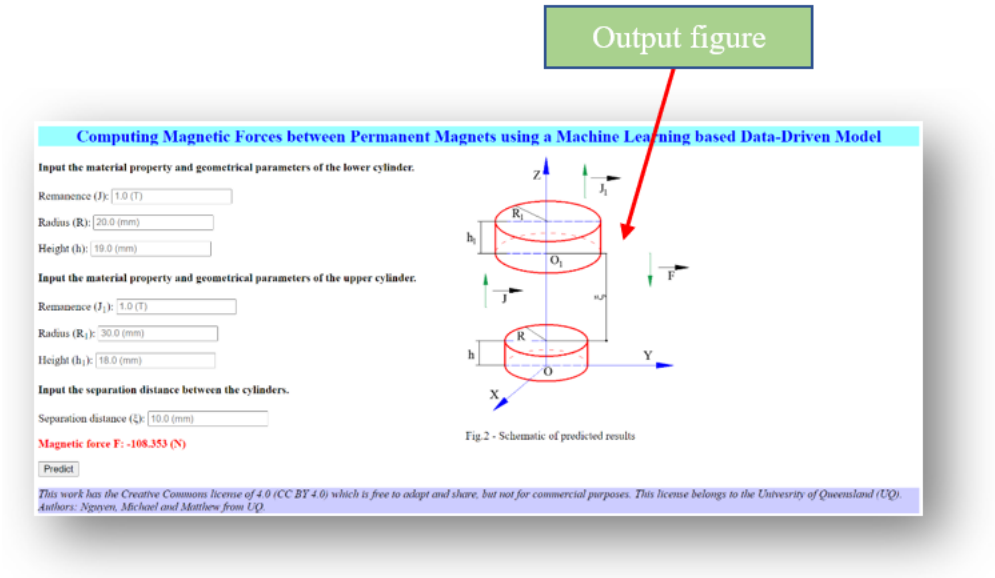

Fig. 14 – Output interface

IX. Conclusion

In this study, a data-driven based model of the magnetic interaction force between permanent magnets which can be rapidly computed is presented. Firstly, the charge model has been applied to develop the semi-analytical model (SAM) to calculate the magnetic force. Using this SAM, the optimized input features of a deep learning model (DLM) have been selected. Furthermore, the training, validation and test datasets are generated based on this SAM. These datasets have been used to train the DLM which is a key element of the data-driven surrogate model (DDSM). As demonstrated in this article, the results of the DDSM are in excellent agreement with those of the semi-analytical and Finite Element Analysis counterparts. However, the computational cost of the DDSM is multiple orders of magnitude lower than the others. This demonstrates the feasibility and effectiveness of applying the deep learning method to describe the magnetic force interaction between permanent magnets, as an alternative method which can potentially replace the existing conventional Finite Element Method and SAM. The permutation feature importance analysis shows that the highest influential feature on the machine learning model is the separation distance between the magnets, and the least are their heights. The deep learning model is generalized regardless of the order of the magnets' geometrical features input ((R_1, h_1, R, h, ξ) or (R, h, R_1, h_1, ξ)). The fast-computed model presented in this article can facilitate the design and optimization processes of permanent magnet systems such as parametric design and optimization of magnetic springs, magnetic levitation systems and soft robots with embedded permanent magnets. In this research, the interaction force between two cylindrical permanent magnets with coaxial orientation has been the focus; nevertheless, the described data-driven approach can be a general guide to develop the magnetic force interaction between permanent magnets of any shape with arbitrary orientation. Using the surrogate model, the magnetic forces between cross-shaped permanent magnets can be computed due to the superposition principle. It is noted that a user-friendly HTML-based software has been developed to compute the magnetic force based on the

surrogate model and publicly available for non-commercial purposes (the link is attached in this article's abstract).

Acknowledgement

The authors would like to thank the EMWORKS Company for issuing the license for the EMS® software to conduct the Finite Element Analyses. Van Tai is grateful to the University of Queensland for providing him with the Research Training Scholarship.

References

[1] E. P. Furlani, "Permanent Magnet and Electromechanical Devices: Materials, Analysis and Applications", Academic Press, 2001.

[2] V. T. Nguyen, T.-F. Lu, W. Robertson and P. Grimshaw, "Magnetic field distribution of an elliptical permanent magnet", Progress in electromagnetics research C, vol. 97, pp. 69-82, 2019.

[3] F. Poltschak and P. Ebetshuber, "Design of integrated magnetic springs for linear oscillatory actuators", IEEE Transactions on industry applications, vol. 54, 2018.

[4] A. Nammari, L. Caskey, J. Negrete and H. Bardaweel, "Fabrication and characterization of non-resonant magneto-mechanical low-frequency vibration energy harvester", Mechanical systems and signal processing, vol. 102, pp. 298-311, 2018.

[5] Z. Zergoune, N. Kacem and N. Bouhaddi, "On the energy localization in weakly coupled oscillators for electromagnetic vibration energy harvesting", Smart materials and structure, vol. 28, pp. 07LT02 (9), 2019.

[6] J. Li, E. S. Barjuei, G. Ciuti, Y. Hao, P. Zhang, A. Menciassi, Q. Huang and P. Dario, "Magnetically-driven medical robots: an analytical magnetic model for endoscopic capsules design", Journal of magnetism and magnetic materials, vol. 452, pp. 278–287, 2018.

[7] S. W. Kwok and S. A. Morin et al., "Magnetic assembly of soft robots with hard components", Advanced functional materials, vol. 24, pp. 2180 – 2187, 2014.

[8] J. M. D. Coey, "Magnetism and magnetic materials," Cambridge University Press, 2009.

[9] M. Roy and O. Wodo, "Data-driven modelling of thermal history in additive manufacturing", Additive manufacturing, vol. 32, 2020.

[10] D. Vokoun and M. Beleggia, "Forces between arrays of permanent magnets of basic geometric shapes", Journal of magnetism and magnetic materials, vol. 350, pp. 174-178, 2014.

[11] W. K. Schomburg, O. Reinertz, J. Sackmann and K. Schmitz, "Equations for the approximate calculation of forces between cuboid magnets", Journal of magnetism and magnetic materials, vol. 506, pp. 166694, 2020.

[12] V. T. Nguyen, “Magnetic field distribution of a conical permanent magnet with an application in magnetic resonance imaging,” Journal of magnetism and magnetic materials, vol. 498, pp. 166136, 2019.

[13] A. Khan, V. Ghorbanian and D. Lowther, “Deep learning for magnetic field estimation”, IEEE Transactions on magnetics, vol. 55, no. 6, 2019.

[14] T. Shan, W. Tang, X. Dang, M. Li, F Yang, S. Xu and J. Wu “Study on a Poisson’s equation solver based on deep learning technique,” Proceedings of IEEE electrical design of advanced packaging and systems (EDAPS), Haining, China, pp. 1–3, 2017.

[15] Goodfellow, I., Bengio, Y., and Courville, “A. Deep learning”, MIT press, 2016.

[16] X. J. Weng, L. C. Shen, H. Tang, G. P. Zhao, J. Xia, F. J. Morvan and J. Zou, “Change of coercivity mechanism with the soft film thickness in hard-soft tri-layers”, Journal of magnetism and magnetic materials, vol. 475, pp. 352-358, 2019.

[17] G. P. Zhao, L. Zhao, L. C. Shen, J. Zou and L. Qiu, “Coercivity mechanisms in nanostructured permanent mangets”, Chinese physics B, vol. 28, no. 7, 077505, 2019.

[18] X. H., Yuan, G. P. Zhao, M. Yue, L. N. Ye, J. Xia, X. C. Zhang and J. Chang, “3D and 1D calculation of hysteresis loops and energy products for anisotropic nanocomposite films with perpendicular anisotropy”, Journal of magnetism and magnetic materials, vol. 343, pp. 245–250, 2013.

[19] W. Zhang, G. P. Zhao, X. H. Yuan, & L. N. Ye, “3D and 1D micromagnetic calculation for hard/soft bilayers with in-plane easy axes”, Journal of magnetism and magnetic materials, vol. 324, pp. 4231–4236, 2012.

[20] A. Forrester and A. Keane et al. “Engineering design via surrogate modelling: a practical guide”, John Wiley & Sons, 2008.

[21] M. Shanker, M. Y. Hu and M. S. Hung, “Effect of Data Standardization on Neural Network Training”, Omega, vol. 24, pp. 385 – 397, 1996.

[22] J. Xiong, G. Zhang, J. Hu and L. Wu, “Bead geometry prediction for robotic GMAW-based rapid manufacturing through a neural network and a second-order regression analysis”, Journal of intelligent manufacturing, vol. 25, pp. 157-163, 2014.

[23] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization”, arXiv preprint, arXiv:1412.6980, 2014.

[24] N. Vinod and E. H. Geoffrey, “Rectified linear units improve restricted Boltzmann machines” International conference on machine learning, pp. 807–814, 2010.

[25] B. Xu, N. Wang, T. Chen and M. Li, “Empirical evaluation of rectified activations in convolutional network”, arXiv preprint, arXiv:1505.00853, 2015.

[26] X. Qian and D. Klabjan, “The impact of the mini-batch size on the variance of gradients in stochastic gradient descent”, arXiv preprint, arXiv:2004.13146, 2020.

[27] V. T. Nguyen and T. -F. Lu, "Analytical expression of the magnetic field created by a permanent magnet with diametrical magnetization," Progress in electromagnetics research C, vol. 87, pp. 163–174, 2018.

[28] V. T. Nguyen and T. -F. Lu, "Modelling of magnetic field distributions of elliptical cylinder permanent magnets with diametrical magnetization," Journal of magnetism and magnetic materials, vol. 491, pp. 165569, 2019.

[29] EMS 2020 User Guide (https://www.emworks.com/portal/download) (latest access 15/11/2020).

[30] L. Breiman, "Random forests", Machine learning, vol. 45, pp. 5-32, 2001.

[31] A. Fisher, C. Rudin, F. Dominici, "All models are wrong, but many are useful: learning a variable's importance by studying an entire class of prediction models simultaneously", Journal of machine learning research, vol. 20, pp. 1-81, 2019.

[32] https://christophm.github.io/interpretable-ml-book/feature-importance.html#fnref35 (latest access 06/11/2020).

[33]https://scikitlearn.org/stable/modules/generated/sklearn.inspection.permutation_importance.html#sklearn.inspection.permutation_importance (latest access 06/11/2020).

[34] https://pypi.org/project/Flask-Language (latest access 18/11/2020).

[35] https://www.tensorflow.org/api_docs/python/tf/keras/optimizers/Adam

Appendix A

Adam (an abbreviation for Adaptive moment estimation) is an efficient momentum-based algorithm in terms of computational complexity (little memory resources required, low execution time, invariant to gradients and well-suited to problems with large datasets/parameters [23]). It was developed to combine the advantages of both well-known algorithm AdaGrad (able to deal with sparse gradients) and RMSprop (able to deal with non-stationary objectives). Adam updates the exponential moving average of the gradients (m_t) and squared gradients (v_t) of loss function ($\mathcal{L}$) w.r.t its weights and biases (denoted as parameters W). The Adam algorithm is as follows:

Step 1: Initialization:

$m_0 \leftarrow 0$ (1st moment vector); $v_0 \leftarrow 0$ (2nd moment vector); $t \leftarrow 0$ (timestep)

Step 2: Recursively perform the following series of computation until the convergence of the parameters W:

Update timestep

$t \leftarrow t + 1;$

Compute gradients of loss function w.r.t parameters W

$G_t \leftarrow grad_W \mathcal{L}^{(t-1)};$

Update the first and second biased moments

$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot G_t;$

$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot G_t^2;$

Compute the adaptive learning rate

$$\alpha_t \leftarrow \alpha \cdot \sqrt{1 - \beta_2^t} \Big/ (1 - \beta_1^t);$$

Update the parameters W

$$W_t \leftarrow W_{t-1} - (\alpha_t \cdot m_t) \Big/ \left(\sqrt{v_t} + \hat{\epsilon}\right).$$

where (by default setting in this study) the learning rate $\alpha = 0.001$; $\beta_1 = 0.9$ and $\beta_2 = 0.999$ are the exponential decay rates for the first and second moments, respectively; $\hat{\epsilon} = 1e\text{-}07$ is a constant to prevent the division by zero [35]. Operations on vectors are carried out elementwise.