OPTIMIZATIONS IN APPLIED MACHINE LEARNING
Research Article* | *Volume 4* | *Issue 4* | *Dec 2024
Received: 1 Nov 2024 | *Revised: 26 Nov 2024*
Accepted: 3 Dec 2024 | *Published Online: 5 December 2024*

An Adaptive Weight-Based Performance Optimization Algorithm for Motor Design Using GNN Representation

Shuguang Xiong, Microsoft Inc., Email: shuxiong@microsoft.com

Huitao Zhang*, Northen Arizona University, Email: hz345@nau.edu

Meng Wang, Newmark Group, Email: wang070210@gmail.com

Ning Zhou, Zhejiang Future Technology LLC, Email: zhouning723@gmail.com

*Corresponding Author

Abstract: Motor design involves a variety of complex parameters, and traditional approaches often rely on experience and experimentation, which can be inefficient and challenging to optimize. With the rapid growth of industries such as electric vehicles and intelligent manufacturing, the demand for improved motor performance continues to rise. Optimizing motor designs under multi-objective and multi-constraint conditions has become a critical challenge. To address this, this paper introduces a motor design performance optimization algorithm that leverages Graph Neural Networks (GNN) and adaptive weighting techniques. GNN, a deep learning model adept at handling complex structured data, is capable of modeling the relationships between multiple parameters in motor design. Its feature propagation mechanism allows for automatic extraction of essential features, effectively addressing the limitations of traditional methods in capturing parameter dependencies. Additionally, Mixed-Integer Linear Programming (MILP) serves as a robust global optimization tool, capable of finding the optimal solution even in the presence of complex decision variables and constraints, overcoming the global convergence issues associated with conventional optimization algorithms. The adaptive weighting mechanism further enhances the algorithm by dynamically adjusting the weights based on the parameters' influence on motor performance, ensuring more accurate and adaptable optimization results across different scenarios. By combining these three techniques, this paper aims to resolve issues related to inefficiency, poor global convergence, and the static nature of parameter weighting in traditional motor design optimization. This approach integrates advanced machine learning models and optimization algorithms to create an efficient framework for motor design performance optimization.

Keywords: *Motor design; GNN; Adaptive weighting; MILP; Performance optimization; Parameter dependency.*

1. Introduction

Motors, as a core component of modern industry and technology development, are widely used in electric vehicles, intelligent manufacturing, aerospace, and other fields, driving the efficient operation and functional realization of modern equipment. Especially with the rapid development of electric vehicles and renewable energy technologies, the requirements for motors have been continuously increasing, demanding not only higher efficiency and lower energy consumption but also the ability to maintain stable performance under multi-objective and multi-constraint conditions[1]. The key performance indicators in motor design include power output, efficiency, thermal management, power-to-weight ratio, and cost, which often have complex trade-offs[2]. Therefore, optimizing motor design while meeting multiple performance requirements has become an important research topic in modern motor design. Most traditional motor design methods rely on designers' experience and repeated experiments, using rule-based simulation or finite element analysis tools for performance prediction and adjustment[3]. While these methods can meet the demands of motor design to some extent, with the increasing complexity of motor designs, traditional methods are gradually exposing two major issues: one is the inefficiency of the design process, relying on repeated experiments and iterations, which is not only time-consuming but also prone to local optima[4]; the second is that traditional methods struggle to cope with the complex dependencies in multi-parameter, multi-objective designs [5]. Especially in aspects such as motor geometry design, material selection, and production processes, multiple design parameters often interact to form complex dependency structures, which traditional methods struggle to capture comprehensively.

To address these challenges, the introduction of machine learning and optimization techniques in recent years has brought new opportunities to the field of motor design. In particular, Graph Neural Networks (GNN)[6], as a deep learning model capable of effectively handling graph-structured data, have shown significant advantages in capturing complex structural relationships and high-dimensional data. The various parameters in motor design and their dependencies can be naturally modeled as a graph structure, with design parameters as nodes and edges representing dependencies between parameters. Through GNN's information propagation mechanism, the interaction features between design parameters can be effectively extracted, providing critical input for optimizing motor performance. This GNN-based design parameter modeling method compensates for the shortcomings of traditional design methods in handling complex multi-parameter dependencies. However, relying solely on GNN for design parameter modeling and feature extraction is still insufficient to solve the global optimization problem in motor design[7]. Motor design often involves a complex combination of discrete and continuous variables, such as material selection and motor structure design dimensions. These variables need to be reconciled within a multi-objective optimization framework. Mixed-Integer Linear Programming (MILP), as a powerful optimization tool capable of handling discrete and continuous variables, offers the ability to find global optima while satisfying multiple constraints, overcoming the problem of traditional optimization algorithms falling into local optima in complex design tasks[8]. By using GNN-extracted design parameter features as input, MILP can further optimize the overall motor design, ensuring that the design not only meets multi-objective performance requirements but also finds the global optimal solution under constraint conditions. Additionally, the importance of different parameters to motor performance varies with design environments and application needs, making the reasonable setting of design weights crucial. Traditional optimization algorithms usually use fixed parameter weights, which are difficult to dynamically adjust the importance of parameters in different design scenarios, resulting in the neglect of certain key parameters in specific contexts. To solve this problem, this paper introduces an adaptive weighting mechanism, which dynamically adjusts the weights of various parameters during the design process, allowing the optimization process to evaluate and adjust the contributions of each parameter in real-time.

This method not only effectively improves the efficiency of motor design but also maintains a high level of optimization performance in different application scenarios, solving the problems of inefficiency, difficulty in global optimization, and lack of dynamic adjustment capability in traditional design methods.

The structure of this paper is as follows: The introduction section provides the background, existing challenges, and research motivation for motor design optimization. Next, the related work section reviews traditional motor design methods, Graph Neural Networks (GNN), Mixed-Integer Linear Programming (MILP), and adaptive weighting applications in optimization, and analyzes the limitations of existing methods. Subsequently, the methodology section elaborates on the motor optimization algorithm based on GNN, adaptive weighting, and MILP, including design parameter modeling, dynamic weight adjustment, and global optimization implementation. The experiments and results section demonstrates the effectiveness of the algorithm through comparative experiments, verifying its application value in motor design. Finally, the conclusion summarizes the main contributions of this paper and provides an outlook for future research directions. The main contributions of this paper can be summarized as follows:

1. This paper proposes a novel method for modeling the multi-parameter relationships in motor design using Graph Neural Networks (GNN). By treating the design parameters in motor design as graph nodes and using GNN's feature propagation mechanism to capture the complex dependencies between these nodes, the method addresses the challenge of traditional methods failing to effectively handle multi-parameter dependencies and provides more accurate data support for performance optimization.
2. This paper combines Mixed-Integer Linear Programming (MILP) technology with GNN-extracted design features, proposing a method for achieving global optimization of motor design under multi-objective and multi-constraint conditions. MILP can handle complex combinations of discrete and continuous variables, ensuring that the global optimal solution is found under multiple constraints, overcoming the problem of traditional optimization algorithms being prone to local optima in complex design tasks.
3. An adaptive weighting mechanism is introduced, enabling the optimization algorithm to dynamically adjust the importance of different parameters based on their actual impact on motor performance. By adaptively adjusting parameter weights, the optimization process can more flexibly adapt to different design scenarios and requirements, ensuring that key parameters are fully considered in the optimization process, thus improving optimization accuracy and model adaptability.

2. Related Work

The optimization of motor design performance is a complex problem involving multiple objectives and constraints, and it has long attracted widespread attention from both academia and industry. To address issues such as parameter dependencies, low optimization efficiency, and poor global convergence in motor design, researchers have explored various methods, including traditional parameter optimization techniques, the application of Graph Neural Networks (GNN) in design optimization, the integration of Mixed-Integer Linear Programming (MILP), and adaptive weight optimization mechanisms[9]. Traditional motor design methods largely rely on the experience of designers and trial-and-error processes, utilizing numerical simulation tools such as Finite Element Analysis (FEA) or Computational Fluid Dynamics (CFD) to predict motor performance. Hameyer et al.[10] proposed a shape optimization method for fractional horsepower DC motors based on stochastic methods. They emphasized the challenges posed by highly complex design parameters and various constraints in the Automatic Optimization Design (AOD) of electromagnetic devices. By combining numerical field computation techniques such as the Finite Element Method (FEM) with stochastic optimization methods, a general and effective solution to these complex technical issues was provided. Huang et al.[11] proposed a thermal design and analysis method for in-wheel motors based on oil spray cooling. Utilizing the flat structural characteristics of the in-wheel motor, they designed an oil spray cooling system and simulated the transient process of oil spraying from the nozzle onto the stator carrier and dripping onto the winding ends using a two-phase CFD method with a volume of fluid model. The effectiveness of this cooling system and the simulation method was validated through prototyping. These methods can provide estimates of various motor performance metrics (such as efficiency, power output, and temperature rise), but due to the complex manual tuning process, they are inefficient and struggle to find optimal solutions in large parameter spaces. In addition, early motor optimization methods typically used heuristic optimization algorithms, such as Genetic Algorithms (GA)[12] and Particle Swarm Optimization (PSO)[13], which simulate biological evolution or natural group behavior to gradually optimize design parameters. However, while heuristic algorithms can handle non-linear problems in motor design, their limitation lies in often finding only local optima, and their efficiency is low when dealing with complex multi-objective problems.

With the development of deep learning technologies, more and more research has applied these techniques to motor design optimization. In particular, Graph Neural Networks (GNNs), which propagate information through the structural features of graphs, can effectively aggregate information between nodes and capture complex parameter relationships. This enables GNNs to automatically extract features in multi-dimensional motor design optimization without the need for manually preset feature engineering. Sabir et al. [14] proposed a GNN-based optimization method, GNN-GA-AST, to address the nonlinear fifth-order induction motor model (FO-IMM) problem. By discretizing the nonlinear FO-IMM with GNN, a fitness function with mean square error as the objective was generated. This method also demonstrated consistency, effectiveness, and rapid convergence in solving the FO-IMM problem through numerical experiments and statistical analysis. Tang et al.[15] proposed a fault diagnosis method for induction motors based on a Graph Cardinality Preserving Attention Network (GCPAT), which can operate under various conditions, including steady-state and transient states. This helps engineers predict potential failure modes during the design phase, optimizing motor structure and material selection, thereby enhancing its reliability and lifespan. However, although GNNs can effectively model complex parameter relationships in motor design, most existing studies focus on single-objective or small-scale design problems. GNN's modeling capabilities and optimization effects still need further improvement when dealing with large-scale design problems with multiple objectives and constraints.

Mixed-Integer Linear Programming (MILP) is a widely used technique for optimization problems, capable of handling both discrete and continuous variables. In motor design, issues such as material selection, geometric design, and manufacturing processes often involve mixed variables. MILP provides an efficient optimization tool that ensures global optimal solutions under complex multi-objective and multi-constraint conditions. By establishing objective functions and linear constraints, it ensures that performance, cost, and manufacturing demands are met throughout the design process. Yamanaka et al.[16] proposed a MILP method for optimizing fuel consumption in Hybrid Electric Vehicle (HEV) power systems. By linearizing non-linear terms and employing piecewise linear and multilayer perceptron regression methods to approximate fuel consumption, the MILP optimization efficiently obtained Lagrange multipliers for design variables, facilitating effective design revision strategies. Robuschi et al.[17] proposed an iterative linear programming algorithm for calculating the optimal fuel energy management strategy of a parallel HEV under specific driving cycles. The method first established a mixed-integer model that included engine start-stop signals and gear shift commands, and by converting the fuel optimization problem into linear programming, the optimal shift trajectory and energy management strategy were quickly calculated, achieving a fuel-optimal control strategy with lower computational burden. However, a major limitation of MILP in motor design lies in its computational complexity. As design parameters increase, MILP's solution time may grow exponentially, especially when dealing with complex nonlinear constraints, making the solving process extremely complex.

Traditional motor design optimization methods often use fixed weights, which cannot adjust the importance of parameters in real-time according to changing design needs, leading to some parameters being overlooked or overly considered in certain scenarios, thus affecting optimization results[18]. The adaptive weight mechanism dynamically adjusts the weights of design parameters during the optimization process by assessing their contribution to the final performance in real-time, allowing the optimization algorithm to respond more flexibly to different design needs and scenarios. In multi-objective optimization problems, adaptive weights can enhance the algorithm's sensitivity to local performance requirements while maintaining global optimization objectives, improving optimization efficiency[19]. However, the application of adaptive weight mechanisms in motor design optimization is still in its early stages. Effectively implementing dynamic weight adjustments and integrating them with other optimization algorithms require further exploration[20]. Although the aforementioned methods have made some progress in motor design optimization, they still have some limitations. Traditional heuristic algorithms tend to fall into local optima, GNNs, while effective at modeling complex parameter dependencies, are not yet mature in large-scale optimization problems. MILP has strong capabilities for solving global optimization problems but suffers from high computational complexity, and adaptive weight mechanisms still face challenges in dynamically adjusting weights during the optimization process. This paper combines GNN, adaptive weights, and MILP to construct a motor design optimization framework that handles complex design parameter relationships, dynamically adjusts optimization weights, and achieves global optimal solutions. By using GNN to model motor design parameters, this approach addresses the issue of traditional optimization methods being unable to effectively capture multi-parameter dependencies. The introduction of an adaptive weight mechanism enhances the model's flexibility across different design scenarios. The integration of MILP ensures the ability to solve for the global optimum, significantly improving the effectiveness and efficiency of motor design optimization.

3. Method

Figure 1 illustrates the overall architecture of the motor optimization design algorithm proposed in this paper. First, the input is transformed into a graph representation, and a Graph Neural Network (GNN) is used to model the complex parameter relationships in motor design. Combined with Mixed-Integer Linear Programming (MILP) for prediction, it outputs the marginal probability of the variables. During this process, an adaptive weighting mechanism dynamically adjusts the importance of each parameter, ensuring that parameters with a significant impact on performance are prioritized in different design scenarios. Next, the algorithm selects key variables based on marginal probability, applies a rounding strategy to obtain an initial solution, and further refines this solution through trust-region search to approach the global optimum. The final output is a near-optimal design solution, achieving multi-objective optimization and efficient optimization under multiple constraints in motor design.

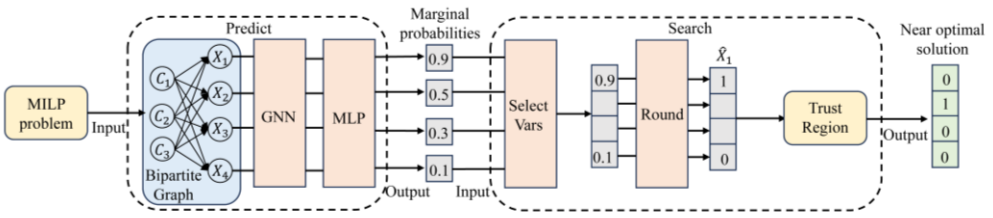

Figure 1. Overall algorithm architecture.

3.1 Graph Neural Network Architecture

In motor design optimization, there are often complex dependencies among design parameters. To effectively model these dependencies, a Graph Neural Network (GNN) is used to extract features and optimize the design solution. The network architecture is shown in Figure 2. First, the motor design parameters are modeled as an undirected graph $G=(V,E)$, where V represents the set of nodes, with each node $v \in V$ corresponding to a design parameter. E represents the set of edges, and an edge $(u,v) \in E$ represents the dependency or interaction between design parameters u and v. Each node v has an initial feature vector $h_v^{(0)} \in R^d$, representing the attributes of the parameter. The weight of the edge w_{uv} represents the strength of the relationship between parameters u and v, which can be set based on physical constraints, empirical rules, or historical data.

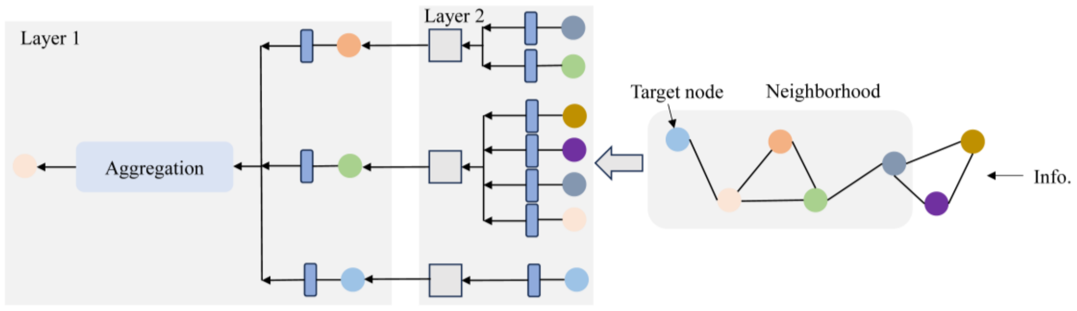

Figure 2. GNN network architecture diagram.

To update and propagate the features of the nodes, this paper uses a Graph Convolutional Network (GCN) to implement feature propagation and aggregation. The basic idea of GCN is to update each node's representation by aggregating the features of its neighboring nodes. The feature vector of each node in the l-th layer is updated through its neighboring nodes' features, with the specific update formula as follows:

$$h_v^{(l+1)} = \sigma\left(\sum_{u\in\mathcal{N}(v)} \frac{w_{uv}}{\sqrt{d_v d_u}} W^{(l)} h_u^{(l)} + W^{(l)} h_v^{(l)}\right) \quad (1)$$

where $h_v^{(l+1)}$ is the feature vector of node v in the $l+1$-th layer, $\mathcal{N}(v)$ is the set of neighbors of node v, w_{uv} is the weight of the edge (u,v), d_v and d_u represent the degrees of nodes v and u respectively, $W^{(l)}$ is the weight matrix of the l-th layer, and σ is a nonlinear activation function.

To avoid numerical instability during feature propagation, a normalized form of the graph Laplacian matrix is used for neighborhood feature aggregation:

$$\tilde{A} = D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \tag{2}$$

where A is the adjacency matrix of the graph, and D is the degree matrix. This ensures stability during information transfer in feature propagation. To capture the global dependencies among design parameters, a multi-layer GNN architecture is used. The features of each node depend on the features of its neighboring nodes in each layer, and through stacking multiple layers, the features of a node can aggregate information from farther neighbors, forming a global feature representation. The feature update formula for the l-th layer is:

$$H^{(l+1)} = \sigma\left(\tilde{A} H^{(l)} W^{(l)}\right) \tag{3}$$

where $H^{(l)}$ is the feature matrix of all nodes, the output of the l-th layer, and $W^{(l)}$ is the learnable weight matrix of the l-th layer. By stacking multiple GNN layers, the features of nodes can aggregate information from farther nodes layer by layer. After multiple layers of GNN feature extraction, the final feature representation of each node contains global information from itself and its neighboring nodes. To achieve global optimization, these node features are further processed into a global feature representation $z \in R^d$, which is used as the input to the subsequent optimization module. The final global feature representation can be obtained by pooling the feature vectors of all nodes:

$$z = \mathrm{Pool}(\{h_v^{(L)} \mid v \in V\}) \tag{4}$$

where $h_v^{(L)}$ is the node feature after L layers of GNN. By using GNN, the complex dependencies among motor design parameters are effectively modeled.

3.2 Mixed-Integer Linear Programming Architecture

In motor design optimization, it is necessary to handle both continuous and discrete decision variables. Therefore, MILP is used to handle these mixed-type variables and achieve global optimization under multiple objectives and constraints. The network architecture is shown in Figure 3. MILP defines an objective function and combines linear constraints to find the optimal solution globally, making it an effective tool for addressing complex motor design optimization problems. The MILP problem can be formulated as the following optimization problem:

$$\min\ f(x, y) = c^T x + d^T y \tag{5}$$

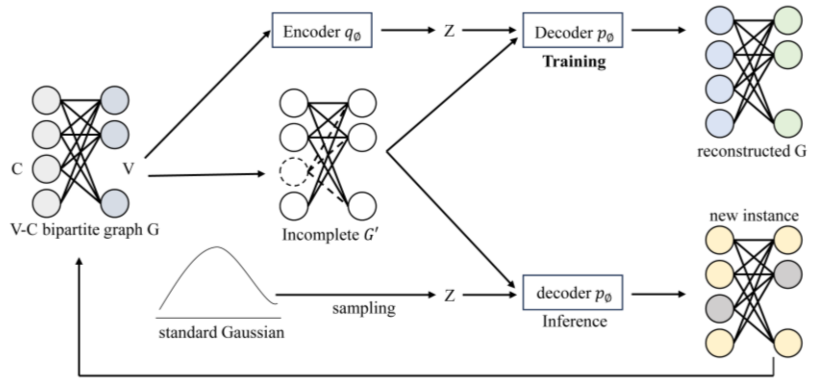

Figure 3. MILP algorithm architecture diagram.

where $x \in R^n$ are continuous variables, $y \in Z^m$ are integer variables, and $c \in R^n$ and $d \in R^m$ are the coefficient vectors of the objective function, representing the design parameters to be optimized. The goal of MILP is to minimize the objective function $f(x,y)$, subject to a series of linear constraints:

$$Ax + By \le b \tag{6}$$

where $A \in R^{p\times n}$, $B \in R^{p\times m}$, and $b \in R^{p}$. These constraints represent the physical, performance, or material limitations during the design process, such as power limits, thermal management requirements, or material properties.

In motor design optimization, the objective function typically includes multiple sub-objectives, such as minimizing loss, maximizing efficiency, and controlling costs. By weighting these objectives, a composite optimization objective is formed:

$$\min\ f(x,y) = \lambda_1 f_1(x,y) + \lambda_2 f_2(x,y) + \cdots + \lambda_k f_k(x,y) \tag{7}$$

where $f_1, f_2, \ldots, f_k$ are different objective functions, and $\lambda_1, \lambda_2, \ldots, \lambda_k$ are the weights of each objective, which can be adjusted according to design needs. The objective function is defined according to different design requirements. The efficiency maximization function improves the motor's energy conversion efficiency and reduces operating losses. This goal is usually achieved by optimizing design parameters such as motor geometry and winding structure. The corresponding objective function can be defined as the negative value of motor efficiency:

$$f_{\text{efficiency}}(x,y) = -\eta(x,y) \tag{8}$$

where $\eta(x,y)$ represents the motor's efficiency, which is a function of design parameters x and y. The cost minimization function reduces the total cost of motor manufacturing, including material and manufacturing costs. The objective function can be expressed as the weighted sum of material and manufacturing costs:

$$f_{\text{cost}}(x,y) = \alpha_{\text{material}} \cdot C_{\text{material}}(x,y) + \alpha_{\text{manufacture}} \cdot C_{\text{manufacture}}(x,y) \tag{9}$$

where $C_{\text{material}}(x,y)$ and $C_{\text{manufacture}}(x,y)$ are the material and manufacturing costs, and α_{material} and $\alpha_{\text{manufacture}}$ are the weight coefficients. The thermal management optimization function controls the motor's temperature rise to prevent damage due to overheating. The corresponding objective function can be defined as:

$$f_{\text{thermal}}(x,y) = T(x,y) \tag{10}$$

where $T(x,y)$ represents the maximum temperature rise of the motor under the given design conditions. To ensure that the optimization results are feasible in real-world applications, power constraints ensure that the motor's output power meets design requirements:

$$P_{\text{output}}(x,y) \ge P_{\text{required}} \tag{11}$$

where $P_{\text{output}}(x,y)$ is the output power of the motor design, and P_{required} is the minimum power requirement. Temperature constraints limit the motor's maximum operating temperature to prevent overheating:

$$T(x,y) \le T_{\max} \tag{12}$$

where $T_{\max}$ is the maximum allowable temperature. Material and geometry constraints limit the selection of materials and geometric dimensions within reasonable ranges:

$$L_{\min} \le L(x, y) \le L_{\max} \tag{13}$$

$$M_{\min} \le M(x, y) \le M_{\max} \tag{14}$$

where $L(x, y)$ and $M(x, y)$ represent the geometric dimensions and mass of the motor. Finally, CPLEX is used to solve the MILP problem and find the optimal design parameters x^*, y^* that minimize the objective function while satisfying all constraints. This method not only captures the complex dependencies among parameters but also performs global optimization, ensuring optimal performance under multi-objective and multi-constraint conditions.

3.3 Adaptive Weighting

The adaptive weighting mechanism dynamically adjusts the weight values by evaluating the contribution of each design parameter to the final objective function in real-time. This ensures that each parameter receives appropriate attention during different optimization stages. By automatically adjusting the importance of parameters according to their impact on performance, the optimization algorithm can flexibly handle complex design scenarios. The adaptive weighting mechanism expresses the optimization of the objective function as follows:

$$\min\ f(x, y, w) = \sum_{i=1}^{k} w_i f_i(x, y) \tag{15}$$

where w_i is the adaptive weight of the i-th objective function $f_i(x, y)$, and $f(x, y, w)$ is the weighted composite objective function. The adaptive weights w_i are dynamically updated during the optimization process.

At the start of the optimization, the weights w_i of all design parameters are initialized based on the design task's priorities. In the absence of specific priorities, the weights of all objective functions can be initialized to the same value:

$$w_i = \frac{1}{k}, \quad \forall i \tag{16}$$

where k is the number of objective functions. If the designer has prior knowledge of the importance of different objectives, the weights w_i can be assigned based on experience:

$$w_i = \frac{\text{priority}_i}{\sum_{j=1}^{k} \text{priority}_j} \tag{17}$$

where priority_i is the priority of the i-th objective function.

As the optimization process progresses, the influence of design parameters on the objective function may change, necessitating dynamic weight adjustment. This paper employs a gradient-based feedback mechanism to adjust the weights by evaluating the impact of each design parameter on the current objective function. The basic idea of weight updating is to adjust the weight corresponding to each objective function based on its rate of change during the optimization process. The weight update formula is as follows:

$$w_i^{(t+1)} = w_i^{(t)} - \alpha \frac{\partial f_i}{\partial x} \cdot \frac{\partial x}{\partial w_i^{(t)}} \tag{18}$$

where $w_i^{(t)}$ is the weight of the i-th objective at the t-th iteration, α is the learning rate controlling the speed of weight update, and $\frac{\partial f_i}{\partial x}$ represents the gradient of the design parameter x with respect to the objective function f_i. Through this gradient descent mechanism, parameters with greater influence on the objective function receive higher weights, while parameters with less impact have their weights reduced. The adaptive weighting mechanism dynamically adjusts parameter weights, enabling the optimization algorithm to effectively balance different design objectives, thus improving overall optimization performance in motor design.

4. Experiment

4.1 Experimental Data

Dataset 1 mainly focuses on optimizing the geometric design parameters of the motor[21], including seven key stator geometric parameters such as tooth head overhang 1, height of tooth head, tangential groove width, stator inner diameter, tooth head overhang 2, tooth width near air gap, and iron length, which vary during the simulation, while other electrical parameters (such as the number of slots, phase voltage, and phase current) remain constant. The generation of Dataset 1 is based on geometric models created using Computer-Aided Design (CAD). These geometric design parameters are then input into simulation software, which converts them into pixelized images, with each pixel representing different motor material components (such as air, metal, magnet, etc.). Through the simulation process, 68,099 samples were generated, and key performance indicators (KPIs) were derived for each design, including active part costs, critical field strength, maximum torque, maximum power, efficiency, and more. Table 1 lists the key performance indicators (KPIs) for Dataset 1, including costs of active parts, critical field strength, maximum torque, maximum power, efficiency, etc. By analyzing these KPIs, the impact of geometric parameter variations on motor performance can be assessed, and the design can be optimized accordingly.

Dataset 2 expands upon the modeling scope of Dataset 1, covering both stator and rotor geometric parameters[22]. The model for Dataset 2 includes 12 variables, representing the design of the full-pole cross-section of the motor. By modeling both the stator and rotor simultaneously, the samples generated from this dataset can more comprehensively reflect the overall performance of the motor. Similar to Dataset 1, these geometric parameters are transformed into pixelized images for simulation. A total of 7,744 samples were generated. Table 2 lists the KPIs for Dataset 2, including total cost, maximum torque, maximum power at maximum rpm, iron losses, copper losses, and the mass of different components. Dataset 2 is particularly suitable for studying the synergy between stator and rotor parameters and provides more detailed feedback during the optimization process.

Table1. Dataset 1 KPI introduction.

KPI	Parameter Description	**Unit**
z_1	Total cost	Euro
z_2	Critical magnetic field	kA/m
z_3	Peak torque	Nm
z_4	Maximum power	W
z_5	Efficiency rating	%
z_6	Torque fluctuation	Nm
z_7	Ripple behavior	-
z_8	Converter losses	W
z_9	Acoustic noise level	dBA
z_{10}	Highest magnet temperature	K
z_{11}	Peak winding temperature	K

Table2. Dataset 2 KPI introduction.

KPI	Parameter Description	**Unit**
q_1	Total cost	Euro
q_2	Peak torque	kA/m
q_3	Maximum power at top speed	Nm
q_4	Iron losses	W
q_5	Copper losses	W
q_6	Maximum torque ripple	Nmp
q_7	Iron mass	Kg
q_8	Copper mass	Kg
q_9	Magnet mass	Kg
q_{10}	Torque ripple characteristics	unitless

4.2 Evaluation Metrics

To evaluate the model's performance, two key evaluation metrics are employed: the dimensionless Mean Relative Error (MRE) and the Pearson Correlation Coefficient (PCC). These two metrics assess the accuracy of the model's predictions and the correlation between the input-output mappings from different perspectives. The Mean Relative Error is used to evaluate the relative error between the predicted and true values of the model. It is suitable for multi-output nonlinear regression problems where each Key Performance Indicator (KPI) has different dimensions. MRE is calculated using the following formula:

$$MRE = \frac{1}{n_{test}} \sum_{i=1}^{n_{test}} \left| \frac{y_j^{(i)} - \hat{y}_j^{(i)}}{y_j^{(i)}} \right| \times 100 \tag{19}$$

where $y_j^{(i)}$ is the true value of the i-th test sample, $\widehat{y_j^{(i)}}$ is the predicted value of the model, and $n_{test}$is the number of test samples. MRE is expressed as a percentage to measure the degree of deviation in the prediction results. This metric quantifies the accuracy of the model's predictions, with lower values indicating more accurate predictions.

The Pearson Correlation Coefficient (PCC) is used to measure the linear correlation between the input parameters and the target output. By calculating the correlation between the true values and the predicted values, PCC reflects the accuracy of the model's mapping to the target output. The formula is:

$$PCC = \frac{\sum_{i=1}^{n_{test}} \left(y_j^{(i)} - \bar{y}_j \right)\left(\hat{y}_j^{(i)} - \bar{\hat{y}}_j \right)}{\sqrt{\sum_{i=1}^{n_{test}} \left(y_j^{(i)} - \bar{y}_j \right)^2 \sum_{i=1}^{n_{test}} \left(\hat{y}_j^{(i)} - \bar{\hat{y}}_j \right)^2}} \tag{20}$$

where $\bar{y}_j$ and $\bar{\hat{y}}_j$ are the mean values of the true and predicted values, respectively. The PCC ranges from [-1, 1], with values closer to 1 indicating a stronger linear correlation between the model's predictions and the true values, implying better model performance.

4.3 Experimental Comparison and Analysis

Verify the effectiveness of the motor design performance optimization algorithm based on graph neural network representation and adaptive weights through experiments. We selected key performance indicators (KPIs) from the motor design task and divided them into Dataset 1 and Dataset 2. Using these datasets, we compared the performance of four models: GNN, MILP, AW, and the final fusion model. The experiments employed Mean Relative Error (MRE) and Pearson Correlation Coefficient (PCC) as evaluation metrics. MRE measures the error between the model's predicted values and the actual values, while PCC assesses the correlation between the predicted results and the actual values.

Table 4 provides a detailed comparison of the four models' performance on Dataset 1, primarily measuring model performance through MRE and PCC. First, the GNN model showed good performance across various indicators, but had a relatively high MRE value. For example, for the z_1 indicator, MRE was 1 and PCC was 0.91, indicating that the GNN model has strong correlation on this indicator but significant prediction error. For other indicators, such as z_3 and z_5, MREs were 0.61 and 0.56, with PCCs of 0.91 and 0.86, respectively, showing that the GNN predicts certain indicators accurately, but overall error still needs improvement. In comparison, the MILP model has certain advantages in handling global optimization problems, but MRE increased for some indicators; for example, for z_2 and z_4, MREs were 1.34 and 1.3, with PCCs of 0.89 and 0.87, indicating that the MILP model tends to get trapped in local optima for these indicators, leading to increased prediction error. The AW model (adaptive weight model) enhances prediction accuracy by dynamically adjusting weights. For indicators z_3, z_5, and z_8, the AW model achieved PCCs of 0.95, 0.92, and 0.93, showing strong correlation for these indicators. However, MRE increased for some indicators, such as z_2, where MRE reached 1.91, indicating that in certain cases, adjusting the adaptive weights may lead to increased prediction error. The final model integrates the advantages of GNN, MILP, and AW, performing well across multiple indicators, significantly reducing MRE and improving PCC. Additionally, Figure 4 clearly demonstrates the final model's predictive capability across different performance indicators, validating the model's effectiveness in motor design performance optimization.

Table 5 presents the performance comparison of each model on Dataset 2. It can be observed that the final model significantly reduced MRE across all indicators, indicating its predictive accuracy is notably superior to other models. Meanwhile, PCC improved to above 0.9 in most cases, suggesting that the final model's predictions have a stronger correlation with actual values. Particularly for indicators q_1 and q_3, the final model saw the greatest reduction in MRE, dropping to 0.43 and 0.19, while PCC increased to 0.96 and 0.93, respectively. This indicates that the final model not only improved prediction accuracy when processing Dataset 2 but also demonstrated more stable performance across different indicators. In contrast, the GNN, MILP, and AW models exhibited more dispersed performance in terms of MRE and PCC, failing to achieve the same level of optimization. Additionally, Figure 5 illustrates the prediction results of the final model across various indicators on Dataset 2, showing the distribution of predicted values compared to actual values, indicating the final model's high prediction accuracy for these indicators.

Table4. Comparison of related indicators on dataset 1.

Model	GNN		MILP		AW		Final Model	
	MRE	PCC	MRE	PCC	MRE	PCC	MRE	PCC
z_1	1	0.91	0.67	0.93	0.91	0.93	0.13	0.98
z_2	1.3	0.82	1.34	0.89	1.91	0.89	0.54	0.94
z_3	0.61	0.91	0.79	0.9	1.02	0.95	0.22	0.99
z_4	0.79	0.92	1.3	0.87	0.68	0.92	0.14	0.96
z_5	0.56	0.86	1.01	0.86	1.65	0.92	0.08	0.94
z_6	1.96	0.89	2.53	0.91	2.91	0.91	1.38	0.98
z_7	1.97	0.83	2.6	0.85	1.86	0.87	1.22	0.95
z_8	0.78	0.9	0.64	0.93	0.56	0.93	0.26	0.98
z_9	1.09	0.88	1.3	0.87	1.21	0.89	0.34	0.97
z_{10}	0.43	0.89	1.41	0.93	1.14	0.96	0.16	0.98
z_{11}	1.22	0.83	1.5	0.89	1.64	0.93	0.42	0.95

Table5. Comparison of related indicators on dataset 2.

Model	GNN		MILP		AW		Final Model	
	MRE	PCC	MRE	PCC	MRE	PCC	MRE	PCC
q_1	0.99	0.9	1.85	0.89	1.64	0.92	0.43	0.96
q_2	0.6	0.88	0.6	0.9	0.95	0.91	0.32	0.94
q_3	1.11	0.84	1.4	0.85	1.73	0.85	0.19	0.93
q_4	0.94	0.9	1.47	0.92	0.96	0.88	0.26	0.96
q_5	0.38	0.83	0.4	0.89	1.66	0.88	0.13	0.95
q_6	1.04	0.89	0.82	0.86	1.39	0.91	0.47	0.94
q_7	1.29	0.81	1.35	0.83	1.73	0.88	1.06	0.92
q_8	1.15	0.9	1.1	0.91	1.54	0.91	0.37	0.96
q_9	1.14	0.85	1.61	0.89	1.54	0.9	0.42	0.94
q_{10}	0.45	0.9	1.33	0.93	1.57	0.9	0.23	0.97

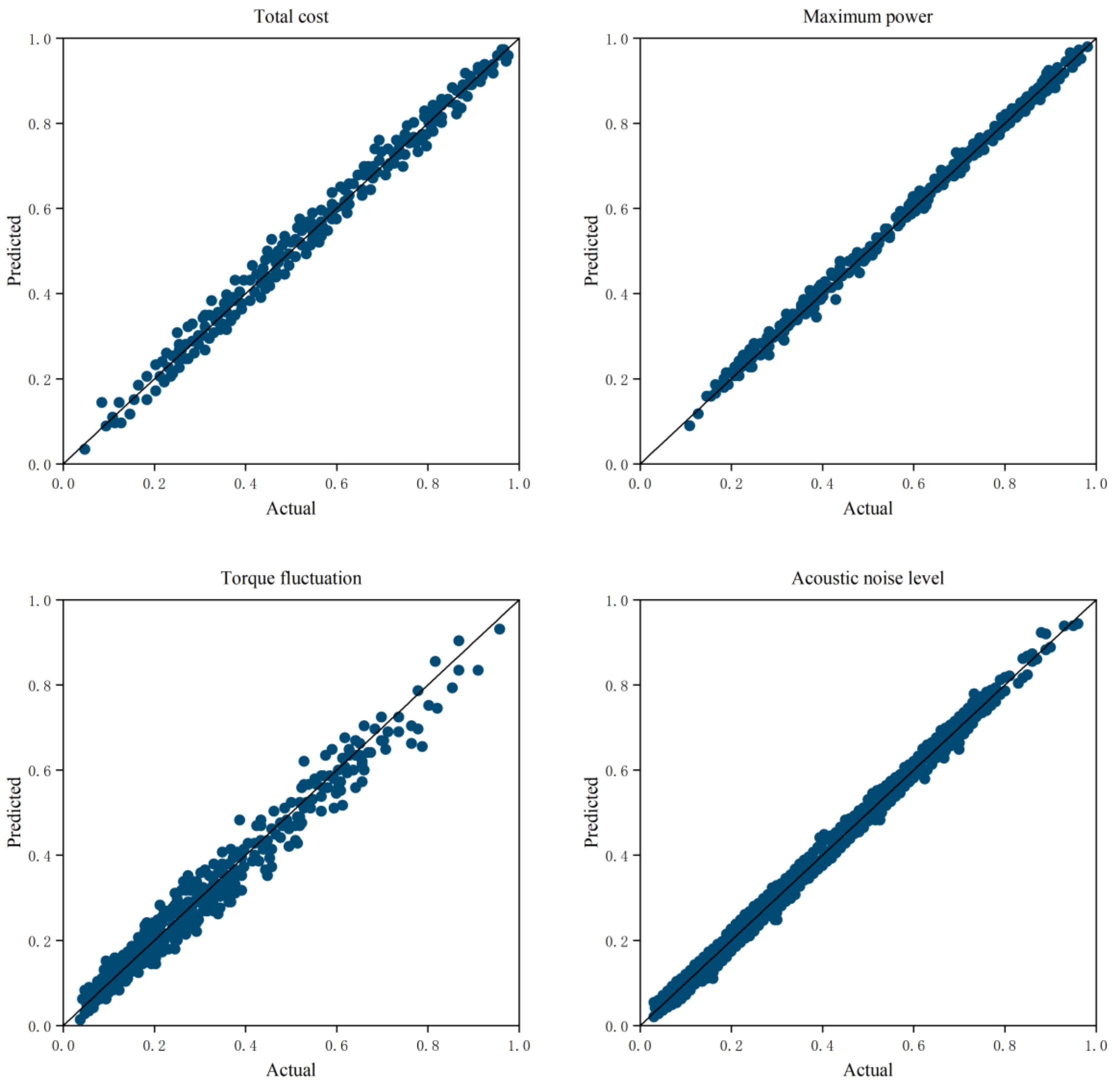

Figure 4. Metrics predictions for the final model on dataset 1.

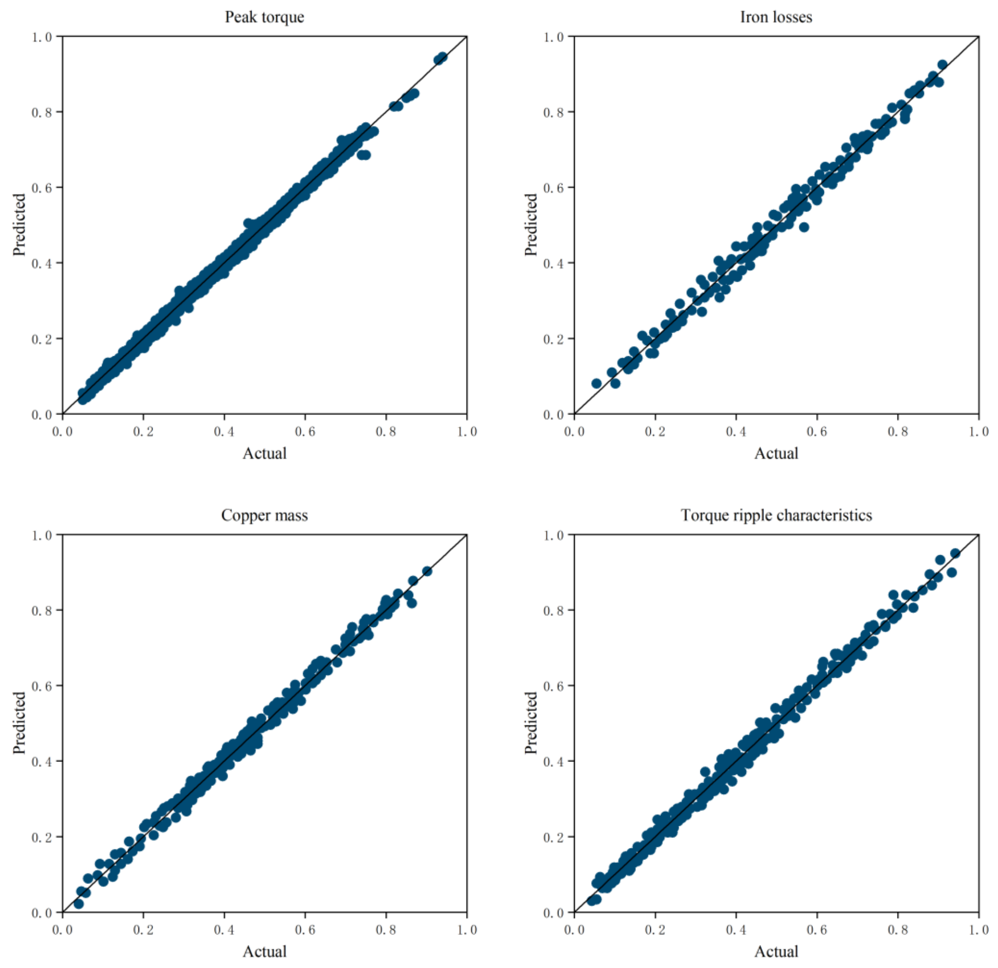

Figure 5. Metrics predictions for the final model on dataset 2.

5. Conclusion

In this paper, we proposed a novel motor design performance optimization algorithm that integrates Graph Neural Networks (GNN), Mixed-Integer Linear Programming (MILP), and an adaptive weighting mechanism. The algorithm addresses several key challenges in modern motor design, including the complexity of multi-parameter dependencies, the difficulty of global optimization, and the need for dynamic weight adjustment in multi-objective scenarios. Through the use of GNN, the algorithm effectively captures the intricate relationships between various design parameters, allowing for a more accurate representation of motor characteristics. MILP ensures global optimization across both continuous and discrete variables, overcoming the limitations of traditional optimization algorithms that often fall into local optima. The introduction of adaptive weighting further enhances the model's ability to adjust the importance of different parameters in real-time, ensuring that the design process remains flexible and adaptive to different performance requirements. Experimental results on two datasets demonstrated that the proposed algorithm significantly improves both accuracy and global optimization performance compared to traditional methods. The final integrated model consistently outperformed standalone GNN, MILP, and adaptive weighting models, achieving lower Mean Relative Error (MRE) and higher Pearson

Correlation Coefficient (PCC) in multiple KPIs. In conclusion, this work presents an efficient and scalable framework for motor design optimization, capable of handling the increasing complexity and performance demands in fields such as electric vehicles and intelligent manufacturing. Future research could explore further improvements in optimization techniques and applications to more diverse motor designs and configurations, enhancing the algorithm's adaptability and generalization across different industrial domains.

Funding
Not applicable

Author Contributions

Conceptualization, S. X. and H. Z.; writing—original draft preparation, M. W. and N. Z.; writing—review and editing, S.X. and M.W.; All of the authors read and agreed to the published the final manuscript.

Institutional Reviewer Board Statement
Not applicable

Informed Consent Statement
Not applicable

Data Availability Statement
Not applicable

Conflict of Interest

The authors declare no conflict of interest.

Reference

[1] T. Orosz et al., "Robust design optimization and emerging technologies for electrical machines: Challenges and open problems," vol. 10, no. 19, p. 6653, 2020.

[2] Y. Li, G. Lei, G. Bramerdorfer, S. Peng, X. Sun, and J. J. A. S. Zhu, "Machine learning for design optimization of electromagnetic devices: Recent developments and future directions," vol. 11, no. 4, p. 1627, 2021.

[3] C. A. Candelo Zuluaga, "Design optimization and performance analysis methodology for PMSMs to improve efficiency in hydraulic applications," 2022.

[4] Y. Zhao, C. Lu, D. Li, X. Zhao, and F. J. E. Yang, "Overview of the optimal design of the electrically excited doubly salient variable reluctance machine," vol. 15, no. 1, p. 228, 2021.

[5] P.-O. Gronwald and T. A. J. I. t. o. t. e. Kern, "Traction motor cooling systems: A literature review and comparative study," vol. 7, no. 4, pp. 2892-2913, 2021.

[6] Y. Li, C. Xue, F. Zargari, and Y. J. I. A. Li, "From Graph Theory to Graph Neural Networks (GNNs): The Opportunities of GNNs in Power Electronics," 2023.

[7] L. Alrahis, J. Knechtel, and O. Sinanoglu, "Graph neural networks: A powerful and versatile tool for advancing design, reliability, and security of ICs," in *Proceedings of the 28th Asia and South Pacific Design Automation Conference*, 2023, pp. 83-90.

[8] M. Wirtz, M. Hahn, T. Schreiber, D. J. E. C. Müller, and Management, "Design optimization of multi-energy systems using mixed-integer linear programming: Which model complexity and level of detail is sufficient?," vol. 240, p. 114249, 2021.

[9] S. Kumar Singh *et al.*, "Deep Learning in Computational Design Synthesis: A Comprehensive Review," vol. 24, no. 4, 2024.

[10] K. Hameyer and M. J. W. T. o. T. B. E. Kasper, "Shape Optimization Of A Fractional Horse-power De-motor By Stochastic Methods," vol. 2, 2024.

[11] C. Huang, L. Xiong, L. Hu, and Y. J. W. E. V. J. Gong, "Thermal Design and Analysis of Oil-Spray-Cooled In-Wheel Motor Using a Two-Phase Computational Fluid Dynamics Method," vol. 14, no. 7, p. 184, 2023.

[12] K. S. R. Rao and A. H. B. Othman, "Design optimization of a BLDC motor by Genetic Algorithm and Simulated Annealing," in *2007 International Conference on Intelligent and Advanced Systems*, 2007, pp. 854-858: IEEE.

[13] E. S. Rahayu, A. Ma'arif, A. J. I. J. o. R. Cakan, and C. Systems, "Particle swarm optimization (PSO) tuning of PID control on DC motor," 2022.

[14] Z. Sabir, M. A. Z. Raja, D. Baleanu, R. Sadat, and M. R. Ali, "Investigations Of Non-Linear Induction Motor Model Using The Gudermannıan Neural Networks," 2022.

[15] Y. Tang *et al.*, "Graph cardinality preserved attention network for fault diagnosis of induction motor under varying speed and load condition," vol. 18, no. 6, pp. 3702-3712, 2021.

[16] G. Yamanaka, M. Kuroishi, and T. J. E. O. Matsumori, "Optimization for the minimum fuel consumption problem of a hybrid electric vehicle using mixed-integer linear programming," vol. 55, no. 9, pp. 1516-1534, 2023.

[17] N. Robuschi, M. Salazar, N. Viscera, F. Braghin, and C. H. J. I. T. o. V. T. Onder, "Minimum-fuel energy management of a hybrid electric vehicle via iterative linear programming," vol. 69, no. 12, pp. 14575-14587, 2020.

[18] F. G. Borges *et al.*, "Metaheuristics-based optimization of a robust GAPID adaptive control applied to a DC motor-driven rotating beam with variable load," vol. 22, no. 16, p. 6094, 2022.

[19] M. Premkumar, P. Jangir, B. S. Kumar, M. A. Alqudah, K. S. J. C. Nisar, Materials, and Continua, "Multi-Objective Grey Wolf Optimization Algorithm for Solving Real-World BLDC Motor Design Problem," vol. 70, no. 2, 2022.

[20] S. Zhang, H. Yan, L. Yang, H. Zhao, X. Du, and J. Zhang, "Optimization design of permanent magnet synchronous motor based on multi-objective artificial hummingbird algorithm," in *Actuators*, 2024, vol. 13, no. 7, p. 243: MDPI.

[21] M. R. Raia, S. Ciceo, F. Chauvicourt, and C. J. A. S. Martis, "Multi-attribute machine learning model for electrical motors performance prediction," vol. 13, no. 3, p. 1395, 2023.

[22] M. Wiesheu, T. Komann, M. Merkel, S. Schöps, S. Ulbrich, and I. C. J. a. p. a. Garcia, "Spline-Based Rotor and Stator Optimization of a Permanent Magnet Synchronous Motor," 2024.