Predicting Permanent Magnet Synchronous Motors Temperature using Different Machine Learning and Deep Learning Models

1st Naheem Olaniyan
Department of Computer Science
Lakehead University
Thunder Bay, Ontario, Canada
nolaniya@lakeheadu.ca

2nd Ravihari Liyanage
Department of Computer Science
Lakehead University
Thunder Bay, Ontario, Canada
cliyanag@lakeheadu.ca

3rd MD Meher Hassan Chowdhury
Department of Computer Science
Lakehead University
Thunder Bay, Ontario, Canada
mchowdh8@lakeheadu.ca

4th Jahin Ahmed
Department of Computer Science
Lakehead University
Thunder Bay, Ontario, Canada
jahmed3@lakeheadu.ca

5th Dr. Thangarajah Akilan
Department of Computer Science
Lakehead University
Thunder Bay, Ontario, Canada
takilan@lakeheadu.ca

***Abstract*—The use of electric vehicles driven by Permanent Magnet Synchronous Motors(PMSM) is very popular in the world. However, due to the complexity of the motor, the monitoring of motor temperature is difficult and costly through traditional thermal estimation techniques. Hence, this study focuses on developing a data-driven temperature prediction model for permanent magnet synchronous motors using several machine learning models. The Ordinary Least Square Method, K-Nearest Neighbor, Decision Tree, Random Forest, and Convolutional Neural Network(CNN) were trained as the models to perform time-series regression analysis, and the performances of these models were evaluated against the Goodness of Fit (R2), Mean Squared Error(MSE) and Mean Absolute Error values. These models were trained to predict four temperature parameters of PMSMs; the temperature of magnet surface, stator's yoke, tooth, and winding based on the values of motor speed, ambient, torque, direct-axis current, quadrature-axis current, direct-axis voltage, quadrature-axis voltage, and coolant temperature. According to the experimental results of the study, the CNN and Random Forest can be considered as the best two models where the CNN has gained the minimum MSE value of 0.0264 and Random Forest presented the highest goodness of fit value of 0.9999 for their testing datasets.**

***Index Terms*—deep learning, machine learning, permanent magnet synchronous motor, temperature prediction**

I. Introduction

The Permanent Magnet Synchronous Motors(PMSMs) are a very popular choice in electric vehicles mainly due to their high torque capabilities and degree of efficiency. However, as there are temperature-sensitive components and they are likely to fail in high temperatures, temperature monitoring is important to ensure the safe operations of such motors.

Even though sensor-based techniques to measure the machine's thermal state is more accurate and efficient, using it for measuring the rotor temperature is costly and technically infeasible due to its rotation and complex structure [1], [2]. The techniques, such as Computational Fluid Dynamics (CFD) and Heat Equation Finite Element Analysis (FEA) require many computational resources and hence are not suitable for real-time thermal observations [3]. Further, classical thermal modeling techniques, such as Lumped-Parameter Thermal Networks (LPTNs) require expert knowledge in deciding model parameters[4]. When considering these limitations, the use of data-driven approaches, such as machine learning and deep learning will be more suitable as they can be developed without expert knowledge and will be efficient and cost-effective.

The purpose of this study is to investigate the performances of several machine learning and deep learning models for predicting temperature in PMSMs. Overall, five models were implemented, including 03 classification models; K-Nearest Neighbor (KNN), Decision Trees and Random Forest, 01 regression model; Least Square Regression, and finally a deep learning model; Convolutional Neural Network (CNN).

We formulated the following hypothesis for our research:

- There is a relationship between the features of Permanent Magnet Synchronous Motors.
- Some features that are highly correlated can be used in the prediction of Permanent Magnet Synchronous Motors temperature using different Machine Learning and Deep Learning Models.
- The features can be grouped to the top ten profile_id occurrences and used to develop classification models.

II. Literature Review

Previous studies have attempted to address the problem of predicting temperatures in PMSMs using machine learning and deep learning approaches. Many previous researchers have focused on using deep neural network approaches to address this problem [1]–[6] and the different types of deep networks used were, Long-Short-Term-Memory(LSTM), CNN, Residual Neural Network (ResNet), etc. However, several researchers among them have used some other machine learning approaches, such as KNN [1], [6], [7], support vector machine (SVM) [1], [8], Random Forests [1], [7]–[9], ordinary least squares[1]. Not only that, another study has conducted experiments with linear regression with selected features and added kernels as the baseline work to reduce the bias in predicting temperatures [6].

The temperature of PMSMs can be measured through the temperatures of different components in the motor, such as stator winding, stator yoke, stator tooth, and permanent magnet. While many studies [2]–[4], [6], [10] have used the temperature of all of these four components as the output features in predicting the PMSM temperature, some studies have used different combinations of them as well. As the predicting feature, the study [1] has only used the temperature of the permanent magnet and the study [5] has only predicted the stator winding temperature.

In another research, author adopted the actor-critic framework of reinforcement learning to predict the temperature of permanent magnet synchronous motors [13]. The actor-critic framework of reinforcement learning simultaneously learns a policy function and a value function. The policy tells us how decisions should be made, while the value function helps to improve the training process for the value function. The author additionally uses the simplified proximal policy optimization algorithm to optimize the value of the prediction temperature of PMSMs. The interactive state of the values comes from the PMSMs dataset. The target function of the actor is dynamically adjusted based on the feedback function, using the Nadam algorithm in the gradient optimization process. A setback to this approach is that the collection of ground-truth value function takes so much time resulting in high computational time and waste of resources.

In paper [14], the author proposed the Pseudo-Siamese Nested LSTM (PSNLSTM) model in predicting the temperature of PMSMs. The PSNLSTM uses relevant features related to PMSMs temperature as input and predicts the temperature of stator yoke, stator tooth, and stator winding. The NLSTM network adds depth to LSTM via nesting as opposed to stacking. In the paper, the author uses the optimization algorithm of learning rate combined with gradual warmup and decay to accelerate the convergence during the training and improve the training performance of the model. The downside of the method is that the adoption of the pseudo-siamese networks might result in high computational time during training. This is because pseudo-siamese networks learn from quadratic pairs and need to see all available information. Also, the method does not give probability estimates but the distance between each class.

III. Methodology

A. Exploratory Data Analysis (EDA)

The dataset used for the experiment was online on [15] and it contains 1.3million data records for 13 different classes. All these records were collected from multiple testing sessions on the same Permanent Magnet Synchronous Motor(PMSM) with 2Hz sampling rate. The table I gives a brief introduction to the set of features. The independent features that affect the motor's temperature are the current and voltage of both d and q-components, ambient, torque, motor speed and coolant temperatures. The temperatures in the motor's permanent magnet surface, stator's tooth, yoke and winding are considered as output parameters. Therefore, eight input and four output parameters have been considered in this study.

TABLE I: Description of Features in the dataset

Feature	Data Type	Description
u_q	float	Voltage in q axis measured in Volts
coolant	float	motor coolant temperature in
stator_winding	float	stator winding temperature in °C
u_d	float	Voltage in d axis measured in Volts
stator_tooth	float	stator tooth temperature in °C
motor speed	float	ambient temperature around the stator in °C
i_d	float	Current in d axis measured in Amps
i_q	float	Current in q axis measured in Amps
pm	float	permanent magnet tooth temperature in °C
stator_yoke	float	Stator yoke temperature in °C
ambient	float	ambient temperature around the stator in °C
torque	float	torque of the motor
profile_id	integer	Id of the measurement session

All the records are grouped according to a particular session using the profile_id and the recording time of each instance is 2 seconds. Figure 1 shows a horizontal bar plot that demonstrates the duration of each session. It shows that the duration of all the sessions was distributed around 1 to 6 hours. However, most of the recordings were taken for less than 3hrs.

We further show the correlation between each of the attributes and explain how each attribute affects the other in figure 2. Higher numbers indicate higher correlation between attributes. Ambient and Coolant have higher positive correlations towards the predictive variables, such as stator_yoke and stator_tooth. A perfect positive correlation of 1.0 was shown in between the i_q and torque while toque and u_d has shown a higher negative correlation of -0.75. Furthermore, there are very strong positive correlations between the stator variables; stator_tooth, stator_yoke and stator_winding.

Lastly, figure 3 shows the attributes and how they were affected under different circumstances with their mean and median respectively.

B. Data Curation and Conditioning

We started with evaluating the skewness of the dataset. The data skewness is depicted in figure 4 using three attributes (stator yoke, stator tooth, and stator winding) and as the length of all three curves are almost similar, it was verified that there is no skewness of the data.

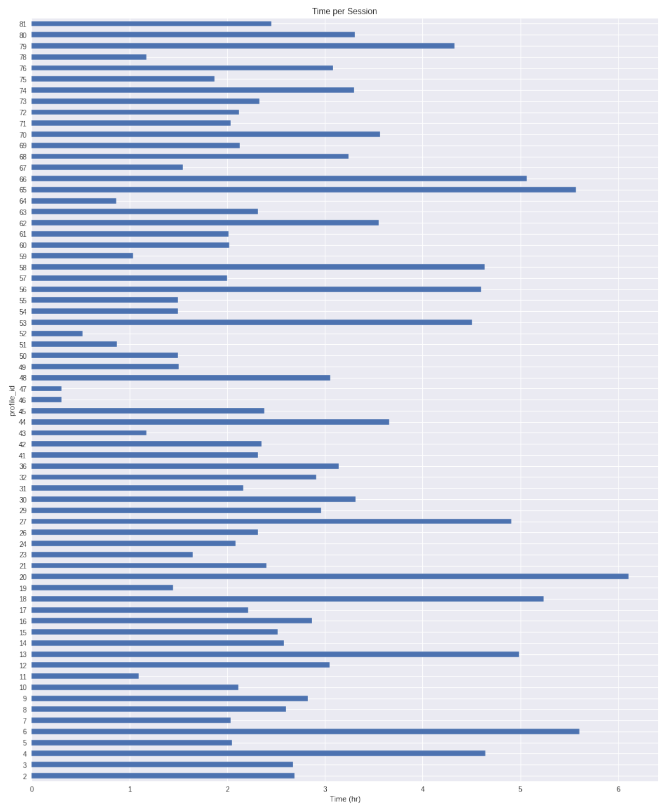

Fig. 1: The measurement of each session based on the profile_ID (recorded in 2HZ)

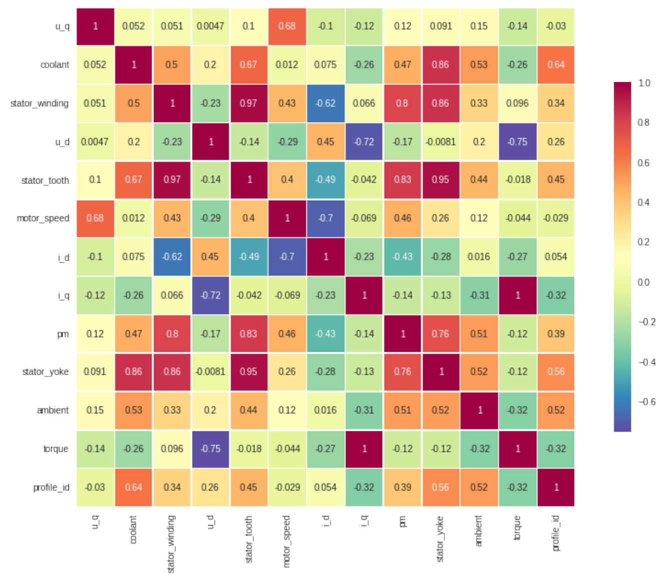

Fig. 2: Correlation between each attribute

We dropped some columns of the dataset because they are not well-correlated with the other attributes. The selection of the best features will helps prevent overfitting, producing better output for our models. We then performed random shuffling to mix up records in the sample well by setting the random state to 3. Next, the standard scaling of features was performed using the MinMaxScaler function to bring the features into a single scale. This will be important for performing distance-based calculations in future steps.

Furthermore, as the performance of different models will be different based on the applied pre-processing techniques, several other pre-processing techniques, such as inverse transformation, dropping columns and principle component analysis(PCA) was conducted during different model development stages.

Since all the variables are measured over time, the later analysis was done based on assuming this as a time-series dataset. Hence, the dataset was split into number of time sequences. The size of the time sequence is an important decision, as higher sequences can lead to higher prediction errors. Therefore, the data has been divided into time period of 5 seconds (10 records) and target value was the temperature of the next 0.5 second (1 record). Further, the profile ids 65 and 72 were separated as the testing datasets.

C. *Assessment of the Prediction Models*

As the numeric measurements of evaluating the model's performance, the Goodness of Fit (R2), Mean Squared Error(MSE) , and Mean Absolute Error(MAE) were used. The R2 is a representation of the relative measure of fit and MSE can be used to represent the absolute fit of the model. The performance of a model will be best when the R2 value is closer to 1 and MSE and MAE are closer to zero. The corresponding equations of these measurements are given below in (1), (2) and (3).

$$R^2 = 1 - \frac{\sum_{i=1}^{m}(Y_{Act} - Y_{Pred})^2}{\sum_{i=1}^{m}(Y_{Act} - Y_{Avg})^2} \tag{1}$$

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(Y_{Act} - Y_{Pred})^2 \tag{2}$$

$$MAE = \frac{\sum_{i=1}^{n}|Y_{Act} - Y_{Pred}|}{n} \tag{3}$$

where:

Y_{Act} = the actual output value
Y_{Pred} = the predicted output value
n = the total number of records

D. *Model Implementation*

We implemented five different models, consisting of both classification and regression approaches. These models' implementation is explained in this part of the paper.

1) Ordinary Least Square (OLS): Ordinary Least Squares (OLS) is a form of supervised learning that is used to estimate unknown parameters by creating a model that reduces the sum of the squared errors between the observed data and the predicted one. In OLS, we calculate the distance from each data point and the regression line, square it, and sum all the

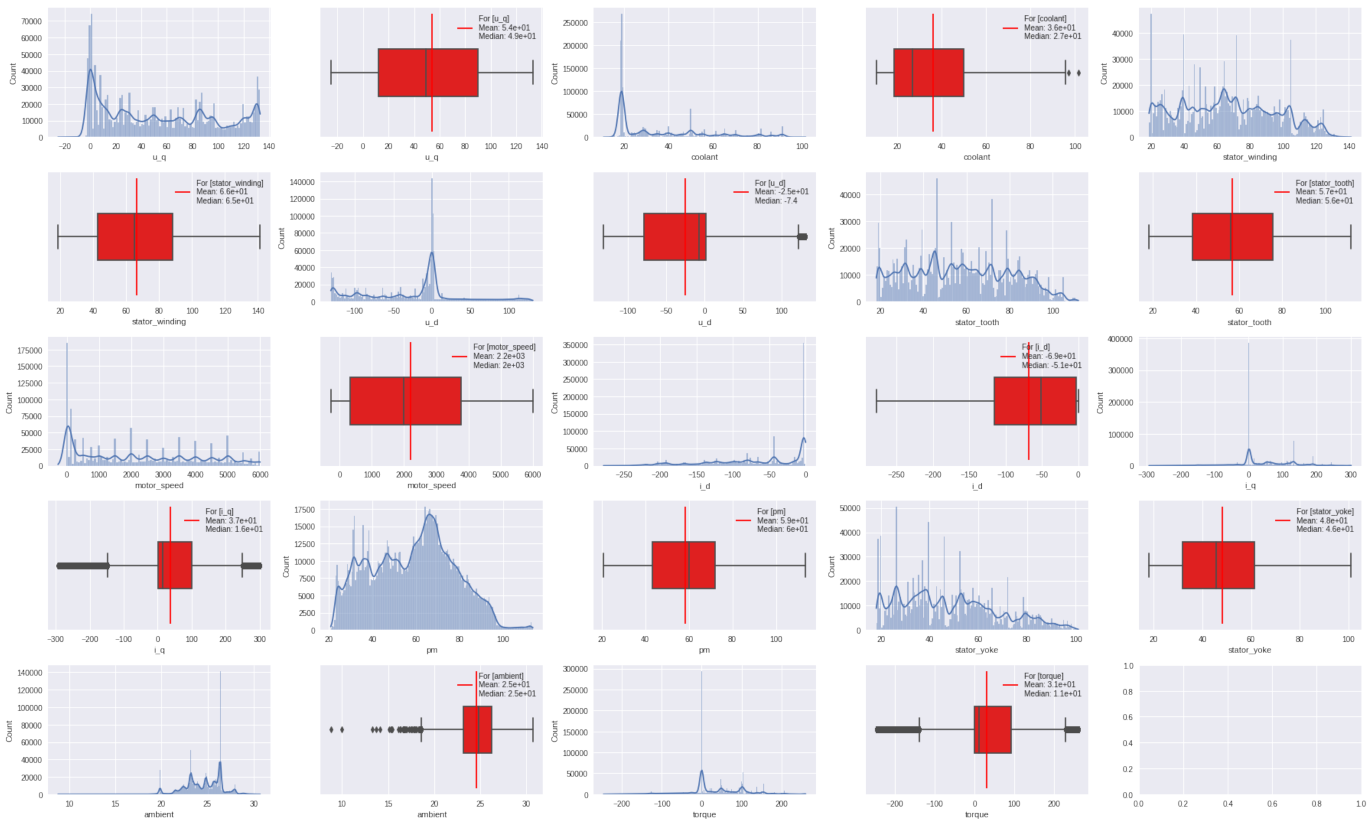

Fig. 3: Distribution of the attributes based on mean and median

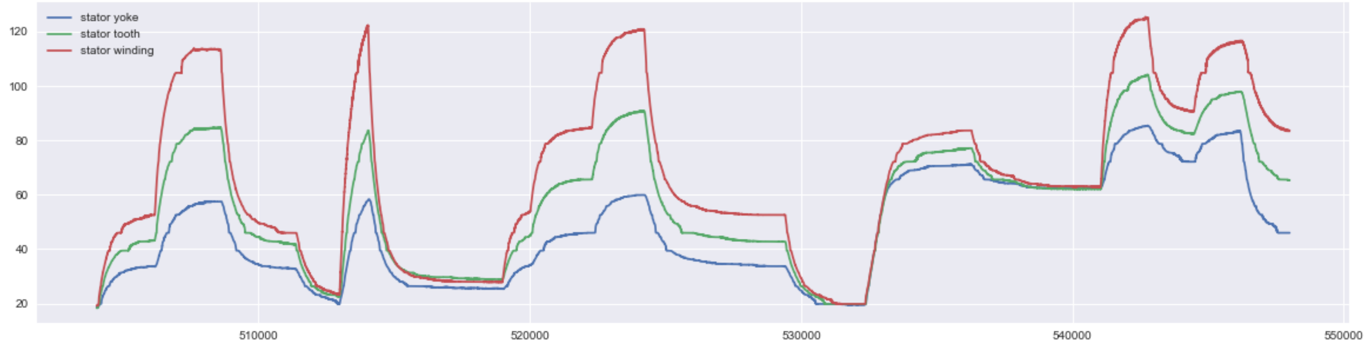

Fig. 4: The Skewness of three attributes in the dataset

squared errors [16]. OLS works for both univariate and multivariate datasets.

$$m = \frac{\sum(x_i - \overline{x})(y_i - \overline{y})}{\sum(x_i - \overline{x})^2} \tag{4}$$

where:

x	independent variables
$\overline{x}$	average of independent variables
y	dependent variables
$\overline{y}$	average of dependent variables

The Ordinary Least Square (OLS) technique was used as a multivariate regression model for predicting the temperature of PMSMs. For this model, seven features were used as the independent variables to predict the temperature of the permanent magnet(PM). As the results of the OLS, the average R2 of 0.609 and the root means squared error (rmse) of 11.9825 and 11.9835 for training and testing sets were gained. As the R2 is closer to 50% and both rmse values are relatively larger, these results were not considered as better results. Hence, to improve the performance of the OLS model, some other pre-processing techniques were performed. For that, some features named, coolant, ambient and i_d, which are not well-distributed within the dataset were manually transformed by changing their values and removing negatives. For this transformed dataset the avarage rmse of the OLS model was 12.2913 and 12.3036 in the training and testing sets respectively. As there is no improvement in the rmse by transforming the data, further experiments were not continued with the transformation.

Next, the PCA technique was used to examine the interrelations among a set of variables in the dataset and after that, the training and testing rmse gained for the model were 12.8174 and 12.8280 respectively. Since the results were further degraded, elimination of some of the features was performed. However, after dropping the i_d and i_q, which are not important features in terms of correlation, the rmse values got further increased to 13.3431 and 13.3495 for training and testing respectively. Finally, another column; motor speed with less correlation was removed and the rmse values could be diminished to 12.3749 and 12.3843. As different pre-processing techniques were unable to improve the performance of the OLS model, the results of the initial model with basic pre-processing steps were recorded as the highest performance.

2) Convolutional Neural Network (CNN): For our deep learning approach, we developed Convolutional Neural Network to solve the regression problem of predicting the temperature of PMSMs. Even though deep CNNs are quite popular in the Image Processing and Computer Vision context, the use of CNN's for time-series analysis problems has been recognized due to their less computational requirement compared to other common sequential data analysis algorithms, such as Recursive Neural Networks (RNN). Here, the pm, stator_yoke, stator_tooth, and stator_winding were selected as target variables, and the 8 other features were selected as input variables. Initially, the dataset was pre-processed using a standard scaler and inverse transformation functions and later divided into time sequences with the purpose of predicting the temperature in time series. The detailed layer diagram of the model with the best performance is shown in figure 5.

The network structure of our model consists of 8 1D convolutional layers with kernel size three, 1 1D max-pooling layer, 1 flatten layer, 2 dense layers, and a dropout layer in-between these two dense layers. A max-pooling layer and a flattened layer were used for pooling and flattening of the input sequence to prepare it as an input to the fully connected dense layer with 128 neurons to learn the pattern in the data sequence. Finally, the last dense layer was set with 4 neurons to predict the output into four target factors. The total number of trainable parameters in the model was 2,454,404. The different hyperparameter settings that were used for model training was given in Table II.

The model training for 30 epochs consumed around 10hrs of time. Later, when we enabled GPUs in a running environment for the training of 50 epochs, the total computational time was reduced to 43 seconds on average. We tried to avoid the over-fitting of the model by using the early stopping technique and by adding dropout layers as model regularization techniques.

3) K-Nearest Neighbor: The k-nearest neighbors (KNN) algorithm is a supervised learning algorithm that can be used to solve both classification and regression problems. KNN algorithm assumes that similar things are close to each other. The algorithm stores all available data and classifies a new data point based on the similarity. The new data can then be classified into an appropriate category. The steps in KNN algorithms are:

- Select the number K of the neighbors
- Calculate the Euclidean distance of K number of neighbors
- Take the K nearest neighbors based on the calculated Euclidean distance (the length of a line segment between the two points).
- Count the number of the data points in each category Among these k neighbors
- Finally, assign the new data points to that category for which the number of the neighbor is maximum.

In order to perform regression operation using the KNN model, first, the 10 dominant profiles with a higher number

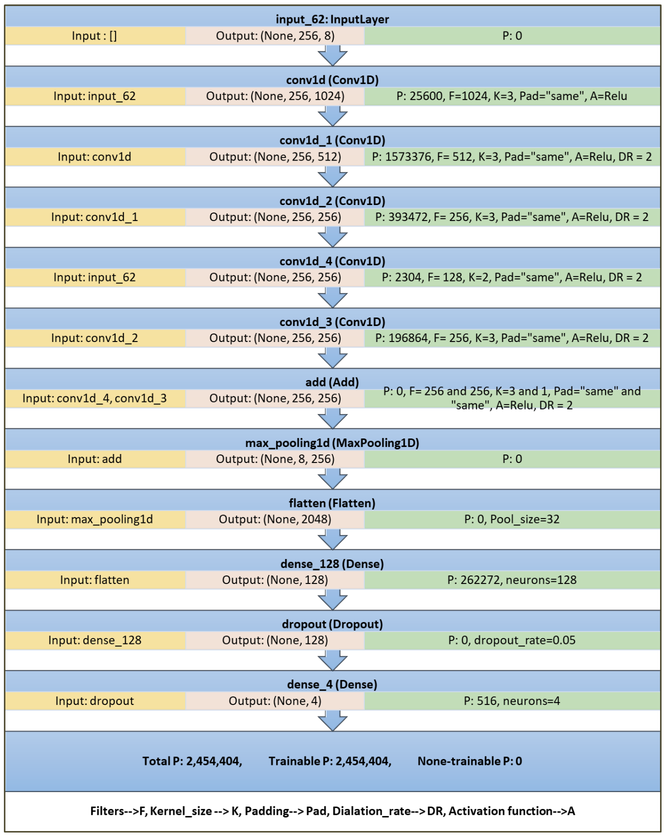

Fig. 5: The CNN Model Architecture.

TABLE II: Hyperparameters used in the CNN model

Hyperparameters	Value
Epochs	50
Batch Size	1000
Learning Rate	0.00001
Loss Function	Mean Squared Error (MSE)
Optimizer	Adam
Activation functions	ReLu
Kernel Size	2 and 3

of records have been filtered. For example, profile_id 17 with 15964 rows. These profiles were used as the classes during the prediction. Next, a hyperparameter tuning was performed for selecting the based number of neighbors. This experiment was conducted by monitoring the performance of the KNN model from 1 to 9 neighbors. The best number of neighbors was found as 1 according to the following graph in figure 8 AND 9. Using neighbor 1 with the best accuracy, the KNN model has 0.9987 accuracies for the testing set.

4) Decision Tree: For classification and regression, Decision Trees (DTs) are a non-parametric supervised learning method [17]. The goal is to learn simple decision rules from data attributes to develop a model that predicts the value of a target variable. A tree is an approximation of a piecewise constant. Decision Trees require less computational power. Before training the decision tree model, the dataset was preprocessed using the Standard Scaler function. The preprocessing and model training was done within a pipeline to automate the workflow.

5) Random Forest: Random Forest is an ensemble learning method where it aggregates the results of many decision trees which trained on different subsets of data. The result of the Random Forest model is the average of prediction results of these individual decision trees[18]. The nonlinear nature of this model will make it better for regression analysis and these Regression Random Forest models can be used in time-series predictions. Further, Random Forests have proved to be good at handling larger datasets efficiently.

The Random Forest model was developed by setting the following hyper-parameters as given in Table III. Since the bootstrap has been set to True, the whole dataset will not be used to build the tree.

TABLE III: Hyper-parameter settings of the best Random Forest Model

Hyperparameters	Description	Value
n_estimators	number of trees in the forest	100
criterion	Function to measure the quality of a split	Gini
max_depth	maximum depth of the tree	None
max_features	number of features to consider when looking for the best split	Auto
bootstrap	Whether bootstrap samples are used when building trees	True

IV. RESULTS AND DISCUSSION

A. Results of the Models

1) Ordinary Least Square: In our OLS model, we got an R2 of 0.609, representing the relative measure of fit. Further, a root means squared error (rmse) of 11.9825 and 11.9835 were obtained for training and testing sets respectively to represent the absolute fit of the model. As the R2 is closer to 50% and both rmse values are relatively larger, these results were not considered as better results. Hence, to improve the performance of the OLS model, some other pre-processing techniques were performed. For that, some features; coolant, ambient and i_d, which are not well-distributed within the dataset were manually transformed by changing their values and removing negatives. For this transformed dataset the terms of the OLS model were 12.2913 and 12.3036 in the training and testing sets respectively. As there is no improvement in the rmse by transforming the data, further experiments were not continued with this data transformation.

TABLE IV: Final Result of OLS Model.

Dep. VARIABLE	PM	R-squared	0.609
Model:	OLS	Adj.R-squared:	0.609
Method:	Least Squares	F-statistic	1.988e+05
No. Observations	892794	Pronb(F-statistic)	0.00
Df Residuals	892786	Log-Likelihood	-3.4840e+06
Df Model	7	AIC:	6.968e+06
Omnibus	31176.980	BIC:	6.986e+06
Prob(Omnibus)	0.000	Durbin Watson	2.002
Skew	0.304		
Rainbow Test			0.8242
Goldfeld Quantile distribution test			0.6803
	ambient		5.145393
	coolant		1.419280
	u_d		3.3383332
Variance inflation factor	u_q		11.987696
	motor speed		5.715799
	i_d		3.556909
	i_q		1.48898

Next, the PCA technique was used to examine the inter-relations among a set of variables in the dataset and after that the training and testing rmse gained for the model were 12.8174 and 12.8280 respectively. Since the results were further degraded, elimination of some of the features were performed. However, after dropping the i_d and i_q, which are not important features in terms of correlation, the rmse values got further increased to 13.3431 and 13.3495 for training and testing respectively. Finally, another column; the motor speed with less correlation was removed and the rmse values could be diminished to 12.3749 and 12.3843. As different pre-processing techniques were unable to improve the performance of the OLS model, the results of the initial model with basic pre-processing steps were recorded as the highest performance.

Some of the statistics from the best output is given in table IV below. Here, the Durbin-Watson (DW) which is a value important in time-series analysis to represent the presence of autocorrelation in the dataset was recorded as 2.002 and this represents that there is very low autocorrelation in the dataset as it is very closer to 2. The skewness value of 0.304, which is between -0.5 and 0.5 represents that the distribution of the dataset is fairly symmetrical.

Moreover, several other tests, such as the rainbow test, Goldfeld Quantile distribution test, and variance inflation factor(vif) were conducted to measure the performance of the OLS model. As the pvalue for rainbow test is greater than 0.05, it will fail to reject the null hypothesis and proves that the data follows linearity. Further, the value for the Goldfeld Quantile distribution test is greater than 0.05, unable to reject the null hypothesis and proves that the data is homoskedastic in nature. Finally, as the vif value for the u_d was higher than 10, it can be concluded that there is significant multicollinearity that needs to be corrected.

2) Convolutional Neural Network: The performance of the developed CNN was monitored with different hyper-parameters settings and different model architectures. The final best model with 8 convolutional layers, Relu activation function was trained for a different number of epochs. The summary of the last three epochs of experiments with and without GPU processing support were summarized in Table VI. Even with a higher number of epochs, the results for the test with GPU support showed comparatively very low computational time compared to the environment without GPU.

According to the results in Table VII, the performance of the CNN model has been improved over epochs as the MSE and MAE values were gradually decreasing and there was no overfitting as the MSE and MAE values of both training and validating are very lower and closer to each other. The testing results of the CNN model were 0.0264 and 0.1122 for the MSE and MAE respectively. The plot of MSE Vs epochs and MAE Vs epochs for 50 epoch test were given in Figures 6 and 7 respectively.

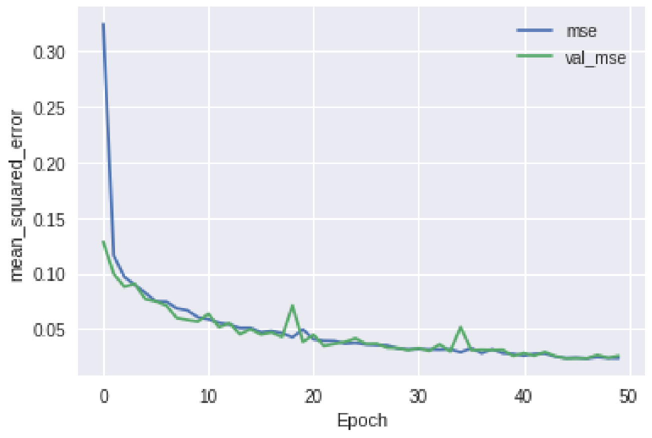

Fig. 6: The mean squared error vs. epochs for the training and validation sets of the CNN model.

3) K-Nearest Neighbor: For finding the best results of the KNN model, a hyperparameter tuning was performed. This experiment was conducted by monitoring the R2 and MSE results of the KNN model by changing the neighbors from 1 to 9. The best number of neighbors was found as 1 according to the following graphs in figure 8 and figure 9. Using the neighbor = 1, the KNN has shown a 0.9715 R2 value, 4.2694 of RMSE, and 0.9249 MAE value for the testing set.

4) Random Forest: The results of the Random forest model were recorded as 0.9999 for R2, 0.2399 for root mean squared error, and 0.0019 for MAE.

5) Decision Tree: The Decision Tree model has shown values of 0.9992 for R2, 0.6940 for RMSE, and 0.0186 for MAE.

B. Model Comparison

All four models, except the OLS method, have shown competitive results in predicting permanent magnet motor temperature. The overall best results of each model have shown in table VIII below.

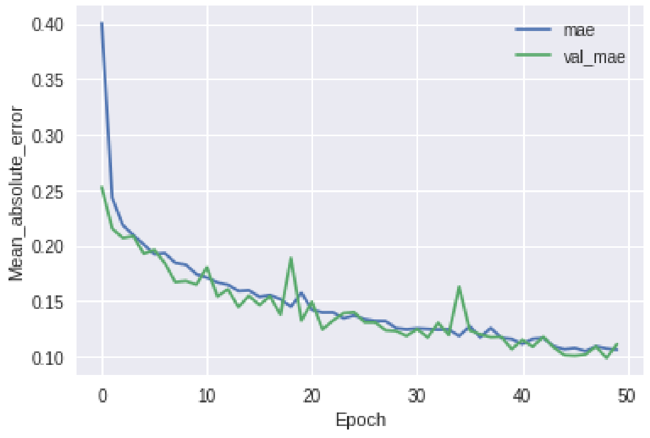

Fig. 7: The mean absolute error vs. epochs for the training and validation sets of the CNN model.

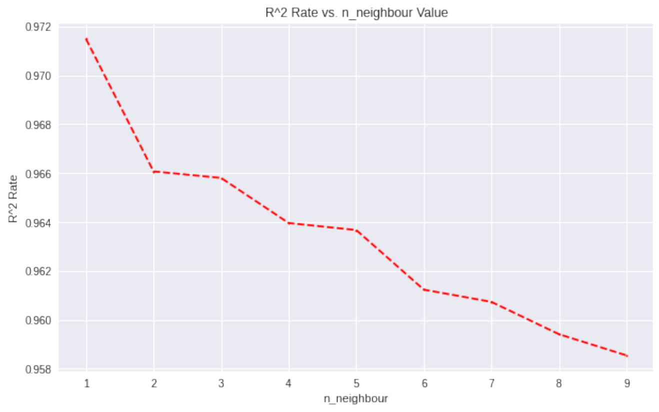

Fig. 8: Plotting the R2 against the number of neighbors hyperparameter during the KNN training phase.

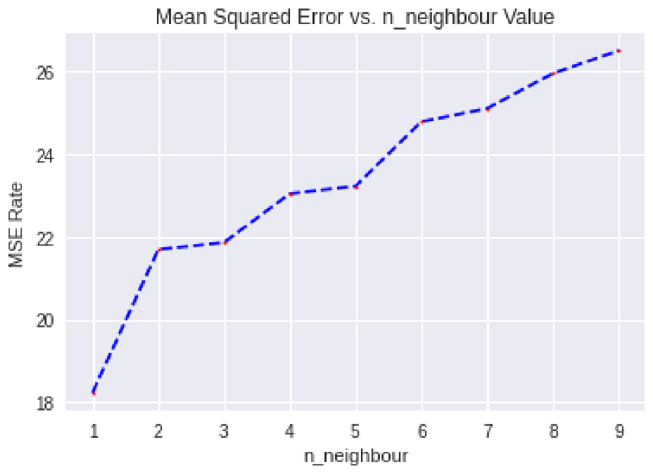

Fig. 9: Plotting the MSE against the number of neighbors hyperparameter during the KNN training phase.

TABLE V: The subset of results of the CNN model training without GPU.

	Training		Validation		
Epochs	MSE	MAE	MSE	MAE	Training time
28	0.0209	0.0981	0.0234	0.1000	1384s
29	0.0197	0.0948	0.0175	0.0820	1334s
30	0.0185	0.0917	0.0174	0.0835	1242s

TABLE VI: The subset of results of the CNN model training with GPU.

	Training		Validation		
Epochs	MSE	MAE	MSE	MAE	Training time
48	0.0247	0.1093	0.0266	0.1094	43s
49	0.0238	0.1075	0.0239	0.0988	43s
50	0.0236	0.1063	0.0258	0.1113	43s

TABLE VII: The final results of all the models.

Model	R2	MSE	MAE
OLS	0.609	143.605	9.2517
CNN	0.9780	0.0264	0.1122
KNN	0.9715	18.2278	0.9248
Decision Tree	0.9992	0.4816	0.0186
Random Forest	0.9999	0.0576	0.0019

According to the results, the CNN model showed the overall best performance in terms of MSE whereas the OLS has shown the least performance compared to all five models with very high MSE value. However, when considering the computational time, the Random Forest can be considered as the best model, as it gives the highest R2 value, the second-lowest MSE, and the lowest MAE compared to all the other models.

V. Conclusion

Predicting the temperature of PMSMs is important to gain good performance of electric vehicles and also to save temperature-sensitive components of the motor. However, the utilization of expert knowledge and technological approaches to monitor the temperature is too costly. Hence, instead of using traditional thermal estimations, it is possible to precisely and efficiently predict motor temperature with different machine learning and deep learning techniques. This study has used five such models to predict four different types of temperatures in PMSMs. Through performing the hyper-parameter tunings and conducting different pre-processing techniques, it was possible to gain achievable results mainly for two models; the CNN and Randon Forest. While the Decision Trees and KNN show moderately good results with higher goodness of fit values and low mean absolute error values, the OLS has shown comparatively poor results with a very high Mean Squared Error value. Finally, by considering the deep learning capabilities and the expandability of the model, the CNN can be selected as the best model for predicting the permanent magnet motor temperature.

VI. Future Work

Due to the difficulty of collecting a comprehensive dataset, most of the previous studies have used the same set for analyzing the performance of machine learning and deep learning model performance for predicting the temperature of PMSMs. Hence, it is still a problem how well different supervised learning models may perform on a different dataset with different motors of the same manufacturer or even among different manufacturers. For this, future studies will need a novel dataset with diversity.

Further, this study and some previous studies have only used pure deep learning techniques, such as CNN, LSTM, ResNet. Hence, the use of hybrid models, such as a combined CNN and LSTM version to gain the advantages of many algorithms, such as analyzing relationships among time-series data through LSTM's memory function would be beneficial.

Acknowledgment

We thank Paderborn University and Dr. Ing Joachim Bocker for the electric temperature data.

References

[1] W. Kirchgässner, O. Wallscheid, and J. Böcker, "Data-Driven Permanent Magnet Temperature Estimation in Synchronous Motors with Supervised Machine Learning: A Benchmark," IEEE Trans. Energy Convers., vol. 36, no. 3, pp. 2059–2067, 2021.

[2] W. Kirchgassner, O. Wallscheid, and J. Bocker, "Estimating Electric Motor Temperatures with Deep Residual Machine Learning," IEEE Trans. Power Electron., vol. 36, no. 7, pp. 7480–7488, 2021.

[3] O. Wallscheid, W. Kirchgassner, and J. Bocker, "Investigation of long short-term memory networks to temperature prediction for permanent magnet synchronous motors," Proc. Int. Jt. Conf. Neural Networks, vol. 2017-May, pp. 1940–1947, 2017.

[4] W. Kirchgassner, O. Wallscheid, and J. Bocker, "Deep residual convolutional and recurrent neural networks for temperature estimation in permanent magnet synchronous motors," 2019 IEEE Int. Electr. Mach. Drives Conf. IEMDC 2019, pp. 1439–1446, 2019.

[5] H. Guo, Q. Ding, Y. Song, H. Tang, L. Wang, and J. Zhao, "Predicting temperature of permanent magnet synchronous motor based on deep neural network," Energies, vol. 13, no. 18, 2020.

[6] R. Le, K. He, and A. Hu, "Motor Temperature Prediction with K-Nearest Neighbors and Convolutional Neural Network," pp. 3–9, 2019.

[7] K. Anuforo and V. Milosavljevic, "Temperature Estimation in Permanent Magnet Synchronous Motor (PMSM) Components using Machine Learning," 2020.

[8] R. Savant, A. A. Kumar, and A. Ghatak, "Prediction and Analysis of Permanent Magnet Synchronous Motor parameters using Machine Learning Algorithms," Proc. 2020 3rd Int. Conf. Adv. Electron. Comput. Commun. ICAECC 2020, 2020.

[9] O. Soprun et al., "Forecasting temperatures of a synchronous motor with permanent magnets using machine learning," CEUR Workshop Proc., vol. 2631, pp. 95–120, 2020.

[10] K. Bingi, B. R. Prusty, A. Kumra, and A. Chawla, “Torque and Temperature Prediction for Permanent Magnet Synchronous Motor Using Neural Networks,” 3rd Int. Conf. Energy, Power Environ. Towar. Clean Energy Technol. ICEPE 2020, 2021.

[11] D. Mukherjee, S. Chakraborty, P. K. Guchhait, and J. Bhunia, “Application of Machine Learning for Speed and Torque Prediction of PMS Motor in Electric Vehicles,” 2020 IEEE Int. Conf. Converg. Eng. ICCE 2020 - Proc., pp. 129–133, 2020.

[12] A. Bilal, A. Waheed, and M. H. Shah, “A comparative study of machine learning algorithms for controlling torque of permanent magnet synchronous motors through a closed loop system,” 2019 2nd Int. Conf. Latest Trends Electr. Eng. Comput. Technol. INTELLECT 2019, pp. 1–6, 2019.

[13] Cen, Y., Zhang, C., Cen, G., Zhang, Y., Zhao, C. (2020). The Temperature Prediction of Permanent Magnet Synchronous Machines Based on Proximal Policy Optimization. Information, 11(11), 495.

[14] Cai, Y., Cen, Y., Cen, G., Yao, X., Zhao, C., Zhang, Y. (2021). Temperature Prediction of PMSMs Using Pseudo-Siamese Nested LSTM. World Electric Vehicle Journal, 12(2), 57.

[15] Joachim Böcker, “Electric Motor Temperature,” 2019. [Online]. Available: https://www.kaggle.com/wkirgsn/electric-motor-temperature. [Accessed: 26-Jan-2022].

[16] “Ordinary least squares regression: Python Data Science,” Data Science. [Online]. Available: http://net-informations.com/ds/mla/ols.htm: :text=Machine

[17] 1.10. Decision Trees. (n.d.). Scikit-Learn. Retrieved March 20, 2022, from https://scikit-learn.org/stable/modules/tree.html

[18] Introduction to Random Forest in Machine Learning. (n.d.). Engineering Education (EngEd) Program — Section. Retrieved March 20, 2022, from https://www.section.io/engineering-education/introduction-to-random-forest-in-machine-learning/