PREDICTING PERMANENT MAGNET RESISTANCE OF ELECTRIC MOTOR USING MACHINE LEARNING

Under the guidance of

Sandeep Naik

Team details:

Team ID : LTVI2024TMID17400

Team size : 4

Team leader : Makkena Akhila

Team member: Karra Santhi Kumari

Team member: Shaik Nazma

Team member: Mande Benhar

CONTENTS:

Abstract:

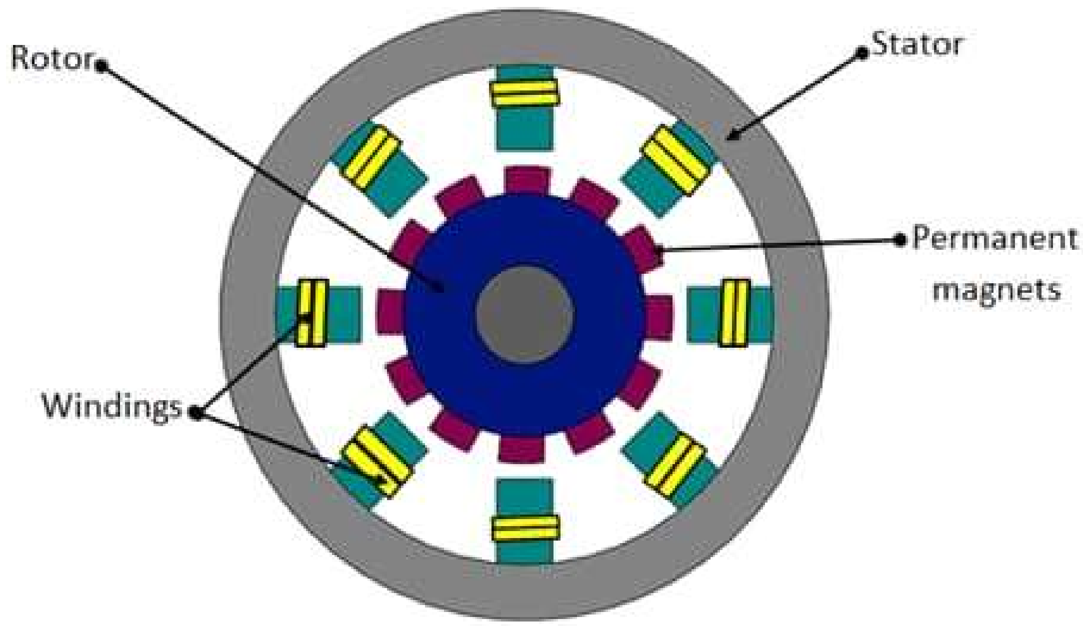
The heat loss and cooling modes of a permanent magnet synchronous motor (PMSM) directly affect the its temperature rise. The accurate evaluation and prediction of stator winding temperature is of great significance to the safety and reliability of PMSMs. In order to study the influencing factors of stator winding temperature and prevent motor insulation ageing, insulation burning, permanent magnet demagnetization and other faults caused by high stator winding temperature, we propose a computer model for PMSM temperature prediction. Ambient temperature, coolant temperature, direct-axis voltage, quadrature-axis voltage, motor speed, torque, direct-axis current, quadrature-axis current, permanent magnet surface temperature, stator yoke temperature, and stator tooth temperature are taken as the input, while the stator winding temperature is taken as the output. A deep neural network (DNN) model for PMSM temperature prediction was constructed. The experimental results showed the prediction error of the model (MAE) was 0.1515, the RMSE was 0.2368, the goodness of fit (R^2) was 0.9439 and the goodness of fit between the predicted data and the measured data was high. Through comparative experiments, the prediction accuracy of the DNN model proposed in this paper was determined to be better than other models. This model can effectively predict the temperature change of stator winding, provide technical support to temperature early warning systems and ensure safe operation of PMSMs.

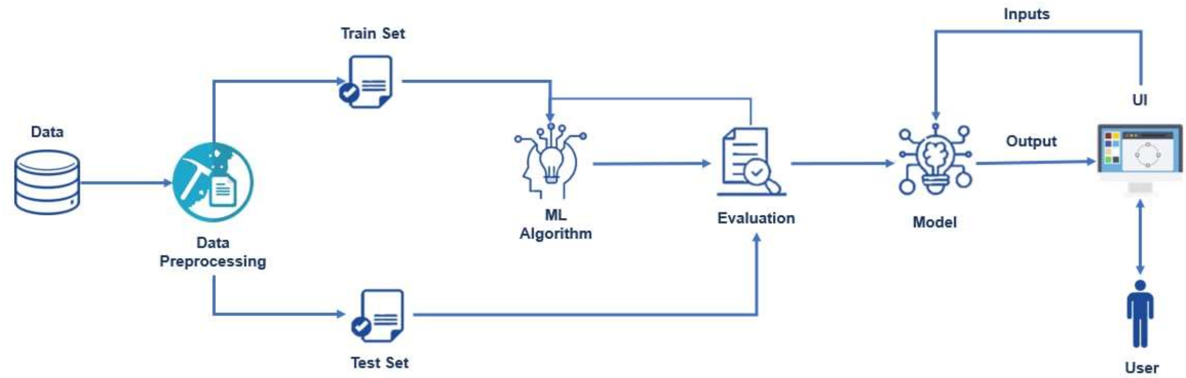
INTRODUCTION:

The permanent-magnet synchronous machine (PMSM) drive is one of the best choices for a full range of motion control applications. For example, the PMSM is widely used in robotics, machine tools, actuators, and it is being considered in high-power applications such as industrial drives and vehicular propulsion. It is also used for residential/commercial applications. The PMSM is known for having low torque ripple, superior dynamic performance, high efficiency, and high-power density.

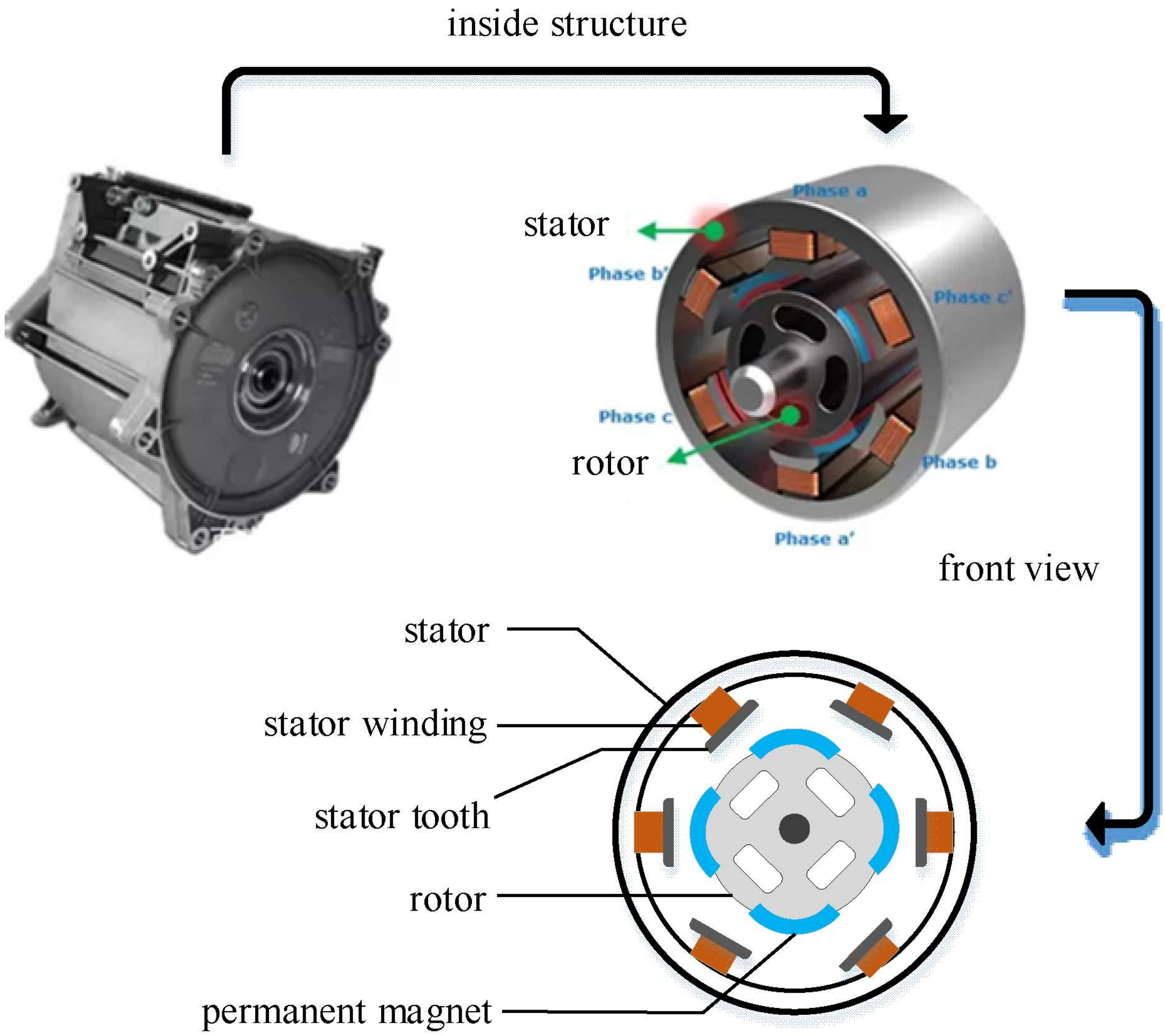
The task is to design a model with appropriate feature engineering that estimates the target temperature of a rotor.

In this project, we will be using algorithms such as Linear Regression, Decision Tree, Random Forest and SVM. We will train and test the data with these algorithms and select the best model. The best algorithm will be selected and saved in pkl format. We will be doing flask integration and IBM deployment.


Rotor
Stator
Permanent
magnets
Windings


Train Set
Inputs
UI
Data
ML
Algorithm
Evaluation
Model
Output
Data
Preprocessing
Test Set
User

VIZUALIZING AND ANALYZING THE DATA:



One of the biggest benefits of machine learning algorithms is expediting the data discovery process. Because they're designed to automatically improve their analysis as they scan information, machine learning tools are ideal for companies that have constant data streams. This real-time visualization can let you see what is happening exactly at all points of your production chain and understand how new factors affect existing data.

More importantly, these algorithms can help you identify outliers and unexpected outcomes. Combining data visualization and machine learning lets you establish baseline metrics for performance before scanning for any situation that breaks that mold. This can help you react faster, more effectively, and avoid downtime across your operations.

[2:59 pm, 22/04/2024] Shanthi: Data visualization is the practice of translating information into a visual context, such as a map or graph, to make data easier for the human brain to understand and pull insights from. The main goal of data visualization is to make it easier to identify patterns, trends and outliers in large data sets. The term is often used interchangeably with information graphics, information visualization and statistical graphics.

Data visualization is one of the steps of the data science process, which states that after data has been collected, processed and modeled, it must be visualized for conclusions to be made. Data visualization is also an element of the broader data presentation architecture discipline, which aims to identify, locate, manipulate, format and deliver data in the most efficient way possible.

Data visualization is important for almost every professional discipline. Teachers use it to display student test results, computer scientists to explore advancements in artificial intelligence (AI) and executives to share information with stakeholders. It also plays an important role in big data projects. As businesses accumulated massive collections of data, they needed a way to get an overview of their data quickly and easily. Visualization tools were a natural fit to provide useful information.

Visualization is central to advanced analytics for similar reasons. When a data scientist is writing advanced predictive analytics or machine learning algorithms, it's important to be able to visualize the outputs to monitor results and ensure that the models are performing as intended. Visualizations of complex algorithms are generally easier to interpret than numerical outputs.

PROGRAM:

Here is a Python code example using machine learning to predict permanent magnetic resistance of an electric motor:

```
import pandas as pd

import numpy as np

from sklearn.model_selection import train_test_split

from sklearn.linear_model import LinearRegression

from sklearn.metrics import mean_squared_error


# Load data

data = pd.read_csv('motor_data.csv')


# Preprocess data

X = data.drop('permanent_magnetic_resistance', axis=1)

y = data['permanent_magnetic_resistance']


# Split data into training and testing sets

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)


# Create and train linear regression model
```

```
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on test data
predictions = model.predict(X_test)

# Evaluate model performance
mse = mean_squared_error(y_test, predictions)
print(f'Mean Squared Error: {mse}')

# Use the model to make predictions on new data
new_data = pd.DataFrame({'input_feature1': [1, 2, 3],
'input_feature2': [4, 5, 6]})
prediction = model.predict(new_data)
print(f'Predicted permanent magnetic resistance:
{prediction}')
```

This code assumes you have a CSV file named motor_data.csv containing the data, with the permanent magnetic resistance as the target variable. The code preprocesses the data, splits it into training and testing sets, trains a linear regression model, makes predictions on the test data, evaluates the model's performance, and uses the model to make predictions on new data.

Note that this is just an example, and you may need to adjust the code to fit your specific data and problem. Additionally, you may want to explore other machine learning algorithms and techniques to improve the accuracy of your predictions.

Also, you can use other machine learning libraries like TensorFlow, PyTorch, Scikit-learn, etc. to implement the model.

You can also use other algorithms like:

- Decision Tree
- Random Forest
- Support Vector Machine (SVM)
- Gradient Boosting
- Neural Networks

It's important to note that the performance of the model will depend on the quality of the data, the features used, and the hyperparameters tuning.

DATA PROCESSING:

Involving quality control to supply chain management, inventory tracking and equipment maintenance, all requires Data Processing. By leveraging Data Analysing techniques, manufacturing companies optimises processes, improves efficiency and reduces costs.

Why is Data Preprocessing Important In Machine Learning?

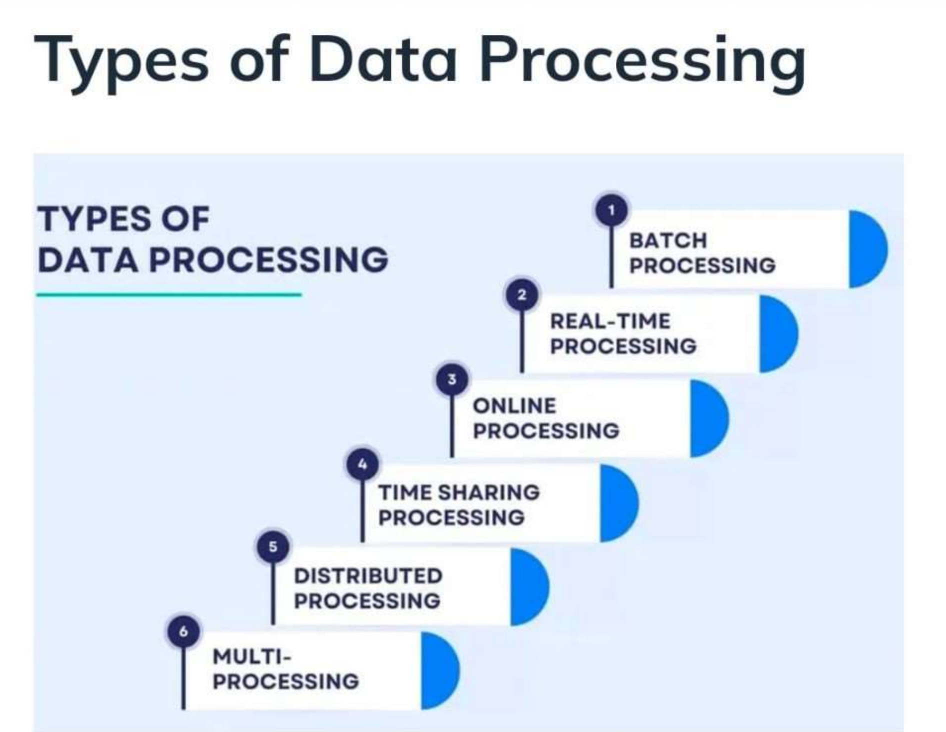
With the help of data pre-processing in Machine Learning, businesses are able to improve operational efficiency. Following are the reasons that can state that Data pre-processing is important in machine learning:

Data Quality: Data pre-processing helps in improving the quality of data by handling the missing values, noisy data and outliers. Accordingly, by addressing the issues, the dataset that is released as the outcome becomes more reliable and accurate. This helps in enabling better performance of the Machine Learning model.

Data Consistency: Data is sourced in the real world from multiple sources which results in these data involving various forms of inconsistencies in formats, units or scales. With the help of data pre-processing techniques, it is possible to ensure that data remains in the standardised and consistent format. It allows and helps in fair comparisons between features and reducing the biases in Machine Learning models.

Feature Engineering: The technique of data pre-processing allows feature engineering which involves creation of new features or transforming existing ones. It helps in improving model performance. With the need of selecting and constructing relevant features, Machine Learning models can help in capturing more meaningful patterns and relationships in the data.

Dimensionality Reduction: For Machine Learning models, high-dimensionality data can be quite challenging. With Data Pre-processing techniques like dimensionality reduction help in reducing the number of features whereby training the most important information. Consequently, it helps in alleviating the challenge of dimensionality and improving the model's efficiency.



Data Pre-processing includes different types with each serving different purposes and therefore catering to specific needs of Machine Learning. Some of the common types of Data Processing are:

Batch Processing: This type of Data Processing involves processing of large volumes of data in batches. The data collected over a long period of time are processes together as a batch. Batch processing is typically useful for non-real-time or offline cases, where the need for instant results is not important. The technique is often used for tasks such as Data Cleaning, aggregation, reporting and generating reports in batches.

Real-Time Processing: This type of immediate processing of data in real-time focuses on the data that arrives immediately and involves handling and analysing data in real-time. It helps organisations to receive instant results and is commonly used in applications where prompt decisions are to be made. Accordingly, these decisions are made on incoming data such as fraud detection, stock market analysis or real-time monitoring systems.

Online Processing: This type of Data Processing involves managing transactional data in real time and focuses on handling individual transaction. It includes transactions like recording sales, processing customer orders, or updating inventory levels. The systems are designed to ensure data integrity, concurrency and quick response times for enabling interactive user transactions. In online analytical processing, operations typically consist of major fractions of large databases. Therefore, today's online analytical systems provide interactive performance and the secret to their success is precomputation.

Time Sharing Processing: In this type of processing, the CPU of a large-scale digital computer helps in interacting with multiple users with the help of different programs simultaneously. With this type of processing, solving several discrete problems during the input/output process is possible because the CPU is faster than most peripheral equipment. This helps the CPU to address each problem sequence-wise. However, remote terminals have the impression that access to and retrieval from time-sharing system enables instant outcomes. This is because the solutions are immediately available as soon as the problem is fully centred.

Distributed Processing: With the help of distributed processing, it is possible to endure analysis of data across multiple interconnected systems or nodes. The type of Data Processing enables division of data and processing tasks among the multiple machines or clusters. The process therefore, helps in improving the scalability and fault tolerance. Distributed processing is commonly in use for big Data Analytics, distributed databases and distributed computing frameworks like Hadoop and Spark.

Multi-processing: It is the type of Data Processing in which two or more processors tend to work on the same dataset at the same time. In this process, multiple processors are housed within the same system. Consequently, data is broken down into frames whereby each frame is processed by two or more CPUs in a single computer system, working in parallel ways.

MODEL BUILDING:

These steps form the backbone to any machine learning process and knowing them will make your life much easier when trying to build ML models.

1. Data Collection

Machine learning requires training data, a lot of it. This data can either be labelled meaning Supervised Learning or not labelled meaning Unsupervised Learning.

Accuracy of the model depends on the quality and quantity of the data. The outcome of this step is generally a representation of data which will be used for training.

Using pre-collected data, by way of datasets from sites like Kaggle, UCI, etc. forms the basis of your Machine learning project. You may also collect data through user-surveys, analysis reports, trends, usage metrics, etc.

2. Data Preparation

We cannot work on raw data. Data needs to be processed by normalization, removing duplicates, errors and biases.

Visualising data can be helpful in searching for patterns and outliers to check if the data collected is right or if it contains missing values. This can be done using libraries like seaborn, matplotlib, etc. Visualize data to help detect relevant relationships between variables or class imbalances, or perform other exploratory analysis.

After performing data wrangling, we need to prepare the data for training. Cleaning of data is done that involves steps like removing duplicates, dealing with missing values, type conversions, correcting errors, normalizing the data, etc.

Not all the above steps are needed to be performed as it depends entirely on the data collected. Some datasets may not require data preparation at all while for some data preparation step takes majority of their ML model build time.

We can also Randomize data, which erases the effects of the particular order in which we collected and/or otherwise prepared our data. Later we can split the data into training, testing and evaluation sets

3. Choose a Model / Algorithm

The third step consists of selecting the right model. There are many models which can be used for many different purposes. Once the model is selected, it needs to meet the business goal.

We need to have an idea about the preparation the model requires along with its accuracy and scalability. Having a complex model does not mean a better model.

Common machine learning algorithms include Decision Trees, Random Forest, Linear Regression, Support Vector Machines (SVM), Logistic Regression, K-means, Principal Component Analysis (PCA), Naïve Bayes, and Neural Networks. Different algorithms need to be applied to different tasks, you need to choose the correct one for your use case.

4. Training the Model

Training a model forms the basis of machine learning. The goal is to use our training data and improve the predictions of our model.

Every cycle in training a model involves updating the weights and biases in each training step. We can use labelled sample data in case supervised machine learning and unlabelled sample data for unsupervised learning.

The goal of training is to evaluate and further improve our model accuracy and performance. Training happens in the form of iterations which is called a training step.

5. Evaluate the Model

After training the model comes evaluating the model. The larger the number of variables in the real world, the bigger the training and test data should be.

Performance metrics are used to measure the performance of the model. These include precision, recall, accuracy, specificity, etc.

The model is then tested against previously unseen data. The unseen data is meant to act as representative of model performance in the real world, but still helps tune the model (as opposed to test data, which does not).

A 70/30 split, or similar, is considered a good train/eval split, which depends on things like data availability, dataset features, domain, etc.

6. Parameter Tuning

The original model parameters need to be tested after evaluating your model. By increasing the training, it can lead to better results.

Parameter tuning is an experimental process and hence we need to define when to stop parameter tuning otherwise it will continue to tweak the model.

Hyperparameter tuning is an art and one that requires patience & experience. Once the model parameters are tuned it can give us better results. Some common hyperparameters include: number of training steps, learning rate, initialization values and distribution, etc.

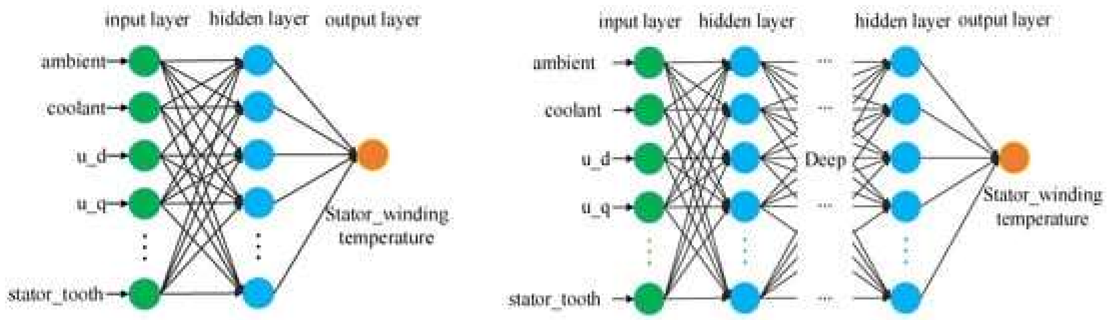
7. Make Predictions

After the processes of collecting data, preparing the data, selecting a machine learning algorithm, training the model and evaluating the model & tuning the parameters, we need to make predictions.

METHOD BASED ON DNN:

There are many factors that after the stator winding temperature of a PMSM. In this paper, 11 variables are regarded as input, including ambient temperature (ambient), coolant temperature (coolant), direct-axis voltage (u_d), quadrature-axis voltage (u_q), motor speed (motor speed), torque, direct-axis current (i_d), quadrature-axis current (i_q), permanent magnet surface temperature (pm), stator yoke temperature (stator_yoke) and stator tooth temperature (stator_tooth). Stator winding temperature (stator winding) is regarded as output. Considering the high dimension of the independent variables, a DNN model was chosen for the prediction.

DNN is an extension of an artificial neural network (ANN), with a structure that is similar to ANN but with a number of hidden layers [20,21,22,23]. Generally, neural networks that have two or more hidden layers can be regarded as a DNN. **FIG** shows the structures of ANNs and DNNs.



(a) (b)

Figure 1. Structures of an artificial neural network (ANN) and a deep neural network (DNN). (**a**) A simple example of an ANN. (**b**) A DNN has two or more hidden layers.

In this paper, the PMSM stator winding temperature prediction model based on a DNN has nine layers. The first layer is the input layer, and the number of its nodes is equal to the number of input variables X. The ninth layer (in other words, the last layer) is the output layer. The number of its nodes is equal to the number of output-dependent variable y, which equals 1. The layers from the second to the eighth are hidden layers, and each of them has 14 nodes, respectively. The nodes of the former layer are connected with each node of the latter layer, one by one, and there are no connections between nodes of the same layer. The activation function of a hidden layer is the ReLU function, and the activation function of the output layer is the tanh function. The loss function of the model is the mean squared error (MSE) function, and the back propagation algorithm is the Adam optimization algorithm, whose learning rate is set to 0.001. FIG shows the DNN model constructed in this paper.

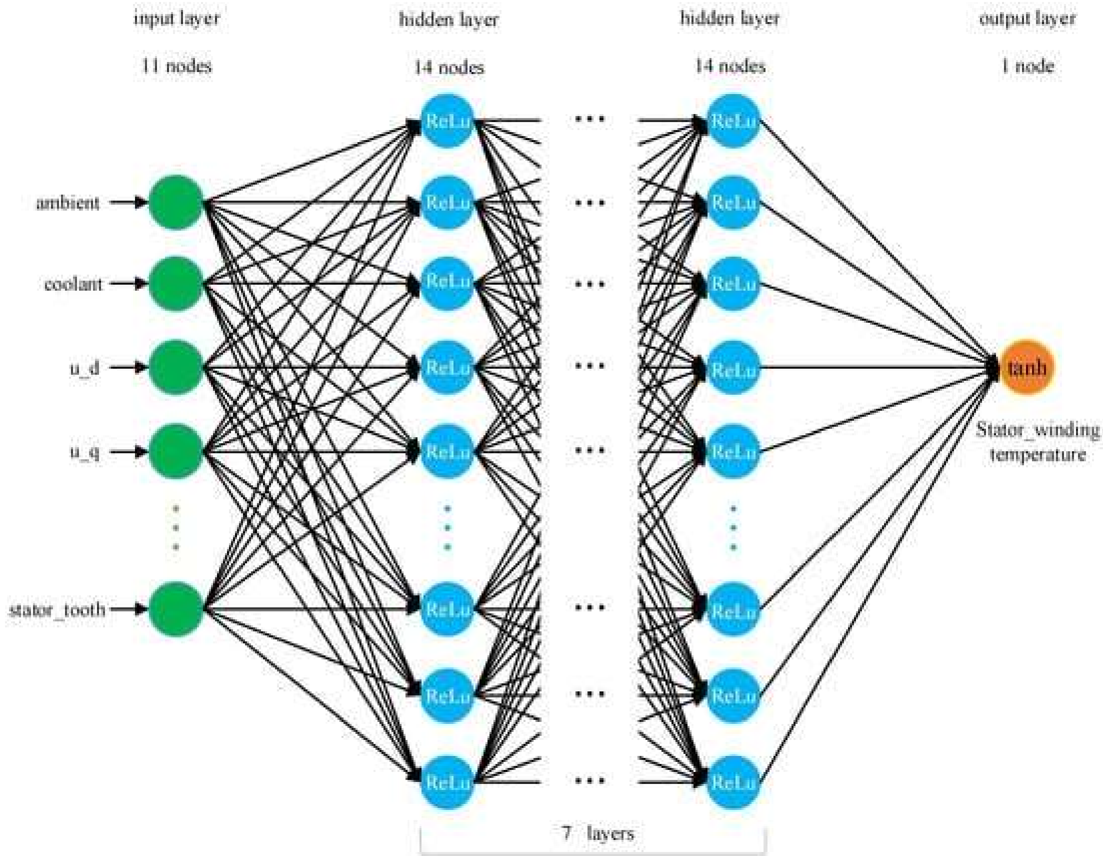


Figure 2. PMSM stator winding temperature prediction model.

The constructed neural network model was used to predict the PMSM stator winding temperature. FIG shows the prediction process. First of all, the data were standardized, then the standardized data were divided into five equal parts by means of five-fold cross validation. Each part was taken as the test set in turn, while the other four parts were used as the training set. The DNN model used the training set to fit the model while the test set was used to predict the stator winding temperature. The results of the DNN model were consolidated; thus, the 6000 pieces of predicted values were obtained. Together with the real values, the metrics of the DNN model could be calculated.

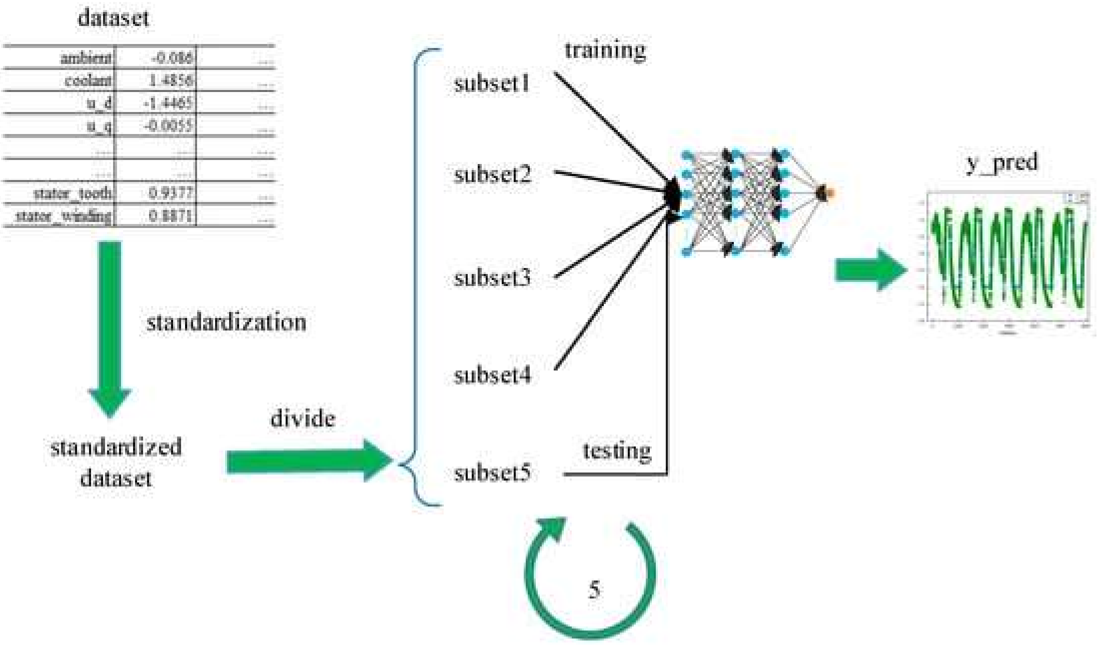


Figure 3. The process of predicting the PMSM stator winding temperature.

CONCLUSION:

The result shows that the prediction is much more accurate than previous methods and algorithms used for prediction and forecasting of temperatures in PMS motor. While traction drives are becoming more dependent on PMSMs, monitoring of latent high dynamic temperatures within the EV motor becomes a critical issue.