IAJIT

Accurate Output Power Prediction in TENG devices using Artificial Intelligence: A Comparative Analysis

Journal Name :	International Arab Journal of Information Technology
Manuscript ID :	259-1696666870
Manuscript Type :	Review Article
Submission Date :	07-Oct-2023

Abstract : The utilization of AI algorithms in the application of processing the experimental data obtained manually relevant to engineering and physics has gained significant attention in the emergence of the field artificial intelligence (AI). Consequently, in the area of triboelectric nanogenerators (TENG) may also adopt AI technology. The analysis of how structural characteristics impact output performance in physical tests can present occasional challenges and limitations. TENGs' structure varies over a narrow range, which makes experiment control challenging. However, it is impractical to conduct experiments encompassing all possible parameter combinations to establish definitive experimental guidelines. For anticipating the performance of triboelectric nanogenerators' outputacross numerous structures, configurations.This study introduces a novel AI algorithm model based on deep neural networks specifically designed for TENGs. The results reveal that the DNN model accurately predicts the expected output power levels for TENGs with the CS mode structure, aligning with the physical experimental data. Subsequently, the DNN model is utilized to predict the output power of TENGs within the parameter range that was not tested in the CS mode structure. This will undoubtedly aid researchers in their analysis of the data's law over a larger range of parameter values, leading to a more accurate experimental law.The utilization of AI algorithms in the fields of engineering and physics has experienced a substantial growth in recent years.AI technology can be helpful in analyzing experimental data, predicting outcomes, and discovering patterns in complex systems.

Keywords : Triboelectric nanogenerators, Regression, DNN, output power

Accurate Output Power Prediction in TENG devices using Artificial Intelligence: A Comparative Analysis

Abstract: The utilization of AI algorithms in the application of processing the experimental data obtained manually relevant to engineering and physics has gained significant attention in the emergence of the field artificial intelligence (AI). Consequently, in the area of triboelectric nanogenerators (TENG) may also adopt AI technology. The analysis of how structural characteristics impact output performance in physical tests can present occasional challenges and limitations. TENGs' structure varies over a narrow range, which makes experiment control challenging. However, it is impractical to conduct experiments encompassing all possible parameter combinations to establish definitive experimental guidelines. For anticipating the performance of triboelectric nanogenerators' outputacross numerous structures, configurations.This study introduces a novel AI algorithm model based on deep neural networks specifically designed for TENGs. The results reveal that the DNN model accurately predicts the expected output power levels for TENGs with the CS mode structure, aligning with the physical experimental data. Subsequently, the DNN model is utilized to predict the output power of TENGs within the parameter range that was not tested in the CS mode structure. This will undoubtedly aid researchers in their analysis of the data's law over a larger range of parameter values, leading to a more accurate experimental law.The utilization of AI algorithms in the fields of engineering and physics has experienced a substantial growth in recent years.AI technology can be helpful in analyzing experimental data, predicting outcomes, and discovering patterns in complex systems.

Keywords: Triboelectric nanogenerators, Regression, DNN, output power

1. Introduction

Numerous efforts are currently underway to develop alternative energy production technologies that can meet the increasing demand for environmentally friendly and sustainable power sources. The extraction of plentiful natural energy from the environment, such as wind energy [3], thermal energy [2], solar energy [1], and mechanical energy [4], has grown in popularity. Traditional energy harvesting systems, such as wind turbines, thermoelectric generators and solar cells, largely depend upon certain environmental parameters for reliable functioning, such as steady sunshine, wind power and temperature [5]. As a result, there is an urgent need to create more ecologically friendly and effective energy acquisition techniques. Recent advancements in the technologies such as Internet of Things (IoT),AI and cloud computing have given rise to user-centric functional wearable electronics, offering new opportunities in this domain. As a result, lightweight, compact, and long-term power sources are always in demand [6][7][8]. These energy sources must also be adaptable, ecologically benign, lightweight, and long-lasting [9].Mechanical energy harvesting has gained attention as a very good method in creating self-powered micro systems which are integrated and wearable. It offers a

solution to the drawbacks associated with traditional energy storage systems like capacitors and batteries, including their limited capacity, short lifespan, frequent recharging, and safety concerns. By harnessing mechanical energy, these systems can overcome these limitations and provide a sustainable and reliable power source for various applications. As a result, there has been a boom in interest in micro/nano scale energy harvesting technology. There has been so much exploration of different technologies in the mechanical energy harvesting realm, including triboelectric nanogenerators (TENGs) [11], electromagnetic micro generators (EMGs) [10] and piezoelectric nanogenerators (PENGs) [10]. Among these, TENG has showed particular potential in turning low-frequency mechanical energy into electricity.The research in this area has been dedicated to achieving high production performance, design that is very lightweight, cost-effectiveness, appropriate selection of the material, design simplicity, and environmental sustainability [13]. The goal is to optimize TENGs for efficient energy generation by exploring different materials, designs, and manufacturing techniques, while also considering their impact on the environment.

The operation of a TENG device is mainly depends on electrostatic induction and contact electrification principles[14][10]. The very fundamental working principle of TENG involves the interaction between two different materials, which become electrically charged upon contact [15]. When these material surfaces come into contact, there is a transfer of charges, leading to the creation of surface charges of both positive and negative in equal amounts. Charge exchange's efficiency is determined by the materials' position that is present in the triboelectric series. This is used to classify the materials; this classification is done by the ability of the device how capable it is while gaining or losing the electrons [16].

Lot of investigations have been carried out by researchers for innovative triboelectric materials for improving energy conversion efficiency while lowering the making cost and complexity of TENG devices. A growing trend is the utilization of waste materials for generating electricity, which contributes to environmental pollution reduction. Several studies have been published on the application of waste materials as novel triboelectric materials for energy harvesting.These waste materials include discarded milk cartons [19], eggshell membranes [20], tea leaves [21], wastages like fish scales [23], used plastic bags [27] and wastage from sugar cane [25]. By repurposing these waste materials for TENG applications, researchers aim to minimize environmental impact while achieving efficient energy harvesting.

Artificial intelligence is the trending technology that has been used in almost every field to open new doors for the better ways to solve real world problems. Deep learning methods come in handy to analyse complex data to extract important features from the given data. With this technology solving complex problems became very effective. Generally deep learning is widely used in the computer-vision applications, NLP which is Natural language processing applications in computer science field but it is not just limited to that field. Deep learning has large scope in the interdisciplinary research also, for instance in the areas of Protein structure prediction[28] , drug discovery and development[29][30], solving the mechanical materials computation problems [31], TENG-based self-powered sensors [32], design of de novo proteins which do not exist yet [33], Liplating on graphite electrode [34]. Jiao Pengcheng [35] proposed that deep learning approach can take crucial role to obtain optimal design of nanogenerators and their predictive tuning in the electromechanical performance. Y. Zhou [36] proposed that huge amount of data can be collected using TENG which can be further used to analyse using artificial intelligence to build deep learning models that can predict our output variable.

Deep learning techniques have shown great potential in various applications, including fault diagnosis of rotating machinery[37]. Deep learning models, known for their capacity to learn from extensive datasets, offer a valuable approach for extracting significant features and patterns from raw sensor signals, thereby enhancing the accuracy and efficiency of fault diagnosis. In the context of rotating machinery, the application of Convolutional Neural Networks (CNNs) proves particularly effective. CNNs excel at capturing spatial features within sensor signals, enabling them to identify and analyze characteristics like vibration patterns' shapes and frequencies. This capability makes CNNs well-suited for such applications.In this context Recurrent Neural Networks can also be useful since they can capture the temporal patterns in the sensor signals over time. Deep Belief Networks (DBNs) can combine the strengths of CNNs and RNNs and can be useful for detecting complex faults that involve both spatial and temporal patterns. The quality of deep learning model is dependent on a number of things. Firstly, the training data that we should be large enough for training our developed model. The training data quality should be very high which significantly impact the model's ability to learn and generalize effectively. Sufficient and diverse data that

accurately represent the problem domain contribute to better performance. Secondly, the selection and configuration of the model architecture and parameters play a vital role. Choosing an appropriate architecture, such as the number and type of layers, activation functions, and connections, can greatly influence the capacity of the model to extract relevant features and make accurate predictions. Additionally, setting the right values for hyper parameters, such as learning rate, batch size, and regularization, is crucial for achieving optimal performance. It is essential to thoroughly consider these factors when developing and training deep learning models to ensure their effectiveness in the intended application. Therefore, it's crucial to carefully design and implement deep learning models for fault diagnosis applications to ensure their effectiveness and reliability.

The proposed approach in the paper is an interesting combination of deep learning and the K-nearest neighbours (KNN) algorithm. In this approach, KNN is first used to cluster similar load profiles, which are then make use of to train the neural network to predict future loads[39]. The idea behind this approach is that load profiles that are similar to each other are likely to exhibit similar load patterns in the future. By clustering similar load profiles together, the KNN algorithm can identify patterns in the data that are not immediately apparent. The DNN is then trained on these similar load profiles to make accurate load forecasts.One potential benefit of this approach is that it may be more effective at handling non-linear load patterns than traditional forecasting methods, such as regression models. In contrast, the KNN algorithm can effectively identify similar load patterns that might go unnoticed by conventional methods.Overall, this approach represents an interesting application of deep learning and KNN in load forecasting, and it will be interesting to see how it performs in practice.

We have built different ML models and DL models and trained those models with our dataset which is obtained through experimental results. Then the models are tested on the test dataset to check the performance of our models. And we have considered the most common performance metrics for the regression analysis like “Mean Square, Mean absolute, and Root mean square errors”, “R2-score”. In deep learning method we have developed Deep Neural network with different number of layers and tested with different optimizers to see which model is giving good results that predicts the output power with minimal error. The problems in TENG design such as high cost and inefficiency can be solved with this type of prediction models and provide good solution for TENG design. Triboelectric series including present work is given in Fig. 1

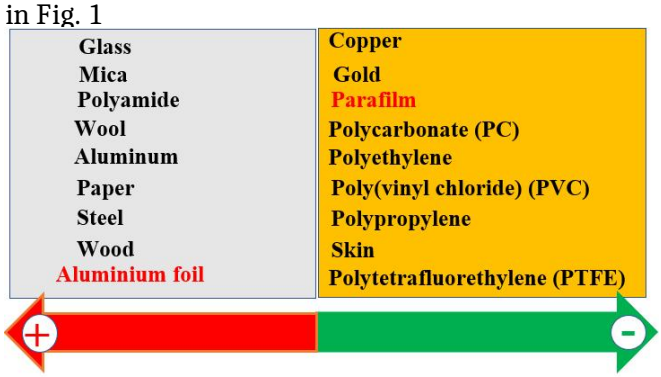

Fig 1. Triboelectric series including present work

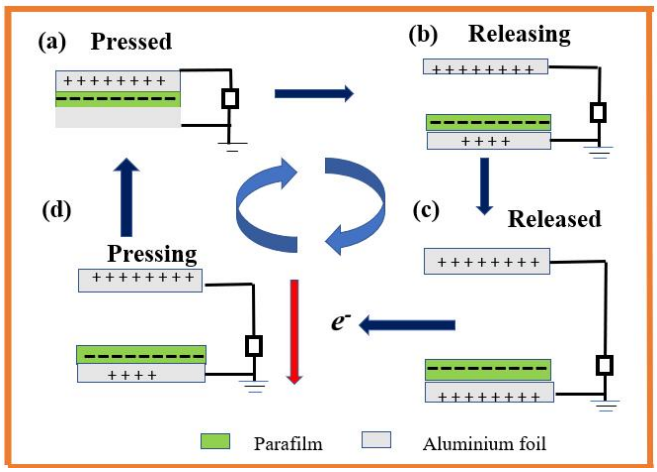

Fig 2. TENG working mechanism

1.1 Theoretical principles of APA TENG

In summary, the initial state of the triboelectric nanogenerator (TENG) is characterized by a balanced condition with no transfer of charge or electric potential. However, when the parafilm comes into contact with the top aluminium electrode, there is a phenomenon of charge exchange. This causes a negative charge to accumulate on the parafilm and an equivalent amount of positive charge to accumulate on the aluminium electrode. So, because of this an electrical potential difference is created, leading the TENG to lose its equilibrium. Charges may be held on the surface of the parafilm, allowing charges to be moved in and out between the electrodes to restore balance. Through periodic interaction and separation of the TENG. On the parafilm, Triboelectric charges present on it cause the passage of electrons that are free on the aluminium electrodes. Electrical output is generated in the external circuit because of electron flow. The TENG working mechanism is given in Fig. 2.

2. Method:

2.1 Data pre-processing:

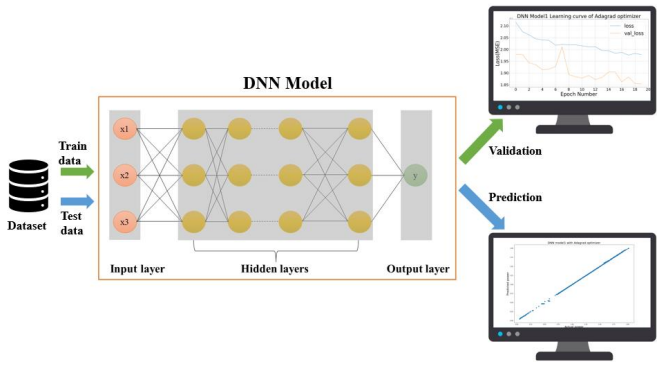

Fig 3. Experimental Setup

In order to build a reliable neural network model, it is important to have a sizable amount of experimental data that is extracted from reliable references or sources[40][41]. So, a good amount of data needs to be collected through experimental results.The experimental setup is shown in Fig. 3. And in our dataset, we have removed the power density and classifier features as they are not contributing much in predicting the output power. After this our dataset is split into 2 parts for making it useful for testing as well so 30% of data is reserved for training and 30% for testing. The training data which is 70% of original data is made use for training our model and test data is the untrained data which our developed model doesn't come across used to test how our model is performing the output power prediction.The layered structure flowchart is shown in below figure Fig. 4.

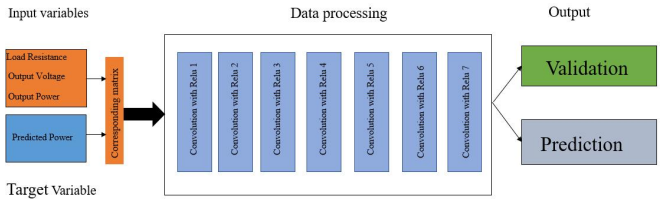

Fig.4Layered structure flow chart for prediction of the output power

2.1. Construction of proposed model and optimization:

By making use of our dataset prepared from experimental results, we can predict the output power by using regression analysis. We have 2 options to solve this regression problem which are ML and DL methods. In our work we analysed both the techniques.The flowchart of overall model is shown in Fig. 5.

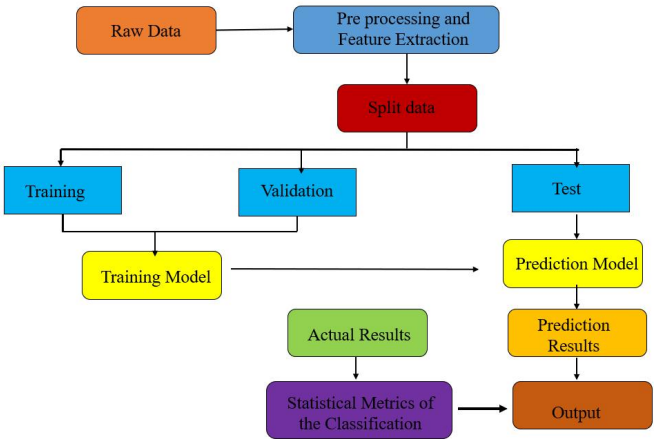

Fig.5Flow chart of overall model from experimental data.

We have proposed a deep neural network architecture. Our proposed model contains one input layer that takes input features data, some hidden layers and one output layer to predict the output variable in our case it is output power. Our model makes use of several weight coefficient vectors w_i where,i = 1, 2, 3, …. N. We need to make use of'b'and 'x' to perform the sequence of linear operations, where 'b' is bias vector and 'x' is an input vector, which is given by$y^{[l]}=w^{[l]} *x^{[l]}+ b^{[l]}$. We have used one of the most widely used activation function which is ReLU.The working of ReLU activation function is mathematically given as g(x) = max (0, x)data from the input layer passes through the hidden layers gives the result at the end of output layer, it is given as$x^{[l]} = g^{[l]}(y^{[l]})$. The loss function L(θ) after completing the forward propagation can be obtained as, $L(\theta) = \frac{1}{N}\sum_{j=1}^{N}(y_j - y_j)^2$.

The model propagation starts at this point through the input of $dx^{[l]}$.we get the output as$dx^{[l-1]}$, $dw^{[l]}$, $db^{[l]}$. Our model parameters get updated in following fashion,

$dx^{[l-1]} = w^{[l]T}\cdot dy[l]$

$dw^{[l]} = dy^{[l]}\cdot x^{[l-1]}$

$db^{[l]} = dy^{[l]}$

$dy^{[l]} = w^{[l+1]T} .dy^{[l+1]}\cdot g^{[l]}(y^{[l]})$.

This workflow is shown in Fig. 6. Obtained results are used for training and checking the overall performance of the model using different performance metrics.The main performance metrics for regression analysis are MSE, RMSE, MAEand R2-score. We have obtained these metrics values for all the models developed and compared the metric values.

a. Mean Square Error (MSE):

The Mean Square Error (MSE) is a common statistic for assessing regression models. The average

squared difference between expected and actual values is calculated under MSE. The disparities between expected and the actual values are squared, summed up, and then divided by the number of data points to produce MSE. The resultant number is the average of the squared differences.

Mathematically, formula for MSE can be expressed as:

$$MSE = \frac{1}{N} \sum_{j=1}^{N} (y_j - y_j)^2 \quad -(1)$$

MSE is a measure of a regression model's prediction accuracy. A lower MSE value indicates that the model is more accurate in predicting real values. The model seeks to provide a more exact fit to the data by minimising the squared discrepancies between anticipated and actual values.

b. Root Mean Square Error (RMSE) :

RMSE, which stands for Root Mean Square Error. It is a generally employed measure for assessing precision of a regression problem. It quantifies the disparity between the predicted values and the original values. The square root of the average of the squared discrepancies between the predicted values and actual values gives the RMSE value. It is given in mathematical form below,

$$RMSE = \sqrt{\frac{1}{N} \sum_{j=1}^{N} (y_j - y_j)^2} \quad -(2)$$

RMSE is a useful metric for comparing regression models and tracking performance over time

.

c. Mean Absolute Error (MAE):

MAE, or Mean Absolute Error, is a statistic often used in regression analysis to calculate the average absolute difference between a dependent variable's predicted and actual values. It is computed by calculating the absolute difference between each anticipated and actual number and then averaging these differences. MAE provides a straightforward and interpretable measure of the magnitude of errors in the predictions. Mathematically, the formula for MAE is:

$$MAE = \frac{1}{N} \sum_{j=1}^{N} |y_j - y_j| \quad -(3)$$

MAE is a popular metric because it is easy to understand and interpret. A lower MAE indicates that the model is more accurate.

d. R2-Score:

R2-Score is given as R2, which reflects the amount of variation explained by the other independent factors for the dependent variable. R^2 gives the insight about how much one variable's variance can explain about the second variable's variance. To understand this let us take an example, if the R^2 of a model is 0.8, then nearly 80% of the observed variation can be clearly explained by the features of that model. It is given in mathematical form as below,

$$R^2 = 1 - \frac{RSS}{TSS} \quad -(4)$$

Where, Sum of squares of Residuals (RSS) = $\sum_{j=1}^{N} (y_j - y_j)^2$

Total Sum of Squares (TSS) = $\sum_{j=1}^{N} (y_j - y)^2$

R2 score is commonly used in linear regression analysis, but it can also be applied to other types of regression models.

2.2.1 Machine learning Techniques:

To analyse and model the connection between dependent and independent variables in the machine learning technique, we have a statistical tool known as Regression analysis. It aims to uncover patterns and quantify the relationship between the variables, allowing for the prediction of the dependent variable based on the independent variables. Here in our dataset dependent variable is the output power which is our target variable to predict how it changes with respect to the changes in independent variables like output current, output voltage and load resistance. Regression is a supervised learning method to draw the correlation between dependant and independent variables. Coming to regression analysis a graph is plotted such that regression line fits the data points on the scatter plot to the best possible way which is used to show overall trend of the data and the how strong the dependent and independent variables are related. In the target-predictor graph we consider the vertical distance. That distance should be measured between the datapoints in the plot and the drawn regression line. So this distances are maintained to be as minimum as possible which indicates the residual error. There are so many types of regressions used for different scenarios. We have tested a few of those methods in our problem to predict the output power. They are:

1. Linear Regression
2. Decision Tree Regression
3. XGB Regression
4. Random Forest Regression

5. Support Vector Regression

I .Linear regression:

It is a technique commonly employed for predictive analysis. It is used to establish a relationship between continuous variables and can be applied to solve regression problems, enabling the prediction of a specific quantity. The method establishes a relationship between the independent, dependent variables. This is a linear relationship between those variables. It enables us for analyzing and prediction of the target variable. In simple linear regression only one input variable is present, but in our case, there are more than one input variable which are output voltage, output current and Load resistance, so our regression problem comes under multiple linear regression.

But when we have more than one independent variable, we need to useMultiple linear regression. In multiple linear regression, the relationship formula becomes:

$$Y = b_0 + b_1X_1 + b_2X_2 + ... + bnXn$$

The coefficients are determined using the least squares approach, that minimises the total of the squared deviations between the actual and expected values of the dependent variable from the regression equation.

II. Decision Tree Regression:

It is one of the supervised learning-basedalgorithms. The advantage with this method is that it can be used for solving both the problems that require classification to be performed oreven for regression problems as well. And this is applicable to both numerical and categorical data. Since our problem is related to regression with numerical data this method can be considered to solve our regression problem. Just like the name of the regression technique it is a structure similar to a tree where each internal node indicates the decisive condition for an attribute, and each branch indicate the result of the decision, and each leaf node indicates the conclusive decision. This regression observes the important features embedded in the existing data and utilize it for training the ML model as in a shape of a tree to predict the output values.

III. XGB regression:

In Regression problemsthe output variable values are continuous values. For building supervised regression models XGBoost is a decent approach. From the XGBoost objective function and the base learners decide our method's performance. So, objective function includesa loss function and the regularization termgives the insight about thedeviation of actual output power and predicted output power. XGBoost is an ensemble learning method, inensemble learning individual models are trained and combined to obtain a single prediction.So, in XGBoost when all the predictions are combined, bad predictions get cancelled and better one adds up to make the final good predictions.

IV. Random forest regression:

Random forest is proficient of both classification and regression tasks.An ensemble learning technique called random forest regression combines numerous decision trees and forecasts the ultimate result using the average of each tree's output. Base models are the combined decision trees, and their formal representation is g(x)= f0(x)+ f1(x)+ f2(x)+.....

Aggregated decision trees are operated in parallel by Random Forest.So they do not interact with one another. By generating random subsets of the dataset, Random Forest regression enables us to prevent Overfitting in the model.

V. Support vector regression:

Support Vector Machine is a supervised learning approach that may be applied to both classification and regression issues. So, it is known as Support Vector Regression(SVR) if we use it to regression-related issues. An effective regression method for continuous data is support vector regression. The basic objective of SVR is to take into account as many datapoints as possible within the boundary lines, and the hyperplane which is the best fit line must include as many datapoints as possible.

2.2.2 Deep learning Techniques:

The main problem with simplelinear regression is that it can only learn the linear relationship between the features and target variable and when there is complex non-linear relationship then the simple linear regression cannot learn. So, to overcome this problem we need a different technique where neural networks come into picture. Artificial neural network is the approach comes under deep learning which has the ability to learn the complex relationship by making use of activation function in each layer of our neural network. Our DNN model summary with the layers used is shown in below table.

DNN with 5 layers:

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 32)	128
dense_1 (Dense)	(None, 16)	528
dense_2 (Dense)	(None, 8)	136
dense_3 (Dense)	(None, 4)	36
dense_4 (Dense)	(None, 1)	5

Total params: 833
Trainable params: 833
Non-trainable params: 0

DNN with 7 layers:

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 128)	512
dense_1 (Dense)	(None, 64)	8256
dense_2 (Dense)	(None, 32)	2080
dense_3 (Dense)	(None, 16)	528
dense_4 (Dense)	(None, 8)	136
dense_5 (Dense)	(None, 4)	36
dense_6 (Dense)	(None, 1)	5

Total params: 11,553
Trainable params: 11,553
Non-trainable params: 0

DNN with 9 layers:

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 512)	3072
dense_1 (Dense)	(None, 256)	131328
dense_2 (Dense)	(None, 128)	32896
dense_3 (Dense)	(None, 64)	8256
dense_4 (Dense)	(None, 32)	2080
dense_5 (Dense)	(None, 16)	528
dense_6 (Dense)	(None, 8)	136
dense_7 (Dense)	(None, 4)	36
dense_8 (Dense)	(None, 1)	5

Total params: 178,337
Trainable params: 178,337
Non-trainable params: 0

DNN with 11 layers:

Model: "sequential"

Layer (type)	Output Shape	Param #
dense (Dense)	(None, 1024)	4096
dense_1 (Dense)	(None, 512)	524800
dense_2 (Dense)	(None, 512)	262656
dense_3 (Dense)	(None, 256)	131328
dense_4 (Dense)	(None, 128)	32896
dense_5 (Dense)	(None, 64)	8256
dense_6 (Dense)	(None, 32)	2080
dense_7 (Dense)	(None, 16)	528
dense_8 (Dense)	(None, 8)	136
dense_9 (Dense)	(None, 4)	36
dense_10 (Dense)	(None, 1)	5

Total params: 966,817
Trainable params: 966,817
Non-trainable params: 0

Number of layers in DNN	Number of total parameters
5	833
7	11,533
9	1,78,337
11	9,66,817

Overall, increasing the number of layers in a deep learning model can provide certain benefits. This can improve the model accuracy, and can lead to challenges in training time, vanishing or exploding gradients, overfitting, and the need for more data and computational resources. The below are few consequences.

1. Increased Model Capacity: With more layers, the model is able to capture.
2. More complex relationships in the data that can lead to increased accuracy in predictions.
3. Increased Training Time: As we are increasing the number of layers,model computational complexity also increases, leading to longer training times.
4. Vanishing or Exploding Gradients: With very deep networks, the gradients (which are used to update the model's parameters during training) may become very small (vanishing gradients) or very large (exploding gradients), making it difficult to train the model effectively.
5. Require More Data: If the number of layers are increased then the neural network may require more training data to prevent overfitting and effectively learn the underlying patterns in the data.
6. Require More Computational Resources: Deeper models require more computational resources, including memory and processing power, making them more expensive.

Deepneural networks are one of the deep learning algorithms that mimics the behaviour of neurons in the human brain. There are different types of artificial neural networks like Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN). RNNs and CNNshave the ability toprocess the unstructured data. artificial neural networks (ANN)consists of one or two hidden layers to process data while DNN mainly contains multiple layers between the input and output layers.

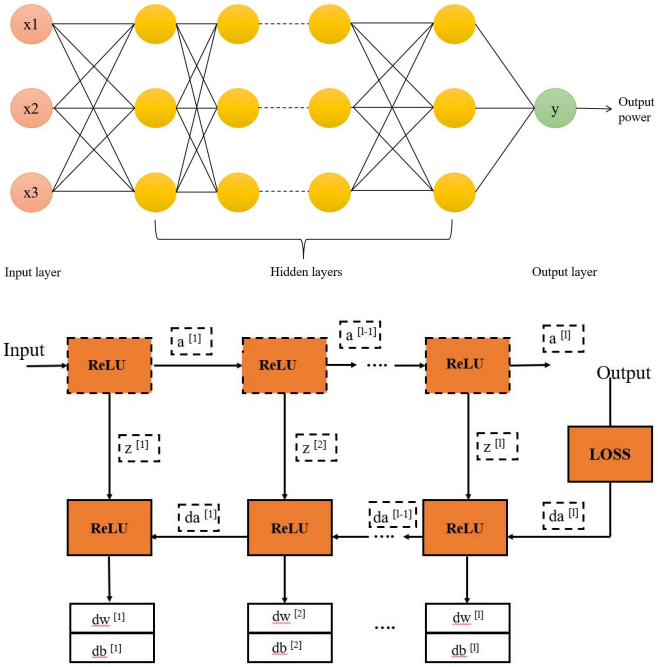

Fig6: Workflow of modelling process.

Before fitting the dataset to the training set, Data is transformed into array representations. In the DNN model, the input for forecasting the power at the output consists of the grating width and sliding velocity. With the help ofReLU function hidden layer is activated, and the loss function is adjusted with clarified variance ratio of every element. Developed model provides the output forecasts for previously unseen data in the CS mode.

There are three layers in artificial neural networks: input, hidden, and output. There may be more than one hidden layer. A layer has n number of neurons in it. Each layer's neurons will each have an associated activation function. The function that introduces non-linearity into the relationship is the activation function. In our situation,a linear activation function is requiredin the output layer. Regularizers may be connected to each layer. The role of regularizers is to prevent over fitting.

Optimizers in deep learning:

During the training of our deep learning model, we must adjust the weights at each epoch and strive to decrease the loss function as much as feasible. We must employ an optimizer for this task, which is an algorithm that modifiesour neural network's weights and its learning rate. So, an optimizer is essential for increasing our model's accuracy and decreasing loss. But picking the right weights for the model is a difficult undertaking. We can use utilize different optimizers to tune the weights and learning rate. We have to choose the optimizer that best suites our application.We have tested different optimizers with our DNN model and observed the performance metrics of all optimizers in predicting the output power.

I. Adam:

The Adam algorithm, short for Adaptive Moment Estimation, is an optimization algorithm used in deep learning. It calculates adaptive learning rates for each parameter during each iteration. The Adam optimizer is widely used and adjusts the learning rate of each weight in the neural network based on the first and second moments of the gradients β1, β2.The algorithm computes an adaptive learning rate for each weight that takes into account the historical gradients and adjusts the learning rate for each weight accordingly. This adaptive learning rate is then utilized for updating the neural network's weights. It enables the algorithm to converge faster and more reliably than traditional stochastic gradient descent. The hyper parameters β1, β2, and ε are used to control the decay rates of the first and second moments and to prevent division by zero in the update equation, respectively. The values of these hyper parameters can affect the performance of the algorithm, and they need to be carefully tuned for each specific problem.

II. SGD:

This is a different Gradient Descent optimizer variation that has the extra ability to operate with data and non-convex optimization problems. With such data, the cost function ends up resting at local minima, which is problematic because they are not ideal for your learning process.
Instead of using batch processing, this optimizer concentrates on carrying out each update one at a time. As a result, it is typically faster, and after each repetition, the cost function decreases. It often updates with a large variance, which results in significant fluctuations in the goal function. As a result, the gradient changes to indicate a possible Global Minima. The cost function may vary about the minimum or even diverge from the global minima if we pick a learning rate that is too big.
SGD can be prone to getting stuck in local minima, which can lead to suboptimal solutions. To address this, researchers have developed various techniques, such as adding momentum or adaptive learning rates, to help SGD escape local minima and converge to better solutions.

III. Adagrad:

This is the adaptive gradient optimization approach, where the updated parameter values are heavily influenced by the learning rate. This optimizer, unlike stochastic gradient descent, employs a variable learning rate for each iteration as opposed to utilising a single learning rate to determine all the parameters.Adagrad is effective in handling sparse data and can make smaller updates for high-frequency features and greater updates for low-frequency features, leading to improved performance and accuracy.

IV. RMSprop:

RMSprop and Adadelta are both optimization algorithms that were created to improve upon Adagrad's problem of having overly aggressive learning rates. Both of these algorithms use a similar technique to calculate the learning rate for each iteration at time t, which involves an exponential weighted average. RMSprop, which was developed by Geoffrey Hinton, is an adaptive learning rate approach that takes an exponentially weighted average of the squared gradients. The learning rate is then divided by this weighted average, which allows for more stable and consistent updates to the model's parameters. This is because the squared gradients can act as a proxy for the curvature of the loss function and help prevent the model from making large updates in the wrong direction. In practice, a value of gamma around 0.95 is often recommended for RMSprop. This value has been found to work well across a range of different datasets and models. However, the optimal value of gamma can depend on the specific problem being addressed, and it may need to be tuned accordingly.

V. Adamax:

The AdaMax algorithm builds on this idea by considering the alleged infinite norm of the previous gradients in the place of the scaled L2

norm. Infinite norm is a way to measure the magnitude of a vector by taking the maximum absolute value of its elements.By considering the infinite norm of the previous gradients, the AdaMax algorithm can make some optimizations more successful. This is because the infinite norm provides a more robust measure of the magnitude of the gradients and can prevent the algorithm from getting stuck in local minima.
Overall, the AdaMax algorithm is a powerful extension of the Adam optimization technique which improve the deep learning model' performance by providing more robust measure of the magnitude of the gradients.

VI. Adadelta:

This modified version of the adaptive gradient optimizer addresses the issue of aggressive reduction of the learning rate in the original algorithm. Instead of using the sum of previous squared gradients, this modification calculates a weighted average of all prior squared gradients. This helps prevent the learning rate from becoming excessively small.

The calculation of the new weight follows the same formula as in Adagrad. However, there are variations in how each iteration determines the learning rate at time step t.
First, the weighted average is computed for each iteration. In this case, the limiting term aids in avoiding the Vanishing Gradient issuedescribes a modification of the adaptive gradient optimizer. This modification addresses a problem with the original algorithm, which tended to aggressively lower the learning rate, causing slow convergence or even divergence of the optimization process.

To reduce the training set's computational cost of the model we used DNN with 7 layers that utilizes very less parameters yet gives very good results. We have compared with our DNN models with 5, 7, 9 and 11 layers but as we increase the number of layers the total number of parameters are increasing so DNN model with 7 layers is optimal for our problem. We have tested and compared with various optimizers as well. We utilized Google Colab as the environment for training and testing our model. The system specifications included Windows 11 Pro, an Intel Core i7-12700 CPU running at 2.1 GHz, and 32.0 GB of memory. We implemented the model using Python and utilized inbuilt libraries such asKeras and TensorFlow.

Our deep neural network (DNN) model consists of one input layer, five hidden layers, and one output layer. The input layer takes the feature parameters as input and computes the hidden layer using the ReLU activation functionwhich is shown in Fig. 7. The output layer provides the predicted results for the output power.During the training process, we adjust the model's parameters based on the difference between the predicted and observed results. We used various optimizers such as Adam, SGD, Adaptive Gradient (Adagrad), RMSprop, Adamax, and Adam. After applying the Adadelta technique to the contact separation, the model achieved a final loss value of 0.0435.

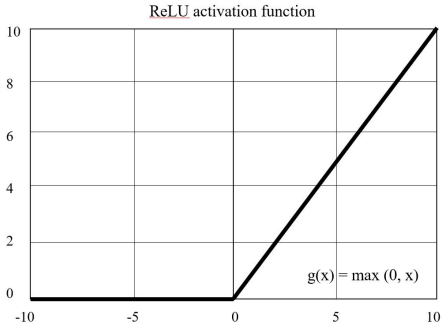

Fig 7: ReLU function

The error between the actual power value and the predicted power value is utilized to update the model's parameters during the training process. This updating of parameters allows the optimizer to improve and refine the model in terms of error and performance. Comparison with the optimizers is shown in below figure. Among the tested optimizers Adagrad and Adadelta are performing better compared to other optimizers in terms of the performance metrics such as MSE, R2-Score, MAE etc.

3. Results and discussion:

Experimental data formed as a dataset is utilized to train the models and is divided them into a training set, and a test set. The train dataset data is used for training ourneural network and the test dataset with is 30% of our total dataset size is used to test the performance our model. Below figures show the experimental output power and the prediction ofRandom Forest regressor which is performing best among other testedmachine learning techniques and the same comparison of our DNN models with different optimizers. We observed that our Deep neural network model performs very well in predicting the output power and the Adadelta optimizer is performing better in terms of all the performance metrics compared to other optimizers. The reasons are,
1. Adaptive Learning Rate: The Adadelta optimizer is an adaptive learning rate optimizer, which means that it can adjust model's learning ratewith respect

to the gradient updates in each iteration. This ensures that the model converges quickly to the optimal solution without overshooting or oscillating.

2. Robustness to Noisy Data: Adadelta is a robust optimizer that can handle noisy data and small gradients. This is because it uses a moving average of the past gradients to update the learning rate, which reduces the impact of individual noisy gradients.

3. No Learning Rate Decay: Unlike other optimizers, such as Adam and RMSprop, Adadelta does not require a learning rate decay schedule. This reduces the need for hyperparameter tuning and makes the optimization process simpler.

4. Faster Convergence: Adadelta optimizer can converge faster than other optimizers, especially in the case of deep neural networks. This is because it uses an efficient gradient update method that requires fewer iterations to reach convergence.

In machine learning techniques Random Forest regression performs better than other regression techniques because of following reasons.

1. Overfitting Reduction: Random Forest Regressor generates many decision trees for prediction decision then integrates all results to get the last prediction, which helps to decrease model overfitting since the average of multiple trees is more stable and accurate than a single tree.

2. Non-Linear Relationships: Random Forest Regressor can gather non-linear relationships between the target variable and the features, which is not always possible in linear regression models.

3. Robustness to Outliers: Random Forest Regressor is robust to outliers as it uses a combination of decision trees to make predictions. A single decision tree can be biased by outliers, but combining multiple trees reduces the effect of any single outlier.

4. Feature Importance: Random Forest Regressor provides an estimate of feature importance, which helps to identify the most significant features that affect the target variable. This can be useful for feature selection and dimensionality reduction.

5. Easy to Use: Random Forest Regressor is relatively easy to use and requires less parameter tuning compared to other complex models like XGB Regressor.

Prediction using Machine learning technique:

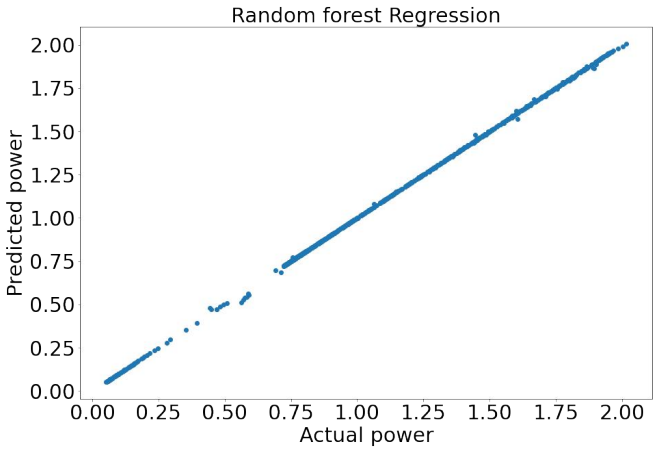

Fig 8. Graph between prediction vs actual power

The output power is predicted for the corresponding Load resistance, Output voltage and Output current using the Random Forest regression technique. The variation of the predicted power to the actual output power is shown graphically in the above figure Fig. 8 and the deviation error is tabulated below.

Table 1: Machine Learning data between actual and predicted under specified load

Load resistance (M Ω)	**Actual power (µW)**	**Predicted power (µW)**	**Difference (µW)**
3	1.013867295	1.013555125	0.00031217
1.3	1.777230769	1.776432734	0.00079803
0.35	1.255875661	1.255664134	0.00021152
4.65	0.747844162	0.747841161	3.00085E-06
0.75	1.611398438	1.614507213	-0.00310877
1.1	1.887768759	1.887976554	-0.00020779
3.2	0.970201259	0.970064257	0.00013699
0.5	1.730939393	1.729671411	0.00126797

1.1	1.88776875 4	1.88797655 4	-0.00020779
2.9	1.04822693 8	1.04824492 5	-1.79866E-05
2.25	1.29856732 1	1.29842436 3	0.00014295 8

Prediction using Deep learning technique (DNN):

Adadelta optimizer:

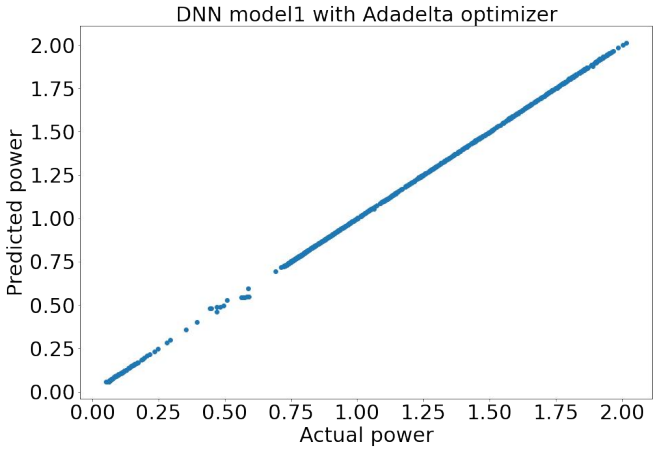

Fig .9DL curve between Actual power and predicted power

Table 2: comparison of DL data for actual and predicted power under specified load

Load resistance (M Ω)	**Actual power (μW)**	**Predicted power (μW)**	**Difference (μW)**
3	1.01386729 5	1.01355512 5	0.00031217
1.3	1.77723076 9	1.77643273 4	0.00079803 5
0.35	1.25587566 1	1.25566413 4	0.00021152 7
4.65	0.74784416 2	0.74784116 1	3.00085E-06
0.75	1.61139843 8	1.61450721 3	-0.00310877 5
3.2	0.97020125	0.97006425 7	0.00013699 3
0.5	1.73093939 4	1.72967141 5	0.00126797 9
2.9	1.04822693 8	1.04824492 5	-1.79866E-05
2.25	1.29856732 1	1.29842436 3	0.00014295 8

Loss fitting curves with different optimizers in DNN model:

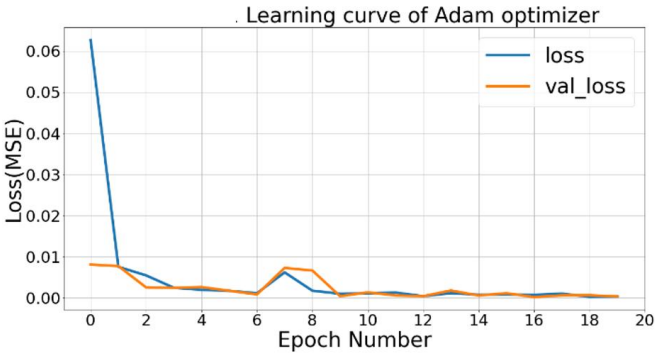

Fig .10 Adam optimizer's Loss fitting curve

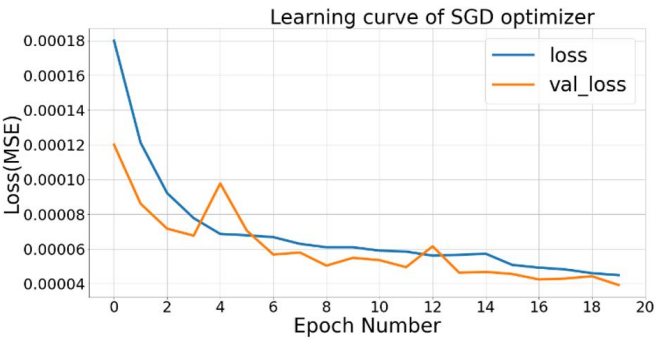

Fig .11 SGD optimizer's Loss fitting curve

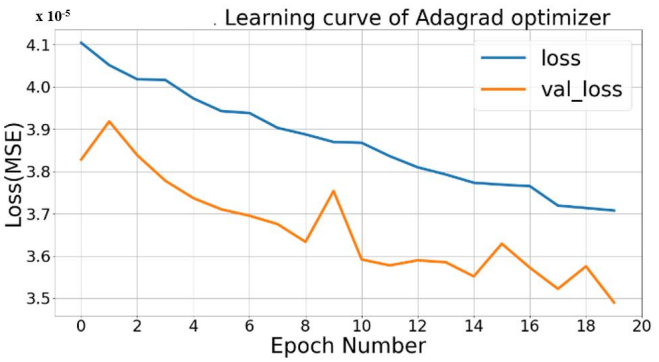

Fig .12 Adagrad optimizer's Loss fitting curve

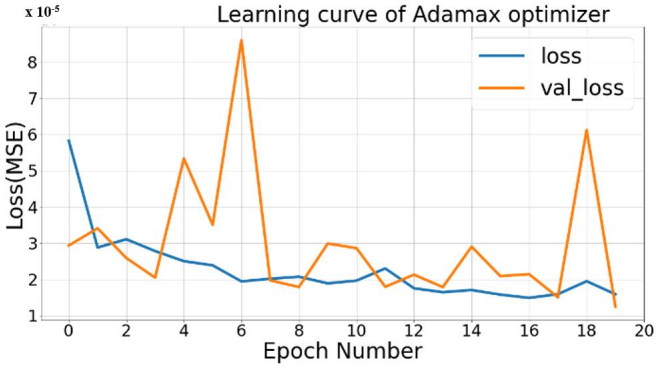

Fig .13 Adamax optimizer's Loss fitting curve

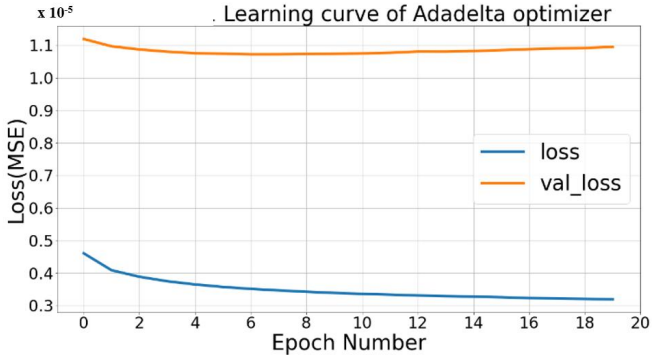

Fig .14 Adadelta optimizer's Loss fitting curve

Loss fitting curves with different optimizers in DNN model are shown in above figures Fig. 10, 11, 12, 13 and 14.

Performance metrics:

Performance metrics for the regression analysis such as MSE, MAE, RMSE, R2-Score are obtained for the ML methods and Deep learning techniques with different optimizers.

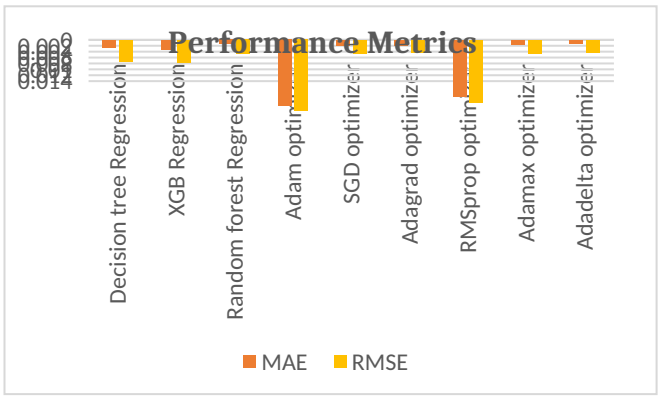

Fig .15Comparison of MAE and RMSE in ML and DL Regression models

As per the above comparison of MAE and RMSE metrics in Fig. 15, Adam and RMSprop optimizers have greater loss compared to other techniques. Among all Adadelta optimizer is giving better MAE and RMSE values.

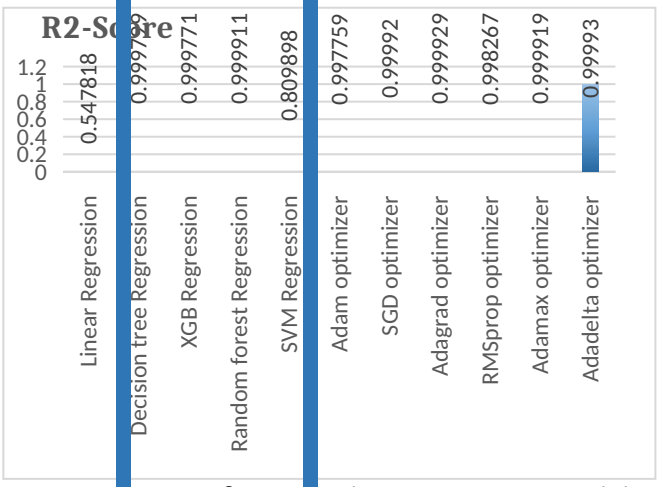

Fig .16 R2 score for ML and DL Regression models

Fig .17 MSE for ML and DL Regression models

As per the above graph in Fig. 17 of MSE metric comparison among the ML and Deep Learning techniques, Linear Regression and SVM Regression are having high MSE values compared to other techniques. Among all Adagrad and Adadelta optimizers are giving very low MSE.

Fig .18 RMSE for ML and DL Regression models

Fig .19 MAE for ML and DL Regression models

The below tables 3 & 4 tells about the related information about ML and DL regression models with their performance metrics. Among the all-regression models in machine learning Random Forest regression giving better results compared to remaining regression techniques present. Table represents the actual and predicted power for a given load resistance, the comparison of powers that are actually getting and our predicted ones in case of both ML regression methods and DL regression methods and in the both cases for ML less error is getting in the case of random forest regression and DL less error is getting in the case of Adamdelta.

Table : Comparison of different performance metrics for ML and DL regression models

Model	R2-Score	MAE	MSE	RMSE
Linear Regression	0.547818	0.251831	0.117537	0.342836
Decision tree Regression	0.999799	0.00257368	5.22E-05	0.00722728
XGB Regression	0.999771	0.0033288	5.95E-05	0.00771113
Random forest Regression	**0.999911**	**0.00139354**	**2.30E-05**	**0.00480066**
SVM Regression	0.809898	0.133898	0.0494138	0.222292
DNN model with Adam	0.997759	0.0223937	0.000582385	0.0241327
DNN model with SGD	0.99992	0.00202658	2.07E-05	0.00455054
DNN model with Adagrad	0.999929	0.0019146	1.85E-05	0.00430573
DNN model with RMSprop	0.998267	0.0191392	0.000450456	0.0212239
DNN model with Adamax	0.999919	0.00150581	2.10E-05	0.00458642
DNN model with Adadelta	**0.99993**	**0.001267**	**1.82E-05**	**0.00426986**

Table : Comparison of Actual and Predicted power for ML and DL regression models

Load resistance (M Ω)	Actual Output power (μW)	Linear regression (μW)	Decision tree (μW)	XGB (μW)	Random forest (μW)	SVM (μW)	Adam (μW)	SGD (μW)	Adagrad (μW)	RMSprop (μW)	Adamax (μW)	Adadelta (μW)
3	1.0139	1.0541	1.0133	1.0124	**1.0136**	1.1212	1.0307	1.0132	**1.0139**	0.9608	1.0142	1.0129
1.3	1.7772	1.3678	1.7747	1.7757	**1.7764**	1.5635	1.7534	1.7767	**1.7771**	1.6922	1.7747	1.7774
0.35	1.2559	1.2505	1.2529	1.2565	**1.2557**	1.7244	1.2422	1.2534	**1.2536**	1.1898	1.2578	1.2562
4.65	0.7478	0.9792	0.7480	0.7478	**0.7478**	0.7459	0.7713	0.7474	**0.7476**	0.7164	0.7498	0.7477
0.75	1.6114	1.2796	1.6075	1.6124	**1.6145**	1.6639	1.6266	1.6312	**1.6310**	1.5544	1.6107	1.6126
1.1	1.8878	1.4210	1.8891	1.8877	**1.8880**	1.6133	1.8663	1.8882	**1.8887**	1.7905	1.8834	1.8872
3.2	0.9702	1.0374	0.9697	0.9705	**0.9701**	1.0784	0.9862	0.9684	**0.9692**	0.9190	0.9713	0.9700
0.5	1.7309	1.4485	1.7230	1.7401	**1.7297**	1.7203	1.7036	1.7307	**1.7328**	1.6153	1.7298	1.7317
2.9	1.0482	1.0677	1.0488	1.0489	**1.0482**	1.1521	1.0654	1.0469	**1.0478**	0.9949	1.0502	1.0489
2.25	1.2986	1.1764	1.2975	1.3037	**1.2984**	1.3201	1.2961	1.2954	**1.2972**	1.2390	1.2988	1.2985

Figure 20 represents the different electronic applications are produced by using designed sensor APA (Aluminium foil parafilm Aluminium foil) TENG which are earlier published in the article [40]. The design of the device is shown in figure 21, and it's construction of real images is shown in figure 22.

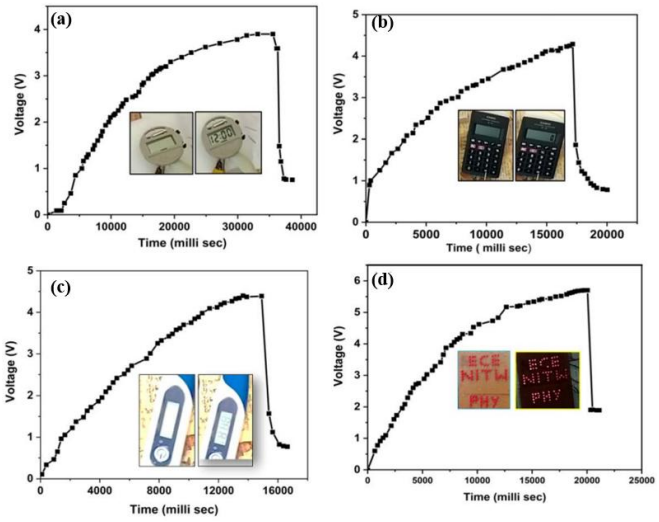

Fig .20 Applications of TENGs (a) glowing of watch (b) calculator (c) Thermometer (d) 85 LEDs [40]

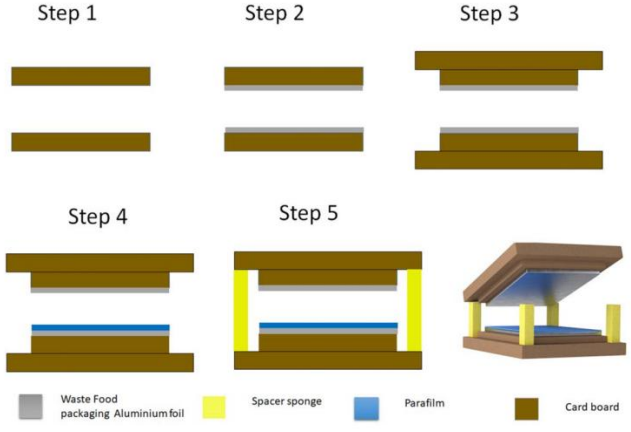

Fig .21 Structure of TENG

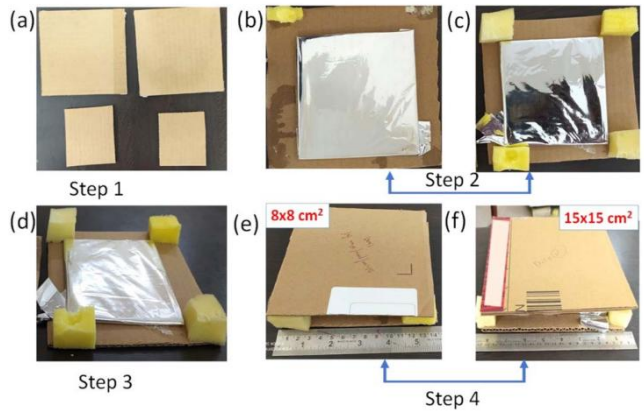

Fig .22 Construction of TENG model

Conclusion:

In this work we have tested both machine learning as well as deep learning methodsfor predicting power at the output based on the load resistance and other parameters. In machine learning we observed that Random Forest regressor gives good predictions compared to other models. In Deep learning techniques, Adadelta optimizer performs better than other optimizers. With the help of this deep learning approach compared to conventional experimentation method for finding the output power, we can predict the output power even without doing the experimentation for getting that result with the help of training the model. This saves lot of cost incurred for experimentation and minimizes the errors. From the results obtained for a Load resistance of 0.5M Ω, the actual output power is 1.7309 μW and the prediction from Random Forest regressor is 1.7297 μW and the predicted output power by our deep neural network with Adagrad optimizer is 1.7328 μW. So in the deep learning technique the error is very small compared to machine learning technique.

Declaration of Competing Interest

The authors state that they have no known conflicting financial or personal interests that may have seemed to affect the work presented in this article.

Acknowledgment

This research was funded byMinistry of Human Resource Development Government of India for awarding the SERB (No. EEQ/2021/000614)

References:

[1] X. Wang, Y. He, X. Liu, L. Shi, and J. Zhu, “Investigation of photothermal heating enabled by plasmonic nanofluids for direct solar steam generation,” *Sol. Energy*, 2017, doi: 10.1016/j.solener.2017.08.015.

[2] R. A. Kishore and S. Priya, “A review on design and performance of thermomagnetic devices,” *Renewable and Sustainable Energy Reviews*. 2018, doi: 10.1016/j.rser.2017.07.035.

[3] J. K. Kaldellis and D. Zafirakis, “The wind energy (r)evolution: A short review of a long history,” *Renewable Energy*. 2011, doi: 10.1016/j.renene.2011.01.002.

[4] F. Invernizzi, S. Dulio, M. Patrini, G. Guizzetti, and P. Mustarelli, “Energy harvesting from human motion: Materials and techniques,” *Chemical Society Reviews*. 2016, doi: 10.1039/c5cs00812c.

[5] W. Kim, D. Bhatia, S. Jeong, and D. Choi, “Mechanical energy conversion systems for triboelectric nanogenerators: Kinematic and vibrational designs,” *Nano Energy*, vol. 56, no. November 2018, pp. 307–321, 2019, doi: 10.1016/j.nanoen.2018.11.056.

[6] L. Atzori, A. Iera, and G. Morabito, “The Internet of Things: A survey,” *Comput. Networks*, 2010, doi: 10.1016/j.comnet.2010.05.010.

[7] M. Ha, J. Park, Y. Lee, and H. Ko, “Triboelectric generators and sensors for self -powered wearable electronics,” *ACS Nano*, 2015, doi: 10.1021/acsnano.5b01478.

[8] J. Luo and Z. L. Wang, “Recent advances in triboelectric nanogenerator based self-charging power systems,” *Energy Storage Mater.*, vol. 23, no. March, pp. 617–628, 2019, doi: 10.1016/j.ensm.2019.03.009.

[9] A. Yu *et al.*, “Core-Shell-Yarn-Based Triboelectric Nanogenerator Textiles as Power Cloths,” *ACS Nano*, 2017, doi: 10.1021/acsnano.7b07534.

[10] N. J. Vickers, “Animal Communication: When I’m Calling You, Will You Answer Too?,” *Current Biology*. 2017, doi: 10.1016/j.cub.2017.05.064.

[11] Z. L. Wang, “Triboelectric nanogenerators as new energy technology and self-powered sensors - Principles, problems and perspectives,” *Faraday Discussions*. 2014, doi: 10.1039/c4fd00159a.

[12] F. R. Fan, Z. Q. Tian, and Z. Lin Wang, “Flexible triboelectric generator,” *Nano Energy*, 2012, doi: 10.1016/j.nanoen.2012.01.004.

[13] G. Zhu *et al.*, “Toward large-scale energy harvesting by a nanoparticle-enhanced triboelectric nanogenerator,” *Nano Lett.*, vol. 13, no. 2, pp. 847–853, 2013, doi: 10.1021/nl4001053.

[14] S. Zhang, M. Bick, X. Xiao, G. Chen, A. Nashalian, and J. Chen, “Leveraging

triboelectric nanogenerators for bioengineering," *Matter*. 2021, doi: 10.1016/j.matt.2021.01.006.

[15] Q. yuan Zhang *et al.*, "A Design of Flexible Triboelectric Generator Integrated with High -Efficiency Energy Storage Unit," *Energy Technol.*, 2021, doi: 10.1002/ente.202000962.

[16] J. J. Shao, T. Jiang, and Z. L. Wang, "Theoretical foundations of triboelectric nanogenerators (TENGs)," *Science China Technological Sciences*. 2020, doi: 10.1007/s11431-020-1604-9.

[17] J. Wang, L. Zhou, C. Zhang, and Z. Lin Wang, "Small-Scale Energy Harvesting from Environment by Triboelectric Nanogenerators," in *A Guide to Small-Scale Energy Harvesting Techniques*, 2020.

[18] S. Mishra, S. Potu, R. S. Puppala, R. K. Rajaboina, P. Kodali, and H. Divi, "A novel ZnS nanosheets-based triboelectric nanogenerator and its applications in sensing, self-powered electronics, and digital systems," *Mater. Today Commun.*, 2022, doi: 10.1016/j.mtcomm.2022.103292.

[19] Y. Li *et al.*, "Low-Cost, Environmentally Friendly, and High-Performance Triboelectric Nanogenerator Based on a Common Waste Material," *ACS Appl. Mater. Interfaces*, 2021, doi: 10.1021/acsami.1c09192.

[20] J. Kaur, R. S. Sawhney, H. Singh, and M. Singh, "Electricity nanogenerator from egg shell membrane: A natural waste bioproduct," *Int. J. Green Energy*, vol. 17, no. 5, pp. 309–318, 2020, doi: 10.1080/15435075.2020.1727482.

[21] K. Xia *et al.*, "A triboelectric nanogenerator based on waste tea leaves and packaging bags for powering electronic office supplies and behavior monitoring," *Nano Energy*, vol. 60, no. March, pp. 61–71, 2019, doi: 10.1016/j.nanoen.2019.03.050.

[22] A. Gaur, S. Tiwari, C. Kumar, and P. Maiti, "Bio -waste orange peel and polymer hybrid for efficient energy harvesting," *Energy Reports*, 2020, doi: 10.1016/j.egyr.2020.02.020.

[23] S. K. Ghosh and D. Mandal, "High-performance bio-piezoelectric nanogenerator made with fish scale," *Appl. Phys. Lett.*, 2016, doi: 10.1063/1.4961623.

[24] J. M. Wu, C. K. Chang, and Y. T. Chang, "High-output current density of the triboelectric nanogenerator made from recycling rice husks," *Nano Energy*, vol. 19, pp. 39–47, 2016, doi: 10.1016/j.nanoen.2015.11.014.

[25] H. Lu, W. Zhao, Z. L. Wang, and X. Cao, "Sugar-based triboelectric nanogenerators for effectively harvesting vibration energy and sugar quality assessment," *Nano Energy*, 2021, doi: 10.1016/j.nanoen.2021.106196.

[26] P. Zhang, W. Zhang, and H. Zhang, "A triboelectric nanogenerator based on waste polyvinyl chloride for Morse code generator," *Sensors Actuators, A Phys.*, 2021, doi: 10.1016/j.sna.2021.112633.

[27] X. Feng, Q. Li, and K. Wang, "Waste Plastic Triboelectric Nanogenerators Using Recycled Plastic Bags for Power Generation," *ACS Appl. Mater. Interfaces*, vol. 13, no. 1, pp. 400–410, Jan. 2021, doi: 10.1021/acsami.0c16489.

[28] A. W. Senior *et al.*, "Improved protein structure prediction using potentials from deep learning," *Nature*, 2020, doi: 10.1038/s41586-019-1923-7.

[29] J. Vamathevan *et al.*, "Applications of machine learning in drug discovery and development," *Nature Reviews Drug Discovery*. 2019, doi: 10.1038/s41573-019-0024-5.

[30] M. Jiang, B. Li, W. Jia, and Z. Zhu, "Predicting output performance of triboelectric nanogenerators using deep learning model," *Nano Energy*, 2022, doi: 10.1016/j.nanoen.2021.106830.

[31] K. Guo, Z. Yang, C. H. Yu, and M. J. Buehler, "Artificial intelligence and machine learning in design of mechanical materials," *Materials Horizons*. 2021, doi: 10.1039/d0mh01451f.

[32] M. Jiang, Y. Lu, Z. Zhu, and W. Jia, "Advances in smart sensing and medical electronics by self-powered sensors based on triboelectric nanogenerators," *Micromachines*. 2021, doi: 10.3390/mi12060698.

[33] C. H. Yu and M. J. Buehler, "Sonification based de novo protein design using artificial intelligence, structure prediction, and analysis using molecular modeling," *APL Bioeng.*, 2020, doi: 10.1063/1.5133026.

[34] W. Mei, L. Jiang, C. Liang, J. Sun, and Q. Wang, "Understanding of Li-plating on graphite electrode: detection, quantification and mechanism revelation," *Energy Storage Mater.*, 2021, doi: 10.1016/j.ensm.2021.06.013.

[35] P. Jiao, "Emerging artificial intelligence in piezoelectric and triboelectric nanogenerators," *Nano Energy*. 2021, doi: 10.1016/j.nanoen.2021.106227.

[36] Y. Zhou, M. Shen, X. Cui, Y. Shao, L. Li, and Y. Zhang, "Triboelectric nanogenerator based self-powered sensor for artificial intelligence," *Nano Energy*. 2021, doi: 10.1016/j.nanoen.2021.105887.

[37] S. Tang, S. Yuan, and Y. Zhu, “Convolutional Neural Network in Intelligent Fault Diagnosis Toward Rotatory Machinery,” *IEEE Access*, 2020, doi: 10.1109/ACCESS.2020.2992692.

[38] C. Lihui, L. Yang, and Z. Donghua, “Fault Diagnosis of the Planetary Gearbox Based on ssDAG-SVM ,” 2018, doi: 10.1016/j.ifacol.2018.09.586.

[39] Y. Dong, X. Ma, and T. Fu, “Electrical load forecasting: A deep learning approach based on K-nearest neighbors,” *Appl. Soft Comput.*, 2021, doi: 10.1016/j.asoc.2020.106900.

[40] P. R. Sankar, K. Prakash, P. Supraja, R. R. Kumar, S. Mishra, and D. Haranath, “A triboelectric nanogenerator based on food packaging Aluminium foil and parafilm for self-powered electronics,” *Phys. Scr.*, 2021, doi: 10.1088/1402-4896/ac2086.

[41] P. Ravi Sankar, P. Supraja, S. Mishra, K. Prakash, R. Rakesh Kumar, and D. Haranath, “A novel triboelectric nanogenerator based on only food packaging aluminium foils,” *Mater. Lett.*, 2022, doi: 10.1016/j.matlet.2021.131474.