Analysis and Prediction of Energy Consumption in Neural Networks Based on Machine Learning

Xiaokun Qi, Tian He*

College of Mechanical and Electrical Engineering, Qingdao University, Qingdao, 266071, China
qixiaokun1210@foxmail.com
**Corresponding author: he_t@x263.net*

Abstract: *In the current technological development, convolutional neural networks have become an important tool for computer vision tasks, especially in mobile devices. However, executing related tasks using complex neural network models often leads to high energy consumption issues. Therefore, energy modeling for neural networks becomes crucial. Through energy modeling, we can better understand the energy consumption of neural networks, and subsequently carry out targeted energy optimization to reduce the energy consumption of devices when performing tasks. This experiment selected nine feature variables from three levels of convolutional neural networks, and used five machine learning algorithms to model the energy consumption of convolutional neural networks. The five machine learning methods are Support Vector Regression (SVR), Neural Network (NN), Decision Tree (DT), Random Forest (RF), and Adaptive Boosting (AdaBoost). To select the best modeling method, this paper introduces Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE) to evaluate the models. The experiment proves that Adaptive Boosting (AdaBoost) has the lowest MSE, RMSE, and MAE, therefore it is the optimal model in this experiment.*

Keywords: *Convolutional Neural Networks, Machine Learning, Energy Modeling, Model Evaluation*

1. Introduction

With the increasing performance requirements of visual systems, neural network algorithms need to process a large amount of input data and carry out complex calculations and inference processes. The models involved are becoming more and more complex, leading to increasingly prominent energy consumption issues. Firstly, mobile devices usually need to complete various tasks, including perception, decision-making, and execution, under the support of limited battery energy. Reasonable energy utilization is the key to ensuring their continuous operation and task execution. High-energy-consuming visual systems will consume a lot of battery energy when working, greatly shortening the use time of the device, and limiting the practical application of the device. Secondly, excessive energy consumption can also lead to overheating and performance degradation of the device, further affecting its stability and reliability. To solve these problems, special cooling devices need to be equipped, which undoubtedly makes the not spacious space inside the miniaturized device more crowded, and the existence of the cooling device will consume some energy, making its activities more restricted. Therefore, energy consumption modeling for neural networks has important practical significance and academic value. By deeply understanding the characteristics and mechanisms of energy consumption, we can provide guidance and support for the design and development of more efficient neural networks.

Energy consumption modeling is a method of quantifying and predicting energy consumption. It can help us understand and analyze the source of energy consumption and find strategies to reduce energy consumption. We compared advanced machine learning algorithms for estimating the energy consumption of convolutional neural networks during operation. We considered the following models: Support Vector Regression (SVR), Neural Network (NN), Decision Tree (DT), Random Forest (RF), and Adaptive Boosting (AdaBoost). Our work is not to propose new machine learning models, but to compare the advantages and disadvantages of 5 models in order to choose the model that best represents energy consumption prediction. The rest of this article is arranged as follows. The second section reviews related research, the third section conducts energy consumption modeling, and the fourth section draws conclusions.

2. Related Research

Research has shown that machine learning methods have been widely applied to energy consumption modeling[1, 2], such as electricity[3], wastewater treatment[4], complex networks[5], sensor parameters[6], and more. Of course, there are also many applications in neural networks. Some researchers have used Gaussian regression to analyze the energy consumption of a specific algorithm, MobileNet, and its results are superior to linear regression and decision trees[7]. Some researchers have run convolutional neural networks (CNN) performance prediction on edge devices. The research compared five widely used machine learning-based methods for predicting the execution time of CNN on two edge GPU platforms. In addition, it also explored the training time of these methods and the time to adjust hyperparameters, and compared the time required to run prediction models on different platforms. Experimental results show that the Extreme Gradient Boosting (XGBoost) method has an average prediction error of less than 14.73% under unknown CNN model structures, and the Random Forest (RF) method has comparable accuracy but requires more training and adjustment time. The accuracy of the other three methods (OLS, MLP, and SVR) for CNN performance estimation is lower. The research results can help designers choose the most effective CNN implementation on a specific edge GPU platform[8]. Different from the above research methods, some researchers have built a fully connected neural network to predict inference time. They believe that deep learning methods can fit more complex functions, adapt to complex scenarios, and have more advantages than traditional machine learning methods. This paper divides neural networks into two types: convolutional neural networks and recurrent neural networks, and extracts layer structure features and hardware parameter feature for the special structures of the fully connected layer, convolutional layer, pooling layer, and recurrent neural network, respectively. The neural network is disassembled into different layers to measure the inference time separately[9]. There are also scholars who combine machine learning and deep learning to seek new methods to reduce overall energy consumption[10]. The combination of k-means and CNN yields better results than using either method alone[11].

3. Energy Consumption Prediction

This chapter will carry out the energy consumption prediction, as shown in Figure 1.

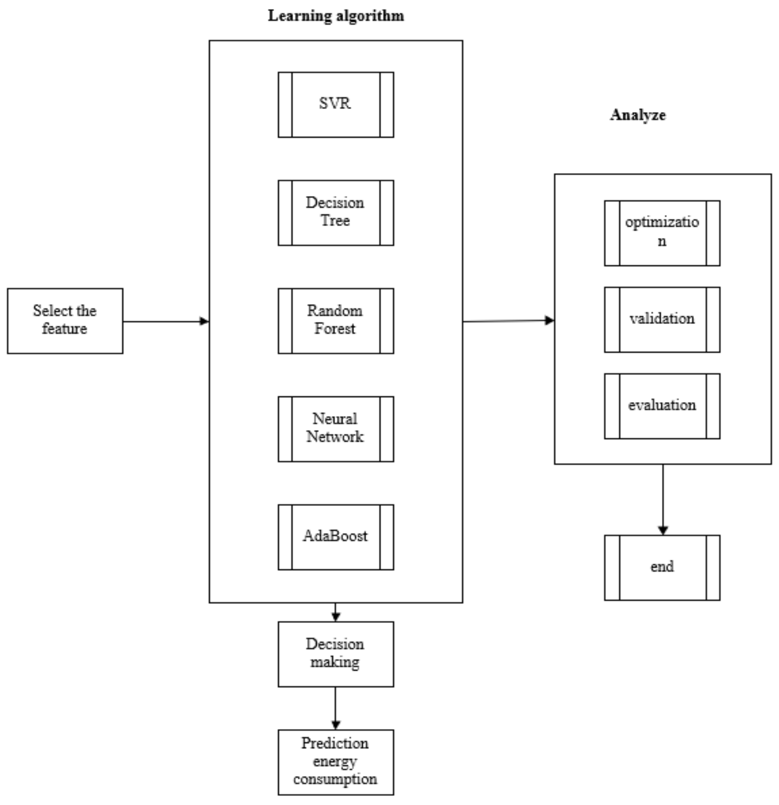


Figure 1: Experimental Procedure

3.1 Feature Selection

This experiment compares the energy consumption of different neural networks, selecting nine features across three levels, as shown in Table 1. The goal is to predict energy consumption through these nine features.

Table 1: Variable Statistics

Variable description	Input or output variable
X1	Number of convolutional layers
X2	The amount of parameters for the convolutional layer
X3	FLOPs of convolutional layers
X4	The number of pooling layers
X5	The amount of parameters for the pooling layer
X6	FLOPs of the pooling layer
X7	The number of fully connected layers
X8	The amount of parameters for the fully connected layer
X9	FLOPs of fully connected layer
Y	Energy consumption

FLOPs (Floating Point Operations Per Second) is a metric for measuring the computational load of a deep learning model, representing the number of floating-point operations performed in a single forward propagation of the model. In the convolutional layer, the number of FLOPs is related to the size of the input feature map, the size of the convolution kernel, and the number of convolution kernels. Larger input feature maps, larger convolution kernels, and more convolution kernels will increase the number of FLOPs in the convolutional layer, thereby increasing energy consumption. The number of FLOPs in the pooling layer is relatively small because pooling operations usually involve selecting the maximum or average value and do not involve multiplication operations. The number of FLOPs in the fully connected layer is proportional to the number of input and output nodes. Each connection requires multiplication and addition operations for weights and biases, so a larger number of connections will lead to a higher number of FLOPs and energy consumption. Therefore, FLOPs is an important metric for evaluating the energy consumption of convolutional layers, pooling layers, and fully connected layers.

However, relying solely on this metric is clearly insufficient, so this article introduces other metrics, namely the number of hotspot layers and the parameter quantity of each hotspot layer. The number of hotspot layers reflects the number of layers in the model with a large amount of computation. More hotspot layers mean that the model may consume more energy during the computation process. The parameter quantity of the hotspot layer refers to the number of parameters that need to be learned in the model. A larger number of parameters will increase the computational load and memory consumption, leading to higher energy consumption. By controlling the parameter quantity of the hotspot layer, we can effectively reduce the energy consumption of the model. By considering FLOPs, the number of hotspot layers, and parameter quantity, we can more comprehensively evaluate the energy consumption of deep learning models. By optimizing the model architecture, performing network pruning, controlling the quantity of parameters, etc., we can reduce the number and parameter quantity of hotspot layers, thereby improving the energy efficiency performance of the model.

3.2 Machine Learning Algorithms for Prediction

In this section, we will use five machine learning algorithms to model the energy consumption of convolutional neural networks. The following will explain the five algorithms, namely Support Vector Regression (SVR), Neural Network (NN), Decision Tree (DT), Random Forest (RF), and Adaptive Boosting (AdaBoost).

3.2.1 Support Vector Regression (SVR)

Support Vector Regression (SVR) is a regression algorithm based on Support Vector Machine (SVM). Compared to traditional regression methods, SVR is more suitable for handling non-linear and complex regression problems. The goal of SVR is to find a function that can accurately predict the continuous values of the output variable given the input features. Similar to SVM, SVR uses support vectors to define the regression model. Support vectors are the samples in the training set that are most relevant to the regression model, and they determine the shape and predictive power of the model. The core idea of

SVR is to find the optimal regression function by maximizing the margin. The margin refers to the sample points closest to the support vectors, and SVR tries to maximize the distance between these sample points and the regression function. This method can effectively handle outliers and noise, improving the robustness of the model. The kernel function of SVR is an important component, which is used to map input features to a high-dimensional feature space for better fitting non-linear relationships. Commonly used kernel functions include linear kernel, polynomial kernel, and Radial Basis Function (RBF) kernel. The training process of SVR involves solving optimization problems, with the goal of minimizing the balance between prediction error and model complexity. To achieve this goal, SVR introduces regularization parameters and tolerance parameters to control the flexibility and fault tolerance of the model. SVR has good generalization ability and robustness, and is suitable for various regression problems, especially when the data features are complex, and the non-linear relationship is strong. However, the training and tuning of SVR are relatively complex, and the requirements for data preprocessing and parameter selection are high.

3.2.2 Neural Network (NN)

Neural Network (NN) is a computational model inspired by the human nervous system, used to simulate and solve complex problems. It consists of many interconnected neurons, which transmit and process information through weights and activation functions. A neural network typically includes an input layer, hidden layers, and an output layer, as shown in Figure 5. The input layer receives external input data, the hidden layers pass and process information through connections between layers, and finally, the output layer produces the final prediction or classification result.

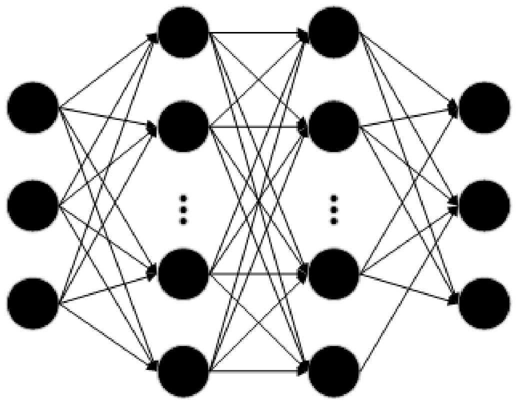


Figure 2: Neural Network Model

The training process of a neural network is implemented through the backpropagation algorithm. This algorithm compares the error between the network output and the expected output, then adjusts the connection weights based on this error, making the network's prediction results gradually approach the expected results. This process is known as training, which allows the neural network to gradually learn and optimize the model's expressive power. One of the advantages of neural networks is their ability to handle non-linear relationships, and they perform well when dealing with large amounts of data and complex problems. In addition, neural networks have good generalization capabilities, allowing them to handle unseen data and make accurate predictions.

3.2.3 Decision Tree (DT)

Decision Tree (DT) is a commonly used machine learning algorithm for solving classification and regression problems. It makes decisions and predictions by constructing a tree-like structure. The tree structure of a decision tree consists of nodes and edges, where each node represents a feature or attribute, and edges represent different values or decision paths. The root node represents the most important feature, while the leaf nodes represent the final classification or regression results. The construction of a decision tree is achieved by recursively splitting the training data. At each node, the decision tree divides the dataset into smaller subsets based on the value of a certain feature until it reaches a stopping condition. The goal of the split is to make the data within each subset purer, that is, data of the same category or similar attributes are as clustered together as possible. The construction of a decision tree can be based on different criteria, common ones include Information Gain, Gini Index, Mean Squared Error, etc. These criteria are used to evaluate the importance of features and the effect of the split, in order to select the best split point. Decision trees have the advantage of being easy to understand and interpret and can generate clear decision rules. In addition, decision trees can handle discrete and continuous features, and have good robustness to outliers and missing data.

3.2.4 Random Forest (RF)

Random Forest (RF) is an ensemble learning method, composed of multiple decision trees. It improves the accuracy and generalization ability of the model by randomly selecting features and samples for training. The construction process of a random forest includes two main steps: random selection of features and random selection of samples. In the training process of each decision tree, the random forest randomly selects a subset of features from the original feature set and uses these features to construct the decision tree. This random selection of features helps to reduce the correlation between features and increase the diversity of the model. In addition, the random forest also randomly selects a subset of samples from the training data for training. This random selection of samples helps to reduce the risk of overfitting and improve the generalization ability of the model. When making predictions, the random forest votes or averages the prediction results of each decision tree to obtain the final prediction result, as shown in equation 1.

$$F(x) = \frac{1}{B}\sum_{b=1}^{B} T_b(x) \quad (1)$$

This ensemble method can reduce the impact of randomness and noise, improving the stability and accuracy of the model. Random Forest is widely used in machine learning for classification and regression problems. It has good performance and robustness and can handle a large number of features and samples. In addition, Random Forest can also evaluate the importance of features, helping us understand the key factors of the problem. However, Random Forest also has some limitations. For example, because the training and prediction processes of multiple decision trees are relatively independent, Random Forest may not be able to capture some complex relationships. In addition, because Random Forest contains multiple decision trees, the interpretability of the model is relatively weak.

3.2.5 Adaptive Boosting (AdaBoost)

Adaptive Boosting (AdaBoost) is an ensemble learning algorithm aimed at improving the performance and accuracy of classification models. It iteratively trains a series of weak classifiers and adjusts the sample weights based on the classification results, making subsequent classifiers pay more attention to misclassified samples. The training process of AdaBoost consists of the following steps: (1) Initialize Sample Weights: Each sample’s weight is initialized to an equal value to start the training process. (2) Iteratively Train Weak Classifiers: A weak classifier is trained, its classification accuracy is slightly higher than random guessing, but it is still relatively weak. Weak classifiers can be simple decision trees, Naive Bayes classifiers, etc. (3) Update Sample Weights: Based on the classification results of the weak classifier, adjust the weights of misclassified samples, so that the next weak classifier pays more attention to these misclassified samples. The weights of correctly classified samples are correspondingly reduced. (4) Update Weak Classifier Weights: Calculate the corresponding weights based on the classification accuracy of the weak classifier. Weak classifiers with high accuracy will get more weight. (5) Combine Weak Classifiers: The weak classifiers obtained through iterative training are combined according to their weights to produce the final classification model. The key idea of AdaBoost is to iteratively train weak classifiers and adjust sample weights based on classification results, making the model pay more attention to misclassified samples. This adaptive training method can effectively improve the performance and accuracy of the model. However, AdaBoost also has some limitations. For example, when there are a large number of noise or outlier samples, AdaBoost is prone to overfit these samples, leading to a decrease in the model’s generalization ability. In addition, AdaBoost is sensitive to noise data and outliers.

3.3 Accuracy Assessment

This section will employ various evaluation methods to assess the machine learning algorithms used. In this study, three research standards were used, namely Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and Mean Absolute Error (MAE). These metrics can help us measure the degree of difference between the model’s predicted values and the actual values.

Firstly, the Mean Squared Error (MSE) measures the performance of the model by calculating the sum of squares of the differences between the predicted and actual values. The smaller the MSE, the closer the model’s prediction results are to the actual values. Secondly, the Root Mean Squared Error (RMSE) is the square root of MSE, used to measure the average deviation between the predicted and actual values. Compared to MSE, RMSE is more sensitive to outliers because it takes the square root of the squared error.

Lastly, the Mean Absolute Error (MAE) measures the performance of the model by calculating the average of the absolute values of the differences between the predicted and actual values. The smaller the MAE, the smaller the average difference between the model's prediction results and the actual values. These three evaluation metrics provide multiple perspectives to assess the predictive accuracy of the model. By comparing the MSE, RMSE, and MAE of different models, we can determine which model performs best in prediction. The formulas for calculating these standards are as follows:

$$MSE = \frac{1}{N}\sum_{i=1}^{n} |y_i - y_i'|^2 \tag{2}$$

$$RMSE = \sqrt{\frac{\sum_{i=}^{N} |y_i - y_i'|^2}{N}} \tag{3}$$

$$MAE = \frac{1}{N}\sum_{i=1}^{n} |y_i - y_i'| \tag{4}$$

Where y_i represents the actual measured value (energy consumption), y_i' represents the predicted value, and N is the number of samples.

3.4 Results

In this section, Support Vector Regression (SVR), Neural Networks (NN), Decision Trees (DT), Random Forests (RF), and Adaptive Boosting (AdaBoost) are used to predict the energy consumption target. The results are shown in Figure 2.

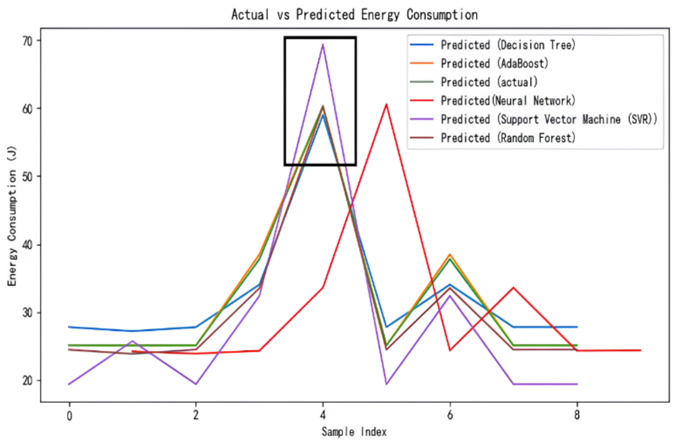


Figure 3: Comparison of Actual Energy Consumption and Predicted Energy Consumption Results

From Figure 3, it can be observed that there is a certain discrepancy between the fitting results obtained by modeling with Support Vector Regression (SVR) and Neural Networks (NN) and the actual data. The results of other algorithms are close to the actual data, but it is not clear from Figure 3 which algorithm performs best. To distinguish similar areas, we magnify Figure 3 locally, and the result is shown in Figure 4.

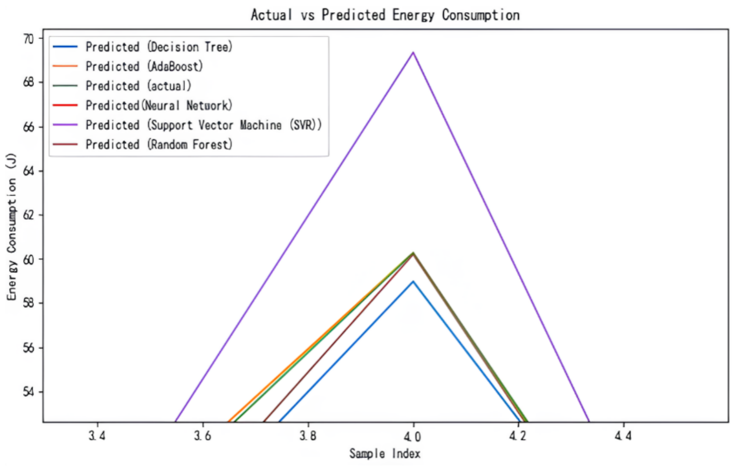


Figure 4: Local Magnification of the Comparison between Actual Energy Consumption and Predicted Energy Consumption Results

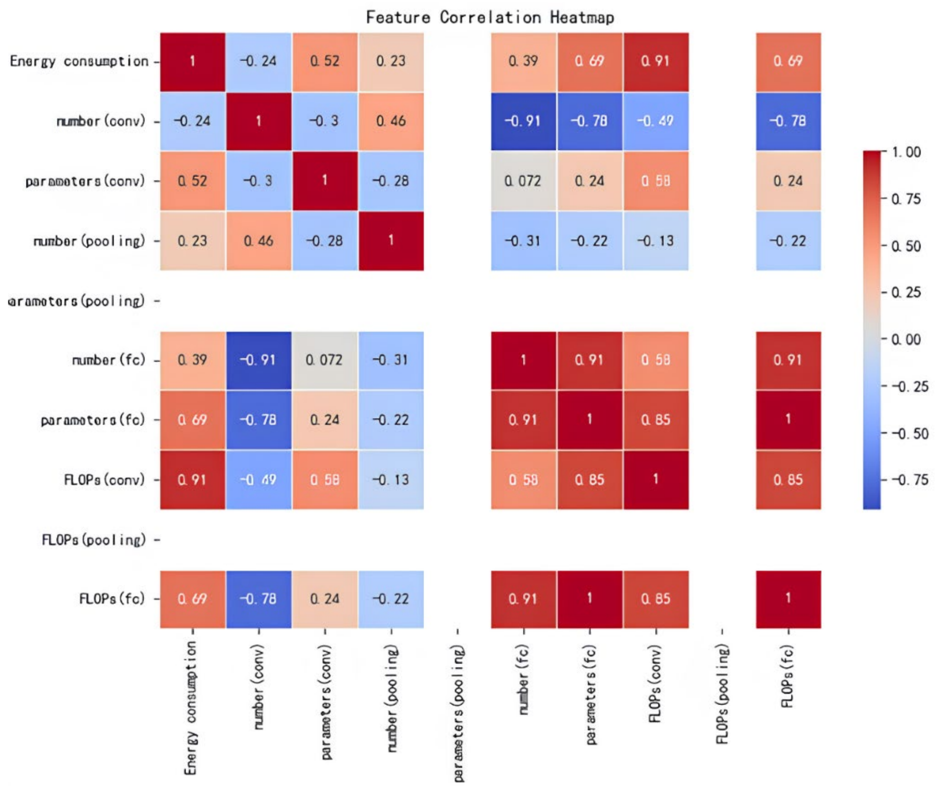
From Figure 4, it is quite evident that, compared to other algorithms, Adaptive Boosting (AdaBoost) is closest to the actual data, suggesting that AdaBoost has a better fitting effect. By observing Figures 3 and 4, we can compare the performance of different algorithms in predicting energy consumption targets. These results can provide information about the prediction accuracy and stability of each algorithm, thereby helping us choose the algorithm that best suits our research purposes. It should be noted that Figures 3 and 4 are just visual presentations of the prediction results. We also need to combine specific evaluation indicators, namely MSE, RMSE, and MAE, to comprehensively evaluate the performance of these algorithms. These indicators will measure the degree of difference between the predicted values and the actual values, thus providing more accurate evaluation results.

Smaller MSE, RMSE, and MAE values indicate that the model's prediction results are closer to the actual values. As can be seen from Table 2, the evaluation results of MSE, RMSE, and MAE are consistent with the results obtained from Figures 3 and 4. The prediction effect of the Adaptive Boosting (AdaBoost) algorithm performs excellently in the three evaluation standards, and its prediction effect is significantly better than other algorithms.

Table 2: Evaluation of Results

Evaluation criteria/ Model	DT	AdaBoost	NN	SVR	RF
MAE	1.69	0.15	4.18	2.57	1.53
MSE	5.86	0.03	22.73	8.22	4.40
RMSE	2.42	0.18	4.77	2.87	2.10

In the previous sections, we proposed a three-tier structure with nine influencing factors for energy consumption modeling. We also analyzed the impact of parameter quantity and FLOPs at each level on energy consumption. It can be inferred that the high computational load of the convolutional layer leads to energy consumption issues. The convolutional layer plays a crucial role in deep learning models, responsible for feature extraction and data processing. Due to the large computational load of the convolutional layer, it accounts for a relatively high proportion of energy consumption in the entire model. This is because the convolutional layer needs to perform a large number of multiplication and addition operations, which contribute significantly to energy consumption. Therefore, this paper speculates that FLOPs are a key factor affecting algorithm energy consumption. To further prove our speculation is correct, we obtained a heatmap as shown in Figure 5.



	Energy consumption	number(conv)	parameters(conv)	number(pooling)	parameters(pooling)	number(fc)	parameters(fc)	FLOPs(conv)	FLOPs(pooling)	FLOPs(fc)
Energy consumption	1	-0.24	0.52	0.23		0.39	0.69	0.91		0.69
number(conv)	-0.24	1	-0.3	0.46		-0.91	-0.78	-0.49		-0.78
parameters(conv)	0.52	-0.3	1	-0.28		0.072	0.24	0.58		0.24
number(pooling)	0.23	0.46	-0.28	1		-0.31	-0.22	-0.13		-0.22
parameters(pooling)										
number(fc)	0.39	-0.91	0.072	-0.31		1	0.91	0.58		0.91
parameters(fc)	0.69	-0.78	0.24	-0.22		0.91	1	0.85		1
FLOPs(conv)	0.91	-0.49	0.58	-0.13		0.58	0.85	1		0.85
FLOPs(pooling)										
FLOPs(fc)	0.69	-0.78	0.24	-0.22		0.91	1	0.85		1

Figure 5: Heatmap

By observing the heatmap in Figure 5, we can intuitively see the contribution of different layers to the energy consumption. If the convolutional layer has a larger proportion in energy consumption, we will see a darker color in the heatmap for this layer, indicating higher energy consumption. Conversely, if other layers have lower energy consumption, we will see lighter colors in the heatmap, indicating lower energy consumption. Through the analysis of the heatmap, we can further confirm the significant contribution of the FLOPs of the convolutional layer to energy consumption. This further validates our conjecture that the computational load of the convolutional layer is large, therefore it occupies a higher proportion in the total energy consumption of the model. It also shows that the FLOPs of each layer are the decisive factor affecting total energy consumption. The results of the heatmap provide us with more intuitive and specific information, helping us to further understand the impact of different layers on energy consumption. This is very helpful for us to choose appropriate strategies to reduce energy consumption and improve the energy efficiency of the model in the design and optimization of deep learning models.

4. Conclusion

In this paper, we compared five advanced machine learning algorithms to evaluate their ability to predict algorithm energy consumption. According to our experimental results and analysis of evaluation indicators, we can conclude that the best algorithm in this study is the Adaptive Boosting (AdaBoost) algorithm. Its fitted data line is closest to the original data line, and the errors of evaluation results such as MSE, RMSE, MAE are the smallest. The research results also show that there is a certain correlation between the level of FLOPs and the energy consumption of the algorithm. Algorithms with higher FLOPs often require more computational resources and energy to execute, so their energy consumption is relatively high. Conversely, algorithms with lower FLOPs require fewer computational resources and energy. This finding helps us to consider FLOPs in algorithm design and optimization to reduce energy consumption and improve algorithm energy efficiency.

References

[1] LIANG H, ZHANG Y, YANG P, et al. Comparison and Analysis of Prediction Models for Locomotive Traction Energy Consumption Based on the Machine Learning [J]. IEEE Access, 2023.
[2] ABDELAZIZ A, SANTOS V, DIAS M S. Convolutional Neural Network With Genetic Algorithm for Predicting Energy Consumption in Public Buildings [J]. IEEE Access, 2023, 11: 64049-69.
[3] CHEN X, CAO B, POURAMINI S. Energy cost and consumption reduction of an office building by Chaotic Satin Bowerbird Optimization Algorithm with model predictive control and artificial neural network: A case study [J]. Energy, 2023, 270: 126874.
[4] ALALI Y, HARROU F, SUN Y. Unlocking the Potential of Wastewater Treatment: Machine Learning Based Energy Consumption Prediction [J]. Water, 2023, 15(13): 2349.
[5] FARD R H, HOSSEINI S. Machine Learning algorithms for prediction of energy consumption and IoT modeling in complex networks [J]. Microprocessors and Microsystems, 2022, 89: 104423.
[6] UYAN O G, AKBAS A, GUNGOR V C. Machine learning approaches for underwater sensor network parameter prediction [J]. Ad Hoc Networks, 2023, 144: 103139.
[7] DAI X, ZHANG P, WU B, et al. Chamnet: Towards efficient network design through platform-aware model adaptation. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, F, 2019 [C].
[8] BOUZIDI H, OUARNOUGHI H, NIAR S, et al. Performance prediction for convolutional neural networks on edge gpus. Proceedings of the 18th ACM International Conference on Computing Frontiers, F, 2021 [C].
[9] JUSTUS D, BRENNAN J, BONNER S, et al. Predicting the computational cost of deep learning models; proceedings of the 2018 IEEE international conference on big data (Big Data), F, 2018 [C]. IEEE.
[10] Brehler M, Camphausen L, Heidebroek B, et al. Making Machine Learning More Energy Efficient by Bringing It Closer to the Sensor [J]. IEEE Micro, 2023.
[11] Bisen D, Lilhore U K, Manoharan P, et al. A Hybrid Deep Learning Model Using CNN and K-Mean Clustering for Energy Efficient Modelling in Mobile EdgeIoT [J]. Electronics, 2023, 12(6): 1384.