Design and Application of Model Compression and Management Platform

Hao Li[1,*], Chuangbo Hao[2]

[1]Beijing Jinghang Computation and Communication Research Institute, Beijing, China
[2]Beijing Jinghang Computation and Communication Research Institute, Beijing, China

*Corresponding author,Email: lihao2021@njust.edu.cn

Abstract. In order to facilitate the deployment of neural network models on edge devices and improve the computational speed of the models, compressing the volume of neural network models has become a key problem. However, currently neural network compression can only be achieved through programming, which is not only inefficient but also inconvenient for model management. Therefore, we design and develop a model compression and management platform that integrates existing model pruning and quantization technologies, while incorporating model management and dataset management functions. It not only facilitates the management of datasets and models, but also meets needs for fast model pruning and quantization.

Keywords: Software design, Model compression, Neural network

1 Introduction

At present, deep neural networks are widely used in Natural language processing, fault diagnosis, prediction and other fields[1]. However, the large number of weights within the deep learning model has caused excessive memory and bandwidth consumption on the device, posing a significant obstacle to the deployment of the deep learning model at the edge and on some embedded platforms. The huge computational workload of the deep learning model also brings a large amount of energy consumption and expensive computing costs[2]. Therefore, it is necessary to prune and quantify the model. However, the current model pruning and quantification mainly relies on running program code, lacking a visual model pruning and quantification operation interface. Therefore, we integrate pruning and quantification technology with software design and development to meet the basic pruning quantification requirements.

The basic idea of model pruning is to prune weight parameters that are not important for model training and testing, and reduce the complexity and volume of the model while ensuring the accuracy of the model as much as possible. In reference [3], a typical pruning process was proposed, and many subsequent pruning studies were basically carried out according to the above process. The main purpose of quantization is to compress the parameters of the model from high bits to low bits[4]. At pre-

A. Rauf et al. (eds.), *Proceedings of the 3rd International Conference on Management Science and Software Engineering (ICMSSE 2023)*, Atlantis Highlights in Engineering 20,
https://doi.org/10.2991/978-94-6463-262-0_64

sent, Tensorflow model quantization technology is widely used for model compression, which can be divided into quantized perception training and post-training quantization[5]. Quantized perception training has higher accuracy compared to post training quantization models, but post training quantization is more convenient to use and has little accuracy loss, making it more widely used in practical applications[6].

We focus on the current demand for visual model compression tools, and combines the aforementioned model pruning and quantification technology to design and develop a model compression and management platform. On the premise of ensuring model accuracy, this platform reduces the memory requirements of the model and improves the computational speed of the model. It facilitates the unified management of models and datasets by users. More importantly, users only need to input and select model compression parameters to perform model pruning and quantization operations, greatly facilitating the processing of model compression.

2 Design of Model Compression and Management Platform

2.1 Requirement Analysis

The application object of our research platform is the Keras model. Based on existing technical conditions, the design requirements for the model compression and management platform of this model are summarized as follows:

(1) The user login must be authenticated.

(2) Equipped with dataset management function, users can upload and manage datasets.

(3) Users can easily manage artificial intelligence models.

(4) Users can perform pruning quantization on artificial intelligence models while only setting pruning quantization parameters, and can download the compressed model locally.

2.2 Functional Design

According to the requirement analysis, the functional design of the model compression and management platform is shown in Figure 1.

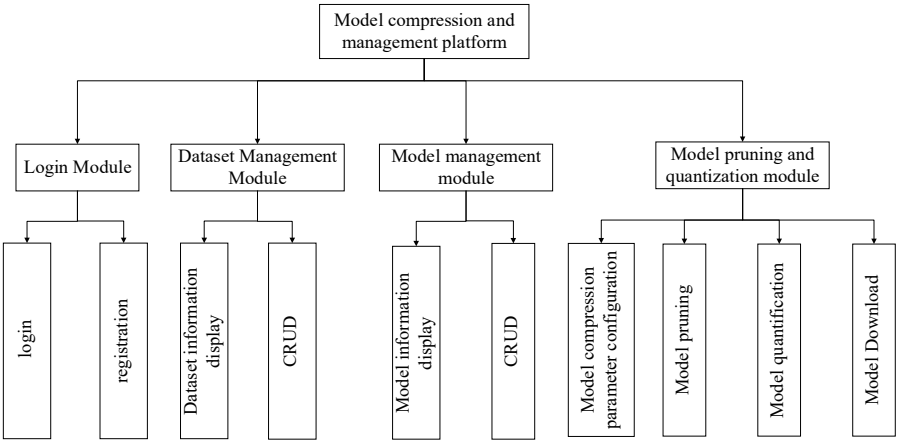


Fig. 1. Functional design diagram of model compression and management platform.

The functional modules of this platform can be specifically divided into the following four parts:

(1) The first part is the login module, which includes user login and registration functions.

(2) The second part is the dataset management module, which includes the display of dataset information and the function of adding, deleting, modifying, and querying datasets.

(3) The third part is the model management module, which includes the information display of the model and the function of adding, deleting, modifying, and querying the model.

(4) The fourth part is the model pruning and quantization module, including model compression parameter configuration, model pruning, model quantization, and model download functions.

2.3 Functional Realization

We conduct front-end and back-end development of the platform based on the above requirement analysis and functional design. The backend development environment used is Anaconda virtual environment, using Django to build the backend framework and Python as the backend development language. The front-end framework is built using Vue, and the front-end style adopts Element UI. The following are the platform development results:

(1) Login module. The most important part of the login module is user login verification. Traditional server based authentication has problems such as high server overhead, poor scalability, cross domain and vulnerability to cross-site request forgery (CSRF) attacks. Using Json Web Token (JWT) will solve these problems[7]. Although both are used to store user information, JWT disperses user information on the client side, reducing server overhead and memory pressure. Therefore, the user login module of this platform uses JWT for login verification. JWT is a digital signature authen-

tication mechanism used for secure transmission of JSON objects. JWT consists of three parts[8]. The first part is the header, which includes the encryption algorithm and token type. The encryption algorithm we use is HS256, and the token type is JWT. The second part is the payload, which includes JSON objects for data declaration. We declare user information and expiration timestamp in the payload. The third part is the signature, which consists of the encoded header , payload, the secret key and the signature algorithm in the header, mainly used to verify whether the information has been changed.

(2) Dataset management module. The main purpose of dataset management function is to facilitate unified management of datasets. The main part of the interface is a list display of dataset information, which mainly displays the ID number, name, storage address, and dataset alias of the dataset.

(3) The main function of model management is to facilitate users to manage models uniformly. The model management function is generally consistent with the dataset management function.

(4) Model pruning and quantization module. Model pruning and quantification functions are key functions of this platform, mainly used for compressing models. We use the form of step bars to gradually prune and quantify the model, and the specific steps are as follows:

Step 1: Select a pre compressed model, where users can choose existing models in model management or upload local models.

Step 2: Select the existing training dataset of the corresponding model or upload the local dataset, and select the optimizer and Loss function for model training.

Step 3: Perform model pruning by configuring a series of pruning parameters such as number of fine-tuning times, training batch, start pruning steps, end pruning steps, validation set proportion, pruning frequency steps, initial sparsity, and target sparsity.

Step 4: We provide three processing methods for users to choose from for the previous model: quantization and conversion, only conversion without quantization, and no conversion without quantization. When users choose to quantify and transform the model, they need to choose the quantization strategy and quantization target of the model.

Step 5: Users can download the model locally through the "Download" button.

3 Application of Model Compression and Management Platform

3.1 Data Source

The vibration data of the rolling bearings used in the experiment is sourced from the PHM2012 data challenge's rolling bearing dataset[9]. This dataset contains vibration acceleration data in both horizontal and vertical directions during the operation of 17 rolling bearings under three different working conditions, with a sampling frequency of 25.6kHz. The data is recorded every 10s, and a single sampling time of 0.1s. We use vibration data from bearings 2-6 for testing.

3.2 Result and Analysis

First, we prune the Convolutional neural networks (CNN) model. The pruning parameter takes 50% of the sparsity as the initial sparsity, the final target sparsity is set to 90%, the starting pruning step is 0, the ending pruning step is 1000, and the pruning frequency is 100. After pruning, we fine tune the model. The comparison of CNN model structure before and after pruning is shown in Figure 2. It can be seen that the model after pruning has reduced the weight parameters by nearly half compared to before pruning.

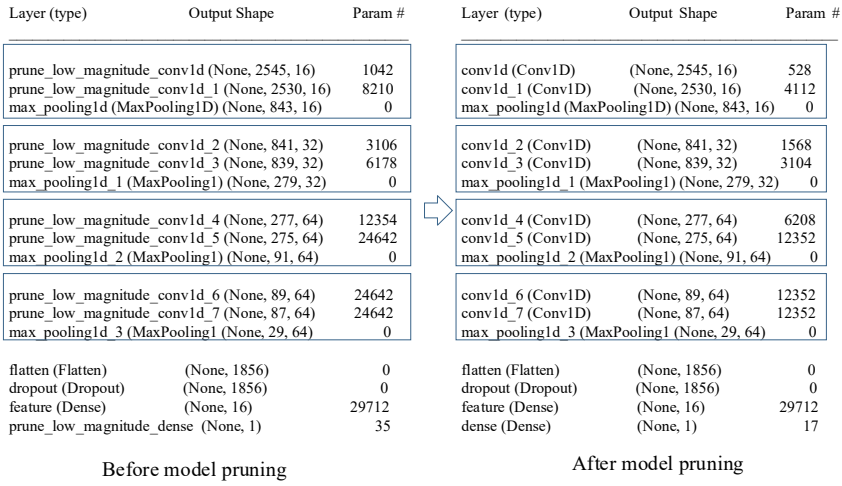


Fig. 2. Comparison diagram of CNN model structure before and after pruning.

For the Long Short-Term Memory (LSTM) model, we set pruning parameters to be consistent with CNN pruning parameters. The comparison of LSTM model structure before and after pruning is shown in Figure 3. From the figure, it can be seen that the pruned LSTM model also reduces the weight parameters by nearly half compared to before pruning.

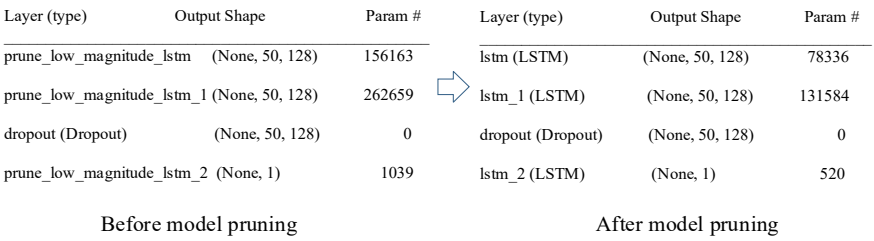


Fig. 3. Comparison diagram of LSTM model structure before and after pruning.

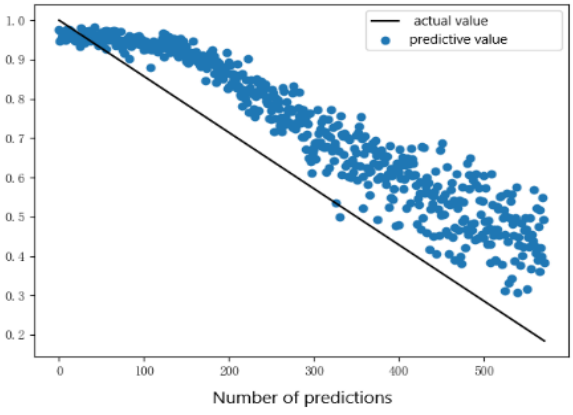
We test the CNN model with bearings 2-6, using model volume, model calculation speed, and model prediction error as evaluation indicators for model compression

performance[10]. The calculation formula for model prediction error is shown in equation (1).

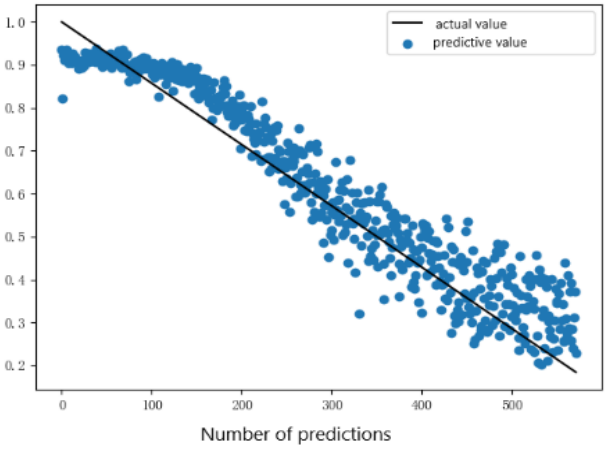
$$E = \frac{P-A}{A} * 100\% \tag{1}$$

In the formula, E represents the model prediction error, P represents the model prediction value, and A represents the actual value.

We input the vibration signals of the bearings into the CNN models before and after compression, and obtain a comparison of the predicted results as shown in Figure 4.



(a)



(b)

Fig. 4. The dashed line represents the predicted degradation state of rolling bearings, while the solid line represents the actual degradation state of rolling bearings. The image labeled (a) shows the predicted results of the CNN model before compression, and the image labeled (b) shows the predicted results of the CNN model after compression.

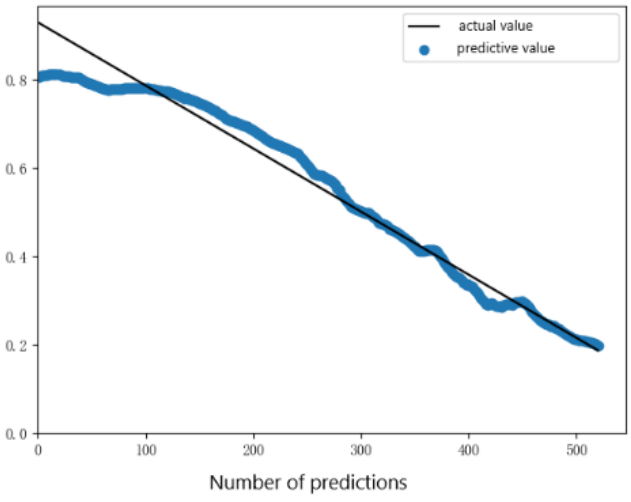
It can be seen that the predicted results of the compressed model are more in line with the actual curve, indicating that the compressed model eliminates redundant parameters and enhances the effective calculation of the model, thereby improving the overall prediction accuracy.

Using a certain input sequence of bearings to test the accuracy of the model, the actual value is known to be 0.3074. The predicted value before model compression is 0.3844, the predicted value after model pruning is 0.2278, and the predicted value after model pruning and quantization is 0.2284. The prediction error is calculated by combining equation (1). Table 1 shows the test results before and after model compression. It can be seen that the volume of the CNN model after pruning and quantization has decreased by 10 times compared to before compression, and the calculation speed has increased by two times. However, the prediction error is not significantly different, proving that the pruning and quantization methods are effective in compressing the CNN model.

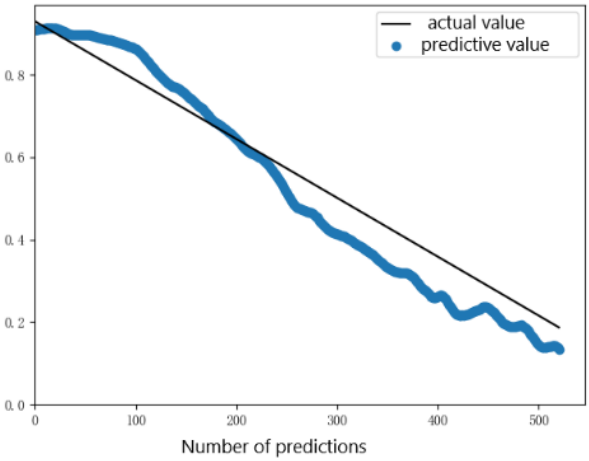
Table 1. CNN model evaluation index value

CNN model	Model volume	Model calculation speed	Model prediction error
initial model	1058KB	271ms	**25.05%**
Model after pruning	366KB	263ms	25.89%
Model after pruning and quantification	**103**KB	**136**ms	25.70%

We continue to test the LSTM model, and the test results are shown in Figure 5. It can be seen that the prediction performance of the LSTM model after compression is not as good as that of the pre compression model in the later stage of degradation, but the overall curve still fits the actual degradation curve relatively well, and the accuracy loss is still within an acceptable range.



(a)



(b)

Fig. 5. The dashed line represents the predicted degradation state of rolling bearings, while the solid line represents the actual degradation state of rolling bearings. The image labeled (a) shows the predicted results of the LSTM model before compression, and the image labeled (b) shows the predicted results of the LSTM model after compression.

We test the LSTM model using a feature sequence of a certain segment of the bearing, and also evaluate the compression performance of the model based on model volume, model calculation speed, and model prediction accuracy. The actual value is known to be 0.1871, the predicted value before model compression is 0.1982, the predicted value after model pruning is 0.2191, and the predicted value after model pruning and quantization is 0.1345. The prediction error is calculated by combining equation (1).

Table 2 shows the calculation results of evaluation indicators for the LSTM model before and after compression. It can be seen that the volume of the pruned and quantized model has decreased by 10 times compared to before compression, and the calculation speed of the model has been greatly improved, but the prediction accuracy of the model has also been slightly reduced. However, in the case of a significant improvement in model calculation speed, the loss of model prediction accuracy is acceptable.

Table 2. LSTM model evaluation index value

LSTM model	Model volume	Model calculation speed	Model prediction error
initial model	2515KB	271ms	**5.93%**
Model after pruning	844KB	263ms	17.10%
Model after pruning and quantification	**238KB**	**0.0130ms**	28.11%

4 Conclusions

We design a model compression and management platform based on existing model pruning and quantification technologies to meet the convenient usage needs of current model pruning and quantification tools. The platform we designed achieves centralized management of models and datasets, as well as model pruning and quantification functions. We also use this platform to prune and quantify CNN and LSTM models. Through comparison, it has been verified that this platform can effectively prune and quantify models, and greatly improves the computational efficiency of models.

ACKNOWLEDGMENT

This work is supported by the National Basic Scientific Research Program of China (Grant No. JCKY2020204C021 and No. JCKY2019602B002). We would like to thank both the editors and the reviewers for providing helpful comments.

REFERENCES

1. Liu X, Tian S, Tao F, et al. A review of artificial neural networks in the constitutive modeling of composite materials[J].Composites, Part B. Engineering, 2021(Nov.1):224.
2. Yu Z, Gao Z, Wang J, et al. Research on Network Resource Management Technology in Multi Access Network Environment of Terminal Communication Access Network[C]//2018.DOI:10.1088/1757-899X/366/1/012054.
3. S. Han, J. Pool, J. Tran, et al. Learning both weights and connections for efficient neural networks[C]. Proceedings of the 29th annual conference on neural information processing systems. 2015, 1135-1143.
4. Tiehu F, Guihe Q, Qi Z. Research on an Improved BP Neural Network Based on Fast Quantized Orthogonal Genetic Algorithm[C]//International Conference on Natural Computation. IEEE Press, 2009.DOI:10.1109/ICNC.2009.254.
5. Ma H, Qiu H, Gao Y, et al. Quantization Backdoors to Deep Learning Models[J]. 2021. DOI:10.48550/arXiv.2108.09187.
6. Chang-Jun S I, Department P T. On the Practical Value of the Training Model of Perceptual Ability for Students Majoring in Air Security[J].Journal of Liaoning Police College, 2018.
7. Shingala K. JSON Web Token (JWT) based client authentication in Message Queuing Telemetry Transport (MQTT)[J]. 2019.DOI:10.48550/arXiv.1903.02895.
8. Rahmatulloh A, Sulastri H, Nugroho R. Keamanan RESTful Web Service Menggunakan JSON Web Token (JWT) HMAC SHA-512[J]. Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI), 2018, 7(2).DOI:10.22146/jnteti.v7i2.417.
9. Cao K, Liu Y, Meng G, et al. An Overview on Edge Computing Research[J]. IEEE Access, 2020, PP(99): 1-1
10. Cheng Y, Wang D, Zhou P, et al. A Survey of Model Compression and Acceleration for Deep Neural Networks[J]. 2017.DOI:10.48550/arXiv.1710.09282.