MobileNet Architecture for Identification of Biomedical Instruments

Prashant Dhope[1]

[1]PG Scholar, Dept. of E&C Engineering, Center for PG Studies, VTU Belagavi, Karnataka, India

---***---

Abstract – *In the medical field, biomedical instruments serve a critical role in assisting physicians in diagnosing and treating patients. There are numerous developments in the field of deep learning in today's day. Using the MobileNet architecture, this study demonstrates the identification of biomedical instruments used in medical areas. For training and validation purposes, the study uses a self-prepared dataset comprising of twelve different biomedical instruments. The proposed methodology delivers training and validation accuracy of 90.61% and 77.42% respectively, according to the experimental evidence.*

Key Words: Biomedical Instruments, Biomedical Engineering, Convolutional Neural Network, MobileNet, Deep Learning, Self Prepared Dataset.

1. INTRODUCTION

Biomedical Instrumentation is a branch of biomedical engineering concerned with the instruments and mechanics that are used to compute, assess, and serve biological systems. In the medical field, a variety of biomedical devices are used for treatment and diagnostics. Multiple sensors are used in biomedical equipment to measure a person's physiological characteristics. Machines and humans are now able to speak with one other because to Artificial Intelligence (AI). Machine Learning (ML) and Deep Learning (DL) have made significant contributions to the development of intelligent healthcare and medical industries. Machines such as robots, for example, may be able to undertake surgeries or surgery without the help of a doctor or physician [1].

The goal of this study is to use the proposed MobileNet architecture to create a model for identifying various biomedical devices. The rest of the paper is organized as follows: section 2 summarizes the related work, section 3 elaborates the proposed methodology for constructing a model, section 4 summarizes the experimental findings, and finally section 5 concludes the study.

2. RELATED WORK

In paper [1], the author proposed a methodology for identification of biomedical instruments based on Convolutional Neural Network (CNN) algorithm. The performance of this study had a training and validation accuracy of 79.56% and 57.42% respectively. Thus, there is a need to improve the performance of the system.

3. PROPOSED METHODOLOGY

Fig -1: Block Diagram of Proposed Methodology

In Fig. 1, the block diagram of proposed methodology is depicted that consists of mainly five stages: (i) Dataset Creation, (ii) Data Pre-processing, (iii) Data Splitting, (iv) Design of MobileNet Architecture and (v) Training & Validation phase.

3.1 Dataset Creation

In this study, the self prepared dataset from study [1] is used, which contains total 517 images of twelve biomedical instruments namely Audiometer, CT scanner, Cannula, Dialyzer, Defibrillator, Digital BP meter, Enema bulb, Needle electrode, Ophthalmoscope, Stethoscope, Syringe and Sphygmomanometer. The Fig. 2 depicts the sample images from the dataset of study [1].

Fig -2: Sample Images from the Dataset of study [1]

3.2 Data Pre-processing

In this stage, all the images from the dataset [1] are resized to a fixed dimension of 224 x 224 (width x height).

3.3 Data Splitting

The complete dataset is split into two groups: Training and Validation. Seventy percent of the data is utilized for training, while the remaining thirty percent is used for validation. The training dataset has 362 images, whereas the validation dataset contains 155 images after the train-validation split.

3.4 Design of MobileNet Architecture

MobileNet is a convolutional neural network (CNN) intended for mobile and embedded vision applications. They are based on a streamlined architecture that builds lightweight deep neural networks (DNN) with low latency for mobile and embedded devices using depth-wise separable convolutions. The Fig. 3 highlights the summary of proposed mobileNet architecture.

Model: "model_4"

Layer (type)	Output Shape	Param #
input_5 (InputLayer)	[(None, 224, 224, 3)]	0
conv1 (Conv2D)	(None, 112, 112, 32)	864
conv1_bn (BatchNormalization	(None, 112, 112, 32)	128
conv1_relu (ReLU)	(None, 112, 112, 32)	0
conv_dw_1 (DepthwiseConv2D)	(None, 112, 112, 32)	288
conv_dw_1_bn (BatchNormaliza	(None, 112, 112, 32)	128
conv_dw_1_relu (ReLU)	(None, 112, 112, 32)	0
conv_pw_1 (Conv2D)	(None, 112, 112, 64)	2048
conv_pw_1_bn (BatchNormaliza	(None, 112, 112, 64)	256
conv_pw_1_relu (ReLU)	(None, 112, 112, 64)	0
conv_pad_2 (ZeroPadding2D)	(None, 113, 113, 64)	0
conv_dw_2 (DepthwiseConv2D)	(None, 56, 56, 64)	576
conv_dw_2_bn (BatchNormaliza	(None, 56, 56, 64)	256
conv_dw_2_relu (ReLU)	(None, 56, 56, 64)	0
conv_pw_2 (Conv2D)	(None, 56, 56, 128)	8192
conv_pw_2_bn (BatchNormaliza	(None, 56, 56, 128)	512
conv_pw_2_relu (ReLU)	(None, 56, 56, 128)	0
conv_dw_3 (DepthwiseConv2D)	(None, 56, 56, 128)	1152
conv_dw_3_bn (BatchNormaliza	(None, 56, 56, 128)	512
conv_dw_3_relu (ReLU)	(None, 56, 56, 128)	0
conv_pw_3 (Conv2D)	(None, 56, 56, 128)	16384
conv_pw_3_bn (BatchNormaliza	(None, 56, 56, 128)	512
conv_pw_3_relu (ReLU)	(None, 56, 56, 128)	0
conv_pad_4 (ZeroPadding2D)	(None, 57, 57, 128)	0
conv_dw_4 (DepthwiseConv2D)	(None, 28, 28, 128)	1152
conv_dw_4_bn (BatchNormaliza	(None, 28, 28, 128)	512
conv_dw_4 (DepthwiseConv2D)	(None, 28, 28, 128)	1152
conv_dw_4_bn (BatchNormaliza	(None, 28, 28, 128)	512
conv_dw_4_relu (ReLU)	(None, 28, 28, 128)	0
conv_pw_4 (Conv2D)	(None, 28, 28, 256)	32768
conv_pw_4_bn (BatchNormaliza	(None, 28, 28, 256)	1024
conv_pw_4_relu (ReLU)	(None, 28, 28, 256)	0
conv_dw_5 (DepthwiseConv2D)	(None, 28, 28, 256)	2304
conv_dw_5_bn (BatchNormaliza	(None, 28, 28, 256)	1024
conv_dw_5_relu (ReLU)	(None, 28, 28, 256)	0
conv_pw_5 (Conv2D)	(None, 28, 28, 256)	65536
conv_pw_5_bn (BatchNormaliza	(None, 28, 28, 256)	1024
conv_pw_5_relu (ReLU)	(None, 28, 28, 256)	0
conv_pad_6 (ZeroPadding2D)	(None, 29, 29, 256)	0
conv_dw_6 (DepthwiseConv2D)	(None, 14, 14, 256)	2304
conv_dw_6_bn (BatchNormaliza	(None, 14, 14, 256)	1024
conv_dw_6_relu (ReLU)	(None, 14, 14, 256)	0
conv_pw_6 (Conv2D)	(None, 14, 14, 512)	131072
conv_pw_6_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_pw_6_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_7 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_7_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_dw_7_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_7 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_7_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_pw_7_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_8 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_8_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_dw_8_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_8 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_8_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_pw_8_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_9 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_9_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_dw_9_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_9 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_9_bn (BatchNormaliza	(None, 14, 14, 512)	2048
conv_pw_9_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_10 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_10_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_dw_10_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_10 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_10_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_pw_10_relu (ReLU)	(None, 14, 14, 512)	0
conv_dw_11 (DepthwiseConv2D)	(None, 14, 14, 512)	4608
conv_dw_11_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_dw_11_relu (ReLU)	(None, 14, 14, 512)	0
conv_pw_11 (Conv2D)	(None, 14, 14, 512)	262144
conv_pw_11_bn (BatchNormaliz	(None, 14, 14, 512)	2048
conv_pw_11_relu (ReLU)	(None, 14, 14, 512)	0
conv_pad_12 (ZeroPadding2D)	(None, 15, 15, 512)	0
conv_dw_12 (DepthwiseConv2D)	(None, 7, 7, 512)	4608
conv_dw_12_bn (BatchNormaliz	(None, 7, 7, 512)	2048

```
conv_dw_12_relu (ReLU)        (None, 7, 7, 512)      0
conv_pw_12 (Conv2D)           (None, 7, 7, 1024)     524288
conv_pw_12_bn (BatchNormaliz  (None, 7, 7, 1024)     4096
conv_pw_12_relu (ReLU)        (None, 7, 7, 1024)     0
conv_dw_13 (DepthwiseConv2D)  (None, 7, 7, 1024)     9216
conv_dw_13_bn (BatchNormaliz  (None, 7, 7, 1024)     4096
conv_dw_13_relu (ReLU)        (None, 7, 7, 1024)     0
conv_pw_13 (Conv2D)           (None, 7, 7, 1024)     1048576
conv_pw_13_bn (BatchNormaliz  (None, 7, 7, 1024)     4096
conv_pw_13_relu (ReLU)        (None, 7, 7, 1024)     0
global_average_pooling2d_4 (  (None, 1024)           0
dense_16 (Dense)              (None, 1024)           1049600
dense_17 (Dense)              (None, 1024)           1049600
dense_18 (Dense)              (None, 512)            524800
dense_19 (Dense)              (None, 12)             6156
=================================================================
Total params: 5,859,020
Trainable params: 5,837,132
```

Fig -3: Summary of Proposed MobileNet Architecture

3.5 Training and Validation Phase

The parameters defined in the training phase of the MobileNet architecture are highlighted in Table 1. The training data is used to develop the prediction model, while the validation data is used to assess the model's performance. During the training phase of MobileNet, callback functions such as ModelCheckpoint(), ReduceLROnPlateau(), and CSVLogger() are employed. After the of the training phase, the prediction model (.h5 file) is created, which is utilised to make biomedical instrument predictions.

Table -1: Training Parameters of MobileNet

Sl. No.	Parameter	Value
1.	Batch Size	8
2.	Number of Epochs	60
3.	Learning Rate	0.001
4.	Metric	Accuracy
5.	Optimizer	Adam

The accuracy and loss rate for the first 10 epochs of the training and validation phase are shown in Fig. 4.

```
Epoch 1/60
45/45 [==============================] - 11s 170ms/step - loss: 2.3677 - accuracy: 0.2514 - val_loss: 4.4912 - val_accuracy: 0.1053

Epoch 00001: val_loss improved from inf to 4.49123, saving model to Biomedical_Instruments_MobilNet.h5
Epoch 2/60
45/45 [==============================] - 7s 156ms/step - loss: 1.7784 - accuracy: 0.3814 - val_loss: 4.0525 - val_accuracy: 0.2829

Epoch 00002: val_loss improved from 4.49123 to 4.05252, saving model to Biomedical_Instruments_MobilNet.h5
Epoch 3/60
45/45 [==============================] - 7s 159ms/step - loss: 1.7894 - accuracy: 0.3729 - val_loss: 3.1733 - val_accuracy: 0.1513

Epoch 00003: val_loss improved from 4.05252 to 3.17333, saving model to Biomedical_Instruments_MobilNet.h5
Epoch 4/60
45/45 [==============================] - 7s 155ms/step - loss: 1.7919 - accuracy: 0.3814 - val_loss: 10.8706 - val_accuracy: 0.1382

Epoch 00004: val_loss did not improve from 3.17333
Epoch 5/60
45/45 [==============================] - 7s 153ms/step - loss: 1.8881 - accuracy: 0.3729 - val_loss: 3.3654 - val_accuracy: 0.3487

Epoch 00005: val_loss did not improve from 3.17333
Epoch 6/60
45/45 [==============================] - 7s 156ms/step - loss: 1.6368 - accuracy: 0.4222 - val_loss: 1.3821 - val_accuracy: 0.5066

Epoch 00006: val_loss improved from 3.17333 to 1.38209, saving model to Biomedical_Instruments_MobilNet.h5
Epoch 7/60
45/45 [==============================] - 7s 155ms/step - loss: 1.5375 - accuracy: 0.4802 - val_loss: 2.3862 - val_accuracy: 0.3224

Epoch 00007: val_loss did not improve from 1.38209
Epoch 8/60
45/45 [==============================] - 7s 159ms/step - loss: 1.5449 - accuracy: 0.4774 - val_loss: 1.7144 - val_accuracy: 0.4342

Epoch 00008: val_loss did not improve from 1.38209
Epoch 9/60
45/45 [==============================] - 7s 156ms/step - loss: 1.5780 - accuracy: 0.4661 - val_loss: 1.8011 - val_accuracy: 0.4145

Epoch 00009: val_loss did not improve from 1.38209
Epoch 10/60
45/45 [==============================] - 7s 158ms/step - loss: 1.5234 - accuracy: 0.4718 - val_loss: 1.8906 - val_accuracy: 0.4145

Epoch 00010: val_loss did not improve from 1.38209
```

Fig -4: Snapshot of Initial 10 Epochs

4. EXPERIMENTAL RESULTS

In this work, the Python programming language is used to train and assess the model using Google Colab notebook. For MobileNet training, the Keras and Tensorflow libraries are used. The accuracy metric is used to assess the prediction model's performance.

The training and validation accuracy of the prediction model for the full 60 epochs is shown in Fig. 5. The training and validation accuracy obtained is 90.61% and 77.42% respectively in the experimental results.

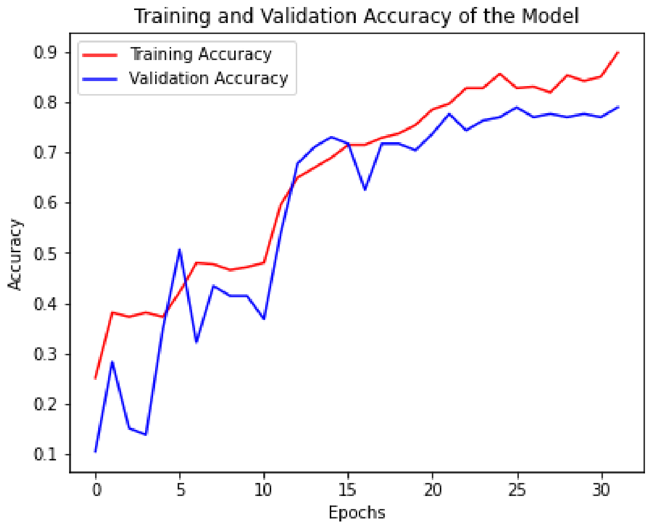

Fig -5: Training and Validation Accuracy of the Model

The training and validation loss of the prediction model for the full 60 epochs is shown in Fig. 6. The training and validation loss obtained is 0.3102 and 0.6895 respectively in the experimental results.

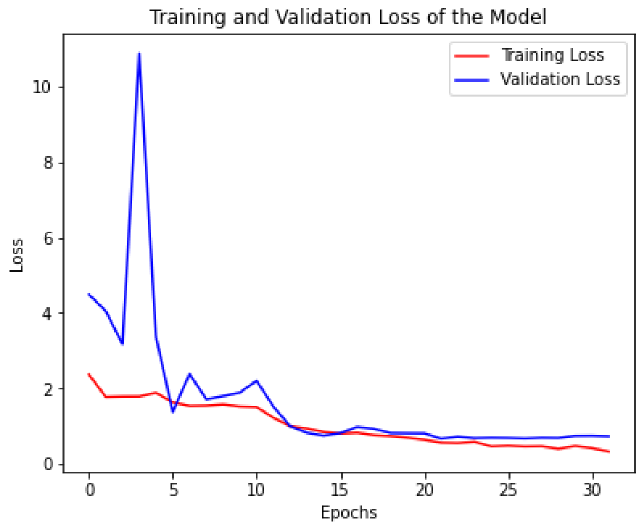

Fig -6: Training and Validation Loss of the Model

The Table 2 is a tabular summary of the above experimental data.

Table -2: Experimental Results of Proposed Methodology

Sl. No.	Performance Metrics	Training	Validation
1.	Accuracy	90.61%	77.42%
2.	Loss	0.3102	0.6895

4.1 Comparison of Results

The Table 3 compares the experimental results of proposed methodology with the study [1]. It is clear that for the identification of biomedical instruments, the proposed methodology using MobileNet architecture provides better results as compared to study [1].

Table -3: Comparison of Results

Reference	Performance Metrics	Training	Validation
[1]	Accuracy	79.56%	57.42%
	Loss	2.9790	4.1020
Proposed Methodology	Accuracy	90.61%	77.42%
	Loss	0.3102	0.6895

5. CONCLUSION

In comparison to study [1], the proposed methodology for identifying biomedical instruments using MobileNet architecture yields excellent results. The proposed study is confined to only 12 biomedical devices; however it can be expanded by integrating a larger dataset of biomedical instruments. In the future, the proposed methodology could aid the robots for identifying biomedical instruments more accurately while performing the medical tasks.

REFERENCES

[1] Prashant Dhope, "Identification of Biomedical Instruments Using CNN Architecture", International Research Journal of Engineering and Technology (IRJET), Volume 8, Issue 8, August 2021, pp. 3926 - 3929, e-ISSN: 2395-0056.

[2] R.S.Khandpur, Handbook of Biomedical Instrumentation, 2^{nd} Edition, Tata McGraw Hill, 2003.

[3] Joseph J. Carr, Biomedical Equipment- Use, Maintenance & Management, Prentice Hall, 1991.

[4] Lan Good fellow and YoshuaBengio, Deep Learning, MIT Press, 2016.

BIOGRAPHY

Prashant Dhope
M.Tech (VLSI Design and Embedded Systems), Department of Electronics & Communication Engineering, Center for PG Studies, VTU Belagavi, Karnataka, India – 590 018.