JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

IMAGE CLASSIFICATION USING CNN WITH CIFAR-10 DATASET

Aravind Nethi, Jayanth Nakka, Vishnu Mitra Mullapudi, Hassan Ali Mohammed, Hanumantha Rao

Student (B. Tech), Student (B. Tech), Student (B. Tech), Student (B. Tech), Assistant Professor
Artificial Intelligence and Machine Learning,
Malla Reddy University, Hyderabad, India

Abstract : Image classification is a fundamental problem in computer vision that involves assigning labels to images based on their content. Convolutional Neural Networks (CNNs) have emerged as a powerful technique for image classification due to their ability to learn hierarchical representations directly from raw pixel data. In this paper, we explore the use of CNNs for image classification using the CIFAR-10 dataset.

Keywords *- Image classification, Convolution neural network (CNN), CIFAR-10 dataset, Computer vision, Deep learning, Image recognition*

I. INTRODUCTION

Image classification is a fundamental problem in the field of computer vision, with a wide range of applications such as object recognition, autonomous driving, and medical imaging. Convolutional Neural Networks (CNNs) have revolutionized the field by providing highly accurate and efficient solutions for image classification tasks. In this paper, we focus on the task of image classification using CNNs with the CIFAR-10 dataset. The CIFAR-10 dataset is a widely used benchmark dataset consisting of 60,000 32x32 color images belonging to ten different classes. Each class represents a specific object category, including common objects such as cars, birds, cats, airplanes, and more. The dataset is divided into a training set with 50,000 images and a test set with 10,000 images, ensuring a comprehensive evaluation of the model's performance.

.

II. EASE OF USE

1) Dataset Accessibility

The availability and accessibility of the CIFAR-10 dataset play a significant role in the ease of use. If the dataset is readily available and well-documented, it becomes easier for researchers and practitioners to access and integrate it into their experiments or applications.

2) Frameworks and Libraries

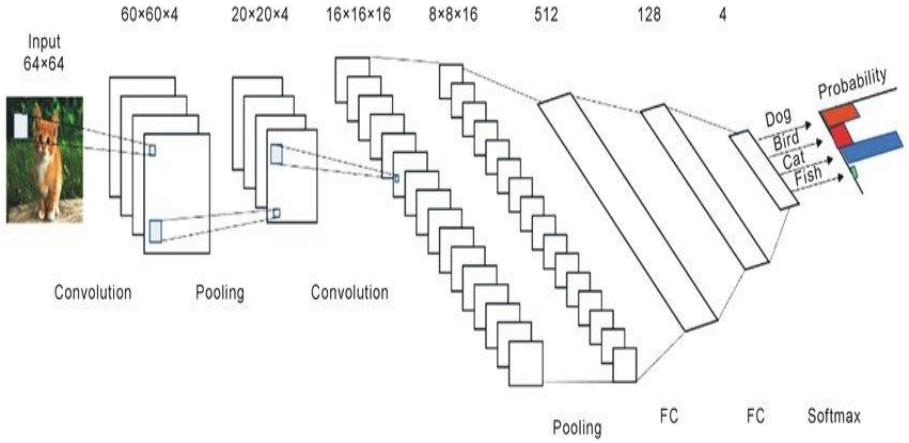
The availability of user-friendly deep learning frameworks and libraries greatly contributes to ease of use. Popular frameworks like TensorFlow and PyTorch offer high-level APIs and pre-implemented CNN architectures, making it easier for users to build and train models without needing an in-depth understanding of low-level implementation details.

III. ALGORITHM

- *Data Preparation*
  Load the CIFAR-10 dataset, consisting of labeled training and test images.

- *CNN Model Architecture*
  Define the architecture of the CNN model, specifying the number and types of layers.

- *Training & Testing*
  Split the preprocessed training dataset into training and validation sets for model evaluation. Pass the test images through the trained model and compute the predicted class labels.



IV. METHODOLOGY

1) Convolutional Neural Networks (CNNs)

CNNs are the primary method used for image classification tasks. They are designed to automatically learn hierarchical representations from raw pixel data, enabling effective feature extraction and classification.

2) Data Augmentation

Data augmentation techniques involve applying random transformations to the training images, such as rotations, translations, flips, and changes in brightness or contrast. Augmentation increases the diversity and variability of the training data, helping the model generalize better to unseen images.

3) Transfer Learning

Transfer learning involves leveraging pre-trained CNN models, such as VGG, ResNet, or Inception, which were trained on large-scale datasets like ImageNet. By utilizing the learned features from these models, one can significantly reduce the training time and improve the performance on the target CIFAR-10 dataset.

4) Regularization Techniques

Regularization techniques, such as Dropout and Batch Normalization, are commonly used to prevent overfitting and improve the generalization capability of the model. Dropout randomly sets a fraction of the input units to zero during training, while Batch Normalization normalizes the activations of the previous layer, reducing internal covariate shift.

5) Hyperparameter Tuning

Fine-tuning the hyperparameters of the CNN model is crucial for achieving optimal performance. This includes tuning parameters such as learning rate, batch size, number of layers, filter sizes, and the number of neurons in the fully connected layers. Techniques like grid search or random search can be employed to explore different hyperparameter combinations.

6) Gradient-based Optimization

Stochastic Gradient Descent (SGD) is a widely used optimization algorithm for training CNN models. It updates the model parameters based on the gradients computed during backpropagation. Other variants of gradient-based optimization methods, such as Adam, RMSprop, or AdaGrad, can also be employed for faster convergence and improved performance.

7) Learning Rate Scheduling

Adjusting the learning rate during training can have a significant impact on model performance. Techniques like learning rate decay or learning rate schedules, where the learning rate is reduced over time, can help the model converge to better solutions and improve generalization.

.

V. IMPLEMENTATION

1) Dataset Preparation

- Download the CIFAR-10 dataset, which consists of labeled training and test images.
- Preprocess the images by resizing them to a fixed size and applying any necessary normalization or augmentation techniques.

2) Model Creation

- Import a deep learning framework, such as TensorFlow or PyTorch, and set up the necessary environment.
- Define the architecture of the CNN model. This includes specifying the number and types of layers, filter sizes, activation functions, and pooling operations.
- Compile the model by specifying the loss function, optimization algorithm, and evaluation metrics to be used during training.

3) Training

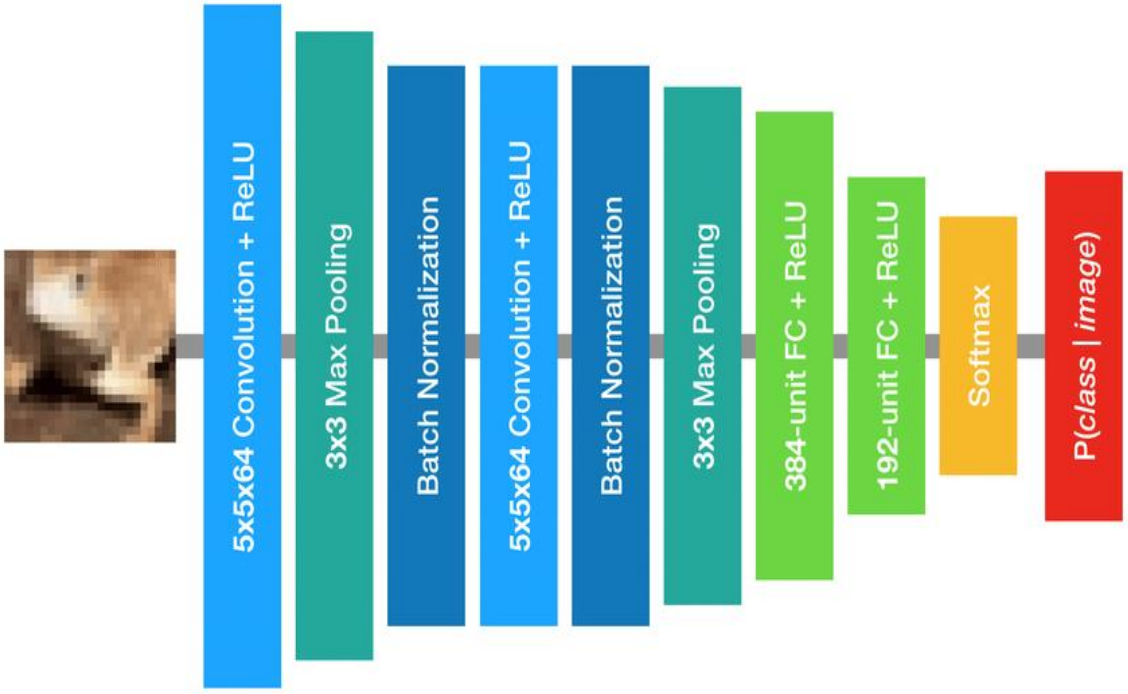
- Split the preprocessed training dataset into training and validation sets.
- Iterate over the training set for a fixed number of epochs:
- Perform forward propagation by passing the input images through the layers of the CNN model.
- Calculate the loss using the defined loss function.
- Perform backward propagation to compute gradients and update the model parameters using the chosen optimization algorithm.
- Evaluate the model's performance on the validation set to monitor its progress.

4) Testing

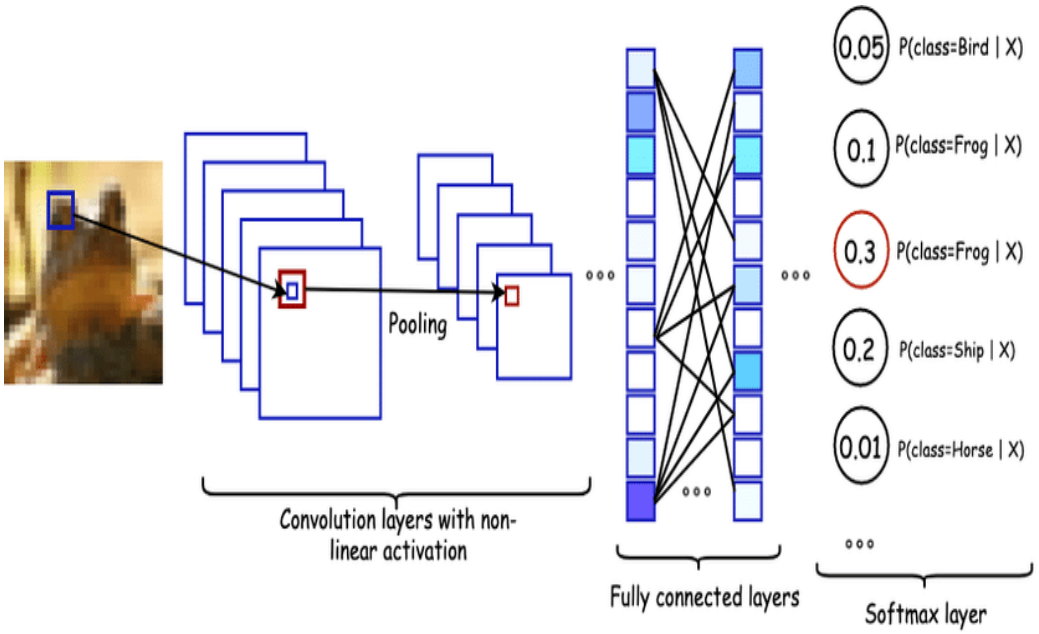
- After training is complete, evaluate the trained model on the test dataset.
- Pass the test images through the trained model and obtain the predicted class labels.
- Calculate evaluation metrics such as accuracy, precision, recall, and F1 score to assess the model's performance.

5) Model Deployment

- Save the trained model parameters for future use or deployment.
- Deploy the model in real-world applications or systems, where it can classify new, unseen images.



VI. ARCHITECTURE



VII. EVALUATION METHODOLOGY

1) Accuracy

Accuracy measures the percentage of correctly classified images out of the total number of images in the dataset. It provides an overall assessment of the model's performance.

2) Precision

Precision is the ratio of true positive predictions to the total number of positive predictions. It measures the model's ability to correctly identify positive classes.

3) Recall

Recall, also known as sensitivity or true positive rate, is the ratio of true positive predictions to the total number of actual positive instances. It measures the model's ability to correctly identify all positive instances.

4) F1 Score

The F1 score is the harmonic mean of precision and recall. It provides a single metric that balances both precision and recall, giving a comprehensive assessment of the model's performance.

5) Confusion Matrix

A confusion matrix provides a detailed breakdown of the model's predictions by showing the number of true positive, true negative, false positive, and false negative predictions for each class. It helps to analyze the performance of the model on different classes and identify potential areas of improvement.

6) Loss

Loss is a measure of the discrepancy between the predicted and actual labels during training. Lower loss values indicate better model convergence and alignment with the ground truth labels.

VIII. RESULTS

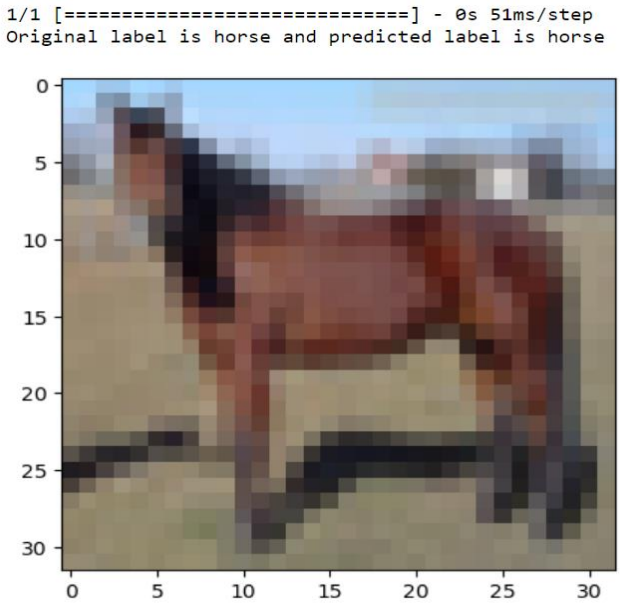


Fig-I

In Fig-I, we have given the orginal label as a horse and the model predicted label is also a horse.

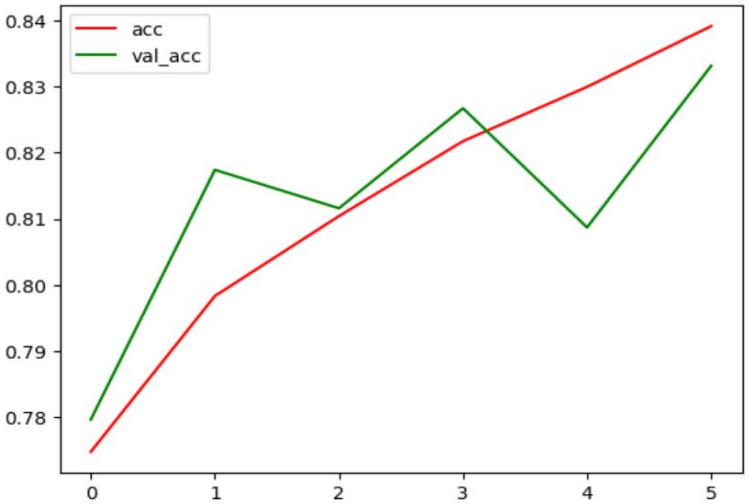


Fig-II

In Fig-II, For 5 epochs we got the validation accuracy as 83% and accuracy is 84%.

IX. CONCLUSIONS

In conclusion, image classification using CNNs with the CIFAR-10 dataset offers a powerful approach for accurately classifying images across various object categories. By leveraging the hierarchical representations learned by CNNs, along with techniques like data augmentation, transfer learning, and regularization, significant improvements in accuracy and performance can be achieved. However, careful consideration of model architecture, hyperparameter tuning, and optimization algorithms is essential. The results obtained from CNN-based image classification demonstrate the effectiveness of these methods in achieving high

accuracy rates. Continued research and exploration of advanced techniques hold promise for further enhancing the performance of image classification systems.

X. Acknowledgment

The authors would like to express their sincere gratitude to Malla Reddy University for the support and resources provided during the course of this research.

XI. References

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems (pp. 1097-1105).

[2] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[3] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[4] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4700-4708).

[5] Lin, M., Chen, Q., & Yan, S. (2013). Network in network. arXiv preprint arXiv:1312.4400.

[6] Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. International conference on learning representations (ICLR).

[7] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[8] Zeiler, M. D., & Fergus, R. (2014). Visualizing and understanding convolutional networks. In European conference on computer vision (pp. 818-833). Springer, Cham.

[9] Hu, J., Shen, L., & Sun, G. (2018). Squeeze-and-excitation networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7132-7141).

[10] Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size. arXiv preprint arXiv:1602.07360.