# Deep Learning with Keras report

**Colab Link:** <https://colab.research.google.com/drive/1mt9wWfJvoPdiCpxFErv8WiKqyB7tdM2-?usp=sharing>

## Introduction

The process of categorising images into various predetermined classes is known as image classification. With the introduction of deep learning algorithms in recent years, the field of picture categorization has been transformed. Deep learning has enabled the creation of complicated models capable of learning patterns and features from photos and making high-accuracy predictions. We offer a study of image categorization using deep learning algorithms on the CIFAR-10 dataset in this publication. To categorise the photos, we utilised a convolutional neural network (CNN) model.

## Model Architecture

We utilised a CNN model to classify images. The model is made up of six convolutional layers, followed by max-pooling layers and two fully linked layers. The first convolutional layer comprises 16 filters with 3x3 filter sizes. The second convolutional layer likewise includes 16 3x3 filters. The third convolutional layer has 32 filters and the fourth convolutional layer has 64 filters, both of which are 3x3. The fifth convolutional layer includes 64 3x3 filters. The final convolutional layer is followed by a fully connected layer with 512 neurons and a softmax activation function for categorization into ten classes.

## Model Training

The stochastic gradient descent (SGD) optimizer with a default learning rate and sparse categorical cross-entropy loss is used to train the model. The model is trained for 50 epochs using 128 batches. The test data is used to validate the model. The losses and metrics have been saved in a history object.

The loss and accuracy charts from training and validation show that the model has learned to classify the photos with high accuracy. The training accuracy is approximately 87%, while the validation accuracy is approximately 62%. However, we see that the validation loss does not reduce significantly after about 10 epochs, implying that the model is overfitting to the training data. Please find the graphs below

Chart, line chart

Description automatically generatedChart, line chart

Description automatically generated A picture containing several

Description automatically generated

## Testing with different learning rates

To determine the best optimizer and learning rate, tried three of them: SGD, Adam, and RMSprop, as well as three learning rates: 0.001, 0.01, and 0.1. Trained the model for 50 epochs with a batch size of 128 for each combination. Before each run, the model's weights were reset with the reinitialize function. The charts below show the training and validation losses and metrics for each run. We can conclude that SGD optimizer with 0.001 learning rate is a best option for the model.

Graphical user interface, application

Description automatically generated A picture containing application

Description automatically generated

## Batch size experiments

To discover the ideal batch size for this model architecture, we experimented with several batch sizes ranging from 32 to 256. To prevent the model from training for too long, it was trained for 50 epochs with an early terminating callback. The validation loss is lowest for a batch size of 32, and it grows when the batch size becomes too small or too large, as shown in the plot above. As a result, we find that 32 is the ideal batch size for our model architecture.

## Regularization experiments

Finally, two types of regularization—dropout and L2 regularization—were added to the model architecture in an update. Using SGD optimizer, 0.001 learning rate, and sparse categorical cross-entropy loss, the model was trained using the optimal parameters identified in earlier tasks. With a batch size of 32 and 50 epochs of training on the CIFAR-10 dataset, early stopping was used to avoid overfitting. A validation accuracy of 79.72% and a training accuracy of 80.69% were attained. The validation loss was .8351, and the training loss was 0.7966. Indicating that regularisation is a useful strategy for enhancing the generalisation of machine learning models, the updated model with regularisation outperformed the preceding model without regularisation.

A picture containing chart

Description automatically generatedA picture containing diagram

Description automatically generated A picture containing several

Description automatically generated

## Conclusion

With the CIFAR-10 dataset, we experimented with several hyperparameters and regularisation strategies to increase the performance of a convolutional neural network. Using the SGD optimizer, we discovered that the ideal hyperparameters for our model were a learning rate and a batch size. To increase generalisation performance, we also inserted L2 regularisation and dropout layers in our final model. Our final model's test accuracy was 79.72%, a significant improvement above the initial model's accuracy of 62.62%. Overall, our findings highlight the relevance of hyperparameter adjustment and regularisation in increasing deep learning model performance.