RV College of Engineering®
Autonomous institution affiliated to Visvesvaraya Technological University, Belagavi
Approved by AICTE, New Delhi, Accredited by NAAC, Bengaluru.
Go, change the world

A Survey of Methods to Prevent Convolutional Neural Network Model Overfitting

A Technical Seminar Report

Submitted by,

Sahana K S **1RV17EC190**
Varsha Kulkarni **1RV17EC177**

Under the guidance of

Prof. Ravishankar Holla
Assistant Professor
Dept. of ECE
RV College of Engineering

In partial fulfillment of the requirements for the degree of
Bachelor of Engineering in
Electronics and Communication Engineering
2020-2021

RV College of Engineering®, Bengaluru

(Autonomous institution affiliated to VTU, Belagavi)

Department of Electronics and Communication Engineering

.

CERTIFICATE

Certified that the Technical Seminar work titled ***A Survey of Methods to Prevent Convolutional Neural Network Model Overfitting*** is carried out by **Sahana K S** (**1RV17EC190**) and **Varsha Kulkarni** (**1RV17EC177**) who are bonafide students of RV College of Engineering, Bengaluru, in partial fulfillment of the requirements for the degree of **Bachelor of Engineering** in **Electronics and Communication Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2020-2021. It is certified that all corrections/suggestions indicated for the Internal Assessment have been incorporated in the Technical Seminar report deposited in the departmental library. The Technical Seminar report has been approved as it satisfies the academic requirements in respect of Technical Seminar work prescribed by the institution for the said degree.

Signature of Guide	Signature of Head of the Department	Signature of Principal
Prof. Ravishankar Holla	Dr. K S Geetha	Dr. K. N. Subramanya

External Viva

Name of Examiners	Signature with Date
1.	
2.	

DECLARATION

We, **Sahana K S** and **Varsha Kulkarni** students of eighth semester B.E., Department of Electronics and Communication Engineering, RV College of Engineering, Bengaluru, hereby declare that the Technical Seminar titled '**A Survey of Methods to Prevent Convolutional Neural Network Model Overfitting**' has been carried out by us and submitted in partial fulfilment for the award of degree of **Bachelor of Engineering** in **Electronics and Communication Engineering** during the year 2020-2021.

Further we declare that the content of the dissertation has not been submitted previously by anybody for the award of any degree or diploma to any other university.

We also declare that any Intellectual Property Rights generated out of this technical seminar carried out at RVCE will be the property of RV College of Engineering, Bengaluru and we will be one of the authors of the same.

Place: Bengaluru

Date:

Name	Signature
1. Sahana K S(1RV17EC190)	
2. Varsha Kulkarni(1RV17EC177)	

ACKNOWLEDGEMENT

We are indebted to our guide, **Prof. Ravishankar Holla**, Assistant Professor, RV College of Engineering . for the wholehearted support, suggestions and invaluable advice throughout this Technical seminar work and also helped in the preparation of this thesis.

We also express our gratitude to our panel members **Dr.Govinda Raju M.**, Assistant Professor and **Prof. Ravishankar Holla**, Assistant Professor, Department of Electronics and Communication Engineering for their valuable comments and suggestions during the phase evaluations.

Our sincere thanks to **Dr. K S Geetha**, Professor and Head, Department of Electronics and Communication Engineering, RVCE for the support and encouragement.

We express sincere gratitude to our beloved Principal, **Dr. K. N. Subramanya** for the appreciation towards this technical seminar work.

We thank all the teaching staff and technical staff of Electronics and Communication Engineering department, RVCE for their help.

Lastly, we take this opportunity to thank our family members and friends who provided all the backup support throughout the technical seminar work.

ABSTRACT

Machine learning is a branch of artificial intelligence (AI) focused on building applications that learn from data and improve their accuracy over time without being programmed to do so. Deep learning is an area of machine learning that deals with artificial neural networks, which are algorithms inspired by the structure and function of the brain. Convolutional Neural Networkss are a type of deep neural network that is frequently used to evaluate visual information. A convolutional neural network contains Deep neural networks like CNN that are prone to overfitting because of the millions or billions of parameters it encloses. A model with these many parameters can overfit the training data because it has sufficient capacity to do so. Overfitting is a statistical modelling error where the function fits closely to the set of data points. Overfitting occurs when the model learns too many features in the training data thereby reducing the generalization. Although the training accuracy is high, as the generalization decreases, the prediction accuracy for unseen data reduces. Therefore, it is essential to prevent model overfitting.

There are several algorithms such as early stopping, data augmentation and regularization techniques that can prevent CNN model overfitting. The objective of this report is to compare the performance of overfitting avoidance algorithms based on training and validation accuracy, training and validation loss and the number of epochs. To achieve this, a neural network is modelled and trained to overfit. Overfitting avoidance techniques are applied to the same model to prevent it from overfitting and the results are compared.

The main software tool used for this report is Google Colab. Software libraries used to train machine learning models are Tensorflow, Keras and NumPy to name a few. A comparison among common overfitting avoidance algorithms like regularisation, data augmentation and early stopping is tabulated. The validation accuracy improved from 65% to 94%, and the validation loss reduced from 2.5 to 0.17. This process can be extended to include more algorithms like cross–validation and more parameters can be considered for comparison.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

CNN Convolutional Neural Networks

GAN Generative adversarial network

RICAP Random Image Cropping and Patching

Chapter 1

Introduction to CNN and model overfitting

CHAPTER 1

INTRODUCTION TO CNN AND MODEL OVERFITTING

1.1 Overview

Machine Learning refers to computer algorithms that learn from past experience or data without requiring explicit programming. Generalization in Machine Learning refers to how well the model learns during training. Training a generalized machine learning model implies that it will work for all unseen data.

Convolutional Neural Networks (CNN)s are neural networks with one or more convolutional layers that are primarily utilised for image processing, classification, segmentation, and other autocorrelated data. they are a class of deep neural network, most commonly applied to analyze visual imagery. They are also known as space invariant artificial neural networks, based on the shared–weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps. CNNs are regularized versions of multilayer perceptrons. Multilayer perceptrons usually mean fully connected networks, that is, each neuron in one layer is connected to all neurons in the next layer. The "full connectivity" of these networks makes them prone to overfitting data.

Overfitting is a modelling error in statistics that occurs when a function is too closely aligned to a limited set of data points. Instead of generalized patterns from the training data, the model tries to fit the training data very closely. Therefore, fluctuations that are specific to the training data are learned, along with outlier information. As a result, the model is useful in reference only to its initial data set, and not to any other data sets. In reality, the data often studied has some degree of error or random noise within it. Thus, attempting to make the model conform too closely to slightly inaccurate data can infect the model with substantial errors and reduce its predictive power.

It is necessary to prevent overfitting in machine learning models. Algorithms such as data augmentation, regularization and early stopping can be used for this purpose.

Accuracy can be improved by combining these algorithms. This report aims to provide a comparative study of the algorithms to prevent overfitting in terms of training accuracy and loss, validation accuracy and loss, test accuracy and loss and the number of epochs.

1.2 Motivation

Artificial Intelligence has made a significant contribution to closing the gap between human and computer capabilities. Machine learning is an Artificial Intelligence area based on the premise that computers can learn from data, recognise patterns, and make judgments with little or no human intervention. While machine learning has proved to be very efficient, due to the fully connected layer, they are prone to model overfitting.

Overfitting occurs when the network tries to learn too many details along with the noise that is present in the training data. A network generally overfits either if the size of the training dataset is very small where the network tries to memorize every single datapoint failing to capture the general trend in the data or if the network is too complex enclosing millions or billions of parameters. Due to this, any future predictions of unseen data will show poor performance.

An overfit model may achieve a training accuracy as high as 100% but perform very poorly on unseen or test data. This behaviour of the model has to be avoided to achieve higher testing accuracy and to align the model performance while training and validation. There are several model overfitting avoidance algorithms like regularization, early stopping and data augmentation which works differently to achieve the same purpose. The presence of comparison among these methods will provide ease in using them for different scenarios.

1.3 Problem statement

To analyze several Neural Network model overfitting avoidance algorithms and provide a comparative study of these algorithms.

1.4 Objectives

1. To implement different overfitting avoidance algorithms like data augmentation, regularization and early stopping.

2. To compare the algorithms in terms of their training and validation accuracy, model training time, training and validation loss.

3. To compare the existing methods with RICAP in terms of accuracy and training time.

1.5 Literature Review

Paper [1] analyses the problems associated with an artificial neural network model overfitting and sets out a case study. It explains that overfitting is detected when a very small training error and a very high validation error occur, which could be caused by one of the following factors: the network's optimal size, the presence of outliers in the input set, the use of too complex resolution algorithms, or the use of too much data in training. It compares several orders of polynomials, demonstrating that choosing the appropriate adjustment is challenging, but the cost function may be very useful in avoiding overfitting and underfitting. A higher degree polynomial may have a high accuracy (or low error) on the training data, but it is likely to perform poorly on the test dataset or when the training data is changed.

Paper [2] discusses overfitting and underfitting modelling errors. A model is said to overfit if it has excessive parameters. In overfitting, the function is closely fit to a set of data points. Overfitting prevents the model from predicting accurately, although the training accuracy is high. Underfitting is another statistical error of a model with fewer parameters. As the information is too little, the model cannot generalize which reduces the accuracy of predictions. Both underfitting and overfitting cause serious issues in predictions.

Paper [3] proposes an activation function, modified sigmoid that prevents the model from overfitting. It also studies the effects of the number of training epochs and learning rates on overfitting. The authors use MNIST datasets to test the accuracy of the algorithm. It was also discovered that increasing the number of hidden layers improves accuracy in the training set but results in overfitting in the test set.

Regularization is a process that adds extra information to reduce the coefficients towards zero. This prevents overfitting in the models. Some of the regularization techniques are noise addition, dropout, weight decay and adversarial training. Paper [4] provides

comparative research on these techniques by evaluating testing errors and computational costs. The comparison is in terms of accuracy of the network, the number of epochs to train the network and the number of operations per input sample.

The excessive number of parameters in deep neural networks is known to cause overfitting. As a result, a variety of regularization strategies aimed at reducing model overfitting have become an essential component of many neural network topologies. However, the most efficient regularization approaches are yet unknown. Paper [5] looks at how regularization affects neural network performance when dealing with unbalanced data. L1, L2, and dropout regularization are the three basic regularization procedures considered. The L1 regularization method has been shown to be a useful technique for preventing overfitting in neural network models for unbalanced data in numerical tests. The results of the experiments show that the L1 regularization outperforms other commonly used techniques.

Early stopping allows the stopping of the training once the model performance stops improving on a hold out validation dataset. It is used while training a model using an iterative method like gradient descent. In iterative methods, each iteration updates the learning model. This improves the performance of the model till a threshold on unseen data. The stopping takes place based on rules. Paper [6] discusses the primary and secondary rules. Primary rules become true at a point during the training. It is a necessary frame for evaluation and is therefore mandatory. However, the secondary rules are not necessary, they have the potential to improve training speed and average accuracy.

Generative adversarial networks can learn generative models of natural image datasets. Data Augmentation is a technique to bypass the lack of labelled training data by synthesizing new samples from the existing data. Paper [7] proposes a method known as Deep Adversarial Data Augmentation. The authors perform simulations on this method proving that it is more effective compared to traditional data augmentation and GAN-based data augmentation. This paper compares the results for Deep Adversarial Data Augmentation and a method that can compose data transformations into a "toolchain".

Paper [8] gives insight into a new data augmentation technique called Random Image Cropping and Patching (RICAP). This technique creates a new training image by cropping four photos at random and patching them together. Furthermore, RICAP combines the four images' class labels, giving the soft labels an edge. This paper achieves a new state-of-the-art test error of 2.19% on the CIFAR-10 dataset. It also shows that deep CNNs with RICAP perform better on CIFAR-100 and ImageNet classification tasks, as well as an image-caption retrieval job using Microsoft COCO and other computer vision tasks. It finally demonstrates that RICAP prevents deep CNNs from overfitting to the most apparent features.

While a model's training should be done in such a way that it has enough examples to train on, it should not be over-fitted, and it must also be recognised that if there aren't enough examples to train on, the model will not be properly trained and will offer poor results when used for testing. Paper [9] examines the differences between the two validation systems and considers whether k-fold cross-validation may be used instead of hold-out validation on large datasets. Experimentation was carried out on four big datasets, and the findings suggest that k-fold cross-validation with varied values of k in relation to the number of instances can actually be utilised for quality classification above hold–out validation up to a specific threshold.

Deep learning models have changed the field of image processing in practically every application area, including agriculture, during the previous two decades. These models have recently been used to successfully identify plant leaf diseases. However, deep learning models are complex in nature and are prone to overfitting. Therefore, the selection of proper training settings is very important to avoid overfitting. From the standpoint of overfitting, paper [10] investigates the impact of data augmentation and various optimization strategies used to partially train a pretrained DenseNet-121 model. In the first phase, tests were done with various combinations of data augmentation approaches, and it was discovered that the network is more prone to overfitting in particular combinations. In the second phase, seven alternative optimizers were evaluated, and the Adadelta optimizer was shown to have the best generalisation ability on the supplied training data.

1.6 Brief Methodology

- A simple classification Convolutional Neural Networks (CNN) model is designed and built.
- The model is trained for an image dataset and its corresponding accuracy and loss for training and validation are recorded.
- CNN model overfitting avoidance algorithms like early stopping, data augmentation, RICAP and regularization techniques are added to the same model and trained. Accuracy and loss values are recorded for every training.
- Few of these algorithms are combined to achieve better accuracy.

The methodology is shown in fig.1.1 below.

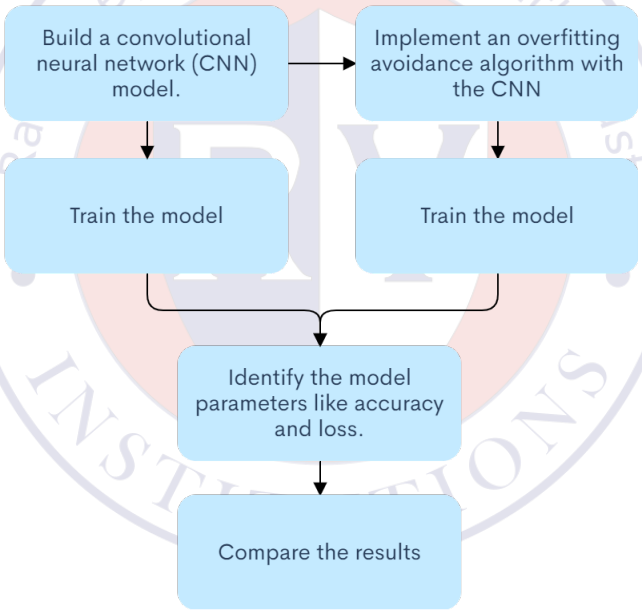


Figure 1.1: Methodology

1.7 Organization of the report

This report is organized as follows.

- Chapter 1 focuses on the introduction, motivation, main objectives, literature review, and the brief methodology of the product.
- Chapter 2 discusses the fundamentals of CNNs and model overfitting

- Chapter 3 provides detailed information on each of the algorithms along with their mathematical models.
- Chapter 4 discusses the algorithms for implementing overfitting avoiding mechanisms.
- Chapter 5 gives insight regarding the results of the different algorithms and compares them.
- Chapter 6 discusses the conclusion and future scope.

Chapter 2

Fundamentals of Convolutional Neural Networks and Modelling errors

CHAPTER 2

FUNDAMENTALS OF CONVOLUTIONAL NEURAL NETWORKS AND MODELLING ERRORS

Convolutional Neural Networks is a type of artificial neural network with one or more convolution layers that are specifically designed to process pixel data. This chapter discusses the fundamentals of Convolutional Neural Networks and highlights the modelling errors in machine learning models. It also provides an introduction to the methods of prevention of model overfitting.

2.1 Convolutional Neural Networks (CNN)

A Convolutional Neural Network (ConvNet/CNN) is a deep learning algorithm that can take an input image, assign significance to various aspects/objects in the image and be able to distinguish one from the other. In comparison to other classification algorithms, the pre–processing required in a CNN is much lower. In CNN, filters are not hand–engineered – with enough training, they have the ability to learn these filters.

The architecture of Convolutional Neural Networks differs from that of regular Neural Networks. In the case of regular Neural Networks, input data is passed through multiple hidden layers where each layer consists of a set of neurons and is entirely connected to all neurons in the layer before, following which there is a final fully–connected layer, the output layer, that represents the predictions. There is a slight difference in the case of Convolutional Neural Networks. Here, the layers are organised in 3 dimensions: width, height and depth. Further, the neurons in one layer do not connect to all the neurons in the next layer but only to a small region of it. Ultimately, the final output will be reduced to a single vector of probability scores, organized along the depth dimension.

2.1.1 Working of Convolutional Neural Networks

A Convolutional Neural network has several layers for processing an image. They include the convolutional layer, pooling layer, and fully connected layer.

Convolution Layer

The convolution layer is the first layer to extract features from an input image. An image is nothing but a matrix of pixel values. Convolution preserves the relationship between pixels by learning image features using small squares of input data. It is a mathematical operation that takes two inputs such as an image matrix and a filter or kernel to produce a feature map. Convolution is executed by sliding the filter over the input. At every location, matrix multiplication is performed between the image matrix and the filter matrix and the result is summed onto the feature map as shown in figures 2.1 and 2.2.

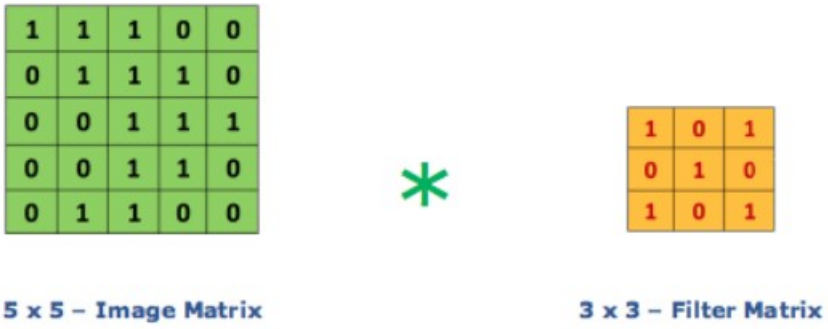


Figure 2.1: Image matrix multiplied with kernel or filter matrix

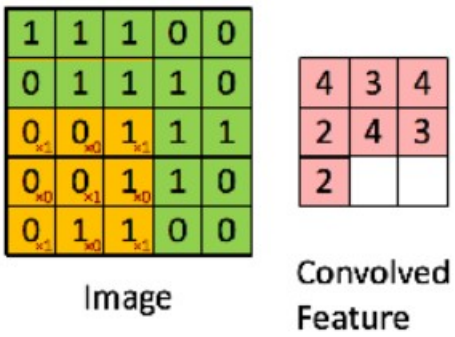


Figure 2.2: Output matrix

Operations such as edge detection, blur and sharpen can be achieved by the convolution of an image with different filters.

Pooling Layer

When the images are too large, pooling layers can reduce the number of parameters. Spatial pooling, also called subsampling or downsampling, reduces the dimensionality of

each map but retains important information. Spatial pooling can be of several types:

1. Max Pooling
2. Average Pooling
3. Sum Pooling

Max Pooling returns the maximum value from the portion of the image covered by the kernel. Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. The Sum of all elements in the feature map is called Sum Pooling. Figure 2.3 shows the operation of max pool with $2X2$ filters and stride 2.

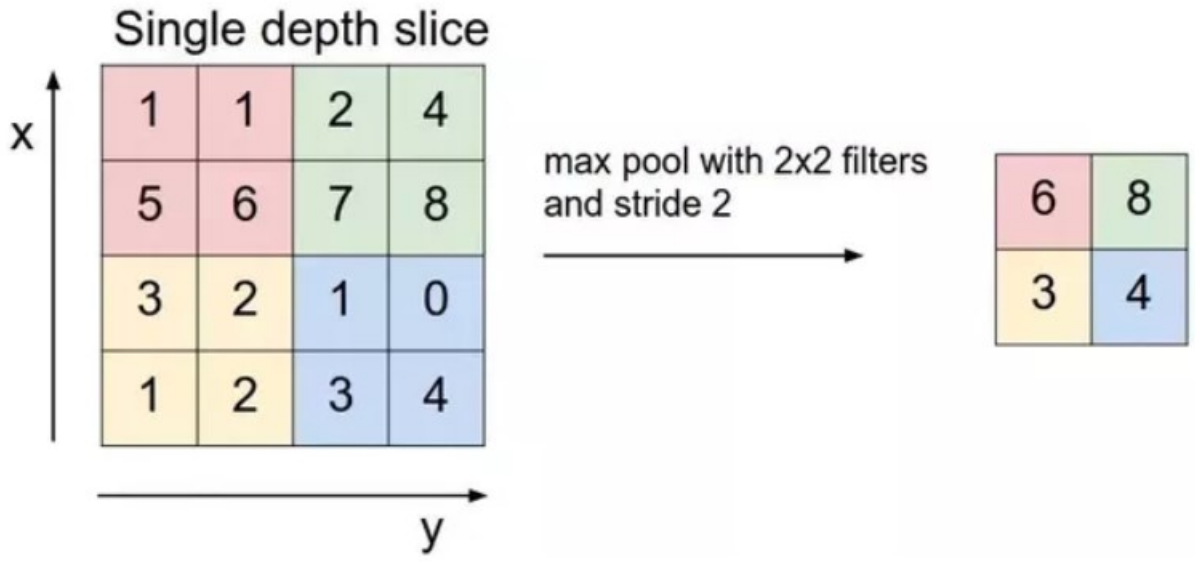


Figure 2.3: Max Pooling

Fully Connected Layer

The input to the fully connected layer is the output from the final Pooling or Convolutional Layer, which is flattened (unroll the output of final (and any) Pooling and Convolutional Layer, which is a 3–dimensional matrix, values into a vector) and then fed into the fully connected layer.

As shown in figure 2.4, the feature map matrix will be converted as vector (x1, x2, x3, ...). With the Fully Connected Layers, these features are combined together to create a model. Finally, there is an activation function such as softmax or sigmoid to classify the outputs as car, truck, cat, dog, etc.

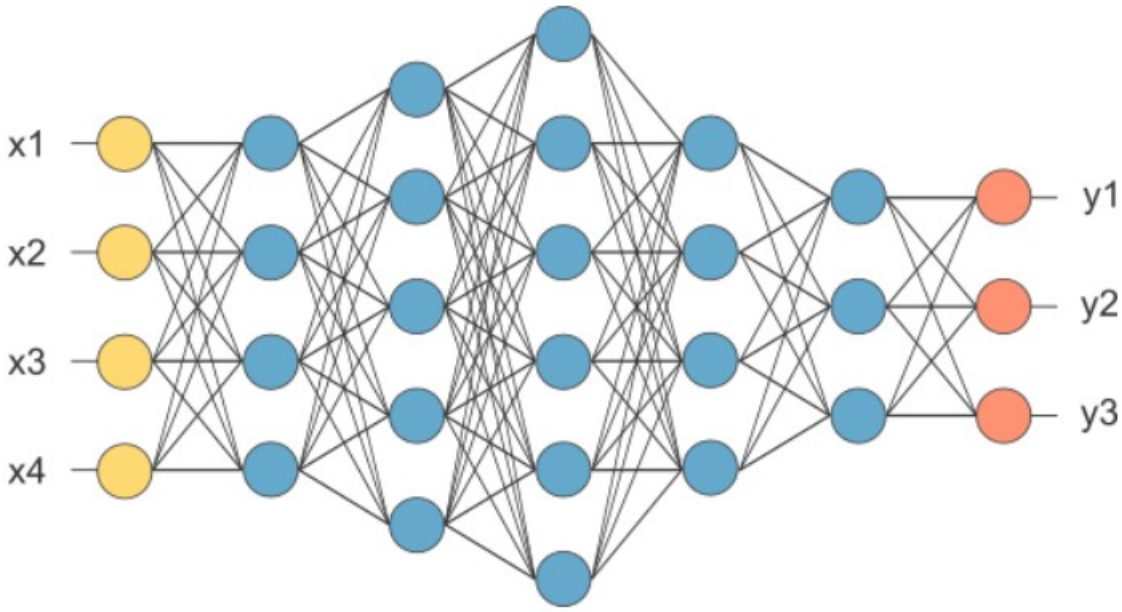


Figure 2.4: After pooling layer, flattened as FC layer

2.2 Modelling errors

Generally, models suffer from two types of statistical modelling errors: Underfitting and overfitting. The model's accuracy is harmed as a result of these inaccuracies. These errors can occur due to the volume of data set used or the number of parameters in the model.

2.2.1 Underfitting

Underfitting refers to a model that can neither model the training data nor generalizes to new data. A machine learning model underfits the data when it cannot capture the underlying trend of the data. A model underfits when there are very few training examples to learn the features of the data. It also occurs while building a linear model with non–linear data.

2.2.2 Overfitting

A model overfits when the function aligns too closely to a set of data points. A model becomes overfit when the network learns many details from the training data. Unlike an ideal situation, the data can also have noise. In an overfit condition, the model learns this noise that is present in the training dataset as well. Overfitting occurs under two scenarios:

- If the training data set is too small, the network tries to learn every detail, which results in the model memorizing the data.
- If the network model consists of many parameters, it becomes complex, causing the model to overfit. Due to overfitting, even though the training accuracy is high, the validation accuracy becomes low.

Fig.2.5 shows how an overfit and underfit model behave for a set of data points.

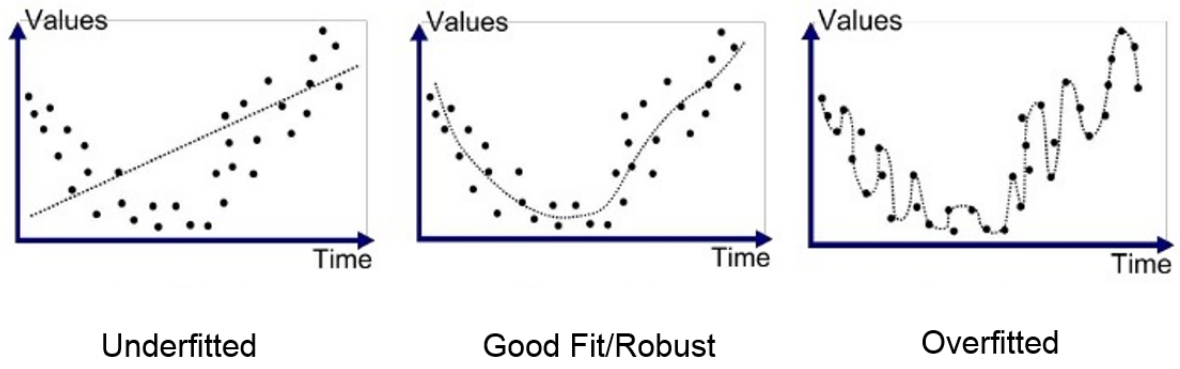


Figure 2.5: Modelling errors

Although both underfitting and overfitting are problematic, underfit models can easily be corrected. As underfitting occurs due to a lack of information, it can be corrected by increasing the size of the data set. Overfitting results in the model focusing on every feature in the function reducing the generalization. This occurs if the data set is too small or due to the presence of a large number of features. Therefore, overfit models require specific algorithms to increase their accuracy.

2.3 Algorithms to prevent Overfitting

As discussed previously, it is necessary to prevent model overfitting. There are general algorithms that can be added to a model to prevent it. The following are the algorithms that can be used:

1. Regularization: It is the process in which extra information is added to the existing data set to improve model fitting across sets of data points.
    - Early Stopping: It is a form of regularization algorithm used while training a model with an iterative method such as gradient descent. The early stopping

model identifies a point after which the generalization starts to reduce and stops training the model. Therefore, the model does not learn excessively providing scope for accurate predictions.

- Drop out: It is a form of regularization in which certain units are randomly dropped during training. Drop out is also known as dilution as it thins out weights.
- L_1 Regularization: It adds penalty equivalent to the magnitude of the coefficients. This limits the size of the coefficients, sometimes reducing it to zero.
- L_2 Regularization: This method adds the sum of squared weight values to the error function. This shrinks all the coefficients evenly unlike L_1 regularization, preventing the coefficients from reducing to zero.
- $L_1 - L_2$ Regularization: This combines both L_1 and L_2 regularization methods performing variable selection and regularization simultaneously.

2. Data Augmentation: The process of adding partially varied copies of existing data into the data set is called data augmentation. This increases the data acting like a regularizer preventing overfitting.

3. Cross–Validation: It is a resampling procedure that groups the data into k groups. It treats one subset of data for training and another for validation. Multiple rounds of cross–validation are performed to reduce the variability.

To summarise, CNNs are Machine learning models primarily used for image classification and recognition. A Convolutional Neural Networks has several layers such as a convolutional layer, pooling layer and fully connected layer. These models are prone to overfitting due to a lack of generalisation. The algorithms used for the prevention of overfitting will be discussed in detail in Chapter 3.

Chapter 3

Algorithms to prevent CNN model overfitting

CHAPTER 3

ALGORITHMS TO PREVENT CNN MODEL OVERFITTING

Overfitting refers to the phenomenon where a neural network models the training data very well but fails when it sees new data from the same problem domain. Overfitting is caused by noise in the training data that the neural network picks up during training and learns as an underlying concept of the data. Algorithms such as regularization, early stopping and data augmentation can help to prevent overfitting of CNN models. This chapter discusses these algorithms in detail along with their mathematical models. It also highlights the software used to implement these techniques.

3.1 Regularization

Regularization is a set of techniques that can prevent overfitting in neural networks and thus improve the accuracy of a Deep Learning model when facing completely new data from the problem domain. It is the process that regularizes or shrinks the coefficients towards zero adding an information term called the regularization term or a penalty to the cost function as shown by equation 3.1 below.

$$cost\ function = loss + Regularization\ term \tag{3.1}$$

The loss term refers to the cost function used for training the model such as binary cross–entropy or categorical cross–entropy. The regularization term is added to reduce the generalization error. Because it is assumed that a neural network with lower weight matrices leads to simpler models, the values of weight matrices decrease when this regularization component is added. As a result, it will significantly reduce overfitting. The following are the types of regularization algorithms.

3.1.1 Early Stopping Algorithm

Early stopping is a regularization method that can prevent overfitting. Too many epochs can lead to overfitting of the training dataset, whereas too few may result in an underfit model. Early stopping is a method that allows to specify an arbitrarily large

number of training epochs and stop training once the model performance stops improving on a hold out validation dataset. It is used while training a model using an iterative method like gradient descent. In iterative methods, each iteration updates the learning model. This improves the performance of the model till a threshold on unseen data. After the threshold, the generalization error increases, reducing the performance accuracy of the model.

Early stopping can help determine the number of iterations that can be run before overfitting starts to occur. By stopping at the threshold point, training the model with additional iterations can be avoided, preventing the overfitting of the model. Early stopping can be used simultaneously with other regularization algorithms.

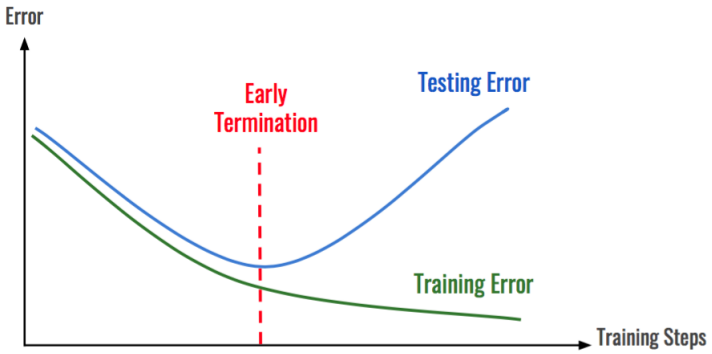


Figure 3.1: Early Stopping

Fig.3.1 shows how the error in validation starts increasing after the optimal point. If the cost function near the optimal point is considered to be a quadratic function, this algorithm is equivalent to L_2 regularization [6] as shown by equation 3.2 below.

$$\tau \approx \frac{1}{\eta\alpha} \tag{3.2}$$

In such a case, the relation between the number of iterations τ, coefficient of the L_2 penalty α, and the learning rate η , is given in equation 3.2. From the equation given, it is evident that high penalties result in a lower number of iterations.[4]

3.1.2 Dropout Algorithm

Dropout is a regularization technique that can reduce overfitting in neural networks. It can prevent complex co–adaptations on training data. The dropout algorithm randomly drops out hidden and visible units during the training.[11]At each training stage,

individual nodes are either dropped out of the net with probability $1 - p$ or kept with probability p, so that a reduced network is left. The dropout algorithm is also known as the dilution algorithm as it can thin the weights.

While training, the dropout algorithm samples from a set of "thinned" networks. At the time of testing, the approximation of the effect of the predictions of all the networks becomes easy by using an "unthinned" network with smaller weights. This prevents neurons from co–adaptation.

Dilution and dropout algorithms are iterative. The network continues to learn even after the algorithm is applied.

Fig.3.2 compares the network before and after the application of the dropout algorithm.

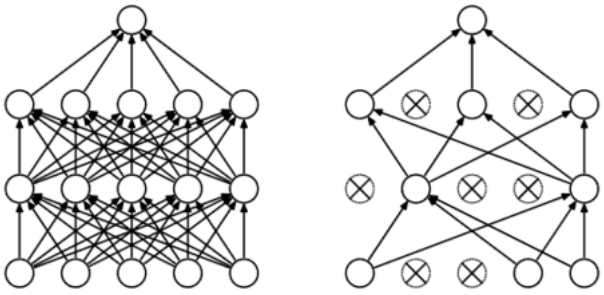


Figure 3.2: Dropout Algorithm

3.1.3 L_1 Regularization or Lasso regression

L_1 regularization is an algorithm that adds an L_1 regularization term ω to the cost function. This term is the sum of the absolute values of the weight parameters in a weight matrix. Adding this penalty limits the size of the coefficients. In L_1 regularization, the weight is always forced towards zero.

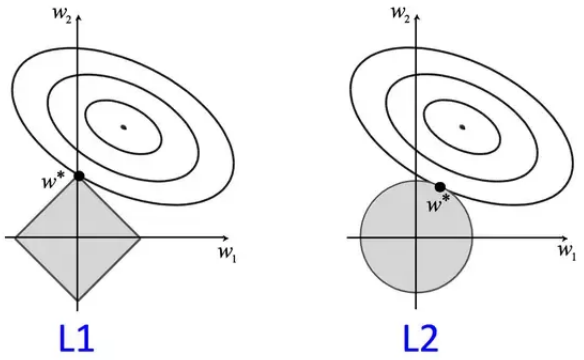


Figure 3.3: L_1 and L_2 regularization

Fig.3.3 shows both L_1 and L_2 regularization methods.

Equations 3.3, 3.4, 3.5 and 3.6 below explain the mathematics behind L_1 regularization [11]. The regularization term is weighted by the scalar alpha (α) divided by two and added to the regular cost function that is chosen for the current task. Alpha is sometimes called the regularization rate and is an additional hyperparameter that determines how much the model is regularised.

$$\hat{L}(W) = L(W) + \frac{\alpha}{2} * \sum(\|w\|) \tag{3.3}$$

where $\hat{L}(W)$ is the new cost function, $L(W)$ is the old cost function and $\sum(\|w\|)$ is the L_1 regularization term. The gradient of the new cost function is computed and these gradients are added to the update rule for the weights as shown by equation 3.4 below:

$$\nabla_W \hat{L}(W) = \nabla_W L(W) + \alpha W \tag{3.4}$$

$$W_{new} = W_{old} - \epsilon(\alpha W_{old} + \nabla_W L(W_{old}) \tag{3.5}$$

$$W_{new} = (1 - \epsilon\alpha) W_{old} - \epsilon \nabla_W L(W_{old}) \tag{3.6}$$

By adding the regularization term, independent of the gradient of the cost function, weights are made a little bit smaller each time an update is performed.

3.1.4 L_2 Regularization or weight decay or Ridge Regression

L_2 regularization involves adding an L_2 penalty term ω to the cost function. This term is the summation of all squared weight values of a weight matrix. The penalty term shrinks all the coefficients by the same factor. Therefore, the coefficients are always a non–zero value[4].

Equation 3.7 below gives the new cost function after adding the L_2 regularization term to the old cost function.

$$\hat{L}(W) = L(W) + \frac{\alpha}{2} * \sum (\|w\|^2) \tag{3.7}$$

where $\hat{L}(W)$ is the new cost function, $L(W)$ is the old cost function and $\sum (\|w\|^2)$ is the L_2 regularization term.

3.1.5 L_1–L_2 Regularization or elastic net regularization

The elastic net regularization is the combination of L_1 and L_2 regularization methods. It includes the L_1 and L_2 penalties as shown by equation 3.8 below.

$$\hat{L}(W) = L(W) + \frac{\alpha}{2} * (\sum (\|w\|^2) + \sum (\|w\|^2)) \tag{3.8}$$

where $\hat{L}(W)$ is the new cost function, $L(W)$ is the old cost function,$\sum (\|w\|)$ is the L_1 regularization term and $\sum (\|w\|^2)$ is the L_2 regularization term.

Limitations of L_1 regularization

The L_1 method uses a penalty function which is given by equation 3.9 below.

$$\|\beta\| = \sum_{j=1}^{p} |\beta_j|. \tag{3.9}$$

The use of this penalty function has several limitations. For example, in the "large p, small n" case (high–dimensional data with few examples), the algorithm selects at most n variables before it saturates. Also if there is a group of highly correlated variables, then the algorithm tends to select one variable from a group and ignore the others. To overcome these limitations, the elastic net adds a quadratic part to the penalty $\|\beta\|^2$, which is

used alone in ridge regression. The elastic net regularization technique overcomes the limitations of the L_1 regularization method. The estimates from the elastic net method are defined by equation 3.10 given below.

$$\hat{\beta} \equiv \underset{\beta}{\operatorname{argmin}}(\|y - X\beta\|^2 + \lambda_2\|\beta\|^2 + \lambda_1\|\beta\|). \tag{3.10}$$

Fig.3.4 shows the improvement in elastic net technique when compared to L_1 regularization.

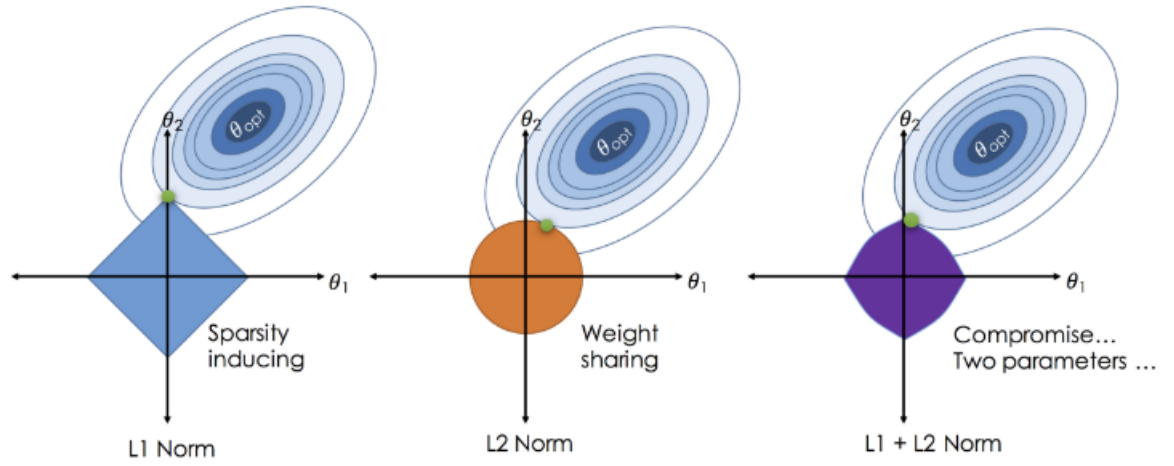


Figure 3.4: Elastic Net Regularization

3.2 Data Augmentation

Data augmentation is a technique that increases the amount of data. This is done by slightly modifying the existing data and adding it to the training dataset. The additional data can also be generated newly by creating synthetic data from the data that is already present. The addition of information acts as a regularizer and helps overcome overfitting.

The following are the two ways to add new data.

Image Transformation

Images can be modified in numerous ways. Geometric transformations such as spatial transformations can be performed on the images. In spatial transformation, every coordinate of an image is mapped to a different coordinate. The images can also be transformed by flipping, modifying the colour, cropping, rotating and injecting noise.

Synthetic Image generation

During data scarcity, image transformation techniques are not very efficient. If the data set is too small, image transformation techniques do not increase the size of the set by a significant amount. Sourcing new synthetic images from existing images solves this problem. Generative adversarial network (GAN)s can be used to create new synthetic images during data augmentation. GANs are algorithmic architectures that use two neural networks, generator and discriminator, which compete against each other (thus the "adversarial") to generate new, synthetic instances of data that can pass for real data. They are used widely in image generation, video generation and voice generation.

Figure 3.5 shows an example of data augmentation.

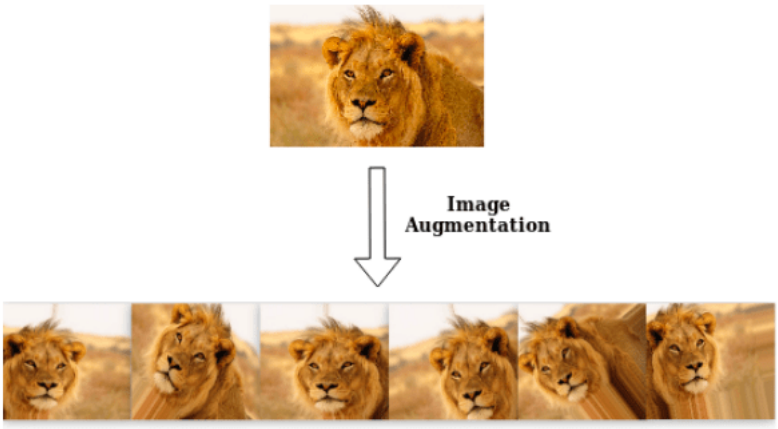


Figure 3.5: Data Augmentation

3.3 Random Image Cropping and Patching

Random Image Cropping and Patching (RICAP) is an algorithm that randomly crops four images and patches them to create a new training image. The algorithm mixes class labels of four images resulting in an advantage of soft labels. Soft labels contain information on the similarity between classes and the ambiguity of each sample. A method known as cutout randomly masks a square region in an image at every step in the training [8]. RICAP shares concepts with cutout and soft labels. RICAP also uses the concept of mixup, which can train Convolutional Neural Networkss on convex combinations of training samples in pairs. A target label c is defined by mixing one–hot coded class labels c_k of the patched images with ratios W_i proportional to their areas in the newly constructed image. Label mixing of RICAP is given by equation 3.11 below.

$$c = \sum_{k \in \{1,2,3,4\}} W_k c_k \tag{3.11}$$
$$for\, W_k = \frac{w_k h_k}{I_x I_y}$$

Where, $w_k h_k$ is the area of the cropped image k and $I_x I_y$ is the area of the original image.

3.4 Software Setup

1. TensorFlow: TensorFlow is a free and open–source software library for machine learning.

2. Keras: Keras is an open–source software library that provides a python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library.

3. Google Colab: Colaboratory, or "Colab" for short, is a product from Google Research. Colab allows anybody to write and execute arbitrary python code through the browser and is especially well suited to machine learning, data analysis and education.

In summary, this chapter discusses the details of algorithms that can prevent CNN model overfitting. Elastic net regularization overcomes the limitations of L_1 regularization by combining it with L_2 regularization technique. Data can be augmented by image transformation or by synthetic image generation which helps in increasing the training dataset thereby preventing overfitting. RICAP is a new technique that is an advancement to traditional data augmentation that improves the classification accuracy by increasing the variety of training images. RICAP is better as it combines the concepts of mixup, cutout, and soft-labelling, and overcomes their limitations.

Chapter 4

Results & Discussions

CHAPTER 4

RESULTS & DISCUSSIONS

The results obtained for a CNN model trained with different overfitting avoidance techniques are shown in this chapter. For every technique, details about the model training, accuracy and loss curves along with results of classification for each technique is mentioned.

4.1 Overfit Model

This section includes training details, model accuracy and results obtained for an overfit CNN model.

4.1.1 Training details

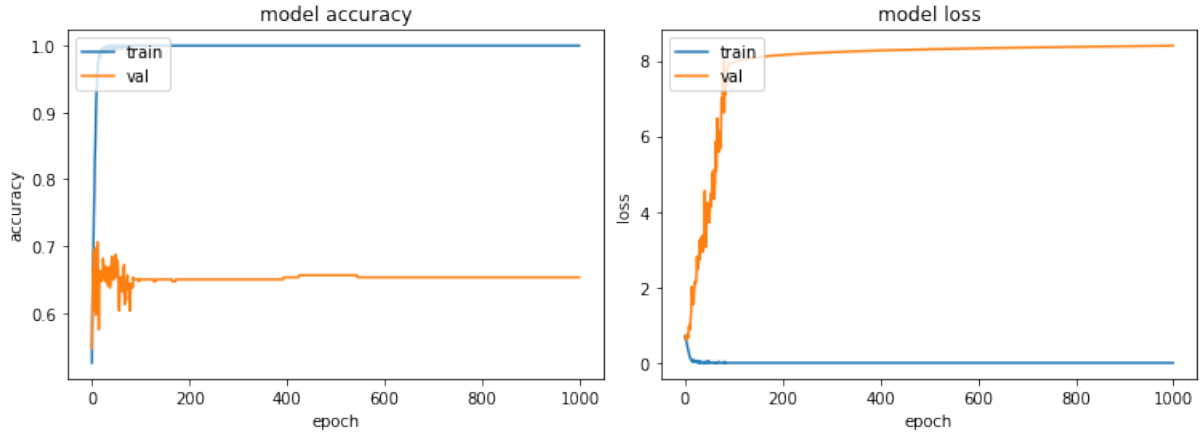
Table 4.1 gives the training details of an overfit CNN model.

Table 4.1: Model overfitting

Number of epochs	1000
Training images used	1600
Validation images used	400
Training accuracy	1
Training loss	0
Validation accuracy	0.65
Validation loss	8.4

4.1.2 Loss and Accuracy

Figure 4.1a shows the training and validation accuracy and figure 4.1b shows the training and validation loss for an overfit CNN model.



(a) Accuracy of an Overfit CNN model (b) Loss of an Overfit CNN model

4.1.3 Result obtained

Figures 4.2 shows the classification result for an overfit model.

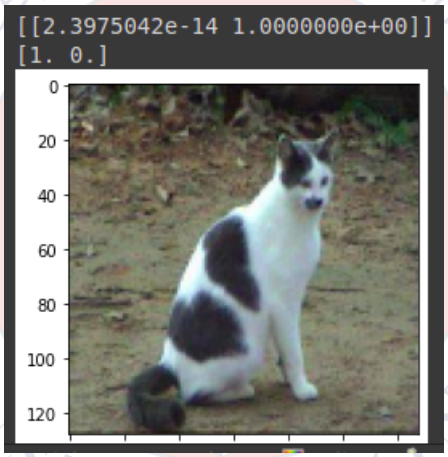


Figure 4.2: Classification result

4.2 Early stopping

This section includes training details, model accuracy and results obtained for a CNN model with early stopping.

4.2.1 Training details

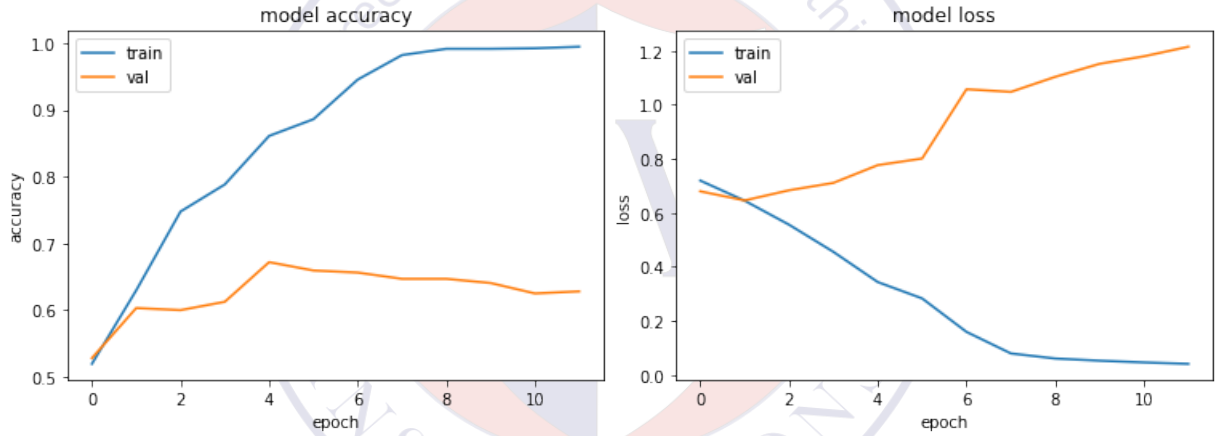
Table 4.2 gives the training details of a CNN model with early stopping.

Table 4.2: Early Stopping

Number of epochs	12
Training images used	1600
Validation images used	400
Training accuracy	0.62
Training loss	0.65
Validation accuracy	0.60
Validation loss	0.64

4.2.2 Loss and Accuracy

Figure 4.3a shows the training and validation accuracy and figure 4.3b shows the training and validation loss for a CNN model with early stopping.



(a) Accuracy of CNN model with early stopping (b) Loss of CNN model with early stopping

4.2.3 Result obtained

Figures 4.4 shows the classification result for a CNN model with early stopping.

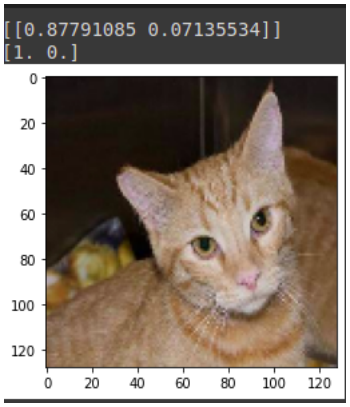


Figure 4.4: Classification result

4.3 L_1 regularization

This section includes training details, model accuracy and results obtained for a CNN model with L_1 regularization.

4.3.1 Training details

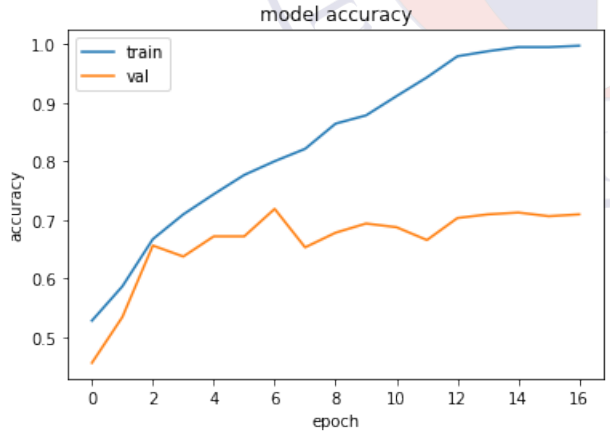
Table 4.3 gives the training details of CNN model with L_1 regularization.

Table 4.3: L_1 regularization

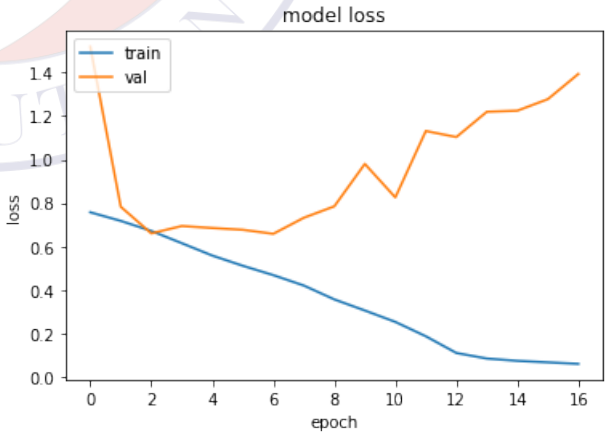
Number of epochs	7
Training images used	1600
Validation images used	400
Training accuracy	0.80
Training loss	0.47
Validation accuracy	0.72
Validation loss	0.65

4.3.2 Loss and Accuracy

Figure 4.5a shows the training and validation accuracy and figure 4.5b shows the training and validation loss for a CNN model with L_1 regularization.



(a) Accuracy of a CNN model with L_1 regularization



(b) Loss of a CNN model with L_1 regularization

4.3.3 Result obtained

Figures 4.6 shows the classification result for a CNN model with L_1 regularization.

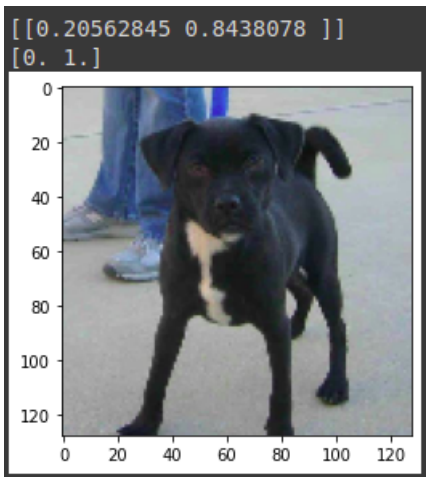


Figure 4.6: Classification result

4.4 L_2 regularization

This section includes training details, model accuracy and results obtained for a CNN model with L_2 regularization.

4.4.1 Training details

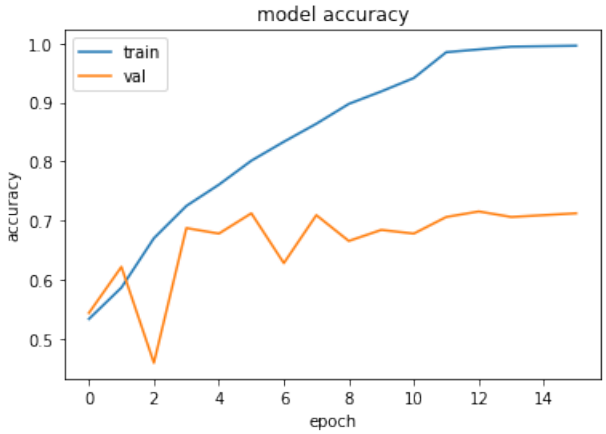
Table 4.4 gives the training details of CNN model with L_2 regularization.

Table 4.4: L_2 regularization

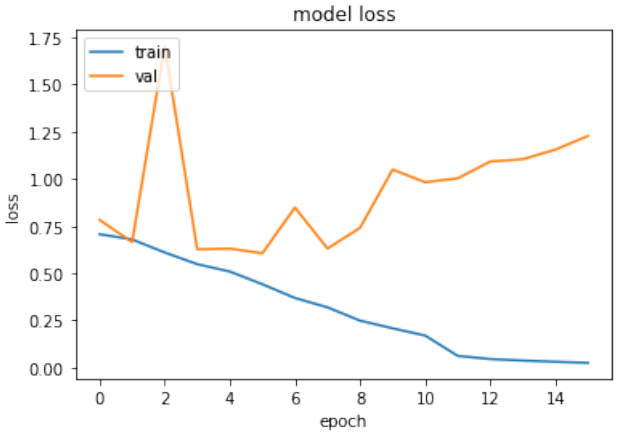
Number of epochs	6
Training images used	1600
Validation images used	400
Training accuracy	0.81
Training loss	0.43
Validation accuracy	0.71
Validation loss	0.60

4.4.2 Loss and Accuracy

Figure 4.7a shows the training and validation accuracy and figure 4.7b shows the training and validation loss for a CNN model with L_2 regularization.



(a) Accuracy of a CNN model with L_2 regularization



(b) Loss of a CNN model with L_2 regularization

4.4.3 Result obtained

Figures 4.8 shows the classification result for a CNN model with L_2 regularization.

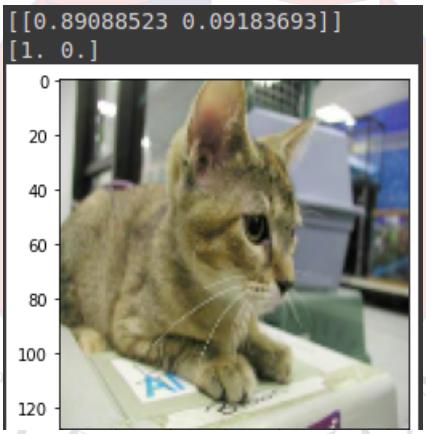


Figure 4.8: Classification result

4.5 L_1 and L_2 regularization

This section includes training details, model accuracy and results obtained for a CNN model with L_1 and L_2 regularizations.

4.5.1 Training details

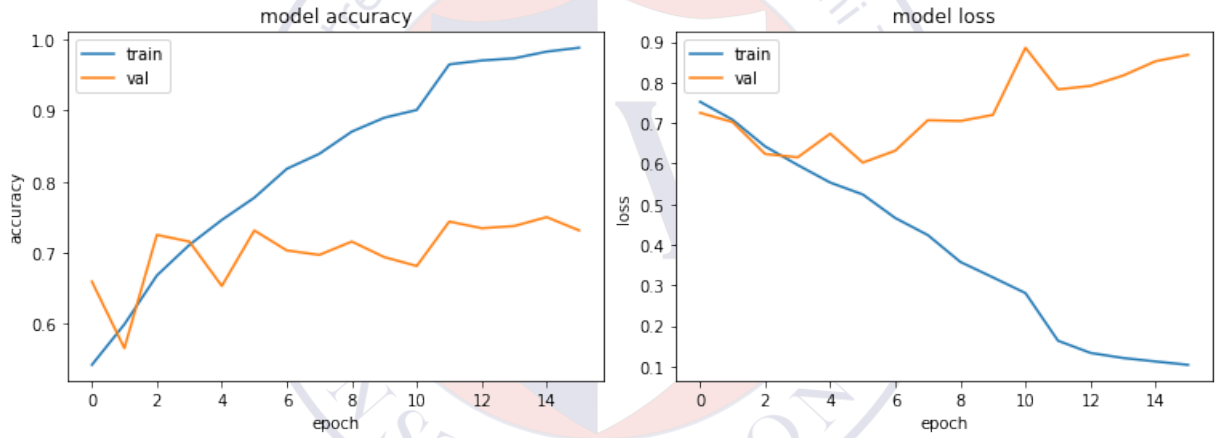
Table 4.5 gives the training details of CNN model with L_1 and L_2 regularizations.

Table 4.5: L_1 L_2 regularization

Number of epochs	6
Training images used	1600
Validation images used	400
Training accuracy	0.77
Training loss	0.52
Validation accuracy	0.73
Validation loss	0.60

4.5.2 Loss and Accuracy

Figure 4.9a shows the training and validation accuracy and figure 4.9b shows the training and validation loss for a CNN model with L_1 and L_2 regularizations.



(a) Accuracy of a CNN model with L_1 and L_2 regularizations

(b) Loss of a CNN model with L_1 and L_2 regularizations

4.5.3 Result obtained

Figures 4.10 shows the classification result for a CNN model with L_1 and L_2 regularizations.

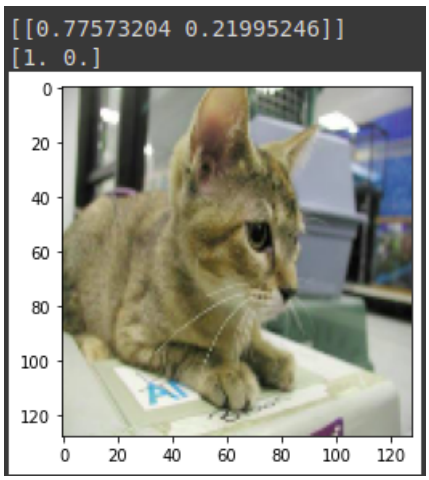


Figure 4.10: Classification result

4.6 Dropout

This section includes training details, model accuracy and results obtained for a CNN model with dropout regularization.

4.6.1 Training details

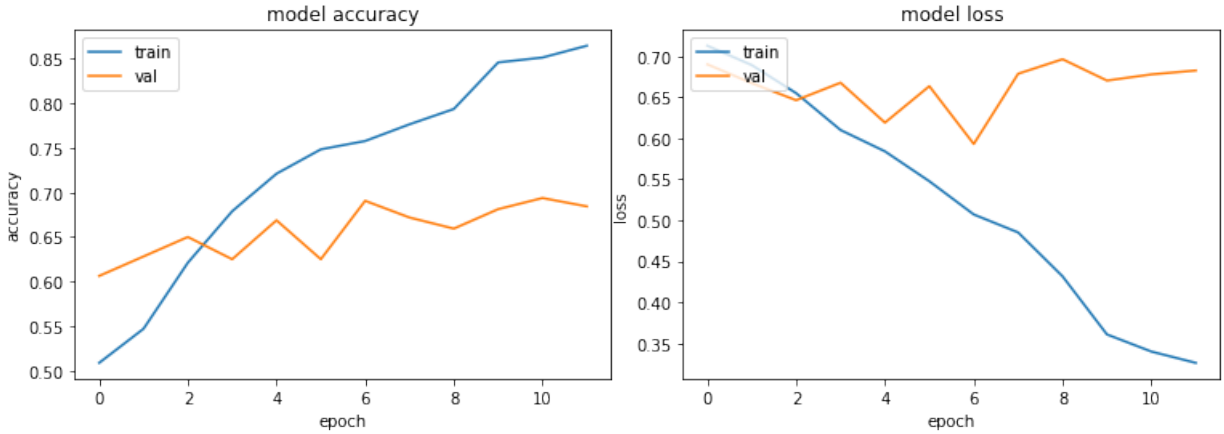
Table 4.6 gives the training details of CNN model with dropout regularization.

Table 4.6: Dropout regularization

Number of epochs	5
Training images used	1600
Validation images used	400
Training accuracy	0.75
Training loss	0.51
Validation accuracy	0.69
Validation loss	0.59

4.6.2 Loss and Accuracy

Figure 4.11a shows the training and validation accuracy and figure 4.11b shows the training and validation loss for a CNN model with dropout regularization.



(a) Accuracy of a CNN model with dropout regularization

(b) Loss of a CNN model with dropout regularization

4.6.3 Result obtained

Figures 4.12 shows the classification result for a CNN model with dropout regularization.

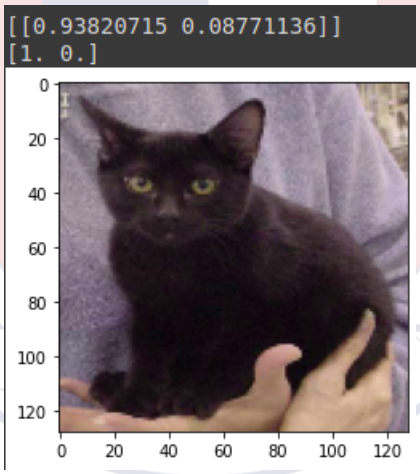


Figure 4.12: Classification result

4.7 L_1 regularization with dropout

This section includes training details, model accuracy and results obtained for a CNN model with L_1 and dropout regularizations.

4.7.1 Training details

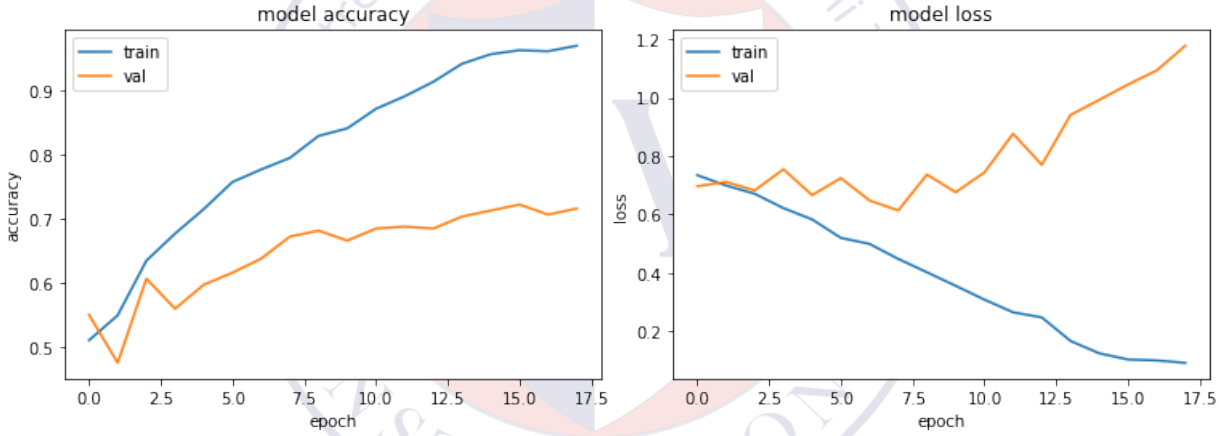
Table 4.7 gives the training details of CNN model with L_1 and dropout regularizations.

Table 4.7: L_1 Dropout regularization

Number of epochs	8
Training images used	1600
Validation images used	400
Training accuracy	0.80
Training loss	0.44
Validation accuracy	0.67
Validation loss	0.61

4.7.2 Loss and Accuracy

Figure 4.13a shows the training and validation accuracy and figure 4.13b shows the training and validation loss for a CNN model with L_1 and dropout regularizations.



(a) Accuracy of a CNN model with L_1 and dropout regularizations.

(b) Loss of a CNN model with L_1 and dropout regularizations.

4.7.3 Result obtained

Figures 4.14 shows the classification result for a CNN model L_1 and dropout regularizations.

Figure 4.14: Classification result

4.8 L_2 regularization with dropout

This section includes training details, model accuracy and results obtained for a CNN model with L_2 and dropout regularizations.

4.8.1 Training details

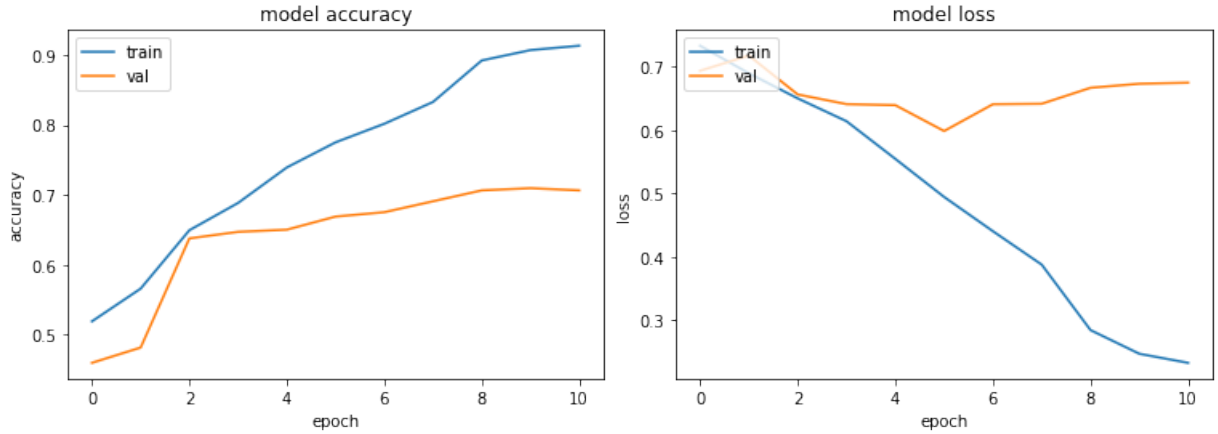
Table 4.8 gives the training details of CNN model with L_2 and dropout regularizations.

Table 4.8: L_2 Dropout regularization

Number of epochs	6
Training images used	1600
Validation images used	400
Training accuracy	0.77
Training loss	0.49
Validation accuracy	0.67
Validation loss	0.60

4.8.2 Loss and Accuracy

Figure 4.15a shows the training and validation accuracy and figure 4.15b shows the training and validation loss for a CNN model with L_2 and dropout regularizations.



(a) Accuracy of a CNN model with L_2 and dropout regularizations.

(b) Loss of a CNN model with L_2 and dropout regularizations.

4.8.3 Result obtained

Figures 4.16 shows the classification result for a CNN model with L_2 and dropout regularizations.

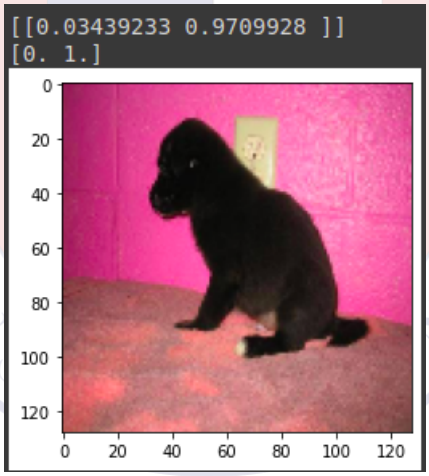


Figure 4.16: Classification result

4.9 $L_1 - L_2$ regularization with dropout

This section includes training details, model accuracy and results obtained for a CNN model with L_1, L_2 and dropout regularizations.

4.9.1 Training details

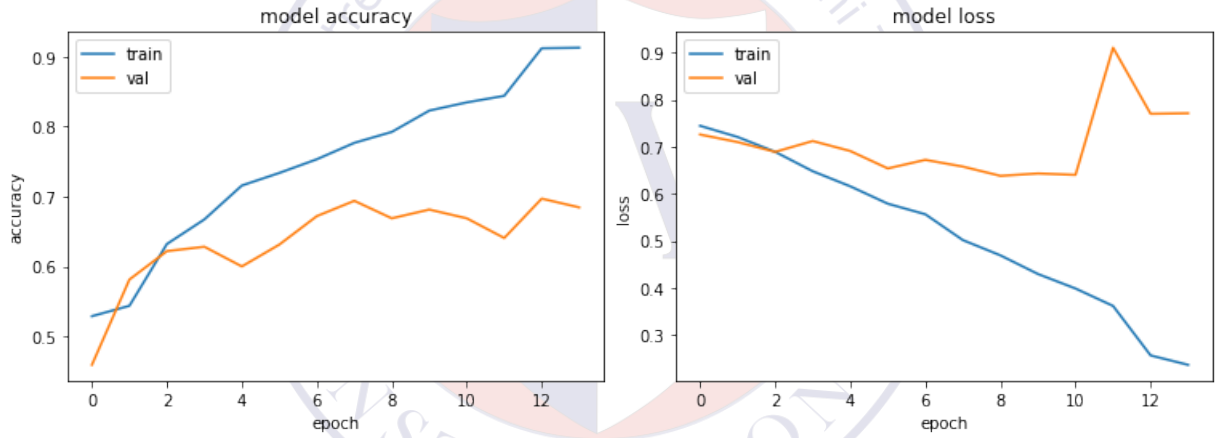
Table 4.9 gives the training details of CNN model with L_1, L_2 and dropout regularizations.

Table 4.9: L_1 L_2 Dropout regularization

Number of epochs	9
Training images used	1600
Validation images used	400
Training accuracy	0.79
Training loss	0.46
Validation accuracy	0.67
Validation loss	0.64

4.9.2 Loss and Accuracy

Figure 4.17a shows the training and validation accuracy and figure 4.17b shows the training and validation loss for a CNN model with L_1, L_2 and dropout regularizations.



(a) Accuracy of a CNN model with L_1, L_2 and dropout regularizations.

(b) Loss of a CNN model with L_1, L_2 and dropout regularizations.

4.9.3 Result obtained

Figures 4.18 shows the classification result for a CNN model with L_1, L_2 and dropout regularizations.

Figure 4.18: Classification result

4.10 Data Augmentation without Early Stopping

This section includes training details, model accuracy and results obtained for a CNN model with data augmentation without early stopping.

4.10.1 Training details

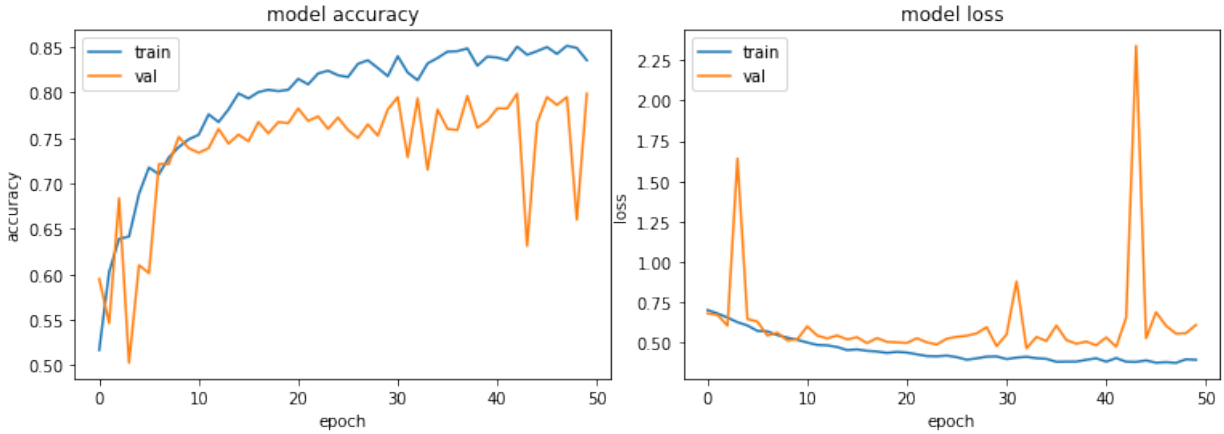
Table 4.10 gives the training details of CNN model with data augmentation without early stopping.

Table 4.10: Data augmentation without early stopping

Number of epochs	50
Training images used	2000
Validation images used	800
Training accuracy	0.84
Training loss	0.37
Validation accuracy	0.80
Validation loss	0.60

4.10.2 Loss and Accuracy

Figure 4.19a shows the training and validation accuracy and figure 4.19b shows the training and validation loss for a CNN model with data augmentation without early stopping.



(a) Accuracy of a CNN model with data augmentation without early stopping.

(b) Loss of a CNN model with data augmentation without early stopping.

4.10.3 Result obtained

Figures 4.20 shows the classification result for a CNN model with data augmentation without early stopping.

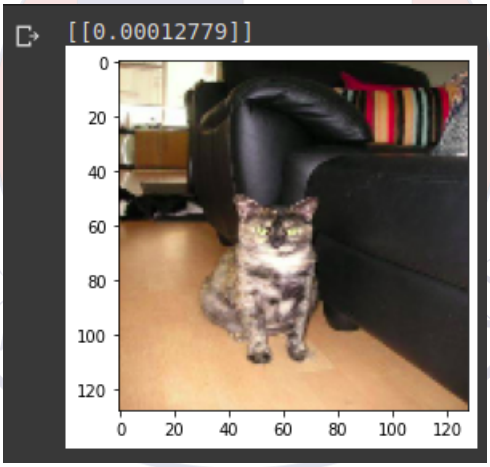


Figure 4.20: Classification result

4.11 Data Augmentation with Early Stopping

This section includes training details, model accuracy and results obtained for a CNN model with data augmentation with early stopping.

4.11.1 Training details

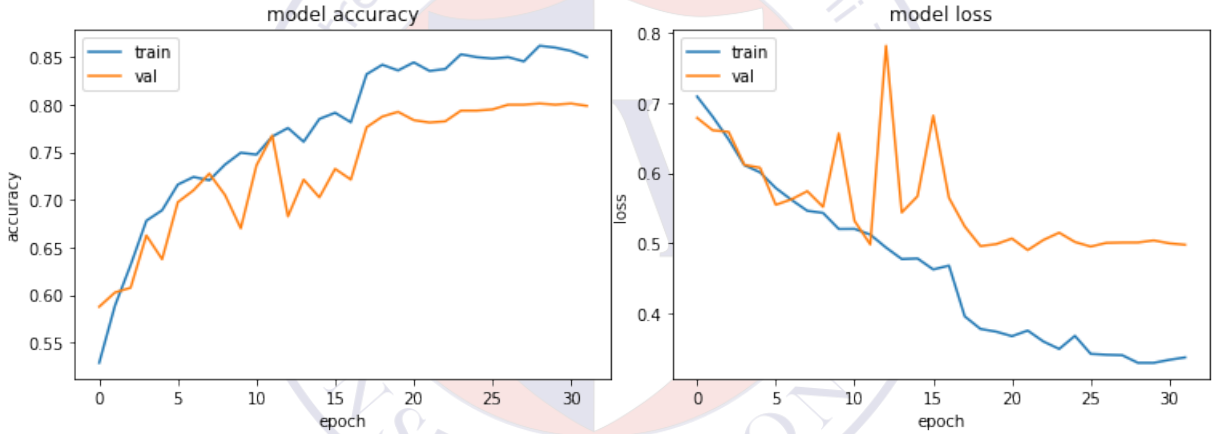
Table 4.11 gives the training details of CNN model with data augmentation with early stopping.

Table 4.11: Data augmentation with early stopping

Number of epochs	22
Training images used	2000
Validation images used	800
Training accuracy	0.85
Training loss	0.37
Validation accuracy	0.78
Validation loss	0.49

4.11.2 Loss and Accuracy

Figure 4.21a shows the training and validation accuracy and figure 4.21b shows the training and validation loss for a CNN model with data augmentation and early stopping.



(a) Accuracy of a CNN model with data augmentation and early stopping.

(b) Loss of a CNN model with data augmentation and early stopping.

4.11.3 Result obtained

Figures 4.22 shows the classification result for a CNN model with data augmentation and early stopping.

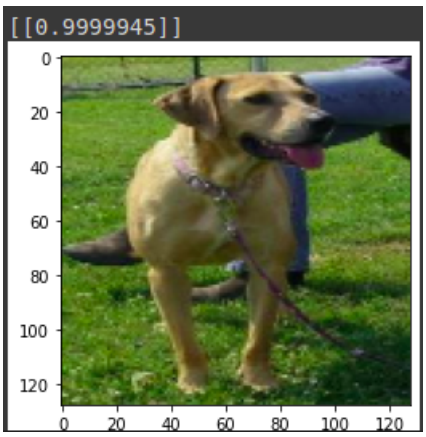


Figure 4.22: Classification result

4.12 Data Augmentation and L_1 regularization

This section includes training details, model accuracy and results obtained for a CNN model with data augmentation and L_1 regularization.

4.12.1 Training details

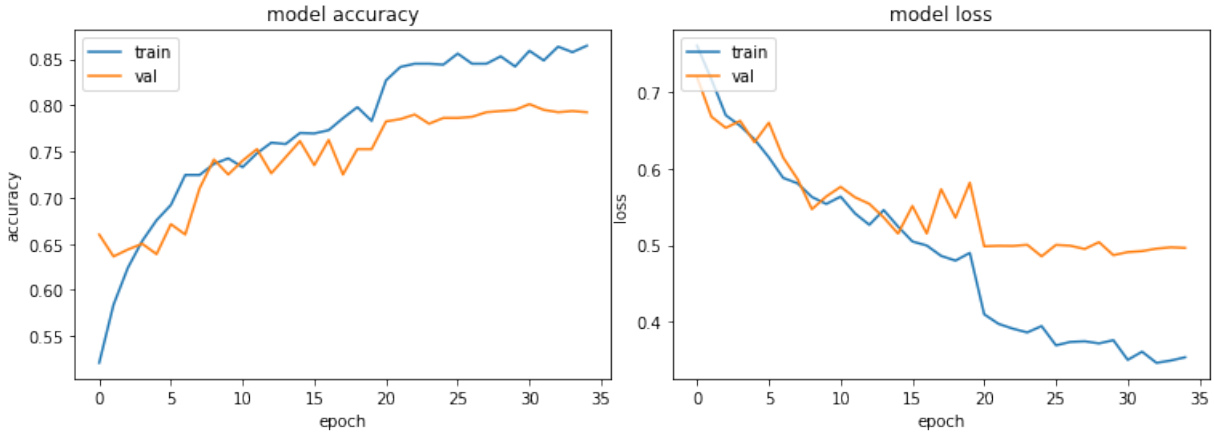
Table 4.12 gives the training details of CNN model with data augmentation and L_1 regularization.

Table 4.12: Data augmentation and L_1 regularization

Number of epochs	25
Training images used	2000
Validation images used	800
Training accuracy	0.84
Training loss	0.39
Validation accuracy	0.79
Validation loss	0.48

4.12.2 Loss and Accuracy

Figure 4.23a shows the training and validation accuracy and figure 4.23b shows the training and validation loss for a CNN model with data augmentation and L_1 regularization.



(a) Accuracy of a CNN model with data augmentation and L_1 regularization.

(b) Loss of a CNN model with data augmentation and L_1 regularization.

4.12.3 Result obtained

Figures 4.24 shows the classification result for a CNN model with data augmentation and L_1 regularization.

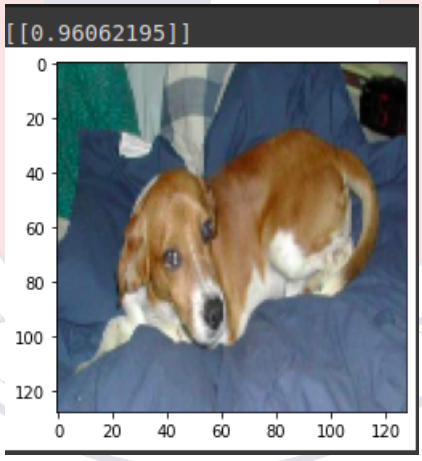


Figure 4.24: Classification result

4.13 Data Augmentation and L_2 regularization

This section includes training details, model accuracy and results obtained for a CNN model with data augmentation and L_2 regularization.

4.13.1 Training details

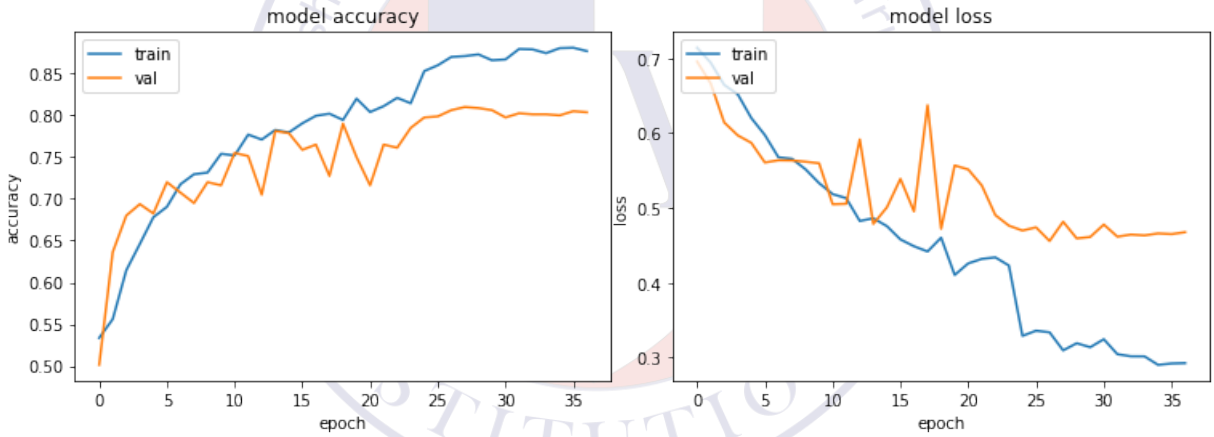
Table 4.13 gives the training details of CNN model with data augmentation and L_2 regularization.

Table 4.13: Data augmentation and L_2 regularization

Number of epochs	27
Training images used	2000
Validation images used	800
Training accuracy	0.87
Training loss	0.32
Validation accuracy	0.80
Validation loss	0.45

4.13.2 Loss and Accuracy

Figure 4.25a shows the training and validation accuracy and figure 4.25b shows the training and validation loss for a CNN model with data augmentation and L_2 regularization.



(a) Accuracy of a CNN model with data augmentation and L_2 regularization.

(b) Loss of a CNN model with data augmentation and L_2 regularization.

4.13.3 Result obtained

Figures 4.26 shows the classification result for a CNN model with data augmentation and L_2 regularization.

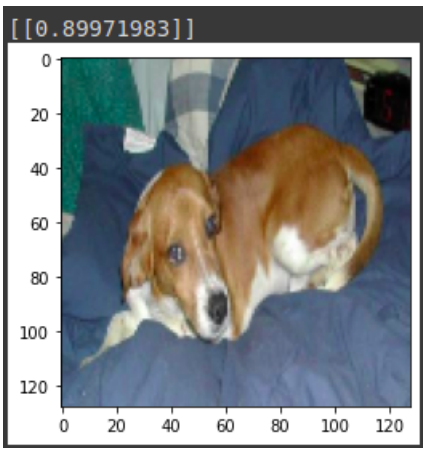


Figure 4.26: Classification result

4.14 Data Augmentation, L_1 and L_2 regularizations

This section includes training details, model accuracy and results obtained for a CNN model with data augmentation, L_1 and L_2 regularizations.

4.14.1 Training details

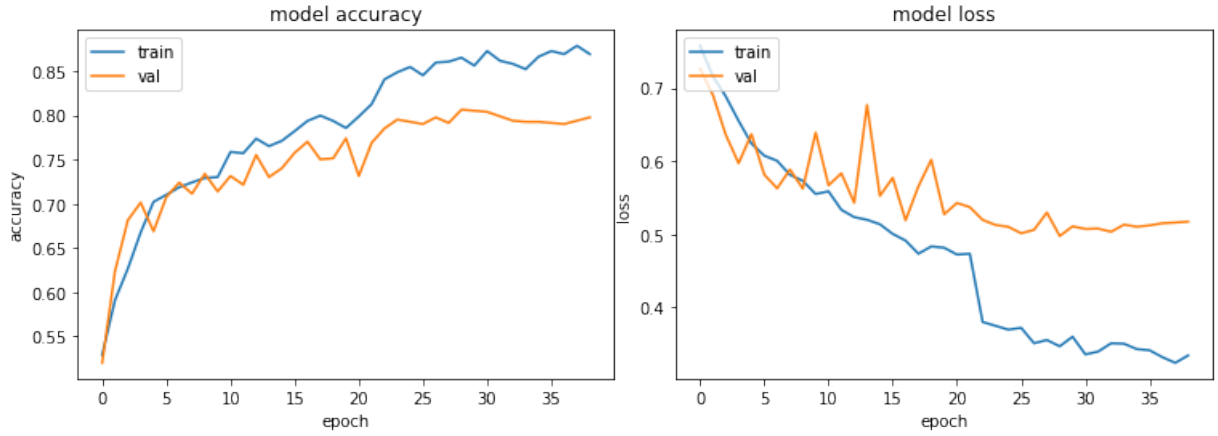
Table 4.14 gives the training details of CNN model with data augmentation, L_1 and L_2 regularizations.

Table 4.14: Data augmentation, L_1 and L_2 regularizations

Number of epochs	29
Training images used	2000
Validation images used	800
Training accuracy	0.87
Training loss	0.34
Validation accuracy	0.81
Validation loss	0.49

4.14.2 Loss and Accuracy

Figure 4.27a shows the training and validation accuracy and figure 4.27b shows the training and validation loss for a CNN model with data augmentation, L_1 and L_2 regularizations.



(a) Accuracy of a CNN model with data augmentation, L_1 and L_2 regularizations.

(b) Loss of a CNN model with data augmentation, L_1 and L_2 regularizations.

4.14.3 Result obtained

Figures 4.28 shows the classification result for a CNN model with data augmentation, L_1 and L_2 regularizations.

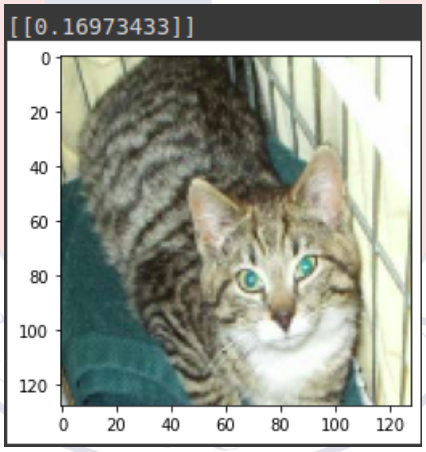


Figure 4.28: Classification result

4.15 Random Image Cropping and Patching (RICAP)

This section includes training details, model accuracy and results obtained for a CNN model with RICAP implementation.

4.15.1 Training details

Table 4.15 gives the training details of CNN model with data augmentation, L_1 and L_2 regularizations.

Table 4.15: RICAP

Number of epochs	266
Training images used	2000
Validation images used	800
Training accuracy	0.83
Training loss	0.48
Validation accuracy	0.87
Validation loss	0.34

4.15.2 Result obtained

Figures 4.29 shows the classification result for a CNN model with RICAP.

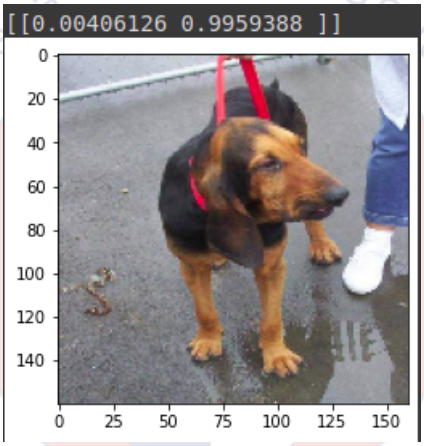


Figure 4.29: Classification result

4.16 Random Image Cropping and Patching (RICAP)– DenseNet

This section includes training details, model accuracy and results obtained for a CNN model with RICAP implementation. A DenseNet is a type of convolutional neural network that utilises dense connections between layers, through Dense Blocks.

4.16.1 Training details

Table 4.16 gives the training details of CNN model with data augmentation, L_1 and L_2 regularizations.

Table 4.16: RICAP with DenseNet

Number of epochs	25
Training images used	2000
Validation images used	800
Training accuracy	0.87
Training loss	0.43
Validation accuracy	0.94
Validation loss	0.17

4.16.2 Result obtained

Figures 4.30 shows the classification result for a CNN model with RICAP.

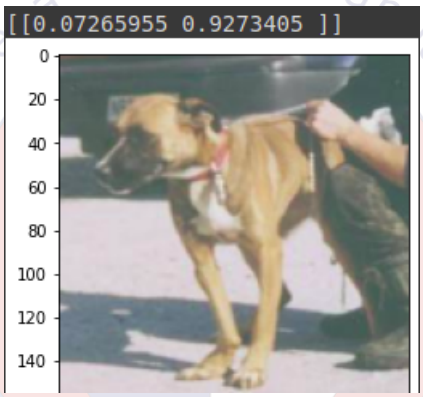


Figure 4.30: Classification result

To summarize, this chapter gives independent results obtained for each overfitting avoidance algorithm. For each technique, training details, accuracy and loss curves and classification results obtained using the corresponding technique is given.

Chapter 5

Conclusion and Future Scope

CHAPTER 5

CONCLUSION AND FUTURE SCOPE

5.1 Conclusion

Machine learning algorithms are efficient, but they are prone to overfitting due to the inclusion of millions or billions of parameters. When a function is closely fitted to a set of data points, this is known as model overfitting. When a model has too many parameters to learn, it suffers from overfitting. As a result, it gathers additional data, limiting the model's generalisation. Generalization is a learning metric that measures a model's ability to make correct predictions based on previously unseen data. As this lowers, the precision of the prediction drops, lowering the model's accuracy. As a result, it's critical to avoid model overfitting. The objectives of this report are to implement the different overfitting avoidance algorithms such as data augmentation and regularization techniques on a CNN model. These techniques are then analysed to make a comparative study. RICAP a data augmentation strategy that integrates mixup, cutoff, and soft labelling techniques, is also implemented.

To achieve the mentioned objectives, initially, a Convolutional Neural Network model is designed and built. This model is trained and the results of overfitting are recorded. After this step, the overfitting avoidance algorithms are included during training and parameters such as accuracy, loss and number of epochs are identified. These parameters are compared for all the overfitting avoidance algorithms.

All of the strategies used were successful in preventing model overfitting. A combination of a couple of these strategies was used to increase the model's validation accuracy, lowering the error on unseen data. The validation accuracy improved from 65% to 94%, and the validation loss reduced from 2.5 to 0.17.

5.2 Future Scope

Model overfitting is one of the most common and known problems faced in Convolutional Neural Networkss. Their prevention is extremely important to improve the overall accuracy of the model.

- The process explained in this report can be extended to included more algorithms

like cross–validation and more combinations of the algorithms mentioned can be analysed.

- Different optimisers like Adam, Adagrad and RMS prop can be implemented and the results can be compared.
- The scope of comparison can be extended to included additional parameters.

5.3 Learning Outcomes of the Technical Seminar

Over the course of the project, many concepts and tools were learnt, and the project was built successfully.

- Understanding the working of Convolutional Neural Networks and its layers.
- Learning statistical modelling errors such as overfitting and underfitting and its causes.
- Understanding the importance of preventing model overfitting.
- Understanding regularization techniques such as Early stopping, Dropout, L_1, L_2 and $L_1 - L_2$.
- Understanding data augmentation techniques like image transformation and synthetic image generation.
- Understanding the RICAP data augmentation method and its advantages over traditional data augmentation.

BIBLIOGRAPHY

[1] I. Bilbao and J. Bilbao, "Overfitting problem and the over-training in the era of data: Particularly for artificial neural networks," in *2017 Eighth International Conference on Intelligent Computing and Information Systems (ICICIS)*, 2017, pp. 173–177. DOI: 10.1109/INTELCIS.2017.8260032.

[2] H. Allamy, "Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study)," Dec. 2014.

[3] H. Li, J. Li, X. Guan, B. Liang, Y. Lai, and X. Luo, "Research on overfitting of deep learning," in *2019 15th International Conference on Computational Intelligence and Security (CIS)*, 2019, pp. 78–81. DOI: 10.1109/CIS.2019.00025.

[4] R. Moradi, R. Berangi, and B. Minaei, "A survey of regularization strategies for deep models," *Artificial Intelligence Review*, vol. 53, Aug. 2020. DOI: 10.1007/s10462-019-09784-7.

[5] F. Kamalov and H. H. Leung, "Deep learning regularization in imbalanced data," in *2020 International Conference on Communications, Computing, Cybersecurity, and Informatics (CCCI)*, 2020, pp. 1–5. DOI: 10.1109/CCCI49893.2020.9256674.

[6] A. Lodwich, Y. Rangoni, and T. Breuel, "Evaluation of robustness and performance of early stopping rules with multi layer perceptrons," in *2009 International Joint Conference on Neural Networks*, 2009, pp. 1877–1884. DOI: 10.1109/IJCNN.2009.5178626.

[7] X. Zhang, Z. Wang, D. Liu, and Q. Ling, "Dada: Deep adversarial data augmentation for extremely low data regime classification," May 2019, pp. 2807–2811. DOI: 10.1109/ICASSP.2019.8683197.

[8] R. Takahashi, T. Matsubara, and K. Uehara, "Data augmentation using random image cropping and patching for deep cnns," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 9, pp. 2917–2931, 2020. DOI: 10.1109/TCSVT.2019.2935128.

[9] S. Yadav and S. Shukla, "Analysis of k-fold cross-validation over hold-out validation on colossal datasets for quality classification," in *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, 2016, pp. 78–83. DOI: 10.1109/IACC.2016.25.

[10] S. K. Noon, M. Amjad, M. A. Qureshi, and A. Mannan, "Overfitting mitigation analysis in deep learning models for plant leaf disease recognition," in *2020 IEEE 23rd International Multitopic Conference (INMIC)*, 2020, pp. 1–5. DOI: 10.1109/INMIC50486.2020.9318044.

[11] A. Oppermann, *Regularization in Deep Learning — L1, L2, and Dropout*, Feb. 2020. [Online]. Available: https://towardsdatascience.com/regularization-in-deep-learning-l1-l2-and-dropout-377e75acc036.