Table of Contents

List of Figures

List of Tables

Abstract

The major difference between humans and computers is intelligence, that is the ability to make the right decisions at the right time based on past experiences. Computer scientists strive to bridge this gap to make computers intelligent. Humans have senses to take input in the form of image, sound, taste, smell, and touch. Computers have mechanical sensors for different types of inputs, for example, a camera for capturing visual input, microphone for capturing sound, light sensors to capture the ambient light intensity and so on. The human brain quickly infers the meaning of the input they receive. For humans, processing visual data is a simple task, on the other hand, when computers are given an image as input, it is a very difficult and complex task for computers to describe or understand the contents and/or focus parts of an image. Now-a-days Deep Learning performs a major role in the manipulation of Visual data with the help of Convolutional Neural Networks (CNN). We designed CNNs to train prediction models which will help us in visual reorganization tasks like Image Caption Generation. In CNN, we train a single model for number of times to attain best results but problem arises when we have a model with different results every time. Our proposed approach will help to overcome the fuzziness of the predictions models.

1 Introduction

The major difference between humans and computers is intelligence, that is the ability to make the right decisions at the right time based on past experiences. Computer scientists strive to bridge this gap to make computers intelligent. Computers are programmed to imitate intelligence by means of mathematical calculations upon previously learned experiences. This is known as Artificial Intelligence (AI). An important aspect of human intelligence is context awareness and perception. Humans have senses to take input in the form of image, sound, taste, smell, and touch. Computers have mechanical sensors for different types of inputs, for example, a camera for capturing visual input, microphone for capturing sound, light sensors to capture the ambient light intensity and so on. After the input is received, the next step is perception. The human brain quickly infers the meaning of the input they receive. Humans majorly rely on their vision to discover the world around them and act accordingly. For humans, processing visual data is a simple task, on the other hand, when computers are given an image as input, it is a very difficult and complex task for computers to describe or understand the contents and/or focus parts of an image. The branch of Computer Science (CS) that encompass with the manipulation of digital images using computers is known as Digital Image Processing (DIP). There are 2 major types of tasks performed in DIP:

- Graphics information enhancement to be evaluated by humans
- Processing images for machine perception without human involvement – *Computer Vision*

1.1 Computer Vision

Computer vision, an inter-disciplinary field combining powers of Machine Vision and Image processing domains. Computer Vision tackles the complexity of the images processing techniques which enables us to extract some meaningful information from image/video which is provided. The main purpose of the computer vision domain is to make computers too smart to understand visual data they gather from the world around. Computer vision scientists are developing different techniques and algorithms in an attempt to make computers achieve autonomous visual perception.

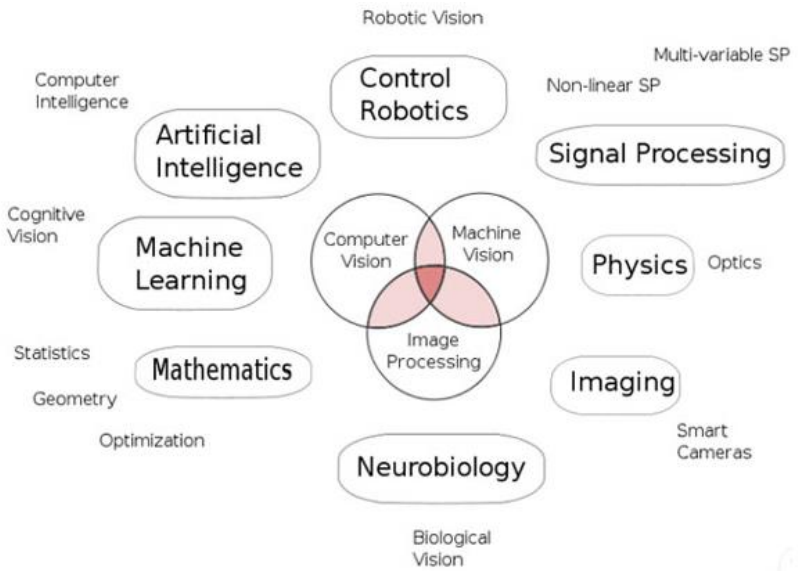

Figure 1. 1 Inter-disciplinary nature of research domain

In recent times due to the vast and easy availability of camera fitted smartphones at affordable prices, capturing and sharing images has become a part of people's daily lives. Due to this change, there has been an influx of images available to the world through the internet. Humans have felt the need for computers to make sense of images as humans can by just looking at them. For example, when a human sees an image of a dog running on the beach, he can instantly tell that it is a dog which is running on beach but for computers, it is not as simple as it sounds because computer sees an image as an array of numbers.

1.1.1 History of Computer Vision

The history of image processing starts from the 1960s for enhancement of images received from satellites, character recognition, medical imaging for assisting doctors in disease diagnostic, image restoration, and improvement or blurred the noisy or damaged image. Computer vision initiated at universities that were working on making intelligent machines. The key purpose at that time was to mimic the human visual system in order to make robots "see" their environment, understand it and make appropriate decisions [1].

The key point difference between computer vision and digital image processing was that computer vision focused on the extraction of 3D features from images in order to achieve complete scene understanding. In the 1970s, studies were carried out that laid

the foundations for several algorithms **feature extraction** that is being used even today, that include edge extractors, line detectors, object representation as inter-connections of tiny structures like corner points, motion rough calculating and optical flow [1].

In the 1980s, studies were carried out based upon more demanding math analysis and quantifiable characteristics of computer vision including scale-space, the inference of object shape using many other factors such as texture, color, focus of the image, shading, and edges. Research Specialist also comprehended that several of these math concepts could be processed similar to the optimization of a framework [2]. By the 1990s, image segmentation problem was solved using a variety of graph cut technique in which a graph of pixels is created and a graph cut with minimum cost is calculated but this technique requires some human input like object location and it is not a fully automated method. Statistics-based machine learning procedures were utilized practically to perform facial recognition in images. These techniques were used to identify patterns in images of each person and learn a function/model that could recognize an unseen image and make it the correct person [1].

In order to solve recognition-based tasks, image representation as key features was a vital step. For this purpose, various types of feature extractors were developed with several hybrid approaches. Once feature extraction was done, then these features are fed into machine learning algorithms for modeling also known as training phase [3]. After the training, we generate a model which we will use afterward.

1.1.2 Applications of Computer Vision

Computer vision can be applied in several different fields which include:

- Biometrics
- Search Engines
- Character Recognition
- Autonomous Vehicles
- Remote Sensing
- Image Classification
- Image Captionization
- Object Detection
- Medical Image Analysis

- Security and Surveillance
- Industrial Quality Inception
- Facial Recognition
- Gesture Analysis

1.1.3 Computer Vision Process

Computer scientists aim to provide the capabilities of human vision to computers or at least try to get as close to that as possible. But this task is not easy as it may sound. At a higher-level Computer Vision tasks may be split into three steps:

- Image Acquisition
- Image Processing
- Image Analysis and Decision

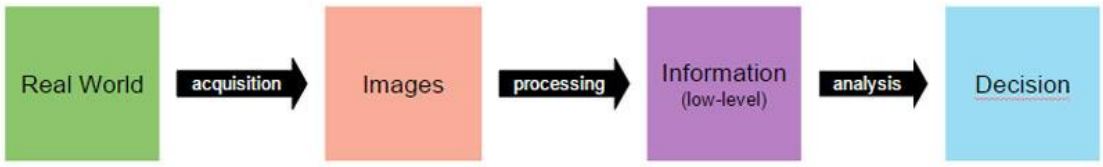

Figure 1. 2 Computer vision process

Image Acquisition

Images are acquired by computer using digital cameras or you may capture the image and put into your computer which is very commonly available these days. The camera is available in a wide range with varying image quality and purpose. Some cameras are special purpose such as thermal imaging cameras which sense heat and create an image whereas ultrasonic cameras use high-frequency sound waves for image creation.

Image Processing

The acquired data is processed by applying algorithms to extract very low-level feature representation of parts of the image. These features are characterized by edges, corner points, colors, textures and objects for example. These are preliminary elements that build up objects in images.

Image analysis and decision making

In this step, the features extracted in the previous step are analyzed for patterns, which is then used in taking a decision. The type of decision to be made is application specific, like biometric verification or image retrieval for a search query, or image label assignment or **image caption generation**.

1.2 Image Classification and Detection.

The most or we can say major important tasks in the computer vision discipline are image classification and object detection as they provide basis for advance tasks like content-based image retrieval, image tagging, **image caption generation**, biometric verification, security and surveillance.

In computer vision domain, image classification is a general work in computer vision where computers try to classify an input image into a collection of finite number of classes. For example, if a classifier model is trained on two classes that is, cats and dogs, given an input image, it should be able to classify it as a dog or cat based on the learned features. Object detection is the next step in computer vision after image classification in which the positions of the objects are detected in the image. For example, if a classifier classifies an image as a cat then a detector's job is to detect the location of the cat in the image.

1.2.1 Challenges in Image Classification

In our real-world scenarios, there are too many factors that make image classification difficult for computers. Following are some of the challenges faced in image classification:

Viewport Variation

All pixels in an image change when the camera moves. It means that the computer sees the same image differently when it is taken from a different angle even human can also not able to understand the meaning.

Figure 1. 3 Different viewport variations in images

Illumination

Images look different when captured under different lighting circumstances.

Figure 1. 4 Effect of different illumination in images

Deformation

A subject can sometimes be found in a shape that is very different from its regular shape.

Figure 1. 5 Effect of subject deformation in images

Occlusion

Sometimes an object is partly visible in the image due to the presence of some other object(s) in an image that may be covering some part of the main object.

Figure 1. 6Effect of subject occlusion in images

Background Clutter

The colors of the subject in an image may be similar to the background making it difficult to differentiate between foreground and background.

Figure 1. 7 Effect of background clutter in images

Intra-class variation

There are often variations among objects of the same class. For example, a cat can be different colors such as brown, black, white, grey and a combination of colors as well.

Figure 1. 8 Intra-class variations among cats

1.2.2 Image Classification Process

The task of image caption generation using a computer can be decomposed into following steps:

1. Image Features Extraction
2. Training Model on Extracted Features
3. Testing Model with Query Images

Traditional Image Features Extraction Techniques

In the past algorithms were written to manually detect edges, corners, points and other shapes in efforts to attempt to extrapolate features that could be used for classification. These algorithms include Harris [4], SURF [5], HOG [6], SIFT [7], BRISK [8] and FREAK [9]. However, these solutions were not scalable. They were affected by factors such as scaling, translation, rotation and background clutter. For example, if a computer was shown straight images of cat and at test time a rotated image of a cat was presented, the computer might not be able to detect it correctly as a cat. Efforts were made to make the algorithms scale, rotation and translation invariant but the results were not good enough to compete human vision.

Traditional Feature Learning Techniques

After the feature extraction phase, the next step is to analyze patterns in these features, so that a computer can “learn” to distinguish among images of different classes. The

domain that deals with pattern recognition and modeling algorithms are known as **Machine Learning** (ML). In this domain, computers learn patterns from existing example data. With the development of fast computers and availability of large labeled datasets in the 2000s, the recent trend after 2010 is to use Machine Learning (ML) algorithms and techniques to solve computer vision problems/tasks. The basic idea in these techniques is to train the computer by showing a sufficiently large number of images along with its ground truth and then after the training phase has completed a test set is given to the computer for prediction and then results are compared to the actual answers to evaluate the algorithm performance. Several machine learning methods exist including Decision Trees, K- Support Vector Machines (SVM) and K-Nearest Neighbors (KNN) but each has its own merits and demerits.

Mimicking Human Vision System

Neurobiology is a sub-discipline or we can say the combination of both biology and neuroscience, that deals with the study of human nervous system as well as the study of human brain cells and the association of these cells into operational circuits that process input data and facilitate behavior. In order to make computers "see" the world, computer scientists took inspiration from the wide-ranging study of the computer vision system comprising of eyes, neurons, and the brain structures dedicated to handling of visual input in humans. These studies and the interest shown by computer scientists has given birth to a subfield within computer vision where artificial systems are devised to imitate the working of biological systems. Neuron-based system of a human brain has motivated the development of **Artificial Neural Networks (ANN).** A neuron in ANN receives information from other neurons, processes the input and sends output to other neuron but the best part is that neurons adjust just themselves according to their previous mistakes – this is the learning part. When a mistake is made, the error is send back to the previous neurons and then they adjust to improve the results, Artificial Neural Networks are the combinations of many layers of artificial neurons. Each neuron layer works on different types of preliminary features such as edges, corners, lines which in turn when combined create a complex feature set such as blobs, shapes and objects for each category of images.

Deep Learning

An artificial neural network with many cascaded layers of artificial neurons is known as Deep Neural Network (DNN) and Deep Learning is the branch of machine learning in which Deep Neural Networks are used to make computers mimic the working of human intelligence by learning manifold stages of features that correspond to diverse stages of abstraction. These multi-level features build up a conceptual hierarchy of the input data. Deep learning models learn to accomplish classification directly from input data such as images, sound or text. No hard-coded rules or feature extraction algorithms are required. The word "deep" suggests the existence of several layers in a network, arranged in a stack (conceptually) where input is passed through the layers one by one to achieve the hierarchical feature representation. Traditional **neural networks have only 2 or 3 layers of neurons** on the other hand, deep networks can have hundreds of layers. Due to its deep nature, deep learning is a computationally intense technique and requires very powerful dedicated hardware to train deep neural networks.

Starting in 2012, a specific type of network introduce which is a kind of deep neural network known as **Convolutional Neural Networks (CNNs)** began dominating the image classification space – and while deep learning with CNN's has yielded amazing results, we continue to look for more computationally efficient, more accurate, and more descriptive models to apply to this task. As more novel Convolutional Neural Network (CNN) architectures are designed, the current industry standard is for them to be benchmarked through the ImageNet Challenge.

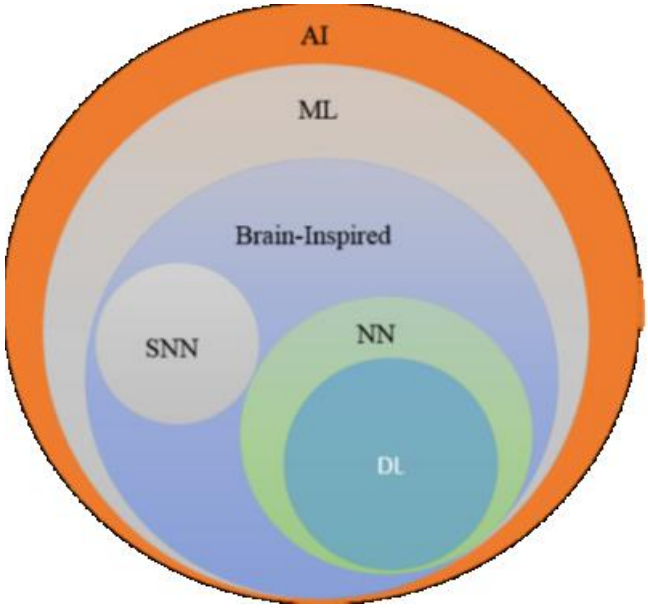

Figure 1. 9 Venn Diagram to Show the Place of Deep Learning [10]

Why Deep Learning Now?

In 1986 Rina Dechter introduced the Deep Learning term to the machine learning community [11] and Igor Aizenberg introduced Artificial Neural Networks in year 2000 [12]. The standard backpropagation algorithm was used by Yann LeCun *et al.* to train a deep neural network for computerized recognition of ZIP codes written by hand on mails [13]. The algorithm achieved its objective, but it did not prove very successful at that time because of the lack of powerful hardware required for training. Two-dimensional image of hand-written digits was recognized using such systems in 1991, while three-dimensional objects were recognized by mapping two-dimensional images with a three-dimensional object model.

It was shown by Brendan Frey in 1995 that using the **wake-sleep algorithm** it makes us eligible to train a network that consisted of 6 layers consisting of hundreds of neurons that were fully connected to each other. Peter Dayan and Hinton developed the wake-sleep algorithm [14] Several factors may cause a slow training speed. Such factors include vanishing gradient problem that was analyzed by Sepp Hochreiter in 1991 [15].

Deep learning started to influence the industry in the early 2000s. In those days Convolutional Neural Networks were being used to process about 10-20% of all the bank checks written by hand in the United States of America. In 2010, deep learning was used for industry level applications of speech recognition.

In recent times, due to the following factors, we can safely say that the computing industry is ready to practically benefit from deep learning:

1. Availability of large labeled datasets
2. Availability of advanced powerful dedicated hardware (on site and cloud-based)
3. Advancement in algorithms

Current Research Trend

In addition to academia, big companies of the computing industry like Google, Microsoft and Facebook are investing resource in developing deep learning models. Google developed GoogLeNet[16] (codenamed Inception v1) in 2014. Microsoft developed ResNet[17] in 2015. Until ResNet, the main idea was that developing deeper networks provided better results but after a certain depth the results started to decline due to vanishing gradient issue. ResNet solved the issue by introducing shortcut

connections with residual blocks. In this project we work on BLEU Score Improvement of the generated caption of image and the stability of the model in training with the help of data augmentation with the help of Flickr8k image dataset Flickr8k. We achieved improved results by data augmentation and using Adamax optimizer.

1.3 Benefits of this research & target audience

This work will benefit the research community as well as application developers working on applications of computer vision such as search engines, self-driving cars, robotics, cameras, smartphones.

Thesis Structure

The rest of this thesis report is arranged as follows: Chapter 2 provides a background insight of deep learning domain, BLEU Score, Data Augmentation and how CNNs work for image classification, Chapter 3 is literature review that provides a comparative analysis of the approaches adopted for improving ResNet. Chapter 4 presents our methodology. Chapter 5 shows our experiments & results and Chapter 6 is consisting of concluding remarks.

2 Background

In order make computers able to predict a caption on an image, computers need to be trained for this task. A computer has to learn with the help of dataset of random images along with captions. We use Flickr8k dataset. But first thing is to learn and there are two main types of ML techniques:

- Supervised Learning

  In this type, the computer is giving a labeled dataset for training, which means that the computer already knows about the inputs and the outputs (with respect to input) are known to the computer at the time of training, the task of the supervised learning algorithm is to understand and create a model for future prediction for the data which was never seen before. After training, there is a portion in a dataset which we used to validate our system or model. If we feed large dataset to our model than the designed model will have higher prediction power.

- Unsupervised Learning

  The computer is given an unlabeled dataset for training, which means the only input data is provided without any labels and it is the unsupervised algorithm's job to learn to group the data on the basis of similarity. So unsupervised learning is not used for classification problems.

Some other hybrid forms of learning such as semi-supervised learning, are also becoming popular which is the combination of both supervised and unsupervised learning.

Commonly used machine learning algorithms used for classification task include:

- Naïve Bayes classifier
- Support vector machines (SVM)
- Nearest neighbors (kNN)
- Decision trees
- Neural Networks

Every algorithm has its own advantages and disadvantages. And algorithm selection depends on the kind of problem at hand.

In starting, neural networks have proved their potential in the image captioning field by providing state-of-the-art results. ImageNet Large Scale Visual Recognition Challenge (ILSVRC) provides a common platform for researchers to measure the performance of their designed algos on a large benchmark dataset consisting of 1.2M images in 1000 categories for only training phase and 50 thousand images for testing phase. Teams compete to design models to achieve higher accuracy on visual recognition tasks such as image categorization, object localization and image caption generation. This has become a standard benchmark in computer vision since 2010, and the competition is held annually. PASCAL Visual Object Classes (VOC) was a similar challenge, in the computer vision domain established in 2005. But it was smaller-scale competition compared to ILVRC as it consisted of only about 20,000 images from twenty object categories. Around 2011, state of the art error rate at ILSVRC was around 25%. In 2012, a deep CNN achieved an error rate of 16%, which was a major breakthrough at that time. In the next few years, error rates dropped to a few percents. By 2015, deep neural networks were able to surpass human accuracy at the ILSVRC tasks as reported in.

In this section we take a deeper look into the neural networks specifically Convolutional Neural Networks (CNN) used for image categorization.

2.1 Artificial Neural Networks

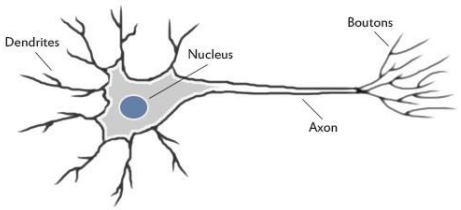

Figure 2. 1 Structure of a biological neuron [18]

An Artificial Neural Network is a mathematical model that is inspired by the working of human bio-neural networks to process data. The human brain uses a large interconnected network of neurons to process data and transmit signal throughout the body. A neuron receives inputs through dendrites from other neurons and then processes inputs. If the outcome of this computation crosses some threshold limit, it sends out a signal to other neurons linked via the axon. In this way signals are propagated to the central nervous system which then sends back corresponding

commands to the concerned neurons. For example, if a finger is burning at a candle flame or a finger is pricked with a needle, the stimuli signal is propagated through neurons to the central nervous system which then sends signals to the arm to move instantly. This all happens so quickly that we do not realize the processing going behind this.

2.1.1 Computational model of a neuron

A neuron is basically a smallest unit in neural network. A neuron is also known as a node or unit. It accepts input from others and process while applying some calculation to produce an output. Based on the relative importance, each input is assigned a weight, say w. The neuron computes a function, say f, using the weighted aggregate of its inputs.

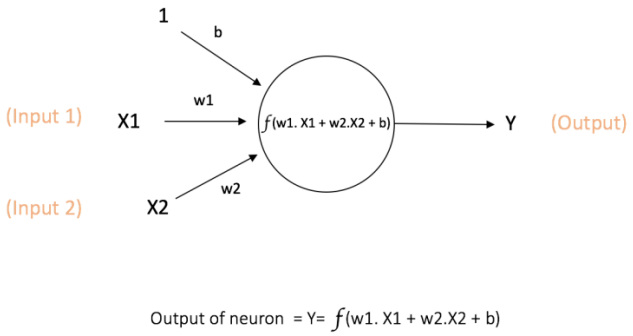

Figure 2. 2 Theoretical models of an artificial neuron [18]

The neuron is shown in Figure 2. 2 takes inputs X1 and X2 with w1 and w2 associated weights respectively. There is a third input labeled as “1” with associated weight b called the bias. The function f of the neuron is called the activation function because its value determines whether the neuron will be activated or not. Usually, a non-linear function is used for a neuron to inculcate non-linearity into the output. Most of the data in real-world scenario are non-linear and cannot be classified using a linear straight-line function, that is why using a non-linear function enables the network to learn these non-linear representations.

Several activation functions exist in literature but here we present a summary of the ones most commonly used with neural networks:

- **Tanh:** takes a numeric input and outputs in the range [-1, 1]. Function definition is as below:

$$f(x) = \sinh(x)/\cosh(x)$$

- **Sigmoid:** takes a numeric input and its output is in the range between 0 and 1. Sigmoid function can be mathematically expressed as follows:

$$f(x) = 1 / (1 + \exp(-x))$$

- **ReLU:** stands for Rectified Linear Unit. It takes a real-valued input and outputs zero for negative values otherwise the output is same as the input

$$f(x) = \max(0, x)$$

ReLu is the most popular function for neural networks due to its good performance but it has a problem known as the "dying ReLu" problem, that is, as the network becomes deeper, the output of ReLU progressively becomes so small (dies) that the neurons stop activating and ultimately the networks stops learning further.

- **Leaky ReLU:** is an effort to solve the dying ReLU issue. Instead of the output being zero for negative inputs, a leaky ReLU has a small negative slope. Function definition is as follows:

$$f(x) = \alpha x \quad \text{when } x < 0$$
$$f(x) = x \quad \text{when } x >= 0$$

where α is a very small constant such as $1e^{-4}$. Another enhancement that can be made in this approach is to make the slope a learnable parameter instead of fix value. This approach is termed is Parametrized ReLU (PReLU) and is introduced in [19].

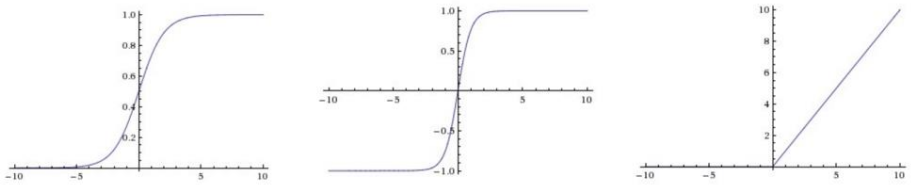

Figure 2. 3 Function graphs for popular activation functions

2.1.2 Feedforward Neural Network

The first and most common type of neural network is the feedforward neural network. This network may contain many neurons which are arranged in layers. Neurons belonging to same layers have connections between them. A numeric weight is associated with each connection.

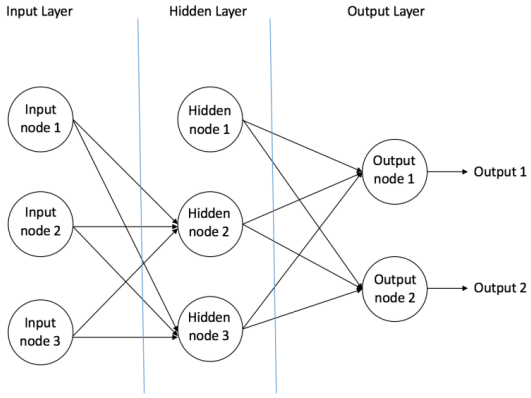

Figure 2. 4 Feedforward Neural Network Structure [20]

Three types of layers may exist in a feedforward neural network:

1. **Input Layer** – this layer receives some input from outside and passes it on to the next layer without performing any computation. This layer determines the dimensions of input acceptable by the network.
2. **Hidden Layer** – this layer receives data from the previous layer which is input layer, performs computations and transfer information to send it to the output layer.
3. **Output Layer** – receives data from hidden layer and performs final. The output of this layer is delivered as the network's output to the outside world.

As the name suggests, in a feedforward network, data travels only in forward direction, from the input nodes, through the hidden nodes finally reaching the output nodes. Feedforward networks can be divided into two main kinds:

1. **Single Layer Perceptron** – It contains only input and output layer; no hidden layers are present.
2. **Multi-Layer Perceptron** – In addition to input and output layers, this type of perceptron contains one or more hidden layers.

Single-Layer Perceptron

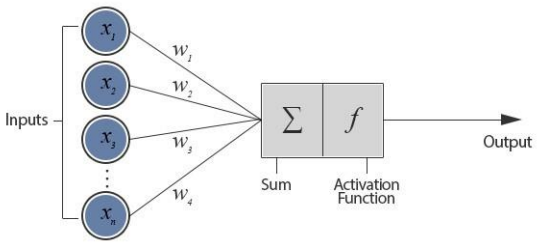

Figure 2. 5 Structure of Single Layer Perceptron

Error! Reference source not found. shows a neuron that receives n real-valued inputs (x_1, x_2, ... x_n) with associated real-valued weights (w_1, w_2, ... w_n). This setup is known as a Perceptron.

As the input is received, a weighted sum of the inputs is calculated and passed on to the activation function. One of the simplest activation functions is the step function that gives 1 as output if the input crosses a certain threshold, otherwise its output will be 0.

Consider the following example:

Inputs:

$x_1 = 0.8$ $x_2 = 1.3$

Weights:

$w_1 = 0.4$ $w_2 = 0.7$

Threshold = 1.1

Weighted sum of inputs = *(0.8 * 1.3) + (0.4 * 0.7) = 1.32*

When this weighted sum is fed to the step function, it will output 1 because the value is greater than the threshold

Training in Perceptron

In order to make a computer learn, we first need to see how a human learns and then we can mimic that process for a computer. If we are to teach a child to recognize cars, we show the child some pictures of cars and some pictures of objects that are not cars. We keep doing this until the child is able to differentiate cars from non-cars in the given pictures. Then we can say the child has learned to recognize cars. In order to make this training more effective, we show the child pictures of different types of cars with variation in shape, size, and colors, rather than pictures of the same type of car.

Similarly, when a set of inputs is provided to a perceptron, one by one, it first tries to calculate the output using existing weight and if the output is wrong, then adjusts its weight in attempt to reach the correct output. Mathematically, this process can be expressed with an equation as follows:

$W(i) = W(i) + a * (C-O) * X(i)$

Where, W is the weight. X is the input. C is the correct output, O is perceptron's output and *a* is the learning rate, for all inputs *i*.

When a perceptron is able to output correctly for each training input then we say that the perceptron has been trained to solve the current problem.

Now, if the perceptron is provided an input that it has never seen before, then it will use its learned weights to output.

The perceptron basically tries to separate the inputs into two regions with a straight line in two-dimensional space depending on the threshold. Points one side of the line are considered in one category whereas the points in the other region are considered in the second category.

Limitation of Perceptron

As it is evident that a perceptron is merely a linear binary classifier but in real-world scenarios data may not be separable using a simple straight line, hence in such situations perceptron will never have power to learn to correctly identify all inputs.

2.1.3 Multi-Layer Perceptron

A Multi-Layer Perceptron (MLP), in contrast to Single Layer Perceptron, has one or more hidden layers along with an input layer and a output layer. MLP is not limited to learning single layer perceptron, rather it has the potential to learn nonlinear models as well which makes it well suited for real-world problems.

Error! Reference source not found. shows a multi-layer perceptron with one input, o ne output and one hidden layer. The input layers receive data from outside world and pass it on to the nodes in the hidden layer, which perform computations on the weighted aggregate of the inputs and their output is sent to the output layer where some additional computation is performed before final output is sent out.

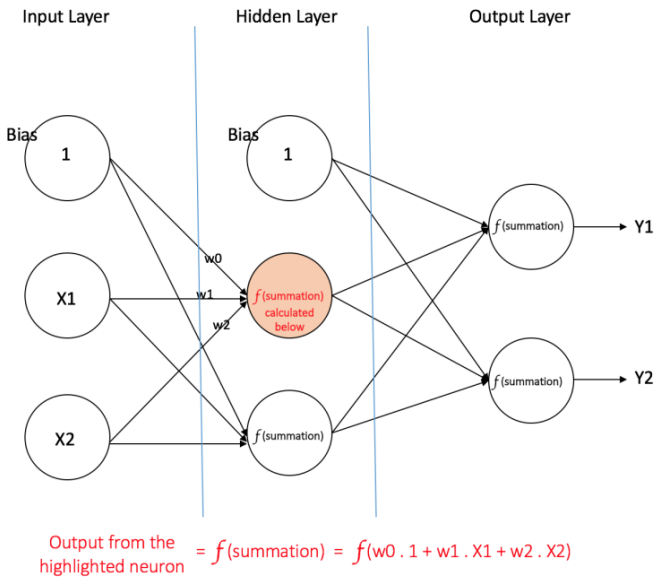

Figure 2. 6 Structure of Multi-Layer Perceptron

Consider an input vector with features **X = (x1, x2, …)** which is going to target **Y**, a Multi-Layer Perceptron can learn the mapping between the inputs and the target, for either classification or regression.

To train a network, a learning algorithm is required that can update the network weight by calculating the amount error, to determine how much change should be made to the weights.

Training MLP using Back-Propagation Algorithm

The final prediction of a network is calculated at the output layer, where this output is compared with the correct output to determine the amount of error and this error value can be used to update weights backward from output to hidden layer but not the input layer, but issue here is that in order to update the weights between the input layer and the hidden layer, the error should be known at the hidden layer. The solution here is to use the "back propagation algorithm" in which the model takes the errors which are calculated at the output layer and proportionally they propagate those errors backward to the all hidden layer.

2.2 Convolutional Neural Networks

Convolutional Neural Networks are lyes in the category of Neural Nets that have proved to be very successful machine learning technique in areas such as speech recognition and computer vision. CNNs have been effective in classifying and recognizing

thousands of different objects. CNNs are being used in real-world applications such as robots, self-driving cars and search engines.

Convolutional Neural Networks are made with of neurons. Every neuron has a learnable weight as well as a bias. Every neuron in CNN receive many inputs from neurons of previous layers which are connected to it and sum all the weights coming from the previous neurons and apply an activation function on the sum and the output of an activation function will be sent to the other neurons which are connected to that neuron. The loss is calculated on the complete network with the help of a loss function which is used to minimize error though backpropagation and weight adjustment as discussed earlier.

So, how are Convolutional Neural Networks different than Neural Networks?

One of the major differences between MLP and a CNN is that CNNs are not fully connected which means that each neuron is connected to only a few neurons in the previous layer and these neurons share weights. Before we go any deeper let's have a look at a relatively simple architecture of a CNN and then we will discuss its components and their working.

2.2.1 The LeNet Architecture (1998)

LeNet was one of the pioneering CNNs which helped boost the field of deep learning. [21] The first LeNet architecture was published in 1989 but after several revisions, the final architecture was published in 1998 and it was named LeNet5. In those days, at that time the main focus of this research was character recognition for computerized processing of zip codes and checks.

This section presents an overview of the working of LeNet architecture and how it learns to classify images. The success of LeNet attracted many researchers towards developing deep-learning based solutions for computer vision building upon the main concepts from the LeNet.

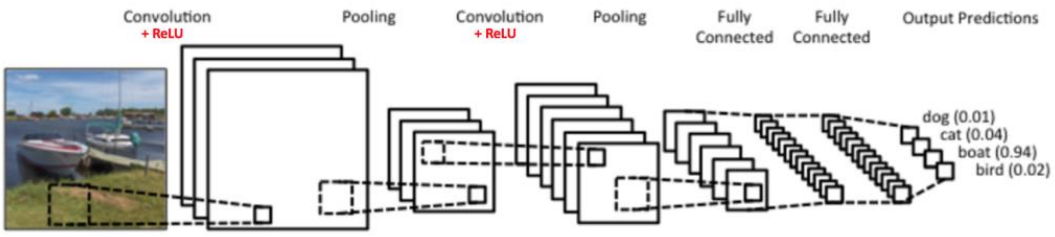

Figure 2. 7 LeNet CNN Architecture

Figure 2. 7 shows a CNN that has an architecture similar to the original LeNet and classifies an input image into one of four classes which are a bird, boat, cat, and dog.

Four main types of layers are used in this CNN:

1. Convolution
2. Pooling or Sub Sampling
3. Non-Linearity (ReLU)
4. Classification (Fully Connected Layer)

A brief overview of the purpose and working of these layers is presented as follows.

2.3 Convolution Layer

As the name of Convolutional Neural Network suggests, Convolution layer is the main building chunk of CNNs. After the input layer, Convolutional Layer is always the first hidden layer in a CNN. At a higher level the basic aim of this layer is feature extraction from an image and by features we mean from lines, edges, and curves to higher level blobs, textures, and shapes. But the question is how does it do that? The first convolutional layer takes the actual input image pixels as its input. So, the input size of this layer is equal to the number of pixels in the input image. For example, if the size of the input image is 28 x 28 x 3 then the number of neuron in this layer is 28*28=1024. The number of channels in this layer will be equal to 3. Working of a convolutional layer can be explained using a concept of sliding window that starts from the top left corner of the image and slides across from left to right and then tops to bottom direction. In machine learning terminology, the window is called a filter and the region covered by the window is called the receptive field. The size of the windows is called filter-size or kernel size. The filter is basically a three-dimensional array of numbers called weights or parameters where the size of third dimension is equal to the input depth. The operation of filter sliding over the image is called convolving. As the filter convolves,

around the input image, it computes the product of the original values of the pixel of the image and the corresponding value in the filter. Then a sum of these multiplications is calculated to achieve a single value that is representative of are summed up to get a single number which is just representative of the area of input currently covered by the filter. This process is repeated for every position in the input where the filter can fit, by moving the filter left to right by a certain step size also known as stride. If the input size is 28 x 28 x 3 and filter size is 5 x 5 x 3 then the size of output after convolution will be 24 x 24 x 3 array of numbers. This output array is called feature map or activation map.

In order to keep things simple, consider a small example of a 5x5 matrix of pixels as input and a convolution layer with filter size of 3 x 3 and weights as follows:

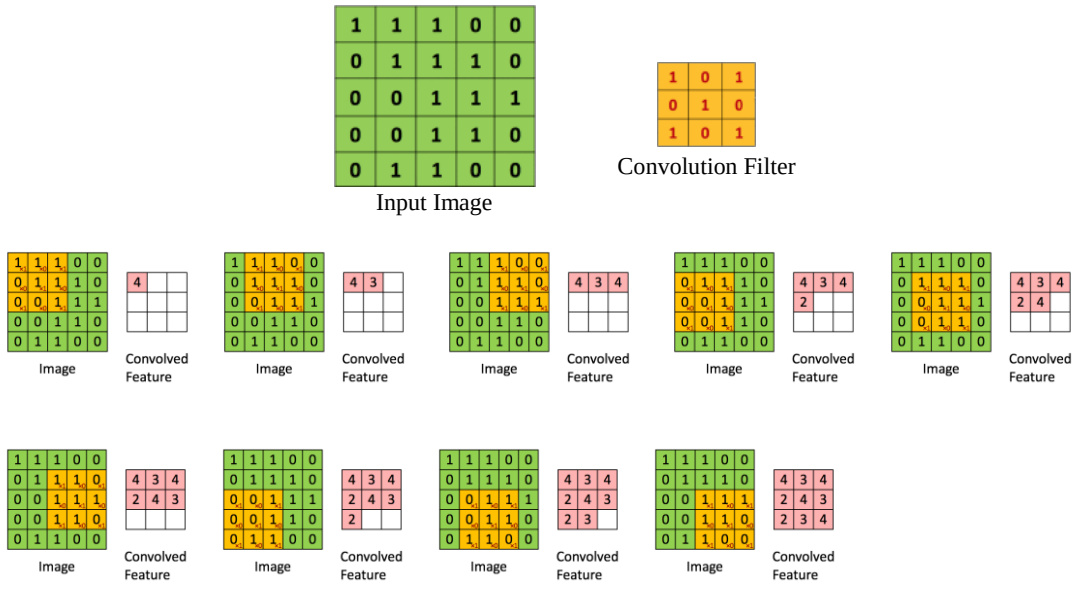

Figure 2. 8 Step by step convolution over 5 x 5 matrix

It is evident from the above images that after the filter has completed sliding over an image a feature map is the output. Taking a closer look, it can be observed that if the values of filter matrix are changed the output feature map will also change consequently, though the input image will be the same. A CNN designer has to specify values for parameters such as filter size, a number of filters, network architecture whereas weights of the filter matrices are adjusted/learned by the CNN during the training phase. A CNN is usually initialized with random weights or using gaussian distribution numbers. During training, a CNN evaluates its error and adjusts accordingly to the weights in an attempt to reduce error.

The size of the feature map is determined by three parameters that are decided by the CNN designer:

- **Depth:** It is a number of filters used in a conv layer. Each feature map output by a filter is stacked as a two-dimensional array, creating rich features maps with an increased number of filters at the cost more computation.
- **Stride:** the total number of pixels by which a filter is moved at each step over the input matrix. There exists an inversely proportional relationship between stride and feature map size. As the stride is increased, the feature map size is reduced.
- **Zero-padding:** Convolving with filter size of greater than 1 results in loss of information at the borders because the filter cannot fit completely at the edges So, an extra layer of zero valued pixels (known as zero-padding) can be added around the image border so that the filter can be applied to bordering elements of the input, preventing information loss at the borders. Zero paddings and feature map size are directly proportional. Increasing zero padding increases the feature map size.

2.3.1 Introducing Non-Linearity (ReLU)

As mentioned earlier that Single Layer Perceptron have the limitation that they can only learn linear functions. So, the Non-Linear activation layer is what allows CNNs to learn complex function and solve non-linear classification. When data cannot be classified using a linear straight line, then we need a non-linear function to model it. In practice, Rectified Linear Unit (ReLU) – a non-linear activation layer and its variants have provided very good results in CNNs. For all negative inputs, ReLu outputs zero. Otherwise, output is equal to input. ReLU function can be defined mathematically as:

$$f(x) = max(0, x)$$

2.3.2 Pooling Layer

Feature maps created after the convolution step are smaller than the actual images but all the values in a feature map are not important. Handling large feature maps can be difficult. The role of pooling is to reduce the size of the feature map while retaining important information. There are many types of pooling like: Max Pooling, Average Pooling, and Sum Pooling.

In the case of Max Pooling, a window is defined of some size like 2 x 2. Then this window is moved across the feature map, starting from the top left corner, and the maximum element from the feature map is taken within that window. In the case of average pooling, the average of all values within a window is taken whereas in case of sum pooling the sum of all values in that window is taken. In general, the improves result is shown by max pooling.

Figure 2. 9 demonstrates max pooling using a window of size 2 x 2.

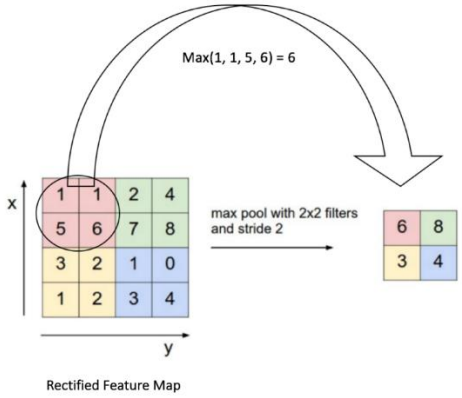

Figure 2. 9 Example of Max Pooling

Pooling step provides the following benefits:

- makes feature maps smaller and easier to manage
- pooling technique decreases learnable parameters quantity and calculations in the network
- makes our the network stable, translations, and transformations in the input image
- assists in achieving scale invariance that loan was the network to detect an object

2.3.3 Fully Connected Layer

The Fully Connected layer is usually used in the output layer of a CNN. This layer uses the softmax activation function. All neurons in this layer is connected to all neurons inthe previous layer, hence the name “full-connected”.

The output from the layer after pooling layer performs its work, we can see the high-level features of the image which was provided. The main propose of this layer is to get features and to classify other images into different classes which are based on the input dataset.

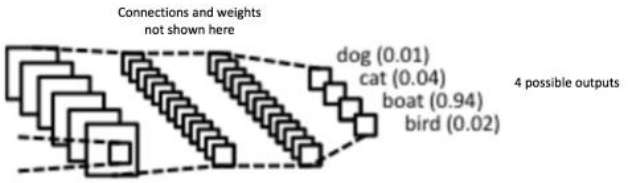

Figure 2. 10 Example of Fully-Connected Layer in a CNN

The sum of output probabilities from the Fully Connected Layer is 1 which is guaranteed if we use softmax as an activation function. The softmax function squashes a vector of arbitrary real-valued scores to a vector of values between zero and one such that all the entries add up to 1.

The Convolution Network training process is listed:

- **Step1:** Initialize network with random values or using any distribution like normal or gaussian distribution
- **Step2:** Feed an image into the network and pass it through all network until it reaches an output layer where softmax calculates the output probabilities for the category. As the weights are randomly assigned for the first time, so maximum chances are that the network will output random probabilities.
- **Step3:** As the correct output is known for training input, compare it with the network's output to calculate the amount of error

  $$Total\ Error = \sum \tfrac{1}{2}\ (target\ probability - output\ probability)^{2}$$

- **Step4:** By using backpropagation, network error is calculated and weights of all network are updated to remove this error in future classification. The weights are adjusted with respect to their contribution to the output.
- **Step5:** Step 2 to 4 is repeated for every image in dataset or training set.

After the training phase is complete, CNN's weights are adjusted ted to correctly classify images.

At the point when another picture that has not been utilized during the training phase is provided as input to the CNN, the network goes through the forward propagation step and yield a likelihood for each class. If our training set is sufficiently expansive, the network will ideally generalize and categorize them into correct classes.

2.3.4 Other layers:

- Addition Layer

- Batch Normalization Layers
- DropOut
- Dense
- Flatten
- Permute

2.4 Other CNN Architectures

CNN has been around since the early 1990s. LeNet earlier was one of the pioneering CNNs. Some other CNN architectures which are very famous nowadays are:

The 1990s to 2012 - 1990s to the early 2010s CNN were in development. With the availability of more labeled datasets and computing power, convolutional neural networks became popular for industry level tasks such as zip codes recognition and hand-written bank checks processing.

the AlexNet (2012) [22] – The ILSVRC in 2012 was won by AlexNet that outperformed all previous state-of-the-art works which were non-neural network based. The error rate was reduced to 16% from the previous record of 25%. AlexNet was based on the foundation laid down by LeNet.

ZF Net (2013) [23] – Matthew Zeiler and Rob Fergus developed a CNN that won the ILSVRC 2013. This network became popular as ZFNet. Basically, it is just an improved version of AlexNet in which author tweaks the hyperparameters.

VGGNet (2014) [24] – The runner-up entry in ILSVRC 2014 was by VGGNet. It lost to GoogLeNet by a very small margin. They developed 2 variants of this network which are VGG-16 and VGG-19. The team's contribution was that they showed that the depth of the CNN plays a vital role in a decent performance.

GoogLeNet (2014) [16] – The winner entry of the ILSVRC 2014 was a CNN developed by Szegedy *et al.* from Google Inc. The main thing in this is the development of Inception Module. In this network, the learnable parameters are reduced. GoogLeNet's consisted of 4 million parameters which are 15 times fewer, compared to 60 million parameters of AlexNet.

ResNet (2015) [17]– ResNet developed by Kaiming He *et al.* from Microsoft Research Team was the winner of ILSVRC in 2015. Their contribution was that they presented a

solution to the issue that occurred in training very deep networks, by introducing shortcut connections. Their winner entry consisted of 1001 layers. Residual Networks are currently state-of-the-art CNN models and are the default choice for using CNNs in practical applications.

DenseNet (2016) [25] –Densely Connected Convolutional Network published by Gao Huang *et al. in 2016*, in this Architecture, every layer is connected to all other layers in a feed-forward manner. The DenseNet has exhibited shows improvements over previous architectures but this performance comes at the cost of the increased number of computation and huge RAM and GPU memory requirements.

2.5 Bilingual Evaluation Understudy (BLEU)

Bilingual Evaluation Understudy [26] is a score, in numeric, for comparing two sentences, one from computer translated and Second is reference. It is also developed for sentences generated from natural language processing. It is a metric to evaluate a machine generated sentence with respect to an actual reference sentence. The BLEU score is not perfect but due to 5 benefits, we use this score as a benchmark:

- This technique is very quick and very cheap in terms of calculation cost.
- This scoring technique is easy to understand by a lay man person.
- The most important benefit of this is that it is independent.
- It looks like the evaluation performed by humans.
- Due to inexpensiveness, this scoring technique is widely adopted.

This scoring technique works while counting and matching n-grams in the generated text to the n-grams in the source text. The comparison is made regardless of orders of the word. If we think that the score is 1 then it's not possible because machine generated description is not exactly what the actual text from the source is. This is also not possible for humans to make the same description as a like reference text.

```
source = [[['hello', 'i', 'am', 'asad'], ['i', 'am' 'asad']]]
generated = [['hello', 'i', 'am', 'asad']]
result = 1.0
```

2.5.1 Cumulative and Individual BLEU Scores

The BLEU score allows you to perform different types of ratings like individual ratings and cumulative rating according to your document.

Individual N-Gram Scores

In individual n-gram scoring, evealuation is happen in just matching grams like 1 gram for single words, 2 grams for matching pair of words. To calculate 1 gram BLEU scoring of a text, we need to specify (1, 0, 0, 0), for 2 gram scoring (0, 1, 0, 0) and so on till 4 gram for example:

```
source = [['hello', 'i', 'am', 'asad']]
generated = ['i', 'am', 'asad']
result = 0.75
```

Cumulative N-Gram Scores

In Cumulative n-gram scoring, we use geometric mean for calculation BLEU scoring. The cumulative 1 gram scoring is equal to the individual 1 gram scoring because in cumulative 1 gram we use (1, 0, 0, 0) which is also use in Individual 1 gram. But in 2 gram cumulative scoring we use (0.5, 0.5, 0, 0), in 3 gram we use (0.3, 0.3, 0.3, 0) and in 4 gram cumulative scorings are (0.25, 0.25, 0.25, 0.25)

```
source = [['hello', 'i', 'am', 'asad']]
generated = ['hi', 'i', 'am', 'asad']


result 1 G: 0.75
result 2 G: 0.50
result 3 G : 0.00
result 4 G: 0.00
```

2.6 Data augmentation

Data augmentation is a helpful way to generate a huge dataset from the small dataset. By using this we can artificially or to used to artificially enlarge our training by making different verities of images.

If we want to make our models skillful to predict or classify than we need more data to train them. So augmentation will be very helpful to us to train our models in a better way.

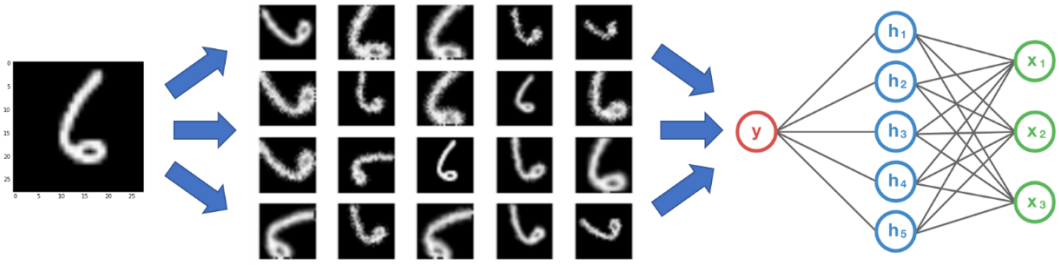

Figure 2. 11 Example of Data Augmentation

2.6.1 Augmentation Techniques

Flip

To augment data, the first approach is to flip image. We can flip it in both directions (vertically or horizontally). The vertical flip is equivalent to the 108-degree rotation of image. Some of the flip examples are shown bellow.

Figure 2. 12 Example of Flip Case in Augmentation

Rotation

Another technique in augmentation is rotation. The important factor in this technique that the width becomes height and the height becomes width. If your image is in the square form then no dimensional effect applies on that image. But if the image is not in the square than the effect will happen if you rotate it in a 90-degree angle. If you rotate it 180 degrees then no effect is made on dimensions.

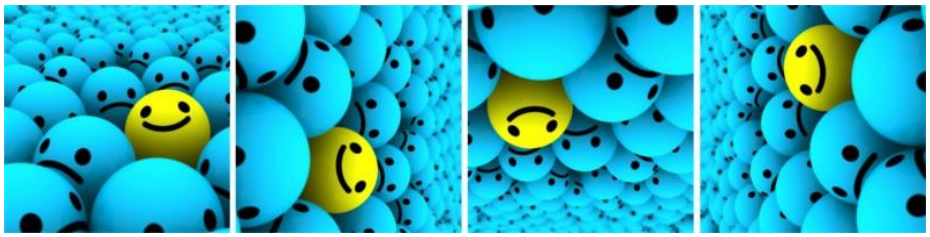

Figure 2. 13 Example of Rotation Case in Augmentation

Scale

Image scaling is also another augmentation technique in which we scale up or scale down the target image to produce different versions of the image. When we scale down the image we need to make image in same dimension so we need to take help with interpolation methods.

Figure 2. 14 Example of Scale Case in Augmentation

Crop

In this technique, we take a part of an image and resize it to the original picture. The part of the image which is going to be selected is random. It looks like the scaling but it is very different from scaling.

Figure 2. 15 Example of Crop Case in Augmentation

Translation

The translation is a process to move the picture into X and Y axes. The below picture is an example. The background of the image is black so we don't need to worry about interpolation. But every image doesn't have a black background. So while adopting this technique, we also need the help of interpolation.

Figure 2. 16 Example of Translation Case in Augmentation

Gaussian Noise

Gaussian noise is a special effect which makes the image noisy which is only happen due to poor light while capturing an image or due to high temperature. The most common type of noise is salt noise and paper noise. In salt noise, small white dots are placed and in paper noise, back dots are placed on the image. The intensity of these dots is controllable.

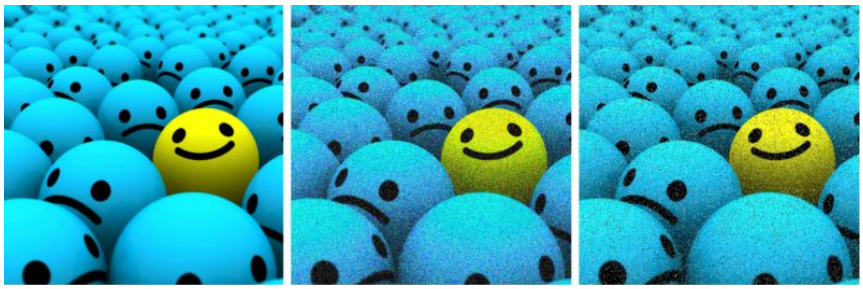

Figure 2. 17 Example of Gaussian Noise Case in Augmentation

2.6.2 Interpolation

After we use augmentation, we need to fix issues produces by augmentation to make out images into the original size. Our image doesn't have any information about what was happened outside the edges of the image. So we need to make this effect cancel by assuming something.

Constant

The simple method is to replace the empty or new generated region with some color value. This technique may not work with a natural image taken from the natural environment but can work with a single color image a monochromatic background

Edge

Another technique is known as "edge" in which edges of the images are extended to fill the empty space of the image.

Reflect

Reflect is another approach, the empty space of the image is filled by reflectiveness of the image. This technique is very useful in natural images like trees and clouds.

Symmetric

This looks like to the reflection, the main difference is that a copy of the edge pixels is made. Normally, reflection and this technique can be similar but the major difference will be seen when we deal with small images.

Wrap

This method is not too famous, but in this, the empty space is replaced with a tiled image. This method makes an error in lots of scenarios so we need to be careful while choosing this.

Figure 2. 18 Example of Interpolitation Cases in Augmentation

3 Literature Review

With the availability of large labeled images data sets like Flickr8k, ImageNet and development of fast, powerful dedicated processing hardware the interest in neural networks revived in the year 2009 onwards.

Up till 2009, all the heavy lifting of training neural networks was being done by CPUs. Researchers were using multiple processors or multi-core processors to train neural nets but CPU is general purpose hardware built to perform generic tasks of a computer. Nvidia, a company is known for manufacturing state of the art GPUs for gaming started working on CUDA cores which are nowadays used in deep neural network training. It was determined by Andrew Ng determined that the speed of deep-learning systems could be increased by 100 times using Graphical Processing Units. This brings along several bundled benefits like ease of scalability and modularity. One can easily add GPU to an existing system, also it is fairly simple to add multiple GPUs and splitting the workload as done in [22]. GPUs consists of hundreds of processing cores that specially designed and optimized calculation in machine learning.

CNN's Review

3.1 AlexNet

In October 2012, **AlexNet**[22] remarkably performed very well with respect to all the previous competitors and won the ILSVRC 2012. The network was built upon the foundations laid down by LeNet but was deeper, with more filters on each layer, and with the sequential order of convolutional layers. AlexNet is trained on 2 Nvidia GPUs and this is the reason behind the splits of the network.

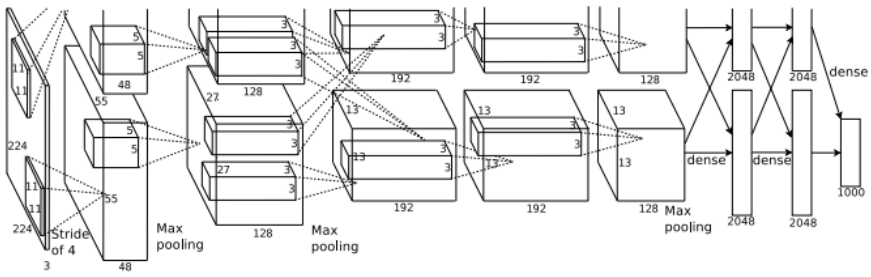

Figure 3. 1 AlexNet Architecture

Since the success of AlexNet, every year the state-of-the-art was a CNN based solution.

3.2 ZFNet

ZFNet(2013) [23] achieved a top-5 error rate of 14.8% which is approximately half of the non-neural error rate: 26.2%. It was built upon the foundations laid down by its predecessor AlextNet. ZFNet achieved this accuracy by just playing with hyper-parameters of AlexNet.

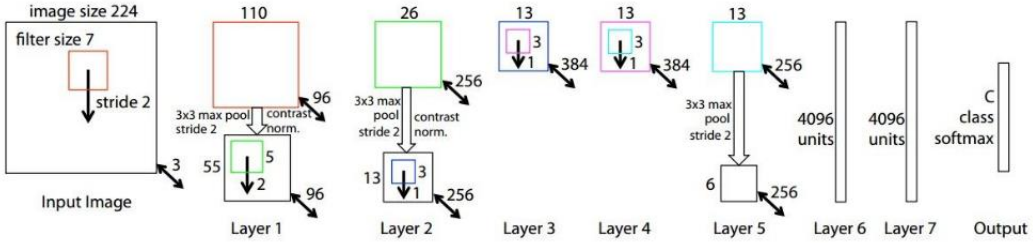

Figure 3. 2 ZFNet Architecture

3.3 GoogleNet

GoogleNet (2014) [16] was the winner CNN at ILSVRC 2014. It was developed by Google and codenamed as Inception. It achieved a top-5 error rate of 6.67% which was very close to human-level accuracy. This CNN built upon the architecture of classic LeNet with the introduction of a special module termed as inception module. This module combines the power of multiple small convolutions along with an average pooling layer, as shown in Figure 3. 3 to enhance the modeling power of CNN and reduce the number of learnable parameters. GoogleNet was of a 22-layer deep CNN with only 4 million learnable parameters.

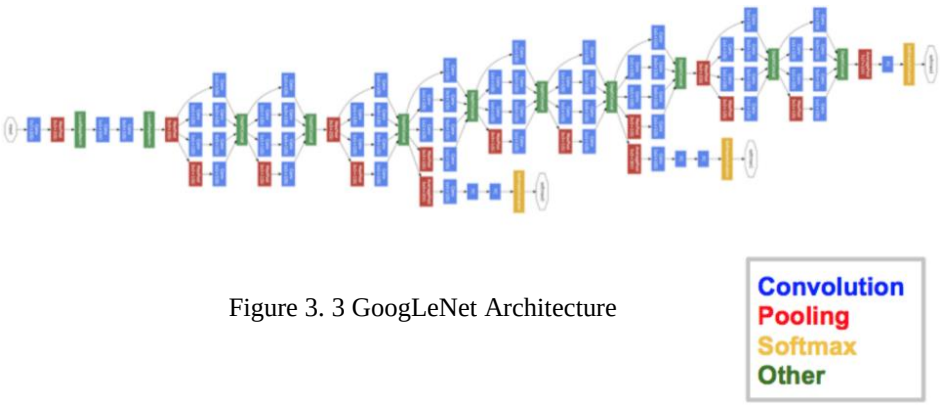

Figure 3. 3 GoogLeNet Architecture

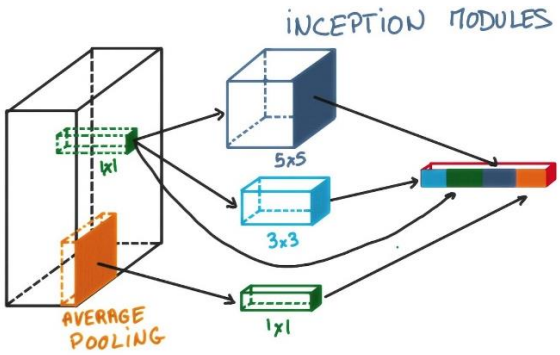

Figure 3. 4 GoogLeNet's Inception Module

3.4 VGGNet

VGGNet (2014) [24] was the runner-up at ILSVRC 2014 losing with a marginal difference from GoogLeNet. VGGNet has two variants known as VGG-16 have 16 convolutional layers whereas VGG-19 have 19 convolutional layers. However, due to a large number of trainable parameters (140 million), it can become a bit difficult to train VGGNet and requires heavy-duty hardware to train.

Figure 3. 5 VGGNet Architecture

One of the main deductions from these studies is that the depth of the networks is crucial parameter and that deeper neural networks tend to have the greater representational capability by using shallow preliminary basic features like edges and corners to form deeper and more complex feature representations like shapes, textures, and objects. For example, in a facial recognition system, pixels from edges and edges form corners. Corners define facial highlights like eyes, nose, mouth, and chin. Facial features compose to define faces.

Theoretically, if we increase the number of layers, it should result in an increase in representation capacity of the network, but that is not the case in practice. CNN's are trained using back-propagation algorithms. One of the most popular back-propagation algorithms is the Stochastic Gradient Descent with Momentum algorithm. First, the neural network is initialized with some weights – usually random or Gaussian normalized. Then a batch of images is fed into the network. The network extracts the features according to current weights and biases and then tries to predict the category

for each image in the batch. Then these predictions are compared with the actual labels of these images. Then this error is propagated back to the earlier layers. The error value is used at each layer calculates partial derivative or gradient with respect to its weights. And then updates its weights accordingly. This amount of change made to the weights is controlled by other factors such as learning rate and momentum. These factors are discussed in detail later. But the network with deep depth is very hard to train, and they suffer from a problem known as vanishing gradient - the issue is that in some instances, the gradient becomes so small that it almost vanishes, consequently preventing the weight from changing its value leading to completely stopping the network from learning further.

In this, a method proposed the Residual Networks [17] to combat the vanishing gradient problem during training very deep convolutional networks while improving performance as well. ResNets have outperformed previous models at a variety of tasks, including as image categorization, object localization and semantic segmentation of images in ILSVRC2015. They are gradually replacing VGGNets [24] in the computer vision community, as the standard feature extractors. A residual neural network, has skip connections that run parallel to the regular stacks of convolutional layers. These skip connections allow the gradients to simply flow back, and as a result, training time is reduced, and accuracy is increased. The Residual Neural Network developed by Microsoft that won the ILSVRC in 2015 has 152 layers, which is 8 times deeper than the last years runner up VGG-19.

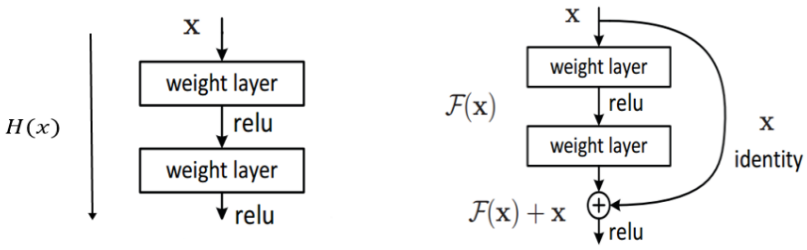

Figure 3. 6 Skip connections in Residual Network

Based on the plain network, shortcut connections are inserted that bypass a block of layers, this block is known as “Residual Block” and the network having such shortcut connections is known as a “Residual Network”. The identity shortcuts can be used directly if the input and output of the shortcut have equal dimensions but when the dimension are different two options are available:

(A) Increase the dimensions of input using zero padding and then perform identity mapping with stride 2. Using this approach, no extra parameters are added.

(B) Use a 1x1 convolution with stride 2 to decrease input dimensions. This is known as projection shortcut.

In [17] two types of residual blocks were proposed. One is known as a "basic block" which is used in ResNet34 and 50 whereas the second architecture is known as "Bottleneck Block" which is used in ResNet-50 and above. We have used ResNet-50 as a base reference model for this project.

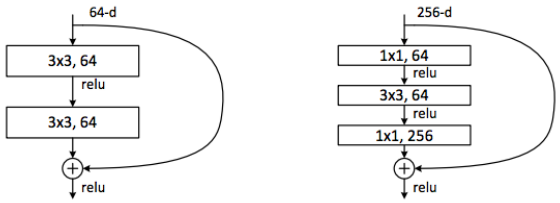

Figure 3. 7 Simple vs. Bottleneck shortcut connection

After the success of Residual Networks, researchers focused their efforts towards this area to further improve their accuracy and efficiency.

3.5 Wide Residual Networks

In [27], the authors highlighted the point that the residual block in [17] with identity mapping that facilitates training very deep networks by mitigating vanishing gradient problem, may also be a potential weak point of residual networks. This issue was expressed as *diminishing feature reuse* in [28]. They increased the width (number of filters) of residual networks and exhibited that the strength of the residual network lies in the residual blocks and that the result of depth is supplementary.

3.6 ResNeXt

In [29], the authors uncovered a new dimension called *cardinality* as an important network parameter besides network depth and width. ResNeXt is a hybrid architecture that combines the strengths of VGGNet, GoogLeNet and ResNet. ResNeXt adopts VGG/ResNets' approach of repeating layers while exploiting Inceptions' split-transform-merge strategy in a simple but extensible way. A module in ResNeXt performs a set of transformations (cardinality is the size of the transformations set), on a low-dimensional embedding, whose outputs are aggregated by summation. The

transformations to be aggregated are all of the same topology. This design allows them to extend to any large number of transformations without specialized design.

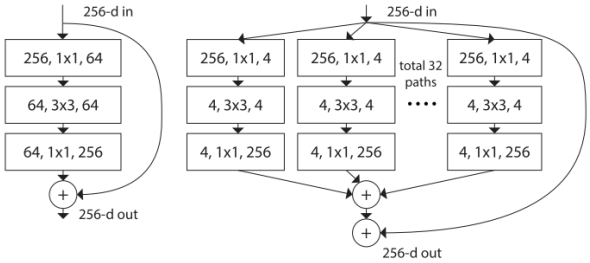

Figure 3. 8 Cardinality in ResNext Network

Parameterized Analysis

Now we present a parameterized analysis of the previously discussed CNNs. First we present a brief definition of each parameter that we have chosen for comparison:

No. of layers – defined as the number of convolutional layers in a neural network

Activation Function – the non-linear activation function used

Top-1 Accuracy on ImageNet – is the conventional accuracy that the model answer (the one with highest probability) must be exactly the expected answer.

Top-5 Accuracy on ImageNet – means that one of the five highest probability answers provided by the model must match the expected answer.

Learning Rate – the initial learning rate used for training

Table 3. 1 Parameterized Analysis of CNN

	No. of Layers	**Activation**	**Best Top-1 accuracy**	**Best Top-5 accuracy**	**Learning Rate**
AlexNet	5	ReLU	61.9%	83.6%	initialized at 0.01 and reduced three times prior to termination
VGGNet	16, 19	ReLU	75.3%	92.7%	initialized at 0.01 and reduced three times prior to termination
GoogLeNet	22	ReLU	-	93.3%	Initially 0.1 and decreased by 4% every 8 epochs
ResNet	18, 34, 50, 101	ReLU	76.1%	96.43	initialized at 0.1, and is divided by 10 when the error plateaus
ResNeXt	50, 101	Leaky ReLU	78.9%	94.4%	initialized at 0.1, and divide it by 10 for three times
Wide ResNet	25	ReLU	78.1%	93.97%	Initialized at 0.01 and dropped at 80 and 120 epochs by 0.1

Papers Review

In [30] the author adopts the retrieval based protocol in which Image query is checked to get a sentence from a pool of reference sentences associated with the image. This protocol can used in any system that can score image and scentences. The idea is image-to-sentence retrieval. To establish the results, they use the Kernel Canonicals Correlation Analysis with the multiple visual and linguistic Kernels to map images and sentence into space where the similarity between them can be computed directly. They train the model using the 6,000 images with real-world captions to each. For further advancement in the results, they ask that they need large training dataset. They provide two reasons for the requirement of the large training dataset.

- Nonlinear image-sentence embedding methods, such as KCCA, tend not to scale to large training sets.
- Obtaining high-quality sentence descriptions for millions of images is a prohibitively expensive task.

They introduce an algo which is called Stacked Auxiliary Embedding that can transfer millions of annotated images on a weakly basis to improve the accuracy of the Retrieval-based Image caption.

[31] based on the work of generating the images descriptions using the Multimodal. They use encoder-decoder pipeline that learns. A multimodal is based on the embedding space of the images and text and a novel language modal that matches distributed representation of the text and images from our space. This multimodal consists of two trained modals which are as follows.

- 1 Neural Language Modal
- 2 Image-Text Modal

The work of the encoder is to rank the images and sentences. On the other hand the work of the decoder is generating the description for the image from scratch. They use LSTM to encode the sentence they match their results with Flickr8K and Flickr30K and without using the object detection. They get their best result using the convolution neural network with 19x layers. But they use three different methods to generate the descriptions of the images.

- Template-based method

- Composition based method
- Neural network based method

For the encoder and decoder method for ranking generation they use the following methods.

- Long short term memory RNNs
- Multimodal distributed representation.
- Log-bilinear NL models
- Multiplicative NL models
- Structure-content NL models

In [32] they use long-term recurrent convolution network (LRCNs). This is the class of the architecture that is used for visual recognition and description generation. It consists of the convolution layers and temporal recursion with long-range and the main benefit is that it is an end to end trainable. They train there modal for the specific video activity recognition and image caption generation. The LRCNs modal is both spatially and temporally deep and flexible enough to be applied for the vision-based task which can be based on sequential input and output. Their results constantly demonstrate learning sequential dynamics with a deep sequence modal. They use the deep neural network like (CNN) for capturing the features from the images and then they add another model LSTM which is used to generate the sequence of words based on the natural language. They combined the both CNN and RNN and under these they use LSTM to generate the description for the images and the videos. The whole system is combined is called LRCNs which contains the features of CNN and RNN and also sequence generator.

In [33] they use CNN modal instead of traditional RNN modal. They use CNN as image "encoder" first they train the CNN for the image classification task and then they use the last hidden layer of the network as an input to the RNN "decoder" that generate the sentence. They call this model natural image caption or NIC. It is a neural network which is fully trainable using the famous technique like stochastic gradient decent. Modal also combines the state of the art sub-networks that perform subtasks like vision and natural language processing. Using these sub modals, they took advantage of pre-training these modal on large datasets. The performance of their system compared to the state of the art models is very good. For example, on the Pascal dataset, NIC BLUE score 59% and the current state of the art model score is 25%, while human performance

reaches 69%. On Flickr30k they improve from 56% to 66% and on SBU from 19% to 28%.

In [34] they describe a new approach to the caption generation that tries to generate the caption using a form of attention with two variants these are

- A mechanism of hard attention
- A mechanism of soft attention

They generate the two attention based modal for the image caption generator under the common framework. A soft deterministic attention-based model which is trained through the back-propagation method. A hard attention-based model is trained by maximizing an approximate e variation lower bound or equivalently by REINFORCE. Datasets they use are Flickr8k [35], Flickr30k and the MS COCO dataset. This new attention based approach gives the state of the art performance on all three benchmark datasets. For evolution, they use BLEU and METEOR metric. They also present how the learned attention can be used into modal generation process and demonstrate that learned alignments correspond very well to the human intuition. This model is not very simple but the result of this model is satisfying and this system can be considerable for the problem solution. In [36] the author uses a different method for caption generation. This technique is different from the previous approaches used by the researcher. They purposed that the description can be represented by collections of nouns, verbs, and prepositional phrases. The object in the image is described as a Noun Phrase. The interaction between the object in the image is encoded both as a verb phrase or may be a preposition. Datasets Flickr30k dataset and COCO are used. In both datasets, every image has a five (or six) sentence descriptions. We can have 559,113 sentences when we combining both datasets. In this, they propose the simplest model that is able to infer different phrases from image samples. From the phrases predicted, their model is able to automatically generate sentences using a statistical language model. Their algorithm, despite being simpler than state-of-the-art models, achieves similar results on this task. Also, their model generates some new sentences which are not generally present in the dataset. They evaluate their model using these methods. They measure the quality of the generated sentences with BLEU score. Human agreement scores are computed by comparing the first ground-truth description against the four others.

In [37] author describe that they use a new method of embedding the visual and the language data. Past work is usually based embedding the whole image and language sentence into the working pipeline space their proposed modal is trained on the following parameters:

- **Learning and inference** they try to retrieve image on the basis of the given query sentence. They train the model on the set of N Images and N correspondence that describe their content. After the completion of training processes, they discarded the training dataset and evaluate the results of the modal on totally unseen dataset.
- **Fragment Embedding** is another variant they use in their work. Image is complex stricter and based on the different objects and the same is with the language sentence so they break it down the image and sentence into fragments and embed these fragments into the vector for validation.

Their modal has some limitations they take a simple phrase like “A cat is black and white” into multiple relations it fails to relate them together. On the other hand on the image side, it takes many persons in the picture as one person. But the overall results and using the state of the art RCNN modal they succeed to some extend in getting their results.

In [35] author focus on the work of associating images with the sentences drawn from the big predefine pool of the images descriptions. These descriptions are not collected from the internet but written by the people who were asked to describe the images descriptions. They provide the alternate method for describing the description for the image which is best suitable to that image. They use a rank system rather than generating the description for the image. The new Rank system works on the basis of the nearest-neighbor search for the image description. The representation in this paper is very simple. They only really on the three different kinds of low-level pixels based perceptual feature that captures color, texture, and shape in the form of SIFT descriptor. They use two different times of kernels one is histogram kernel and other is the pyramid kernel. In both cases, they compute separate kernel for each of three types of images feature and average their results. They draw similarities between these kernels. They are String Kernels with Lexical Similarities, The Lin Similarity Kernel, and Distributional Similarity.

In [38] the author proposed a new query expansion approach which is used in automatic image captioning. The main idea is to translate the visual query into distributed semantics. It is generated by the average of sentence vectors that are generated from the captions of the visual images that are similar to the input images. In this paper, they used three image captioning standard datasets and shows that their technique is better and give more accurate results. Automatic image captioning is very popular in computer vision and language processing. The data-driven method, automatic metrics and subjective evaluation are the techniques which are discussed and compared in this research paper.

The first approach generates novel captions from images directly. In this approach, computer vision techniques like object detection and classification use their outputs to extract the visual contents of the input image and generate captions. These studies combine the convolutional neural network with a recurrent neural network to generate a description for images. The second approach used joint representations of images and captions. They employ machine learning techniques to form common embedding space for visual and textual data and perform image-sentence in that intermediate space to find the most proper captions for a given image. The third technique follows a data-driven approach and treats image captioning as a caption transfer problem.

In [39] the author presents a model that generates natural language descriptions of images. Their proposed model is based on a novel combination of Convolution Neural Networks over image regions and bidirectional Recurrent Neural Network over sentences. After that in this paper, the author describes the Multimodal Recurrent Neural Network architecture that uses the inferred alignments to learn to generate novel descriptions of images. Here they use datasets of Flickr8K, Flickr30K, and MSCOCO. The previous work was in visual recognition and that focused on labeling images with visual categories. They evaluated the output of Multimodal Recurrent Neural Network architecture on both full frame and region-level experiments and shows that in both cases Multimodal RNN output forms the retrieval baselines.

Table 3. 2 Parameterized Analysis of Different Approches which uses BLEU scores witn Flickr dataset

	Approach	Datasets	Measures	Augmentation
Gong et al. (2014)	MultRetrieval	SBU1M, Flickr30K	R@k	X
Kiros et al. (2015)	MultRetrieval	Flickr8K/30K	R@k	X
Donahue et al. (2015)	MultRetrieval	Flickr30K, COCO	Human, BLEU, mRank, R@k	X
Vinyals et al. (2015)	MultRetrieval	Pascal1K, SBU1M, Flickr8K/30K	BLEU, Meteor, CIDEr, mRank, R@k	X
Xu et al. (2015)	MultRetrieval	Flickr8K/30K, COCO	BLEU, Meteor	X
Hodosh et al. (2013)	MultRetrieval	Pascal1K, Flickr8K	Human, BLEU, ROUGE, mRank, R@k	X
Lebret et al. (2015)	MultRetrieval	Flickr30K, COCO	BLEU, R@k	X
Karpathy et al. (2014)	MultRetrieval	Flickr8K/30K, COCO	BLEU, Meteor, CIDEr	X
Yagcioglu et al. (2015)	VisRetrieval	Flickr8K/30K, COCO	Human, BLEU, Meteor, CIDEr	X
Karpathy and Fei-Fei (2015)	MultRetrieval	Flickr8K/30K, COCO	BLEU, Meteor, CIDEr, mRank, R@k	X

4 Methodology

The aim of this research is two-fold. First is to get better the BLEU scores and the second is to bring the stability of prediction models.

4.1 Parameter Identification and Optimization

For better BLEU score, we used the pre-trained VGG16 model introduced with Anaconda. It is trained on 1.2 million images of ImageNet dataset. It is designed to accept input images of size 224 x 224 x 3. We used the model and remove the last layer which is responsible for the final classification of input to 1000 categories of the dataset. These last layer is *Dense Layer*. We then get the features from VGG16 and then take these features into out model as an input to train out model with different parameters. Following are the parameters that we considered:

- Learning Rate
- Decay
- Model Optimizer
- Number of Epochs

4.2 Proposed Network Design

As discussed earlier, the idea of skip last layer in VGG16 pre-trained model is just to utilize that for feature extraction. After feature extraction we feed these features to train our model. The uniqueness of this research is the implementation of data augmentation.

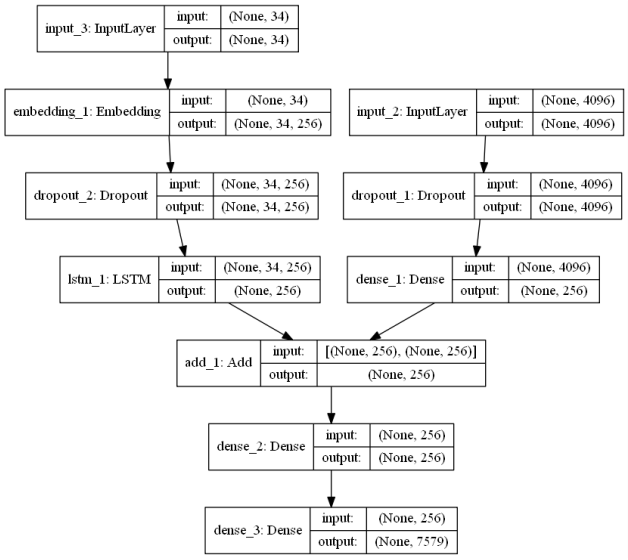

Figure 4. 1 Proposed Model

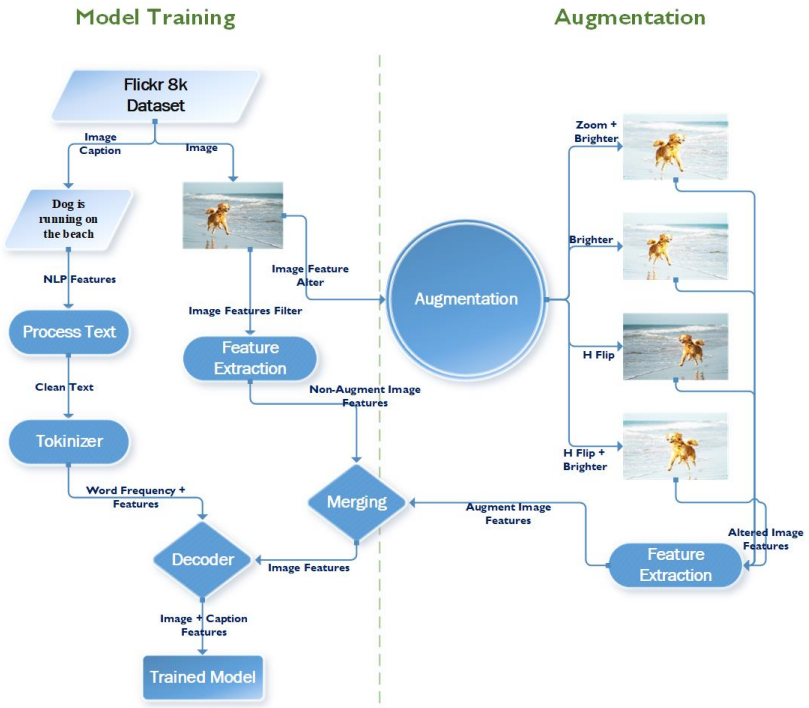

Figure 4. 2 Full Working Diagram

In an attempt to improve BLEU Score, we experimented with different layer orders and also with different types of layers. Specifically, we were interested in changing the Optimizer. Optimizers are the algorithms which may increase or decrease the objective function (error function) which is dependent on models learnable parameters. Optimizer play an important role to minimize the loss during training process.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

Figure 4. 3 VGG16 Model

4.3 Implementation

Now we present a brief summary of the benchmark dataset and tools utilized for this research.

4.3.1 Datasets

The convention in the deep learning research community has been to ImageNet-1K dataset as a benchmark for comparison. ImageNet-1K dataset consists of 1.2 million images across 1000 categories. In order to train a CNN on such big data, it requires a few high-end GPUs hooked up to high-end systems and then the training should complete in about a week [16]. Due to unavailability of such resources, we perform our experiments on smaller benchmark dataset and the target is to show a comparison between results achieved by original dataset and our proposed method (augmented dataset) under same circumstances. We used the following benchmark dataset for this research:

Flickr8k

Flickr8k dataset has become a standard for sentence-based image captionization. This research is done on flickr8k dataset which contains 8k images. Every single image has 5 different captions which describe it clearly. These images belong to 6 different groups from Flickr. The main thing while choosing these images is that the image should not contain any well-known person or place.

4.3.2 Tools Utilized

In light of its extensive application for both research and development on deep learning, Anaconda (Python & Tensorflow) is utilized all through this undertaking and every one of the outcomes are in this manner got from it. Due to lack of resources we used a dataset with less number of images and executed the experiments for limited number of epochs. That is why our results should not be compared directly with the results published by [33]. For augmentation, we develop our own code because this work can never be happening before and we couldn’t find such a code. For model training and testing, we use code from many sources.

A machine with a core-i5 processor and a single 2GB NVidia GPU is used for all experiments in this research.

5 Experiment & Results

This section presents our experimentation outcomes by applying data augmentation on Dataset of Flickr8k. We implemented the original VGG16 and used it as our base reference model to extract features, but we didn't classify the output of the VGG16 model to 1000 categories. We use transfer learning concept and remove the prediction layer in VGG16 and save the raw output. Then we implemented our proposed model on raw output of VGG16 model and we present the impact of data augmentation obtained on the mentioned dataset using the base and proposed model.

Flickr8k

We use adamax optimizer with learning rate 1e-3 with decay factor 1e-6. These models are trained with a minibatch size of 32 on one Nvidia GPU. The small batch size is selected due to GPU memory constraints. We have total 8k images on which we have 6k training images, 1k testing images and 1k images are used for validation purpose. We follow the simple data augmentation with zoom effect, horizontal effect and brightness effect to produce augmented data.

BLEU Score Results

5.1 BLEU-1

The below picture shows the result of our experiment. The blue line show us the BLEU-1 percentage in augmented data and the yellow one shows the un-augmented data. It is clearly seen that the augmented data creates more stable models to computer BLEU-1 score

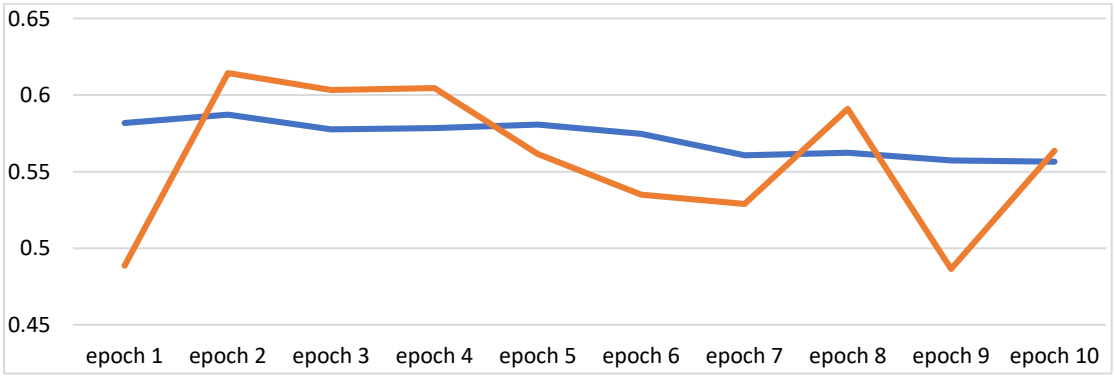

Figure 5. 1 Calculated Results of BLEU-1 after Experiment

5.2 BLEU-2

The below picture shows the result of our experiment. The blue line show us the BLEU-2 (which is (0.50, 0.50, 0, 0)) percentage in augmented data and the yellow one shows the un-augmented data. It is clearly seen that the augmented data creates more stable models to computer BLEU-2 score as well as when we see the BLEU-1 figure, we came to know that the BLEU-2 score of augmentation looks more steady than BLEU-1

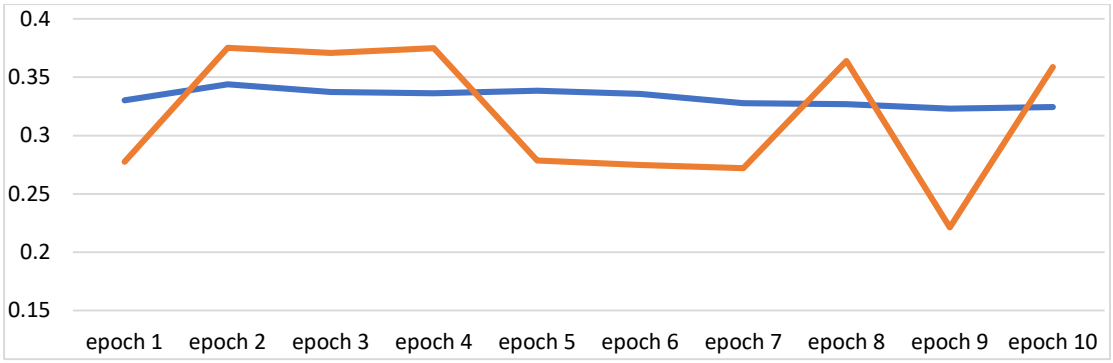

Figure 5. 2 Calculated Results of BLEU-2 after Experiment

5.3 BLEU-3

The below picture shows the result of our experiment. The blue line show us the BLEU-3 (which is (0.33, 0.33, 0.33, 0)) percentage in augmented data and the yellow one shows the un-augmented data. It is clearly seen that the augmented data creates more stable models to computer BLEU-3 score

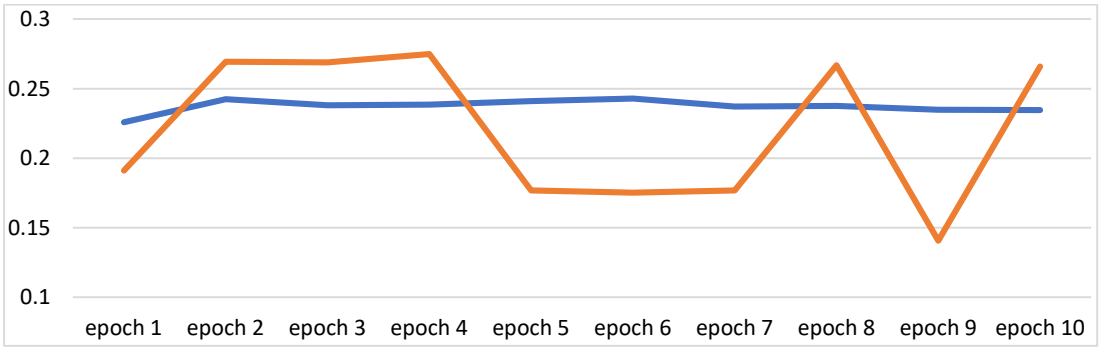

Figure 5. 3 Calculated Results of BLEU-3 after Experiment

5.4 BLEU-4

The below picture shows the result of our experiment. The blue line show us the BLEU-4 (which is (0.25, 0.25, 0.25, 0.25)) percentage in augmented data and the yellow one shows the un-augmented data. It is clearly seen that the augmented data creates more stable models to computer BLEU-4 score

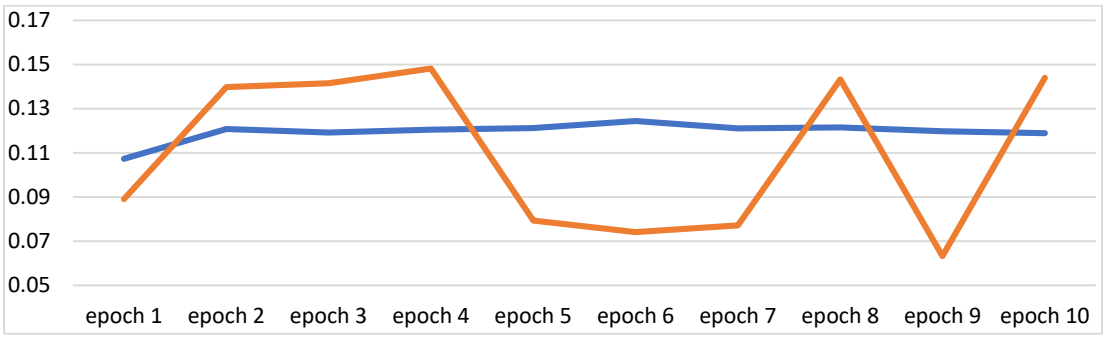

Figure 5. 4 Calculated Results of BLEU-4 after Experiment

Image Captioning Results

Here are some results generated by our model.

Dog is running on the beach

Man is sitting on the water

Man in black shirt is sitting on the street

White dog is running through the grass

Figure 5. 5 Results of Caption Generation

6 Conclusion

Image classification and detection are the core tasks in computer vision systems which lead us to the all other important tasks like Image Captionization. This research introduces an important factor through which we can generate more stable versions of our predictions models. The proposed method outperforms the other approaches in experiments carried out at benchmark subsets. The introduced method reports stable prediction model generation. Although we have a limited resource of GPU and RAM memory, out proposed method performs much better with respect to the hardware. We believe that my proposed method can pass the results produced by others if the high capacity of RAM and GPU memory available.

Limitations

One limitation that we noticed in our proposed method is that it takes times (directally propotoipnal to the size of augmented data) more training time as compared to the original dataset, because dataset is increased due to data augmentation.

Future Work

The future directions are to reduce the model over fitting and to make it optimized for mobile devices and bring benefits of this technique to develop intelligent mobile apps as well as try this method on other datasets.

References

1. Szeliski, R., *Computer vision: algorithms and applications*. 2010: Springer Science & Business Media.
2. Kanade, T., *Three-dimensional machine vision*. Vol. 21. 2012: Springer Science & Business Media.
3. Sebe, N., *Machine learning in computer vision*. Vol. 29. 2005: Springer Science & Business Media.
4. Harris, C. and M. Stephens. *A combined corner and edge detector*. in *Alvey vision conference*. 1988. Citeseer.
5. Bay, H., T. Tuytelaars, and L. Van Gool. *Surf: Speeded up robust features*. in *European conference on computer vision*. 2006. Springer.
6. Dalal, N. and B. Triggs. *Histograms of oriented gradients for human detection*. in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. 2005. IEEE.
7. Lowe, D.G., *Distinctive image features from scale-invariant keypoints.* International journal of computer vision, 2004. **60**(2): p. 91-110.
8. Leutenegger, S., M. Chli, and R.Y. Siegwart. *BRISK: Binary robust invariant scalable keypoints*. in *Computer Vision (ICCV), 2011 IEEE International Conference on*. 2011. IEEE.
9. Alahi, A., R. Ortiz, and P. Vandergheynst. *Freak: Fast retina keypoint*. in *Computer vision and pattern recognition (CVPR), 2012 IEEE conference on*. 2012. Ieee.
10. Alom, M.Z., et al., *The history began from AlexNet: a comprehensive survey on deep learning approaches.* 2018.
11. Dechter, R., *Learning while searching in constraint-satisfaction problems*. 1986: University of California, Computer Science Department, Cognitive Systems Laboratory.
12. Aizenberg, I., N.N. Aizenberg, and J.P. Vandewalle, *Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications*. 2013: Springer Science & Business Media.
13. LeCun, Y., et al., *Backpropagation applied to handwritten zip code recognition.* Neural computation, 1989. **1**(4): p. 541-551.

14. Hinton, G.E., et al., *The" wake-sleep" algorithm for unsupervised neural networks.* Science, 1995. **268**(5214): p. 1158-1161.
15. Hochreiter, S., et al., *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*. 2001, A field guide to dynamical recurrent neural networks. IEEE Press.
16. Szegedy, C., et al. *Going deeper with convolutions*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
17. He, K., et al. *Deep residual learning for image recognition*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.
18. Zurada, J.M., *Introduction to artificial neural systems*. Vol. 8. 1992: West publishing company St. Paul.
19. He, K., et al. *Delving deep into rectifiers: Surpassing human-level performance on imagenet classification*. in *Proceedings of the IEEE international conference on computer vision*. 2015.
20. Glorot, X. and Y. Bengio. *Understanding the difficulty of training deep feedforward neural networks*. in *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 2010.
21. LeCun, Y., et al., *Gradient-based learning applied to document recognition.* Proceedings of the IEEE, 1998. **86**(11): p. 2278-2324.
22. Krizhevsky, A., I. Sutskever, and G.E. Hinton. *Imagenet classification with deep convolutional neural networks*. in *Advances in neural information processing systems*. 2012.
23. Zeiler, M.D. and R. Fergus. *Visualizing and understanding convolutional networks*. in *European conference on computer vision*. 2014. Springer.
24. Simonyan, K. and A. Zisserman, *Very deep convolutional networks for large-scale image recognition.* arXiv preprint arXiv:1409.1556, 2014.
25. Huang, G., et al. *Densely connected convolutional networks*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
26. Papineni, K., et al. *BLEU: a method for automatic evaluation of machine translation*. in *Proceedings of the 40th annual meeting on association for computational linguistics*. 2002. Association for Computational Linguistics.

27. Zagoruyko, S. and N. Komodakis, *Wide residual networks.* arXiv preprint arXiv:1605.07146, 2016.
28. Srivastava, R.K., K. Greff, and J. Schmidhuber, *Highway networks.* arXiv preprint arXiv:1505.00387, 2015.
29. Xie, S., et al. *Aggregated residual transformations for deep neural networks*. in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017. IEEE.
30. Gong, Y., et al. *Improving image-sentence embeddings using large weakly annotated photo collections*. in *European conference on computer vision*. 2014. Springer.
31. Kiros, R., R. Salakhutdinov, and R.S.J.a.p.a. Zemel, *Unifying visual-semantic embeddings with multimodal neural language models.* 2014.
32. Donahue, J., et al. *Long-term recurrent convolutional networks for visual recognition and description*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
33. Vinyals, O., et al. *Show and tell: A neural image caption generator*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.
34. Xu, K., et al. *Show, attend and tell: Neural image caption generation with visual attention*. in *International conference on machine learning*. 2015.
35. Hodosh, M., P. Young, and J.J.J.o.A.I.R. Hockenmaier, *Framing image description as a ranking task: Data, models and evaluation metrics.* 2013. **47**: p. 853-899.
36. Lebret, R., P.O. Pinheiro, and R.J.a.p.a. Collobert, *Phrase-based image captioning.* 2015.
37. Karpathy, A., A. Joulin, and L.F. Fei-Fei. *Deep fragment embeddings for bidirectional image sentence mapping*. in *Advances in neural information processing systems*. 2014.
38. Yagcioglu, S., et al. *A distributed representation based query expansion approach for image captioning*. in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. 2015.
39. Karpathy, A. and L. Fei-Fei. *Deep visual-semantic alignments for generating image descriptions*. in *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.