International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

# Alexnet Technology

**Naseela K K[1], Praveena K P[2], Shanifa P S[3] , Sruthy K S[4]**

[1,2,3,4] *Department of Computer Science*
[1,2,3,4] *College of Applied Science Chelakkara*

---

**ABSTRACT:** *Alexnet is the name given to a Convolutional Neural Network Architecture. Alexnet solves the problem of image classification with subset of ImageNet dataset with roughly 1.2 million training images, 50,000 validation images, and 150,000 testing images. The input is an image of one of 1000 different classes and output is a vector of 1000 numbers. The neural network, which has 60 million parameters and 650,000 neurons, consists of 5 convolutional layers, some of which are followed by max-pooling layers, and 3 fully connected layers. The unique advantage of AlexNet is the direct image input to the classification model. The results of AlexNet show that a large, deep convolutional neural network. It is capable of achieving record-breaking results on a highly challenging dataset using purely supervised learning. An important feature of the AlexNet is the use of ReLU(Rectified Linear Unit) Nonlinearity and a very efficient GPU implementation of the convolution operations. Alexnet Architecture employed regularization methods such as Dropout and Data Augmentation to reduce overfitting in the fully connected layers.*

***Key Words***: Maxpooling , Relu, dropout, data augmentation.

## 1.INTRODUCTION

A few years back, we still used small datasets like CIFAR and NORB consisting of tens of thousands of images. These datasets were sufficient for machine learning models to learn basic recognition tasks. However, real life is never simple and has many more variables than are captured in these small datasets. Convolutional Neural Networks (CNNs) had always been the go-to model for object recognition — they're strong models that are easy to control and even easier to train. They don't experience overfitting at any alarming scales when being used on millions of images. Their performance is almost identical to standard feedforward neural networks of the same size. The only problem: they're hard to apply to high resolution images. At the ImageNet scale, there needed to be an innovation that would be optimized for GPUs and cut down on training times while improving performance. The recent availability of large datasets like ImageNet, which consist of hundreds of thousands to millions of labeled images, have pushed the need for an extremely capable deep learning model called Alexnet. AlexNet was a deep neural network that was developed by Alex Krizhevsky and others in 2012. It was designed to classify images for the ImageNet [3] ILSVRC-2012 competition, where it achieved the state of the art results. Deep Learning based Convolutional Neural Networks (CNN) have given exceptional outcomes in Computer Vision Domain, and AlexNet has proven to be the leading architecture in image classification and object detection problems. AlexNet was one of the first CNN models implemented on GPUs that truly connected the growing computation power at that time with deep learning. They created a deeper and more complex CNN model that had kernels of various sizes (like 11*11, 5*5, and 3*3) and significantly more number of channels than LeNet. They also started using ReLU activations instead of sigmoid or tanh, which helped train better models. AlexNet not only won the Imagenet classification challenge in 2012, but beat the runner up with margins that suddenly made non-neural models almost obsolete.

## 2. LITERATURE REVIEW

Image classification has been a fundamental research area in pattern recognition and computer vision for several years. Earlier approaches mainly depended on traditional machine learning techniques, where feature extraction was performed manually using methods such as edge detection, texture analysis, and color-based descriptors. These features were then classified using algorithms like Support Vector Machines (SVM) and k-Nearest Neighbour (k-NN). Although these techniques provided reasonable results, their performance was often limited due to the complexity of designing effective feature representations.

A major advancement in this domain was achieved with the introduction of the AlexNet model by Alex Krizhevsky et al. (2012). This work demonstrated the

effectiveness of deep convolutional neural networks in automatically learning features from raw images. By utilizing multiple convolutional layers, non-linear activation functions, and GPU-based training, the model significantly improved classification accuracy on large-scale datasets and outperformed conventional approaches.

Subsequent research focused on comparing AlexNet with other CNN architectures such as LeNet, VGG, GoogleNet, and ResNet. These studies highlighted that increasing network depth and improving training strategies led to better performance. It was observed that CNNs are capable of learning hierarchical representations, where initial layers detect simple patterns and deeper layers capture more complex image structures. AlexNet served as a foundation for the development of these advanced architectures.

AlexNet is a deep convolutional neural network architecture designed for large-scale image classification tasks using datasets such as ImageNet, which contains millions of labeled images across multiple categories.

## 3. METHODOLOGY

The proposed system for image classification is based on the AlexNet architecture, leverages deep convolutional layers, efficient training techniques, and regularization strategies to achieve high accuracy in image classification tasks, demonstrating the effectiveness of AlexNet in handling large and complex datasets.

## II. ALEXNET

AlexNet is the name given to a Convolutional Neural Network Architecture that won the LSVRC competition. LSVRC (Large Scale Visual Recognition Challenge) is a competition where research teams evaluate their algorithms on a huge dataset of labeled images namely ImageNet and compete to achieve higher accuracy on several visual recognition tasks. AlexNet was the pioneer in Convolutional Neural Network and open the whole new research era. AlexNet implementation is very easy after the releasing of so many deep learning libraries such as PyTorch, TensorFlow, Kera's etc. The input to AlexNet is an RGB image of size 256×256. This means all images in the training set and all test images need to be of size 256×256. If the input image is not 256×256, it needs to be converted to 256×256 before using it for training the network. To achieve this, the smaller dimension is resized to 256 and then the resulting image is cropped to obtain a 256×256 image. The figure below shows an example:


If the input image is grayscale, it is converted to an RGB image by replicating the single channel to obtain a 3-channel RGB image. Random crops of size 227×227 were generated from inside the 256×256 images to feed the first layer of AlexNet.

## III.DATASET

ImageNet is a dataset of over 15 million labeled high-resolution images belonging to roughly 22,000categories. The images were collected from the web and labeled by human labelers using Amazon's Mechanical Turk crowd-sourcing tool. Starting in 2010, as part of the Pascal Visual Object Challenge, an annual competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) has been held. ILSVRC uses a subset of ImageNet with roughly 1000 images in each of1000 categories. In all, there are roughly 1.2 million training images, 50,000 validation images, and150,000 testing images.

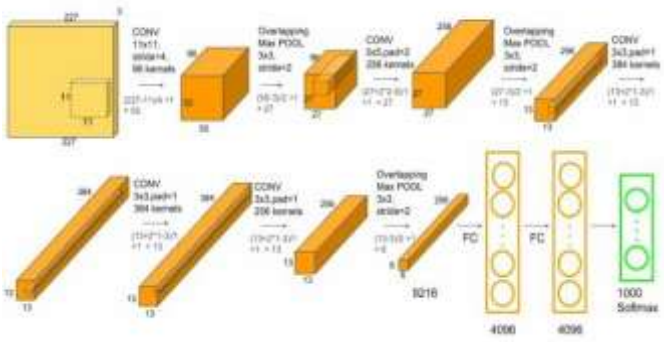
ImageNet consists of variable-resolution images, while the system requires a constant input dimensionality. Therefore, it down-sampled the images to a fixed resolution of 256 × 256. Given a rectangular image, First it rescaled the image such that the shorter side was of length 256, and then cropped out the central 256×256 patch from the resulting image. It did not pre-process the images in any other way, except for subtracting the mean activity over the training set from each pixel. So, it trained the network on the (centered) raw RGB values of the pixels.

## IV. ARCHITECTURE

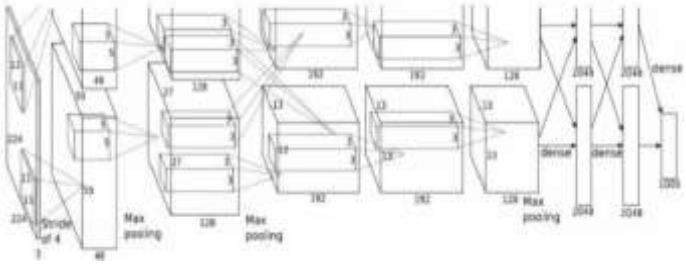
AlexNet was much larger than previous CNNs used for computer vision tasks (e.g., Yann LeCun's LeNet paper in 1998). It has 60 million parameters and 650,000 neurons and took five to six days to train on two GTX 580 3GB GPUs.The Alexnet has eight layers with learnable parameters. The model consists of five convolutional layers

with a combination of max pooling followed by 3 fully connected layers. They use Relu activation in each of these layers except the output layer. They found out that using the Relu as an activation function accelerated the speed of the training process by almost six times. They also used the Dropout layers, that prevented their model from overfitting. Further, the model is trained on the ImageNet dataset. The ImageNet dataset has almost 14 million images across a thousand classes.

AlexNet architecture is shown below:



Let's see the architectural details using a network diagram:



Alexnet is a deep architecture that take input as RGB images of size 227X227X3.

Convolution and Maxpooling layers

| Layer | # filters / neurons | Filter size | Stride | Padding | Size of feature map | Activation function |
|---|---|---|---|---|---|---|
| Input | - | - | - | - | 227 x 227 x 3 | - |
| Conv 1 | 96 | 11 x 11 | 4 | - | 55 x 55 x 96 | ReLU |
| Max Pool 1 | - | 3 x 3 | 2 | - | 27 x 27 x 96 | - |
| Conv 2 | 256 | 5 x 5 | 1 | 2 | 27 x 27 x 256 | ReLU |
| Max Pool 2 | - | 3 x 3 | 2 | - | 13 x 13 x 256 | - |
| Conv 3 | 384 | 3 x 3 | 1 | 1 | 13 x 13 x 384 | ReLU |
| Conv 4 | 384 | 3 x 3 | 1 | 1 | 13 x 13 x 384 | ReLU |
| Conv 5 | 256 | 3 x 3 | 1 | 1 | 13 x 13 x 256 | ReLU |
| Max Pool 3 | - | 3 x 3 | 2 | - | 6 x 6 x 256 | - |
| Dropout 1 | rate = 0.5 | - | - | - | 6 x 6 x 256 | - |

In Alexnet Architecture we apply the first convolution layer with 96 filters of size 11X11 with stride 4. The activation function used in this layer is Relu. The output feature map is 55X55X96.

To calculate the output size of a convolution layer:

output= ((Input-filter size)/ stride)+1

Also, the number of filters becomes the channel in the output feature map.

Next, we have the first Maxpooling layer, of size 3X3 and stride 2. Then we get the resulting feature map with the size 27X27X96.After this, we apply the second convolution operation. This time the filter size is reduced to 5X5 and we have 256 such filters. The stride is 1 and padding 2. The activation function used is again Relu. Now the output size we get is 27X27X256.Again we applied a max-pooling layer of size 3X3 with stride 2. The resulting feature map is of shape 13X13X256.

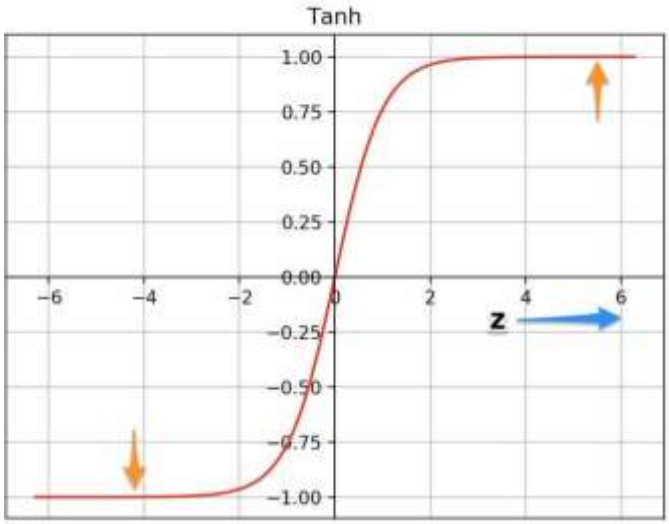
Now we apply the third convolution operation with 384 filters of size 3X3 stride 1 and also padding 1. Again, the activation function used is Relu. The output feature map is of shape 13X13X384.Then we have the fourth convolution operation with 384 filters of size 3X3. The stride along with the padding is 1. On top of that activation function used is Relu. Now the output size remains unchanged i.e., 13X13X384.After this, we have the final convolution layer of size 3X3 with 256 such filters. The stride and padding are set to one also the activation function is Relu. The resulting feature map is of shape 13X13X256.Here,the number of filters is increasing as we are going deeper. Hence it is extracting more features as we move deeper into the architecture. Also, the filter size is reducing, which means the initial filter was larger and as we go ahead the filter size is decreasing, resulting in a decrease in the feature map shape.

Next, we apply the third max-pooling layer of size 3X3 and stride 2. Resulting in the feature map of the shape 6X6X256.

Fully Connected and Dropout Layers:

| Layer | # filters / neurons | Filter size | Stride | Padding | Size of feature map | Activation function |
|---|---|---|---|---|---|---|
| - | - | - | - | - | - | - |
| - | - | - | - | - | - | - |
| - | - | - | - | - | - | - |
| Dropout 1 | rate = 0.5 | - | - | - | 6 x 6 x 256 | - |
| Fully Connected 1 | - | - | - | - | 4096 | ReLU |
| Dropout 2 | rate = 0.5 | - | - | - | 4096 | - |
| Fully Connected 2 | - | - | - | - | 4096 | ReLU |
| Fully Connected 3 | - | - | - | - | 1000 | Softmax |

After this, we have our first dropout layer. The drop-out




rate is set to be 0.5.

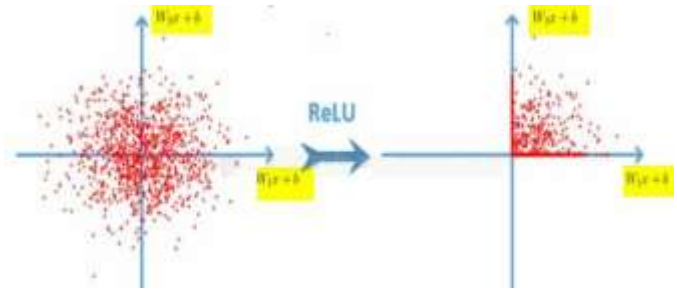
Then we have the first fully connected layer with a Relu activation function. The size of the output is 4096. Next comes another dropout layer with the dropout rate fixed at 0.5.This followed by a second fully connected layer with 4096 neurons and Relu activation.

Finally, we have the last fully connected layer or output layer with 1000 neurons as we have 10000 classes in the data set. The activation function used at this layer is Softmax.

## V. WHY DOES ALEXNET ACHIEVE BETTER RESULTS?

Some approaches makes alexnet special are :

### 5.1. RELU ACTIVATION FUNCTION IS USED:




An important feature of the AlexNet is the use of ReLU(Rectified Linear Unit) Nonlinearity's or sigmoid activation functions used to be the usual way to train a neural network model. AlexNet showed that using ReLU nonlinearity, deep CNNs could be trained much faster than using the saturating activation functions like tanh or sigmoid tested on the CIFAR-10 dataset.

Let's see why it trains faster with the ReLUs.

The ReLU function is given by

f(x) = max(0,x)

plots of the two functions —

1. tanh
2. ReLU.

Tanh

The tanh function saturates at very high or very low values of z. In these regions, the slope of the function goes very close to zero. This can slow down gradient descent.

Relu

The ReLU function's slope is not close to zero for higher positive values of z. This helps the optimization to converge faster. For negative values of z, the slope is still zero, but most of the neurons in a neural network usually end up having positive values.

### 5.2 MULTIPLE GPUS

Back in the day, GPUs were still rolling around with 3 gigabytes of memory (nowadays those kinds of memory would be rookie numbers). This was especially bad because the training set had 1.2 million images. AlexNet allows for multi-GPU training by putting half of the model's neurons on one GPU and the other half on another GPU. Not only does this mean that a bigger model can be trained, but it also cuts down on the training time. A single GTX 580 GPU has only 3GB of memory, which limits the maximum size of the networks that can be trained on it. It turns out that 1.2 million training examples are enough to train networks which are too big to fit on one GPU. Therefore, we spread the net across two GPUs. Current GPUs are particularly well-suited to cross-GPU parallelization, as they are able to read from and write to one another's memory directly, without going through host machine memory. The parallelization scheme that we employ essentially puts half of the kernels (or neurons) on each GPU, with one additional trick: the GPUs communicate only in certain layers. This means that, for example, the kernels of layer 3 take input from all kernel maps in layer 2. However, kernels in layer 4 take input only from those kernel maps in layer 3 which reside on the

same GPU. Choosing the pattern of connectivity is a problem for cross-validation, but this allows us to precisely tune the amount of communication until it is an acceptable fraction of the amount of computation. The resultant architecture is somewhat similar to that of the "columnar" CNN employed by Ciresan et al. [5], except that our columns are not independent. This scheme reduces our top-1 and top-5 error rates by 1.7% and 1.2%, respectively, as compared with a net with half as many kernels in each convolutional layer trained on one GPU. The two-GPU net takes slightly less time to train than the one-GPU net.

### 5.3 LOCAL RESPONSE NORMALIZATION

ReLUs have the desirable property that they do not require input normalization to prevent them from saturating. If at least some training examples produce a positive input to a ReLU, learning will happen in that neuron. However, we still find that the following local normalization scheme aids generalization. Denoting by a i x,y the activity of a neuron computed by applying kernel i at position (x, y) and then applying the ReLU nonlinearity, the response-normalized activity b i x,y is

$$b^i_{x,y} = a^i_{x,y} / \left( k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta}$$

given by the expression.

where the sum runs over n "adjacent" kernel maps at the same spatial position, and N is the total number of kernels in the layer. The ordering of the kernel maps is of course arbitrary and determined before training begins. This sort of response normalization implements a form of lateral inhibition inspired by the type found in real neurons, creating competition for big activities amongst neuron outputs computed using different kernels. The constants k, n, α, and β are hyper-parameters whose values are determined using a validation set; we used k = 2, n = 5, α = 10−4, and β = 0.75. We applied this normalization after applying the ReLU nonlinearity in certain layers.

This scheme bears some resemblance to the local contrast normalization scheme of Jarrett et al. [11],but ours would be more correctly termed "brightness normalization", since we do not subtract the mean activity. Response normalization reduces our top-1 and top-5 error rates by 1.4% and 1.2%, respectively. We also verified the effectiveness of this scheme on the CIFAR-10 dataset: a four-layer CNN achieved a 13% test error rate without normalization and 11% with normalization.

### 5.4 OVERLAPPING POOLING

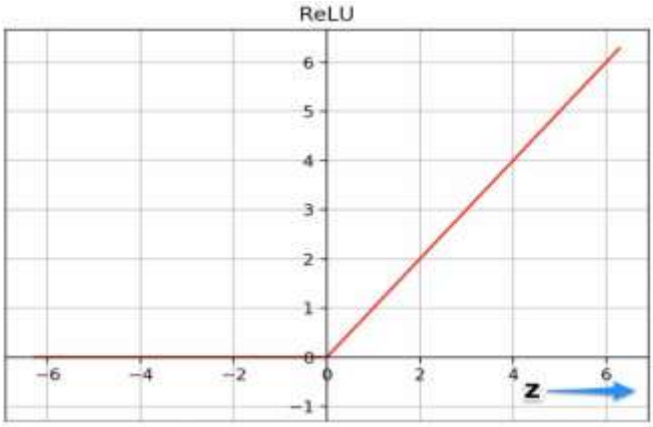
CNNs traditionally "pool" outputs of neighboring groups of neurons with no overlapping. However, when there introduced overlap, it makes a reduction in error by about 0.5% and found that models with overlapping pooling generally find it harder to overfit.

## VI.THE OVERFITTING PROBLEM

AlexNet had 60 million parameters, a major issue in terms of overfitting. Two methods were employed to reduce overfitting:

### 6.1 Data Augmentation

The easiest and most common method to reduce overfitting on image data is to artificially enlarge the dataset using label-preserving transformations. We employ two distinct forms of data augmentation, both of which allow transformed images to be produced from the

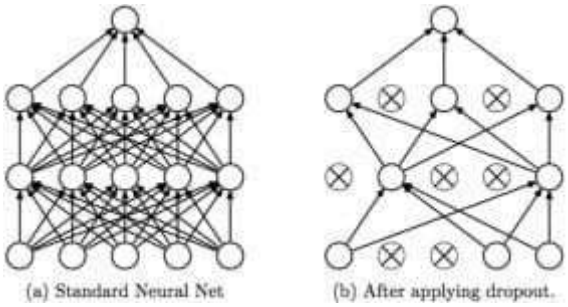



original images with very little computation, so the transformed images do not need to be stored on disk. The first form of data augmentation consists of generating image translations and horizontal reflection. This is done by extracting random 224 × 224 patches (and their horizontal reflections) from the256×256 images and training our network on these extracted patches. This increases the size of training set by a factor of 2048, though the resulting training examples are, of course, highly inter-dependent. Without this scheme, the network suffers from substantial overfitting, which would have forced to use much smaller networks. At test time, the network makes a prediction by extracting five 224 × 224 patches (the four corner patches and the center patch) as well as their horizontal reflections (hence ten patches in all), and averaging the predictions made by the network's softmax layer on the ten patches. The second form of data augmentation consists of altering the intensities of the RGB channels in training images. Specifically, we

perform PCA on the set of RGB pixel values throughout the ImageNet training set.

### 6.2 Dropout

The second technique that AlexNet used to avoid overfitting was a dropout. It consists of setting to zero the output of each hidden neuron with a probability of 0.5. The neurons which are "dropped out" in this way do not contribute to the forward pass and do not participate in backpropagation. So, every time an input is presented, the neural network samples a different architecture. This technique consists of turning off neurons with a predetermined probability. This means that every iteration, the neurons "turned off" do not contribute to the forward pass and do not participate in backpropagation.



(a) Standard Neural Net (b) After applying dropout.

## VII. COMPARISON WITH OTHER CNN MODELS

### 7. 1. LeNet-5 (1998)

LeNet-5, a pioneering 7-level convolutional network by LeCun et al in 1998, that classifies digits, was applied by several banks to recognise hand-written numbers on checks (cheques) digitized in 32x32 pixel greyscale input images. The ability to process higher resolution images requires larger and more convolutional layers, so this technique is constrained by the availability of computing resources.

### 7.2. AlexNet (2012)

In 2012, AlexNet significantly outperformed all the prior competitors and won the challenge by reducing the top-5 error from 26% to 15.3%. The second place top-5 error rate, which was not a CNN variation, was around 26.2%.

### 7.3 ZFNet(2013)

ILSVRC 2013 winner was also a CNN which became known as ZFNet. It achieved a top-5 error rate of 14.8% which is now already half of the prior mentioned non-neural error rate. It was mostly an achievement by tweaking the hyper-parameters of AlexNet while maintaining the same structure.

### 7.4 GoogLeNet/Inception(2014)

The winner of the ILSVRC 2014 competition was GoogLeNet(a.k.a. Inception V1) from Google. It achieved a top-5 error rate of 6.67%! This was very close to human level performance which the organisers of the challenge were now forced to evaluate. As it turns out, this was actually rather hard to do and required some human training in order to beat GoogLeNets accuracy. After a few days of training, the human expert (Andrej Karpathy) was able to achieve a top-5 error rate of 5.1%(single model) and 3.6%(ensemble).

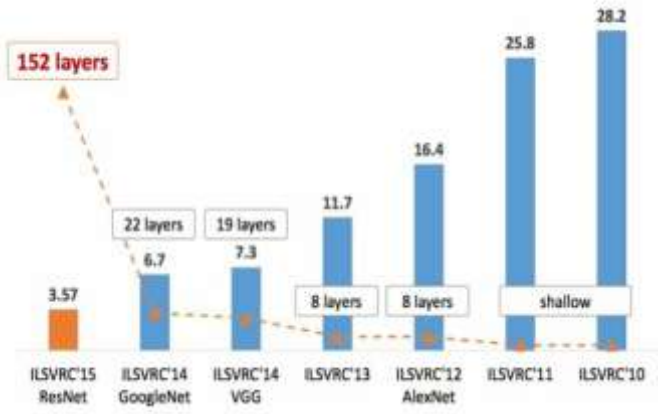
### 7.5 VGGNet (2014)

The runner-up at the ILSVRC 2014 competition is dubbed VGGNet by the community and was developed by Simonyan and Zisserman. VGGNet consists of 16 convolutional layers and is very appealing because of its very uniform architecture. Similar to AlexNet, only 3x3 convolutions, but lots of filters. Trained on 4 GPUs for 2–3 weeks. It is currently the most preferred choice in the community for extracting features from images.

### 7.6 ResNet(2015)

At last, at the ILSVRC 2015, the so-called Residual Neural Network (ResNet) by Kaiming He et al introduced anovel architecture with "skip connections" and features heavy batch normalization. Such skip connections are also known as gated units or gated recurrent units and have a strong similarity to recent successful elements applied in RNNs.

| Year | CNN | Developed by | Place | Top-5 error rate | No. of parameters |
|---|---|---|---|---|---|
| 1998 | LeNet(8) | Yann LeCun et al | | | 60 thousand |
| 2012 | AlexNet(7) | Alex Krizhevsky, Geoffrey Hinton, Ilya Sutskever | 1st | 15.3% | 60 million |
| 2013 | ZFNet() | Matthew Zeiler and Rob Fergus | 1st | 14.8% | |
| 2014 | GoogLeNet(19) | Google | 1st | 6.67% | 4 million |
| 2014 | VGG Net(16) | Simonyan, Zisserman | 2nd | 7.3% | 138 million |
| 2015 | ResNet(152) | Kaiming He | 1st | 3.6% | |




**THE OVERFITTING**

## 5. CONCLUSIONS

AlexNet is a work of supervised learning and got excellent results. It was also important for selecting methods like dropout and data augmentation that helped the network's performance.

The AlexNet made revolutionary implementation on ConvNets that continues nowadays, such as ReLU and dropout. It is not easy to have low classification errors without having overfitting. However, removing any of the convolutional layers will drastically degrade AlexNet's performance. AlexNet is a leading architecture for any object-detection task and may have huge applications in the computer vision sector of artificial intelligence problems. In the future, AlexNet may be adopted more than CNNs for image tasks. As a milestone in making deep learning more widely-applicable, AlexNet can also be credited with bringing deep learning to adjacent fields such as natural language processing and medical image analysis.

## ACKNOWLEDGEMENT

The authors would like to express their sincere appreciation to all individuals who provided guidance, encouragement, and valuable suggestions during the course of this research work. Their support and constructive feedback greatly contributed to the completion of this study.

## REFERENCES

*[1]Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012).ImageNet Classification with Deep Convolutional Neural Networks.In Advances in Neural Information Processing Systems (NeurIPS 2012).*

*[2]https://medium.com/analytics-vidhya/concept-ofalexnet-convolutional-neural-network-6e73b4f9ee30*

*[3]https://www.kaggle.com/code/blurredmachine/alexnetarchitecture-a-complete-guide/notebook*

*[4]https://learnopencv.com/understanding-alexnet/*

*[5]https://papers.nips.cc/paper/4824-image - classification - with -deep- convolutional- Neural-networks.SPdf*

*[6] Sermanet, P., et al. (2013).*
*OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks.*

*[7] Simonyan, K., & Zisserman, A. (2014).*
*Very Deep Convolutional Networks for Large-Scale Image Recognition.*

*[8] Szegedy, C., et al. (2015).*
*Going Deeper with Convolutions.*

*[9] He, K., et al. (2016).*
*Deep Residual Learning for Image Recognition.*

*[10] Alom, M. Z., et al. (2018).*
*The History Began from AlexNet: A Comprehensive Survey on Deep Learning Approaches.*

*[11] Kataoka, H., et al. (2015).*
*Feature Evaluation of Deep CNNs for Object Recognition*