A Survey on Computer Vision Architectures for Large Scale Image Classification using Deep Learning

D. Dakshayani Himabindu[1]
Department of IT, VNRVJIET
Hyderabad-90, T.S, India

S. Praveen Kumar[2]
Department of Computer Science,
GITAM, Visakhapatnam-45,
Andhra Pradesh, India

***Abstract*—The advancement in deep learning is increasing day-by-day from image classification to language understanding tasks. In particular, the convolution neural networks are revived and shown their performance in multiple fields such as natural language understanding, signal processing, and computer vision. The property of translational invariance for convolutions has made a huge advantage in the field of computer vision to extract feature invariances appropriately. When these convolutions trained using back-propagation tend to prove their results ability to outperform existing machine vision techniques by overcoming the various hand-engineered machine vision models. Hence, a clear understanding of current deep learning methods is crucial. These convolution neural networks have proven to show their performance by attaining state-of-the-art performance in computer vision over years when applied on humongous data. Hence in this survey, we detail a set of state-of-the-art models in image classification evolved from the birth of convolutions to present ongoing research. Each state-of-the-art model evolved in the successive year is illustrated with architecture schema, implementation details, parametric tuning and their performance. It is observed that the neural architecture construction i.e. a supervised approach for an image classification problem is evolved as data construction with cautious augmentations i.e., a self-supervised approach. A detailed evolution from neural architecture construction to augmentation construction is illustrated by provided appropriate suggestions to improve the performance. Additionally, the implementation details and the appropriate source for the execution and reproducibility of results are tabulated.**

***Keywords*—Image classification; deep learning; computer vision survey; convolution neural networks; IMAGENET dataset**

I. Introduction

Previous machine vision methods mostly use hand-engineered features. They mostly rely on the morphology of the image sometimes [1]. This can eventually cause a problem in designing a model to capture essential features. To overcome this deep learning models are adapted. Deep learning is advancing in numerous domains such as image recognition, speech recognition [2-6], signal processing [7-12], language processing [13-18], and graphs [19-24]. This leverage in the use of deep learning-initiated advancements in the development of highly scalable hardware architectures which perform large computations. The availability of huge data with high computing resources eventually helped in developing deep architectures which are utilized for large scale tasks. Specifically, in computer vision, deep learning has advanced in numerous subdomains such as image classification [25-30], object recognition [31-43], pose estimation [44-48], image segmentation [49-54], and visual question answering [55-60]. The previous research states that these advancements are held on a large scale to attain state-of-the-art results. In most of the tasks, the generic method applied is convolution neural networks (convnets). There are variant hyperparameters involved in building an effective neural architecture. There are definite properties of convolution neural networks and these properties act as advancements to the current research building large scale architectures. Hence, in this introduction a certain set of relevant concepts regarding Convnets are detailed. In the next section, a set of contributions are detailed explicitly.

II. Contribution

The contributions of this survey to the present existing literature are described as,

1) Firstly, a prerequisite introduction to convnets is provided and the successive advancements and the individual parameters involved in architecture are detailed.
2) The evolution of the convnets from its beginning is explained and a sequential state-of-the-art advancement in image classification utilizing the convnets are elaborated in detail.
3) Finally, a set of recommendations are provided to enhance the neural architectures to obtain successive state-of-the-art performance and pave a path to future advancements.

III. Organization of the Survey

The organization of this survey is described in three phases. Further, Fig. 1 describes the complete flow of this survey.

1) The first phase gives a complete description of the convolution neural networks i.e. specifically describing the components involved in convolutions and their visual illustrations are provided equivalently. This section provides a clear intuition of the working of convnets with a glimpse of the terminology used. Finally, the advantages and disadvantages are equally provided to understand where convnets can perform best and fail.

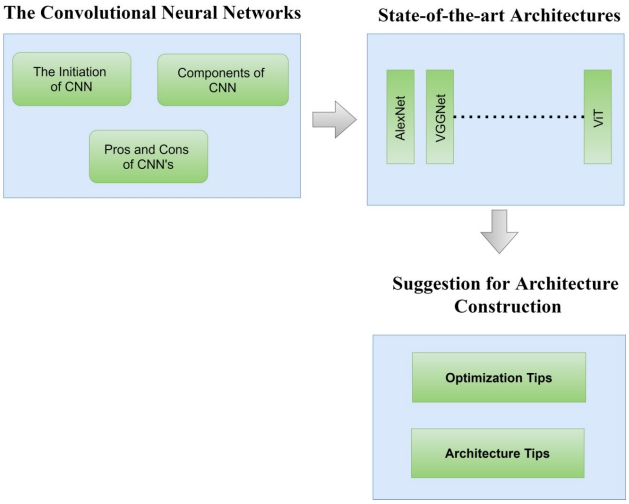

Fig. 1. The Visual Illustration of the Complete Organization of the Survey.

2) In the second phase, a clear understanding of the state-of-the-art networks is provided. Each architecture is described in detail by detailing the method implied and hyperparameters tuned for variant settings. This gives insights to the reader to understand the flow and the evolution of convnets and its developing aspects in the current research.
3) 3. The final phase provides suggestions to construct a novel architecture to provide a good transferability of features with low computational expense by considering various factors.

IV. The Convolution Neural Networks

First, it is aimed to discuss the mathematical intuition of convolution neural networks and next, the first implementation of convnets is described. Next, a set of components involved in the construction of convolution architecture are described accordingly. Subsequently, a set of properties for convnets are detailed. Finally, the advantages and the disadvantages carried by convolution neural networks are specifically mentioned [61, 62].

A. The Initiation of Convnets

The convnets are inspired by the convolution theorem. Convolution is a combinatory operating between two functions where their arguments are real.

$$Conv(y) \leftarrow \int f(x).g(y-x)dx$$

The equation above, Conv(.) is a convolution operation. This convolution operation is mentioned typically[1] as,

$$Conv(y) \leftarrow (f * g)(y)$$

The first function f(.), denotes a probability density function which is referred to as input. The second argument, g(.) is referred to as kernel . Hence, these mathematical implications are helped in building the first convolution neural network.

The first convolution neural network was observed in the literature by LeCun. Y et al. [63]. The main object of this research was to implement a convnets to recognize handwritten postal zip codes. To train the model, backpropagation was implied and then successively able to extract variant features. The complete architecture has 1 input layer, two convolution layers and two fully connected layers. This first work helped revolutionize the convnets to a greater extent.

Subsequently, work by LeCun. Y et al. [64] implemented multi-layered NN by training the model end-to-end using backpropagation. This helped to learn and implement gradient-based optimization. In addition to the previous work, this work implemented a graph transformer network for language understanding which utilizes convnets by training with global techniques. The convolution architecture proposed is known as LeNet-5 which had 4 convolution layers and 3 fully connected layers. The final fully connected layer i.e. final activations are Gaussian connections. This initial conceptualization of convnets produced rigorous outcomes after the evolution of large computational devices to obtain state-of-the-art performance every year in large scale visual recognition challenge ILSVRC-12.

B. Components in Convnets

There are a set of components involved in convnets and this help understand the terminology regarding the convnets. A visual illustration of individual components is provided accordingly.

a) Kernel: The kernel is described as a grid or a matrix that convolves on the input.

b) Stride: The stride is a step taken after each convolution i.e., the number of steps moved by the kernel on the input.

c) Feature Map: The feature map is considered as the output activation incurred after completion of the convolution operation.

d) Padding: The padding is the process of filling the borders of the input equivalently in every dimension i.e., mathematically the input is surrounded by zero eventually increasing the size of the input.

Hence, In Fig. 2 these components involved in convolution are explained in detail. The blue component which is of size 2x2 is input. The grey 3x3 size matrix refers to the kernel. Next, the dotted square grid bordered around the input is called padding. The dark green component which is projected on the top of the input is a feature map obtained. Hence, the convolution operation is carried by moving the kernel onto the input. This kernel applies dot product on the input and a set of values are obtained. Further, these values are aggregated using a sum function. Then, a feature map is obtained accordingly. This process is iterated till the complete input is convolved. In a convnet, kernel size determines the shape of the kernel to perform the convolution operation. The number of kernels determines the number of variant types of kernels with varying values inserted into them. Next, padding

[1]The * mentioned denotes the convolution operation.

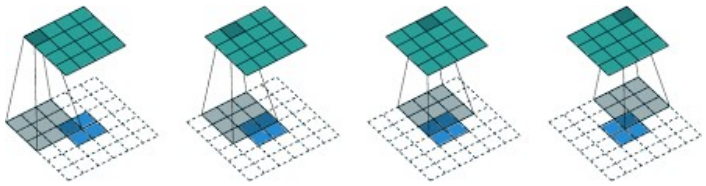

Fig. 2. Visual Illustration of Convolution Operation.

is generally used to produce similar dimensional output. In the next section, a set of advantages and disadvantages involved in convolutions are detailed. Further, a detailed explanation is provided by considering varying situations and altering the above-mentioned components in the work [65].

C. Pros and Cons of Convnets

The convnets do have certain abilities which provide higher performance on multi-domain tasks. Even having definite advantages, convolutions carry a set of disadvantages which are discussed in detail.

a) **Advantages:**

- **Transferability**: The previous deep networks such as Restricted Boltzmann Machines, deep belief networks and fully connected neural networks do not persist with transferability of weights. But, the convnets are provided with transferability of features. Hence, a certain layer in architectures can be extracted to reproduce the weights for variant tasks. It can be further implicated in architecture pruning for improving the feature extraction for variant tasks.
- **Sparse Connections**:The connections in most of the previous existing neural networks have dense connections i.e. having an extreme number of connections which in turn increases the computational budget of the model. But, whereas convnets have sparse connections reducing the redundant connectivity and reducing the computational expense.

b) **Disadvantages:**

- **Rotational Sensitivity**:The convnets cannot extract the features of the entity residing an input which is rotated until and unless the objects in the images are rotationally symmetrical. Hence, to overcome these many techniques are implied such as augmentation. Horizontal flip, vertical flip and angular rotations are provided to an individual image to extract features even having rotational changes.
- **Time-Variant signals**: The convolutions lack in understanding the signal processed during a variant time pattern to that of a non-linear system. This can lead to the problem is speech specifically problem underlying the acoustic detections. But this problem is not seen in image recognition.

V. State-of-the-art Vision Models

A. Alex-Net

Krizhevsky et al. [66] proposed an end-to-end trainable deep convolutional network for large scale image classification i.e., on IN12. They observed the problem of using ML methods for image classification. They developed an eight layered deep NN which has 5 conv layers and 3 fully connected layers. The kernel size and stride implied are clearly illustrated in figure.3.

Firstly, they used relu [67] as non-linearity to forward the activations from one layer to another where they observed speeding up of convergence when relu is used as non-linearity. Second, they used GPU's for training their network in which, two GPU's are used with parallelized computing and having communication mutually layer to layer. This improved the performance of the model by reducing T-1 error and T-5 error by 0.017 and 0.012 respectively. Next, for normalization a technique (which is a similar normalization technique to that of [68]), named local response normalisation, is operated for conv layers which are tuned while validation procedure. This improved the performance of the model by reducing the T-1 error and T-5 error by 0.014 and 0.012 respectively. Next, the overlapping pooling technique is utilized which pools the pixels which are not only adjacent but also which are overlapping with correspondence. It is achieved by reducing the step during convolution. This reduced error-rate of T-1 and T-5 activations by 0.004 and 0.003 respectively.

The constructed architecture has consumed 60 M parameters which are mentioned in Fig. 3. To have good generalization a sequence of tasks was done to reduce the problem of overfitting in the networks. Firstly, data augmentation is done. In this step, the samples regarding an image are increased either translating the image in horizontal (or vertical) directions or the pixels of the images are changed in terms of colour intensities. This is done by considering the principal components of images and adding weight to pixels accordingly. This procedure led to an increment of T-1 accuracy by 1%. Secondly, the fully connected layers are attached with two dropout layers [69] (for the last two layers excluding class activations) with a drop ratio of 50% i.e. 50% of the neurons are inactive during the training and the network tend to learn during validation.

The complete model was trained on 90 epochs. It is optimized using SGD with an initial learning rate of 10^{-2} and 9×10^{-1} as momentum through 128 batches (batches considered per iteration). When no convergence invalidation was observed in the learning, the rate was decreased 10 times to that of initial learning. The model achieved a T-1 error rate of 37.5% and a T-5 error rate of 17%. Further, the model was altered in various types and different accuracy score are obtained. These details are tabulated in the Table I.

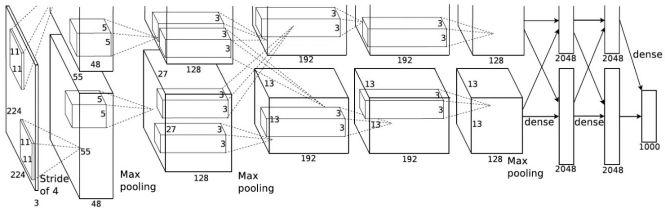

Fig. 3. Alex-Net Architecture with Varying Strides and Filters.

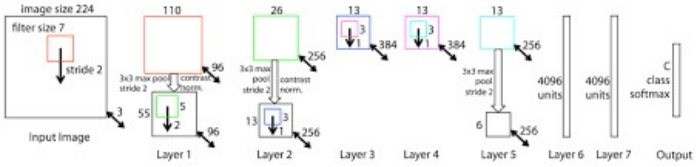

Fig. 4. Ze-Net Architecture with Varying Strides and Filters [5].

B. Ze-Net

Matthew D. Zeiler et al. [70] proposed a Conv NN which is very similar to Alex-Net by visualizing the feature maps and kernels for better understanding internal computations of the convnets. They developed an eight layered deep NN which has 5 conv layers and 3 fully connected layers. The kernel size and stride implied are clearly illustrated in figure.4.

The architecture of the model is designed with a decoder and an encoder which extracts latent representations and reconstructs the image respectively. Several conv layers are used to extract the spatial features with ReLu as activation throughout the network. The decoder helps in unspooling the visual representations by a switch variable. This switch variable is used for memorizing the pooled information in the encoder structure and mapping it on the decoder structure. Further, to observe feature extraction, translation scaling and rotation mechanisms are performed in which the convnets were invariant to translation and scaling but not for the rotation. Finally, to observe the localization ability of convnets a certain part of the image consisting of important feature is occluded. It is observed that convnets significantly degraded in terms of performance due to occlusion.

The model implied is very similar to that of AlexNet with two variations, the filter size is reduced in the first layer from 11x11 to 7x7 and stride 4 of convolution is reduced to stride 2. Augmentation is performed by subtracting the input with individual pixel mean and used 10 variant sub crops techniques such as horizontal flip, vertical flip etc. The learning rate with which the SGD optimizer was initialized as 0.01. A momentum of 0.9 was implied for faster training. The bias components were initialized with zero and 50% of the densely connected layers are dropped during the training process. The model acquired T-1 and T-5 error rates of 36% and 14.7% respectively. Further, the network pre-trained on ImageNet is implied on Caltech-101 dataset with an accuracy score of 83.8 for 15 images per class whereas increasing 30 images per class it obtained an accuracy score of 86.5%.

C. OverFeat

This work was inspired by the standard concepts that injected good improvement in the field of classification [71-74]. Sermanet et al. [75] proposed a framework implying CNN's not only for classification but also for detection and localization. The novel localization criterion in this work is obtained by capturing and aggregated to multiple object boundaries. When the localization task is performed on ImageNet the best performing OverFeat model secured the first position in the 2013 challenge.

The main objective of the OverFeat is to perform classification by simultaneously locating and detecting the objects with the use of a single conv architecture. A novel method is implied to detect and localize the bounding boxes of the image which is predicted by the neural architecture. With a combination of various localization predictions, the process of detection acquires good features and hence the performance is increased and eventually training time can be reduced. This method not only helps to provide less computation but also with greater performance acquiring higher accuracy scores.

The complete OverFeat model has three ideologies and the methodology is implemented accordingly,

1) Initially, a conv net is applied at variant locations captured in the specified image. Sequentially, a sliding window approach is implied using different scales. This eventually helped to provide a better classification model but, the localization performance was degraded.
2) The system was not only trained to produce distribution for the set of categories but also improved localization by properly constructing the size of the bounding box to capture the region of interest for that specified category.
3) Lastly, a proof of concept was provided for a specific category at individual locations.

The implementation works by training a conv net by using a sliding window as the decision box by choosing the centre pixel and classifying it accordingly to a definite object. The advantages of this method are the bounding contours utilized for localization need not be rectangular. The disadvantage of the model is that it acquires numerous pixel-level labels which in turn increases computations cost. This work was the first implementation of localization, and the detection task for ImageNet by using a unified framework. The localization and detection task performed by overfeat is done by allowing the model to guess the labels for the specified object five times and if the probability of the guess turns to be 0.5 and above (matching the ground truth label) then, a definite label for the object is assigned to definite class accordingly. The five times guess the pattern is chosen to specify the correct object in the presence of multiple objects without labels.

During the construction of the OverFeat, a set of hyperparameters are tuned and are mentioned individually. The optimizer implied in this method is SGD with an initial learning rate of 5×10^{-2}. A momentum was used to faster the training procedure (an initial momentum of 0.6 was implied). Weight decay for the L2 regularization is initialized as 10^{-5}. ReLU is used as an activation function at the utmost every layer. The initial five layers of the model implied are very similar to AlexNet with ReLU activations and successive pooling layers (max-pooling layers). But with many similarities, certain differences are to be noted and they are mentioned. No local response normalization is utilized in this work as it did not improve performance. The pooling layers implemented do not overlap as they depicted better performance. Further, implying small stride in the first two layers, better invariances was obtained i.e., large stride speedups the training process but performance in terms of accuracy can be degraded.

The OverFeat proposed 8 models of which, two are ensemble models. The fast ensemble model with four scales and fine stride acquires a T-1 error rate of 35.10% and a T-5 error

rate of 13.86%. Whereas, an accurate model acquired a T-1 error rate of 33.96% and a T-5 error rate of 13.24%.

D. VGGNets

Simonyan et al. [76] worked on deep neural networks with different depth of layer's in them to know the changing rate of accuracy concerning the depth of the neural network. The depth of the neural networks proposed in this paper varied from 11 to 19 layers. Six different types of networks were used by the author to know how the models perform based on different configurations in it. The kernel size or the receptive field is set to the size of 3X3 rather than 5X5 or 7X7 because a smaller receptive field help in capturing the details of the image in a more specific way and use fewer parameters. The six types of networks built in this paper have been given the following names A, A-LRN, B, C, D and E. These networks differ by the depth of layers. An A-LRN is the two networks with a depth of 11 layers, the only difference is that in A-LRN, Local Response Normalization (LRN) is used to check how the accuracy is varied when LRN is used in a network. It was observed that adding LRN to the network was not much of use to improve the accuracy score. B has a depth of 13 layers, C is just an extension of B where there are 3 extra 1X1 convolutional layers in it. D and E have 16 and 19 layers of depth in their network configurations respectively.

In the aforementioned networks, a max-pooling layer is present after a few convolutional layers or a block of these layers. Inside each block, there is a combination of 3X3 and 1X1 convolutional layers accordingly. The input image is of size 224X224 pixels, which is downsampled by the convolution and max-pooling layers next the extracted features are passed into the dense layer for the classification or detection task of the image. These architectures used Stochastic Gradient Descent (SGD) with 0.9 momentum and has a batch size of 256. Drop out was also used in two fully connected layers followed by a dense layer and a softmax layer to predict the class of the image. The learning rate was set to 10-2 and this was decreased by a factor of 10 if the accuracy got saturated at a point. Training of these networks was completed after 74 epochs. During the training time, Lr was decreased by a factor of 10 for 3 times in total. First, the networks (A, A-LRN, B) were trained on a single scale of 256 and the remaining networks (C, D, E) were trained using multiple scaled images (scale jittering) with the scale ranging from 256 to 512. It was observed that the performance of these networks improved significantly with the use of scale jittering and by increasing the depth of the network, E convnet got a top-5 Val-error of 8% which is a competitive score.

To further assess the capabilities of the network, the VGG team used scale jittering even more aggressively on the train-test set this time and saw that convnets D and E got a top-5 Val-error of 7.5%. Multiple crops were also used in the next experiment and it was compared with the dense evaluation method. From this experiment, it is concluded that the multiple crop method outperforms the dense method. The testing method shown by the VGG team was very different from the previously mentioned works, where the last FC layer was converted into a convolutional layer and this receptive field was put on a whole image and then obtained a single vector with the individual class score. The vector was pushed into the softmax layer to get the prediction score.

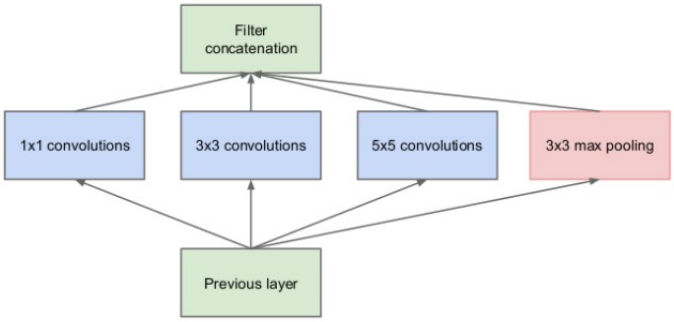

Fig. 5. Naive Inception Module.

An ensemble of all there convnets was made and it was seen that the seven networks ensemble model has a test error of 7.3% and the ensemble of D and E convnets had a test error of 6.8%. The 2 convnet ensemble networks secured second place in the ILSVRCV-2014 challenge. But the margin between the scores was very close when compared to Google Net (first place). The single net performance of VGG architecture outperforms all the other architectures (even Google Net) with a large margin of 0.9%.

E. Google-Net and InceptionV2

Szegedy et al. [77] presented a deep learning model which has an inception module in it. Google-Net mainly focuses was to develop a deep neural network architecture with a less computational expense. As the network goes deeper the arithmetic operations performed by the models also increases and this gives scope for newer error that occurs with computing gradients. Because of the previously mentioned reasons the author suggests creating a sparse network rather than a fully connected network. The goal is very simple all we have to do is find optimal weights through a sparse network that could approximate or predict an image. Translation invariances added in this work by building a network through several convolution layers.

Fig. 5 shows the naïve inception module which applies convolution to the input image with a kernel size of 1x1,3x3,

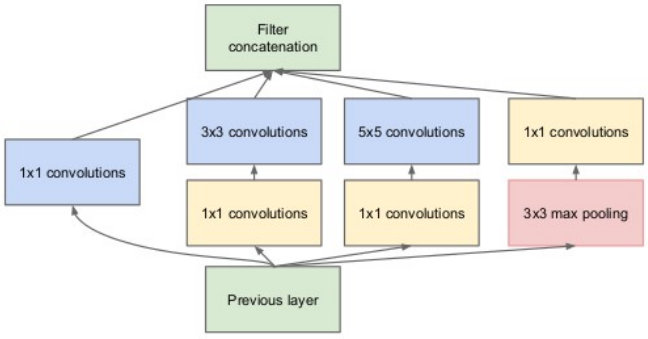

Fig. 6. Dimensionality Reduced Inception Module.

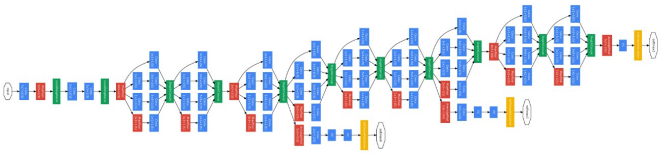

Fig. 7. Detailed Architecture of Google-Net.

and 5x5. Next, pooling is done to the input image and these activations are concatenated using correlation statistics instead of stacking up the layers can increase computational expense. With this understanding, a new inception module is created, and the dimensions are reduced by bottleneck convolutions i.e., 1x1 convolution kernel to the input image and it is observed that lower-dimensional space preserve the information of the corresponding image. These dimensionality reduced inception modules are now stacked on each other by applying max pooling layer of stride 2 in between the modules occasionally. The proposed Google-Net architecture consists of 22 layers in total. An ensemble of 7 such models was created and tested on IL-14 for classification as well as detection.

Fig. 6 shows the architecture of GoogleNet. In between this network for few inception modules, a classifier was assigned to them. This has helped to generalize images more precisely. These classifiers contain a 1x1 convolutional filter with 128 filters in it. Next, the convolutional layer is stacked with a fully connected layer with 1024 neurons in it. Followed by a dropout layer and a SoftMax layer to provide the probabilities of each class and then predict the image class. In this network, every layer uses the ReLU non-linearity function for the activation of each neuron in the network.

Seven distinct types of networks were built based on the new inception module to train them on the ImageNet dataset with different learning rates and sampling methodologies. The probabilities of all these networks were averaged to get the output. With this ensemble method, Google Net got a top-5 error rate of 6.67% on testing and validating set. An ensemble of 6 models was used in the ILSVRC 2014 detection challenge which achieved map of 43.9% and secured first place in both the classification and detection challenges. From this work, it can be deduced that the sparse network can be useful in deep neural networks to know the deep representation of the image while using less computational resources. Kindly refer Fig. 7 for detailed understanding of architecture.

A certain problem, covariant shift is observed while training a deep neural network is addressed and solved by implementing the Batch Normalization (BN) procedure. This paradigm was proposed by Sergey Ioffe and Christian Szegedy [78-80]. BN procedure was implemented on Inception ensemble with an Image resolution of 224x224 produced a T-1 accuracy score of 79.9% and T-5 accuracy score of 95.1%.

F. InceptionV3

Szegedy et al [81] implied the aforementioned Inception architecture and scaled the convolution layer to provide higher performance by decreasing computational expense. This is the upgraded version of GoogleNet and maintained appropriate convolution by doing defiant regularization throughout the network. The authors illustrated the work by defining a set of principles and scale the conv layer by optimizing techniques. The principles are defined in such a way that they performed experimentation on different datasets by considering much architecture. The principles of the network are

- A cautious decrement in representation is preferable, instead of bottleneck layers at the beginning of the network.
- Higher dimensions in the network are easier to process with piling up the activations in a conv network for extracting invariant features.
- Even though pooling provides faster learning, spatial aggregation in the network holds the representational features without any loss in the lower dimensions.
- The width and depth of the network must be optimally selected with a balanced criterion.

Generally, a 5x5 or large conv layers can capture the activations of the previous layers. Reducing the feature map size would decrease the no of parameters, training time and computational cost of the network. The inception module consists of 5x5 conv layers, instead of these, the authors have replaced 5x5 conv layers with two 3x3 conv layers which are shown in Fig. 8 with 28% relative gain. But this method has a problem of loss of expressiveness or using a low filter size (below 3x3) which may produce the best outcome. So, the authors have come up with an idea of using asymmetric convolutions. The concept of asymmetric convolution is any nxn convolution can be replaced by 1xn convolution which is followed by nx1 convolution. As n increases the computation of the model decreases. The 3x3 convolutions in the network are replaced by 1x3 and 3x1 as shown in the figure.9. By using this method, it reduces the computation cost by 33%. The activation maps in the network filters are improved because to get rid of the bottleneck representation. The network consists of 42 layers and has 2.5% more computation than GoogLeNet.

The concept of asymmetric convolution is any nxn convolution can be replaced by 1xn convolution which is followed by nx1 convolution. As n increases the computation of the model decreases. The 3x3 convolutions in the network are replaced by 1x3 and 3x1 as shown in the figure.9. By using this method, it reduces the computation cost by 33%. The activation maps in the network filters are improved because to get rid of the bottleneck representation. The network consists of 42 layers and has 2.5% more computation than GoogLeNet.

The model takes SGD as an optimizer with a batch size of 32 which is trained across 100 epochs by considering a learning rate of 0.045. The model achieves the state-of-the-art results with T-1 and a T-5 error rate of 21.2% and 5.6% respectively. By ensembling 4 Inception-v3 models they got a T-1 and T-5 error rate of 17.2% and 3.58% respectively.

G. Inception-v4, Inception-ResNet

Szegedy et al [82]. has extended the idea of Inception-v3 by combining residual connections to it with accelerating training. This model has won the 2015 ILSVRC challenge by acquiring state of the art performance. The authors have

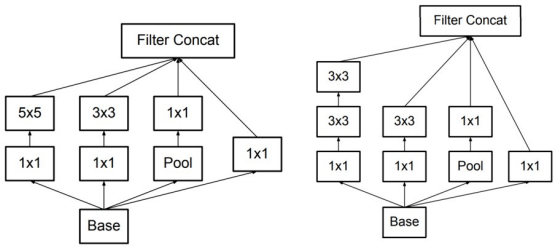

Fig. 8. Illustration of the Variation of Bottleneck from 5x5 to two 3x3 Convolution Blocks.

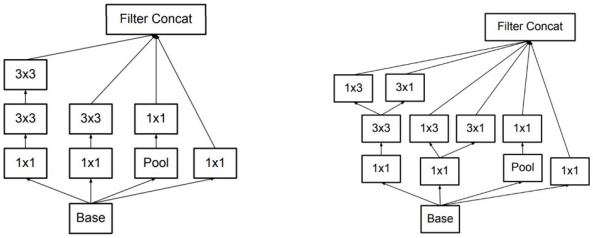

Fig. 9. Illustration of the Variation of Bottleneck from a 3x3 to 1x3 and 3x1 Convolution Blocks.

provided proof of the residual connections speed up the training of Inception networks. As the Inception networks are very deep, the filter concatenation stages in the network are replaced by residual connections. By increasing the depth and width of the Inception-v3 networks they proposed another network called Inception-v4. To provide an optimised network, the layers are tuned cautiously. To connect Residual versions with the Inception network, a cheaper Inception block is implied rather than the original Inception. Fig. 10 shows the whole architecture with residual connections inside an Inception network. There are filter expansion layers (1x1 convolutions with no activation) inside the network. Batch-normalizations are omitted on the top of the network and overall the no of inception blocks was added subsequently. While experimentation the authors have found that if the networks have more than 1000 filters, the model has died before the training has started. There is no use in increasing the batch size or lowering the learning rate. It seemed to stabilize the training process by scaling the residuals and then adding to the before layers.

Using RMSProp [83] as an optimizer and learning rate of 0.045 they achieved a T-1 and T-5 error rate of 19.9% and 4.9% respectively on ILSVRC 2012 by considering Inception-ResNet-V2 as the base model. By combining three residual and one Inception-v4 they achieved a T-5 error rate of 3.08% on the ImageNet classification challenge.

H. ResNext

Saining Xie et al [84] has developed a model succeeding the ResNet model which is known as "ResNext". This model is the 1^{st} runner-up in ILSVRC 2016 competition. This model contains extra dimensionality called cardinality which deals with the depth and width of the network. The ResNext

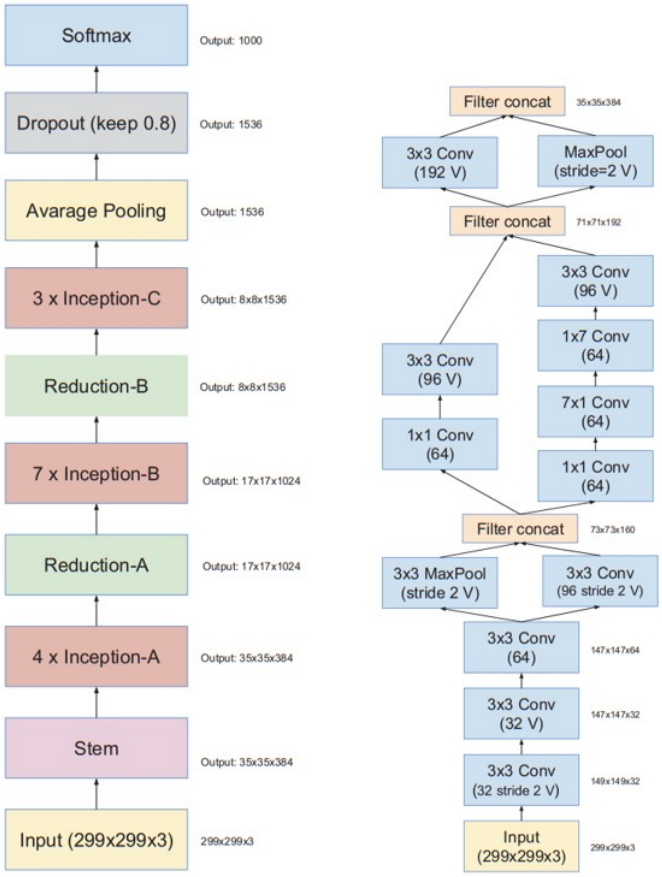

Fig. 10. The above Figure Illustrates the Complete Architectural Details of Inceptionv4.

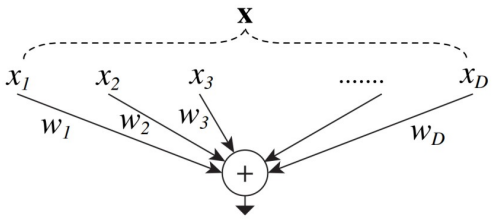

Fig. 11. The Neural Weight Transformation with Varying Inputs and Weights.

architecture consists of aggregated transformations by splitting, transforming and aggregating a single neuron.

The model embraces the method of repeating layers in VGG and ResNet's by making use of the spilt-transform-merge strategy in the Inception model. The neuron in the network splits the input and transforms the weighted sum to low dimensions by aggregating through summation fig(11).

Each neuron in the network carries out a non-linear function due to the addition of the new dimension (cardinality). The ResNext model replaces the elementary transformation with a signified function and constructed by combing a set of residual

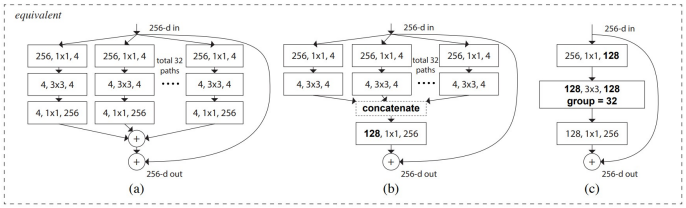

Fig. 12. Three Variant Architecture Designs Implemented in the ResNeXt.

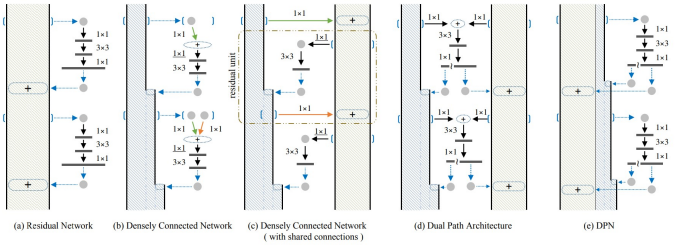

Fig. 13. Visual Illustration of Networks from Deep Residual Network to Dual Path Networks (DPN).

blocks which are subjected by two rules first, maintaining the same shape in the spatial maps i.e. ensuring width size and a filter size of each block are the same. Second, it maintains the complexity of the network where width is multiplied by 2 when the spatial maps are down sampled. This model takes fewer parameters when compared to existing ResNet's with $4.2x10^n$ FLPOs.

Each block in the ResNext network has the same number of internal dimensions. ResNext-50 (32x4d) indicates four internal dimensions with 32 paths (cardinality=32). When compared with the Inception-ResNet block ResNext model is designed with less effort in each path and implemented in different forms illustrated in the figure.12. The third form of the network is chosen because it is much faster and has grouped convolutions than the other two models. The grouped convolutions consist of 32 convolutions with input and output of 4 dimensions. The experiments were carried out with increasing the cardinality and width which results in the increase of FLOPs by a factor of 2. By increasing the cardinality, the error is reduced by 1.3% to 20.7% rather than increasing the width of the network. The ResNext-101 (64x4d) has obtained a T-1 error rate of 20.4% and a T-5 error rate of 5.3% with an image size of 224x224. They also evaluated this model on different dataset like ImageNet-5K and got an error rate of 40.1% which reduces the error by 2.3% when compared to ResNet-101.

I. Dual Path Networks

Y. Chen et al [85] proposed an architecture Dual path network (DPN). It is the combination of a residual network (ResNet) and a Densely connected network (DenseNet). The proposed architecture takes the feature reusage from ResNet and feature exploration from DenseNet by maintaining low complexity and more number of parameters. DPN includes higher-order recurrent neural networks (HORNN) which benefit from sharing weights throughout the network and also proves that ResNets and DenseNets are the same by using HORNN. By optimizing the network they had achieved a state of the art results on ImageNet-1k.

To understand the connection between the two networks they formulated the HORNN as
$i^j = p^j \sum_{Q=0}^{j-1} R_Q{}^j(i^Q)$

Where i^j is the state which is hidden in RNN at a particular step which is denoted by Q, the current step is indicated by j. $R_Q^j(.)$ function is for extracting features. $R_Q^j(.)$ and $p^j(.)$ do not share weights but extracts the same features more times. So that it may lead to feature redundancy this is one of the drawbacks of the network. ResNet has the problem with finding new features while DenseNet has the problem with high feature redundancy. The DPN architecture has a 1x1 conv layer, 3x3 conv layer and 1x1 conv layer as last layer figure. 13. The output of the network has divided into two parts first, the element-wise addition of residual combinations. Second, adjoined with DenseNets to improve the learning ability of individual micro-block. The second conv layer in the network is replaced by ResNext. ResNet is widely used so the authors choose it as the main part of the network which is combined with DenseNets to construct the architecture. A DPN can implement either by adding a "slice layer" or "concat layer" to the residual network which consumes extra memory usage and computational cost. DPN has 26% fewer parameters when compared to ResNext-101 (64x4d). The model is implemented on 40 k80 graphic cards with a batch size of 32 on individual GPU. The proposed architecture DPN-131 (40x4d) has got a T-1 error rate of 18.55% and a T-5 error rate of 4.16%. The model is also evaluated on different dataset like places 365 standard datasets where it got T-1 and T-5 accuracy scores of 56.84% and 86.69

J. NASNets

Zoph et al. [86] contribute a new search space for constructing neural architectures by transferring the weights from a smaller dataset to that of the larger one. This research introduces a new regularization method (known as scheduled-drop-path) for the models developed through their proposed search space which improves generalization. The efficient model developed through this search space attains SOTA results in classification (IN-12 dataset). Additionally, utilizing the R-CNN framework the learned representations are captured through the best model attains SOTA on the CoCo dataset. The proposed NAS (Neural Architecture Search) [87] implements a reinforcement strategy to optimize the configurations to design a good neural architecture. This method implies 2 different cells with a similar structure and separate weights. These cells are normal and reduction which is shown in Fig. 14. The normal cell input and output are of the same dimensions whereas, the reduction cell reduces the shape of input dimensions to half the previous input (i.e. stride 2 is applied). These cells provide faster and efficient search with appropriate generalization. The NAS which is mentioned in Fig. 14. has a controller block is a recurrent neural network that predicts multiple architectures with multiple probabilities. Then a small network (child) is trained to reach convergence with a certain accuracy score. The gradients of multiple probabilities attained are scaled in such a way to attain new accuracy scores and are updated to the controller.

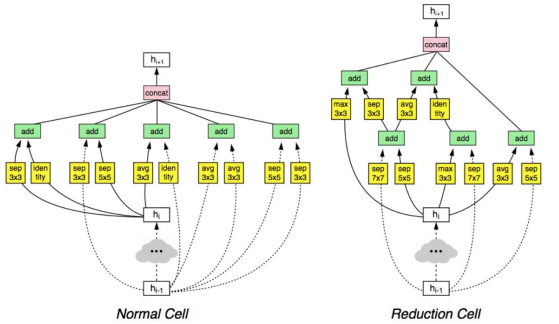

Fig. 14. Detailed Architecture of Normal and Reduction cell of NAS-Net.

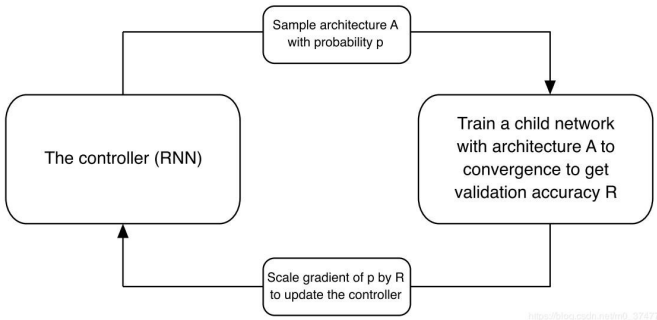

Fig. 15. A Brief Illustration of Neural Architecture Search.

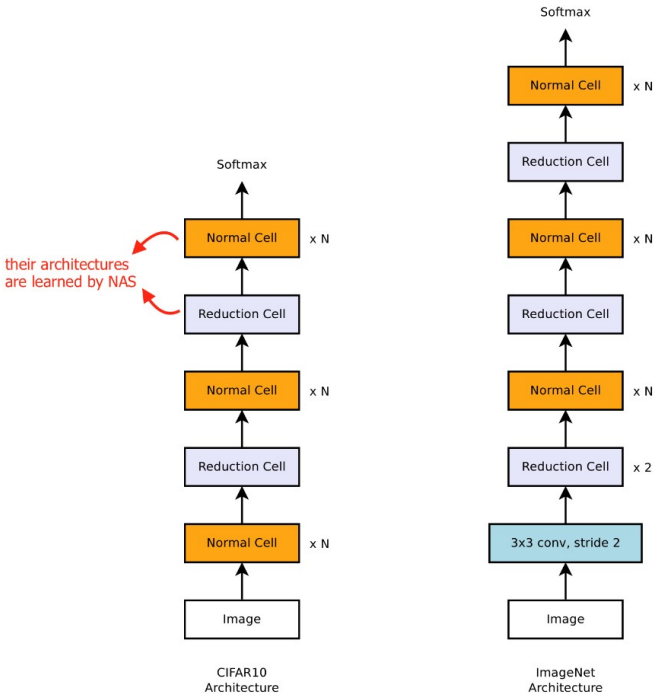

Fig. 16. Visual Illustration of Normal and Reduction Cell in NAS-Net.

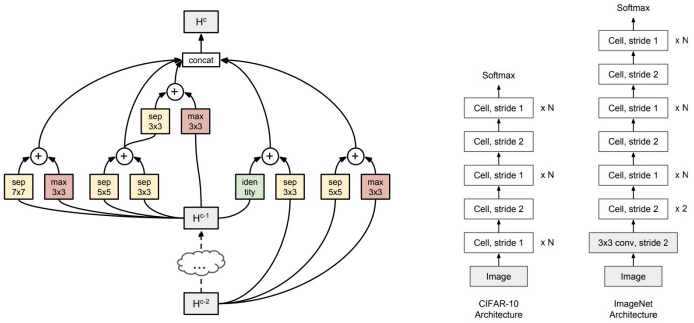

Fig. 17. The Architecture Details of the PNASNet.

Observing the Fig. 14 the cells have two hidden states. The input of the hidden states is passed from the output of the preceding cells. If there are no previous cells then each hidden state takes an image as input. The architecture is formed by predicting subsequent convolution which can be formed using those two hidden states. The complete algorithm for NAS is determined [87]. Instead of random search, NAS provides a reinforming learning strategy to construct a deep architecture. The random search lack in providing significant result only for CIFAR-10 dataset and Fig. 15 illustrates the NAS search architecture.

The architectures which attained greater performance for the ImageNet, as well as CIFAR-10, are mentioned in Fig. 16. The controller is trained on the PPO criterion [88]. The learning rate was set as $3510^{(}-5)$. All the activations of the convolution are fed using relu as non-linearity with successive batch normalization layers. Additionally, implied bottleneck convolutions i.e. 1x1 convolutions and implied RMS prop as the optimizer. The best performing model takes 331x331 input image size and attains a T-1 accuracy score of 82.7% and T-5 accuracy score of 96.2% with 88.9 Million parameters. As a note, for object detection NAS-Net implied in Faster-RCNN obtained state-of-the-art mAP of 43.1%.

K. PNASNets

C. Liu et al [89] proposed a network by using reinforcement learning and different algorithms. Sequential model-based optimization (SMBO) is used in the model which finds for structures in the network by increasing complexity with simultaneous learning. The model is compared with the previous method which is efficient up to 5 times within the same search space. The architecture consists of a search algorithm where it finds the best conv "cell". Each cell includes a certain number of blocks where it consists of two input tensors with a combination operator. These blocks are stacked and determined based on the training time this approach easy transfer datasets from one to another. The search space in the network is based on the heuristic approach which starts with a basic model and improved complexity as the search goes on. The detailed architecture of the model is shown in Fig. 17.

The architecture details of the PNASNet.

- Considering simple structures, the training of the model becomes faster and inherit the process quickly.
- A set of surrogates (proxy) are requested to obtain the predictions of the quality of the structures which tend to be higher from the input is received.
- The search space is factorized by multiplying smaller search spaces which give the advantage of finding

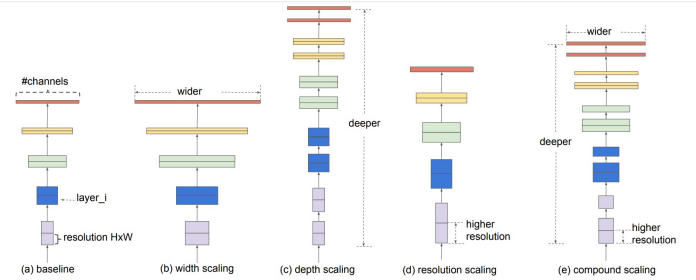

Fig. 18. The above Figure Illustrates the Scaling of Variant Components Involved in Convnets.

more block with the precise model.

The model is trained on ImageNet with RMSProp as an optimizer, an initial learning rate of 0.04 and decayed after 2.2 epochs with a batch size of 32. The architecture consists of 86.1M parameters which are smaller than NASNet. The model has achieved a state of the art results with T-1 and T-5 accuracy of 82.9% and 96.2% respectively.

L. EfficientNet

M. Tan and Quoc V. Le proposed a model by scaling the depth of the network, width of the network and resolution of the image [90]. The model is developed by using a combination of MobileNets and ResNets. By scaling those parameters, the model led to better performance with less computation cost. The scaling of the network is done in such a way they maintained a constant ratio throughout the network this scaling method is known as effective compound scaling. The scaling of the model is done as shown in Fig. 18. Due to various resources, they face a problem while scaling the convnet. So, they increased the depth, width and resolution of the image by a factor of P^k, Q^k and R^k respectively where P, Q, R are small grid constant coefficients. By increasing the depth of the network, a convnet can apprehend more complicated features. Here, a problem of vanishing gradients arises and it is very complicated to train the network. By scaling the depth with a coefficient P, they maintained balance in the network. The next constraint is to balance the width of the network which is usually done in very small models. Increasing the width can capture more fine-grained features and takes less time for training. Their experimentations have shown, the wider the network, the more is the drop in accuracy. The resolution of the image is scaled by a factor R because of the higher resolution of the image takes more time for training. The proposed method enhances the accuracy and optimizes the FLOPS. By examining the depth, width and resolution values of the network to be 1.4, 1.2 and 1.3 respectively are found to be accurate with 2.3B FLOPS.

The model (EfficientNet-B7) achieved a T-1 and T-5 accuracy of 84.4% and 97.1% respectively with 66M parameters which are 8.4x smaller than the previous state-of-the-network.

M. FixResNeXt

Hugo et al. [91] performed augmentation trails for acquiring better generalization by choosing appropriate train and test size for a network. During experimentation, it is justified that, lower training resolution for an image and higher testing resolution eventually improved the performance to a greater extent by reducing training computational cost. This procedure was implemented on ResNeXt-101 by outperforming the existing models and obtained state-of-the-art performance on 2019 ILSVRC. There was a good significant shift in the model when the training and the testing methods are fine-tuned separately. A joint optimization is done by scaling the train-test resolutions equivalently by maintaining individual RoC (region of classification) sampling. To overcome the distribution, shift the first two layers of the model are prioritized to fine-tune by varying the crop resolution. A detailed analysis is done to pre-process the model by increasing the crop resolution at the testing phase and during training, roc sampling is done appropriately. This eventually, acquired a good generalization by providing lower train resolutions and higher test resolutions. The computation is reduced by 3-fold by halving training resolution which in turn speed up the training procedure. Implying larger batches for training impacted a good performance with saving GPU memory. A further modification is done to the model by adjusting activation statistics of the layer which is preceding the global average pooling (GAP) layer. When these techniques are implemented on ResNet-50 by varying the test size the results obtained are mentioned (CR is equivalent to crop resolution). First, with 64 as CR, the model obtained an accuracy score of 29.4% on ImageNet. Further, with an increase in resolution by 128 the model obtained an accuracy score of 65.4%. A higher accuracy score of 78.4% was obtained for 288 as CR.

It is observed that increasing test resolution further (more than 288) the accuracy score was gradually decayed. Even after assigning appropriate test resolution a set of skewed activations were observed and they were addressed by two methods. First, a parametric adaption is chosen and the other is an adaption by tuning appropriately i.e., fine-tuning. Hence the parameters of the architecture are to be addressed in detail with experimental results. Instead of performing the train-test method for generalization 10-fold cross-validation is implied with mean and standard deviation for each execution. During the training process, extra training data was provided for most of the implementations. The best performing model (ResNeXt-101) acquired parameters of 829 M. While training ResNet-50 learning rate was initialized as 0.1 and is decayed by 10 for every 30 epochs. Initially, 512 samples were fed into the network as a batch with a horizontal flip, color jittering and random resize crop as augmentation parameters. The experimentation was performed on eight Tesla V100 GPUs. Subsequently, a set of 80 CPU clusters were inserted along with GPUs. The experimentation was carried out on standard pre-trained networks such as ResNet-50, ResNeXt-101 and PNASNet. Large network classification was done by complete fine-tuning PNASNet-5-Large with a train resolution of 331x331 which obtained the highest T-1 accuracy and T-5 accuracy of 83.7% and 98.0% on 480x480 test resolution respectively. Whereas, ResNeXt-101 was trained on 224x224 as a resolution to obtain a state-of-the-art accuracy of 86.4% with 320x320 as test image resolution. Further, this method was effective even on various transfer learning tasks and it obtained state-of-the-art performance for CUB-200-2011 and Birdsnap datasets.

TABLE I. IMPLEMENTATION DETAILS AND SOURCE CODE REGARDING STATE-OF-THE-ART MODELS

SOTA Works	Source Code with Implementation Details
AlexNet	https://worksheets.codalab.org/worksheets/0xfafccca55b584e6eb1cf71979ad8e778
ZeNet	https://github.com/atriumlts/subpixel
VGGNet	https://github.com/tensorflow/models/blob/master/research/slim/nets/vgg.py
InceptionV2	https://github.com/tensorflow/models/blob/master/research/slim/nets/inception_v2.py
InceptionV3	https://github.com/tensorflow/models/blob/master/research/slim/nets/inception_v3.py
InceptionV4	https://github.com/tensorflow/models/blob/master/research/slim/nets/inception_v4.py
ResNeXt	https://github.com/facebookresearch/ResNeXt
DPN	https://github.com/rwightman/pytorch-image-models
PNAS*	https://github.com/chenxi116/PNASNet.pytorch
NASNet	https://github.com/tensorflow/models/blob/master/research/slim/nets/nasnet/nasnet.py
NoisyStudent	https://github.com/google-research/noisystudent
EfficientNet*	https://github.com/tensorflow/tpu/tree/master/models/official/amoeba_net
FixResNext	https://github.com/facebookresearch/FixRes
BiT	https://github.com/google-research/big_transfer
ViT	https://github.com/google-research/vision_transformer

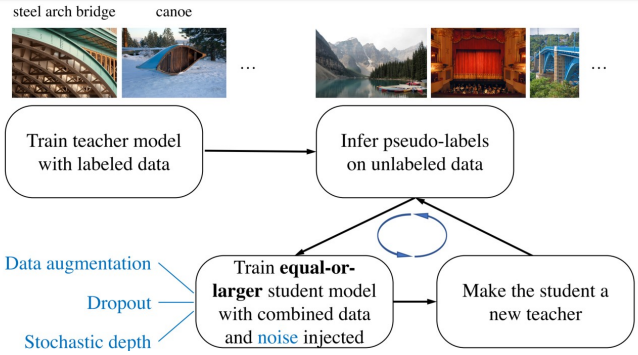

Fig. 19. The Working Principle behind the NoisyStudent Procedure.

N. NoiseStudent

Q. Xie et al. [92] implied self-supervised for training large scale images. This approach is based on a student-teacher learning paradigm. First, EfficientNet is trained on a set of labelled images(as a teacher model) of ImageNetand then produced pseudo labels by evaluating on a different data set which consists of 300 Images. Second, the larger EfficientNet model is considered as a student model and this is trained on the grouped labels i.e., pseudo labelled and labelled images. Next, the student model is replaced with the teacher and this process is iterated to attain significant performance. It is observed that the teacher model dose does not contain nosy labels as they were trained through a supervised approach. In the student model, a noise component such as dropout, stochastic depth, and random augmentations are implied. These implementations helped the student model to have greater generalization to that of the teacher model.

There are certain hyperparameters involved in tuning the model. The batch size is assigned as 2048 as default. A varying batch was implied i.e. 512, 1024, and 2048 to the EfficientNet model and all the batches turned out to have the same performance. The student model was trained for 350 epochs and smaller student models were trained for 700 epochs. The noise implied to student model with a dropout of 50%. Further, the random augmentation [18 for STNS] provided a magnitude of 27 for two operations. Finally, the probability of survival is set to 0.8 for the stochastic depth. The Noisy Student model beats the current state-of-the-art BiT Large with a 0.9% increment in accuracy i.e., the best performing NoisyStudent acquired an accuracy score of 88.4% T-1 accuracy and 98.7% T-5 accuracy respectively. This model consumed 480 Million parameters and which is approximately half the computational resource of the previous state-of-the-art by training the model with 300 unlabelled samples considered from the JFT dataset. The best performing model considered EfficentNet-L2 as the backbone to imply the NoisyStudent approach as mentioned in the Fig. 19. Further, the importance of adding a noise component in training the student model is discussed and evaluated. The training signal tends to vanish if the student samples were trained in a similar approach to that of a teacher by attaining zero cross-entropy loss. The T-1 accuracy obtained on ImageNet is 83.9%. This indeed shows large variation from the proposed method i.e., high variance from the current state-of-the-art. The co-training helps in segregating the two disjoint segments and training two models in a student-teacher self-supervised fashion helped in improving the performance to a greater extent.

O. BiT (Big Transfer)

Kolesnikov et al [93], performed transfer learning on large scale image recognition to improve tuning of hyperparameters and sample efficiency. The parameters are tuned cautiously by focusing on certain components for various vision tasks to improve performance with feature reproducibility. To provide greater performance transferability is provided on large scale vision tasks and performed transfer learning to produce three variant models BiT-Small, BiT-Medium and BiT-Large. The models were trained by fixing the architecture and varying the size of the data. Where small is performed on ILSVRC-2012 consisting of 1.2 million samples with 1000 classes. The medium is trained on full ImageNet with 14.2 million samples

with 21 thousand labels. Finally, the large utilized JFT dataset consisting of around 300 million samples and approximately 1.2 labels per sample. A set of tricks are considered by understanding certain components to attain higher performance for a neural network. They addressed two necessary components to build an effective neural architecture which is upstream and downstream components.

Upstream Components: Upstream components are implied for pre-training definite task. The components considered during up-stream pre-training are scale, Group normalization, and Weight standardization. Properly adjusting these components led to having a lower computational budget and greater efficiency. Further, group normalization and weight standardization obliged faster training over large batch structures.

Downstream Components: Whereas, Downstream components are applied for fine-tuning a similar visual task. In this, a heuristic rule is applied by discarding computationally expensive hyperparameters. Simple image pre-processing techniques such as resizing input to squared shape, cropping a short square randomly, and performing horizontal flip at training time. The parameters tuned while pre-training the model, at upstream and downstream are discussed independently. Most of the BiT models utilize ResNetV2 as backbone architecture to imply transferability. The upstream models utilize SGD as an optimizer and initializing the learning rate by 3x10-2. Additionally, a momentum of 0.9 was induced for faster convergence. The input samples were isotopically resized to 224x224 shape. Next, the small and medium models were trained with 90 epochs. But the training procedure was different as the learning rate was reduced by 10 after 30, 60 and 80 epochs. Subsequently, the large model was trained by decaying learning rate after 25%, 57.5%, 75% and 92.5% of the training progress. Similarly, for the downstream task, the SGD was implied as an optimizer with a learning initializer of 0.03 and to progress convergence, a momentum of 0.9 is added. The input shapes were reshaped appropriately to the context of the dataset. In a large scale visual classification challenge, the T1 accuracy obtained by the BiT-Large model on ImageNet-1K is 87.54% (with a standard deviation of 0.02). It remained a state-of-the-art model not only for ImageNet but also, for multiple standard data sets such as CIFAR-10, CIFAR-100, Pets, Flowers, VTAB. Further, the BiT was analysed on object detection, which implied RetinaNet as the backbone. This attained a state-of-the-art average precision of 43.8. With the BiT transferability, the object detection model attained an improvement of around 7.3%.

P. ViT (Visual Transformer)

Dosovitskiy et al. [94] utilized a transformer, the standard neural architecture for natural language processing onto computer vision task to drive self-attention for large scale visual recognition. This visual transformer was able to drive present state-of-the-art with lower computational cost to that of Convnets. The transformer is implied invariant fields depicting its performance. A Visual Transformer (ViT) is trained by appropriately setting the input embedding to the transformer to extract visual representations. The patch embeddings are obtained by resizing the image of a 2D image into sequential 2D patches. These embeddings are inserted into the transformer

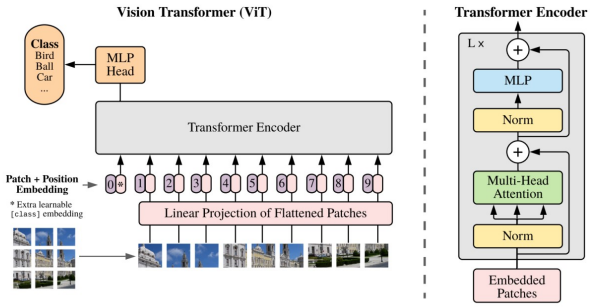

Fig. 20. The above Figure Illustrates the Architecture and the Patch Embedding of the Vision Transformer.

TABLE II. IT GIVES DETAILS ABOUT THE ACCURACY SCORES OBTAINED BY SOTA MODELS.

SOTA Methods	Top-1 accuracy	Top-5 accuracy
AlexNet	62.5	83
ZeNet	64.o	85.3
Overfeat	66.0	86.7
VGGNet	76.3	93.2
InceptionV2	79.9	95.1
InceptionV3	78.8 9	4.4
InceptionV4	80.1	95.1
ResNeXt	79.6	94.7
DPN	81.4	95.8
PNASNet	82.9	96.2
NASNet	82.7	96.2
NoisyStudent	88.4	98.7
EfficientNet	84.4	97.1
FixResNext	83.7	98.0
BiT	87.5	-
ViT	85.5	-

similar to that of BERT's model class token. The architecture and patch embeddings are visually described in the Fig. 20.

The ViT model and the models considered for comparison were trained on certain parameters. The optimizer implied is adam with an initial learning rate set to default (0.001). Further, the β,and β_2 were set as 0.9 and 0.999 respectively. A weight decay of 0.1 was applied and it helped to construct good performance. Further, to fine-tune the model was initiated with a batch size of 512 and the optimizer as SGD. A small momentum was applied to improve the training speed. A maximum dropout of 0.1 was used for the ViT model trained on a large ImageNet dataset. Self-attention is provided by the transformer helped to combine the features extraction at the lower layer on focusing on the definite set of entities residing in the image.

VI. SUGGESTIONS FOR ARCHITECTURE CONSTRUCTION

Observing the state-of-the-art literature in the convnets there are certain factors observed in the construction of a novel architecture with greater performance and lower computational cost. These certain factors are constructed by analysing the minute parameters providing a better model. The performance

of various SOTA models is produced in Table II.

A. Architecture Tips

The architecture tips fairly include all the factors influencing to develop a resilient architecture that extracts invariant features. Larger kernel size in the beginning layers of the convolution provides loss of information which degrades performance but, speeds the training process. Similarly, the higher the stride faster the model is trained but the accuracy degrades successively. Without adding residual connections developing a model just by increasing the depth can lead to the problem of degradation. A network architecture without bottleneck activations can explode in terms of computational cost hence, a set of bottleneck activations are to be implanted into the networks. The varying dimensionality of the receptive field can provide invariant features. An architecture trained on very small data cannot perform well on most of the unseen samples. Hence, solutions for these problems are explicitly provided for building a resilient convnet.

- A small receptive field provides a set of variant abstract features which carry detailed invariances.
- A smaller stride can eventually provide good representation by reducing the loss of the information through excessive pooling.
- To skip the problem of degradation, residual connections can be implied accordingly. Further, this improves the performance and also reduces the computational cost for deeper architectures
- A set of bottleneck connections can provide a generic feature representation and reduce computational effort while developing a convent width-wise.
- The asymmetric receptive fields with an appropriate bottleneck layer provide a greater representation of features.
- Finally, a model trained on multiple tasks with an appropriate set of samples can eventually improve in terms of performance acquiring state-of-the-art without much effort in parametric tuning.

B. Optimization Tips

The optimization tips include developing representation in convnets by altering the hyperparameters and indicating their right implementation. The hyperparameters which are widely implied in the deep learning paradigm to observe a conventional change in the model behaviour during stochastic optimization are described in detail.

- **Dropout**: Dropout helps in generalizing the model by halting a set of neurons during training and releasing them during the validation or testing time. Hence, selecting the percentage of dropout is crucial. According to the present implementations, most of the research implies 50%. But it can be varied from 30-50% and choosing it in this interval provides good generalisation is densely connected networks.
- **Normalization**: Local response normalization implemented in AlexNet did not perform well in most of the instances. As it has a huge number of hyperparameters it is a complicated task to imply such a normalization technique. Further, the Batch normalization technique was implied and it provided a great deal of succession in convnets by solving the problem of covariate shift. It is mostly utilized in the present research as it does not include very few parameters to tune and it works globally for variant architectures. Next, some problems were addressed in batch normalization and overridden by group normalization. It shows very minute performance variation when incurred on a smaller task but has a good variety when applied on large scale. Hence, group normalization can be used while developing a deeper model and for a small architecture batch normalization and group normalization works equivariantly.
- To skip the problem of degradation, residual connections can be implied accordingly. Further, this improves the performance and also reduces the computational cost for deeper architectures
- Lastly selecting optimizer and scheduling the learning rates is still tedious. Hence, most of the research imply SGD with varying learning rate based on the problem and varying momentum by observing the convergence. Hence, for building a small scale convnets Adam optimizer with small learning rates and high batch size is provided for good performance. Whereas, training a large-scale model the parameters might vary from the architecture and choice of dataset.

VII. Conclusion

A detailed survey regarding the previous state-of-the-art is conducted. Additionally, a section explicitly gives an intuition of developing a good model with high performance and less computational power. This illustrates developing resilient architecture by tuning specific hyperparameters which as insightful in developing deep models.

Further, a set of details are not mentioned in this survey are to be described and held as our future direction. There a variant model which is developed in between these high-performance models which are not mentioned in this work. A set of small-scale models which resolve the problems in convolutions (i.e. Capsule Networks) does not describe explicitly. A detailed set of implementation framework which can reduce the effort of the implicit utility of architectures is not provided. These are taken as a challenge for the successive research and designing a framework overhauling these problems is chosen as future scope of work.

Acknowledgment

We kindly thank the Department of IT for providing extensive support during the research. Further, we thank GITAM University for support and appropriate guidance.

References

[1] O'Mahony N. et al. (2020) Deep Learning vs. Traditional Computer Vision. In: Arai K., Kapoor S. (eds) Advances in Computer Vision. CVC 2019. Advances in Intelligent Systems and Computing, vol 943. Springer, Cham. https://doi.org/10.1007/978-3-030-17795-9_10.

[2] T. Kohonen, "The self-organizing map," Proceedings of the IEEE, vol. 78, no. 9, pp. 1464–1480, 1990.

[3] G. Hinton, L. Deng, D. Yu, G. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T. Sainath, and B. Kingsbury, "Deep Neural Networks for Acoustic Modeling in Speech Recognition: The Shared Views of Four Research Groups," IEEE Signal Processing Magazine, vol. 29, no. 6, pp. 82–97, Nov. 2012.

[4] Rabiner, Lawrence. "Fundamentals of speech recognition." Fundamentals of speech recognition (1993).

[5] Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K. and Kuksa, P., 2011. Natural language processing (almost) from scratch. Journal of machine learning research, 12(ARTICLE), pp.2493-2537.

[6] A. Graves, A. Mohamed, and G. Hinton, "Speech recognition with deep recurrent neural networks," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, May 2013.

[7] Stearns, S.D., 1985. of Aldapfive Signal Processing.

[8] Stanley, W.D., Dougherty, G.R., Dougherty, R. and Saunders, H., 1988. Digital signal processing.

[9] Bruderlin, A. and Williams, L., 1995, September. Motion signal processing. In Proceedings of the 22nd annual conference on Computer graphics and interactive techniques (pp. 97-104).

[10] Ye, Jong Chul, Yoseob Han, and Eunju Cha. "Deep convolutional framelets: A general deep learning framework for inverse problems." SIAM Journal on Imaging Sciences 11.2 (2018): 991-1048.

[11] He, Miao, and David He. "Deep learning based approach for bearing fault diagnosis." IEEE Transactions on Industry Applications 53.3 (2017): 3057-3065.

[12] He, M. and He, D., 2020. A new hybrid deep signal processing approach for bearing fault diagnosis using vibration signals. Neurocomputing, 396, pp.542-555.

[13] J. Pennington, R. Socher, and C. Manning, "Glove: Global Vectors for Word Representation," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014.

[14] G. A. Miller, "WordNet," Communications of the ACM, vol. 38, no. 11, pp. 39–41, Nov. 1995.

[15] Manning, C. and Schutze, H., 1999. Foundations of statistical natural language processing. MIT press.

[16] L. A. Zadeh, "Outline of a New Approach to the Analysis of Complex Systems and Decision Processes," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 1, pp. 28–44, 1973.

[17] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L. and Polosukhin, I., 2017. Attention is all, in "Advances in neural information processing systems 2017".

[18] Manning, C.D., Surdeanu, M., Bauer, J., Finkel, J.R., Bethard, S. and McClosky, D., 2014, June. The Stanford CoreNLP natural language processing toolkit. In Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations (pp. 55-60).

[19] S. BOCCALETTI, V. LATORA, Y. MORENO, M. CHAVEZ, and D. HWANG, "Complex networks: Structure and dynamics," Physics Reports, vol. 424, no. 4–5, pp. 175–308, Feb. 2006.

[20] Xu, K., Hu, W., Leskovec, J. and Jegelka, S., 2018. How powerful are graph neural networks? arXiv preprint arXiv:1810.00826.

[21] Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C. and Philip, S.Y., 2020. A comprehensive survey on graph neural networks. IEEE transactions on neural networks and learning systems.

[22] Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M. and Monfardini, G., 2008. The graph neural network model. IEEE transactions on neural networks, 20(1), pp.61-80.

[23] Qu, M., Bengio, Y. and Tang, J., 2019, May. Gmnn: Graph markov neural networks. In International conference on machine learning (pp. 5241-5250). PMLR.

[24] Dwivedi, V.P., Joshi, C.K., Laurent, T., Bengio, Y. and Bresson, X., 2020. Benchmarking graph neural networks. arXiv preprint arXiv:2003.00982.

[25] R. M. Haralick, K. Shanmugam, and I. Dinstein, "Textural Features for Image Classification," IEEE Transactions on Systems, Man, and Cybernetics, vol. SMC-3, no. 6, pp. 610–621, Nov. 1973.

[26] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971–987, Jul. 2002.

[27] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A Fast Learning Algorithm for Deep Belief Nets," Neural Computation, vol. 18, no. 7, pp. 1527–1554, Jul. 2006.

[28] Lu, D. and Weng, Q., 2007. A survey of image classification methods and techniques for improving classification performance. International journal of Remote sensing, 28(5), pp.823-870.

[29] Wang, F., Jiang, M., Qian, C., Yang, S., Li, C., Zhang, H., Wang, X. and Tang, X., 2017. Residual attention network for image classification. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3156-3164).

[30] Chapelle, O., Haffner, P. and Vapnik, V.N., 1999. Support vector machines for histogram-based image classification. IEEE transactions on Neural Networks, 10(5), pp.1055-1064.

[31] Dalal, N., & Triggs, B. (n.d.). Histograms of Oriented Gradients for Human Detection. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). doi:10.1109/cvpr.2005.177

[32] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision, 115(3), 211–252. doi:10.1007/s11263-015-0816-y.

[33] Viola, P., & Jones, M. (n.d.). Rapid object detection using a boosted cascade of simple features. Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition.

[34] Girshick, Ross. "Fast r-cnn." Proceedings of the IEEE international conference on computer vision. 2015.

[35] He, Kaiming, et al. "Mask r-cnn." Proceedings of the IEEE international conference on computer vision. 2017.

[36] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017.

[37] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool, "Speeded-Up Robust Features (SURF)," Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346–359, Jun. 2008.

[38] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common Objects in Context," Lecture Notes in Computer Science, pp. 740–755, 2014.

[39] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," International Journal of Computer Vision, vol. 88, no. 2, pp. 303–338, Sep. 2009.

[40] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object Detection with Discriminatively Trained Part-Based Models," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 32, no. 9, pp. 1627–1645, Sep. 2010.

[41] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2016.

[42] V. Caselles, R. Kimmel, and G. Sapiro, International Journal of Computer Vision, vol. 22, no. 1, pp. 61–79, 1997.

[43] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," 2012 IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2012.

[44] Toshev, A. and Szegedy, C., 2014. Deeppose: Human pose estimation via deep neural networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1653-1660).

[45] Murphy-Chutorian, E. and Trivedi, M.M., 2008. Head pose estimation in computer vision: A survey. IEEE transactions on pattern analysis and machine intelligence, 31(4), pp.607-626.

[46] Cao, Z., Simon, T., Wei, S.E. and Sheikh, Y., 2017. Realtime multi-person 2d pose estimation using part affinity fields. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7291-7299).

[47] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," CVPR 2011, Jun. 2011.

[48] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," Computer Vision and Image Understanding, vol. 104, no. 2–3, pp. 90–126, Nov. 2006.

[49] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015, pp. 234–241, 2015.

[50] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Jun. 2015.

[51] Jianbo Shi and J. Malik, "Normalized cuts and image segmentation," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8, pp. 888–905, 2000.

[52] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Jun. 2014.

[53] Haralick, R.M. and Shapiro, L.G., 1985. Image segmentation techniques. Computer vision, graphics, and image processing, 29(1), pp.100-132.

[54] Zhang, Y.J., 1996. A survey on evaluation methods for image segmentation. Pattern recognition, 29(8), pp.1335-1346.

[55] Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C.L. and Parikh, D., 2015. Vqa: Visual question answering. In Proceedings of the IEEE international conference on computer vision (pp. 2425-2433).

[56] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization," 2017 IEEE International Conference on Computer Vision (ICCV), Oct. 2017.

[57] D. Kahneman and D. T. Miller, "Norm theory: Comparing reality to its alternatives.," Psychological Review, vol. 93, no. 2, pp. 136–153, Apr. 1986.

[58] V. D. Calhoun, T. Adali, G. D. Pearlson, and J. J. Pekar, "A method for making group inferences from functional MRI data using independent component analysis," Human Brain Mapping, vol. 14, no. 3, pp. 140–151, 2001.

[59] R. Krishna, Y. Zhu, O. Groth, J. Johnson, K. Hata, J. Kravitz, S. Chen, Y. Kalantidis, L.-J. Li, D. A. Shamma, M. S. Bernstein, and L. Fei-Fei, "Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations," International Journal of Computer Vision, vol. 123, no. 1, pp. 32–73, Feb. 2017.

[60] P. Anderson, X. He, C. Buehler, D. Teney, M. Johnson, S. Gould, and L. Zhang, "Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Jun. 2018.

[61] LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. Nature, 521(7553), 436–444. doi:10.1038/nature14539.

[62] Goodfellow, Ian, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. Deep learning. Vol. 1, no. 2. Cambridge: MIT press, 2016.

[63] LeCun, Y., Boser, B., Denker, J.S., Henderson, D., Howard, R.E., Hubbard, W. and Jackel, L.D., 1989. Backpropagation applied to handwritten zip code recognition. Neural computation, 1(4), pp.541-551.

[64] LeCun, Yann, Y. Bengio et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.

[65] Dumoulin, V. and Visin, F., 2016. A guide to convolution arithmetic for deep learning. arXiv preprint arXiv:1603.07285.

[66] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." Advances in neural information processing systems 25 (2012): 1097-1105.

[67] Nair, V. and Hinton, G.E., 2010, January. Rectified linear units improve restricted boltzmann machines. In ICML.

[68] Jarrett, Kevin, KorayKavukcuoglu, Marc'AurelioRanzato, and Yann LeCun. "What is the best multi-stage architecture for object recognition?." In 2009 IEEE 12th international conference on computer vision, pp. 2146-2153. IEEE, 2009.

[69] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R., 2014. Dropout: a simple way to prevent neural networks from overfitting. The journal of machine learning research, 15(1), pp.1929-1958.

[70] Zeiler M.D., Fergus R. (2014) Visualizing and Understanding Convolutional Networks. In: Fleet D., Pajdla T., Schiele B., Tuytelaars T. (eds) Computer Vision – ECCV 2014. ECCV 2014. Lecture Notes in Computer Science, vol 8689. Springer, Cham.

[71] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[72] He, K., Zhang, X., Ren, S. and Sun, J., 2016, October. Identity mappings in deep residual networks. In European conference on computer vision (pp. 630-645). Springer, Cham.

[73] G. Huang, Z. Liu, L. Van Der Maaten and K. Q. Weinberger, "Densely Connected Convolutional Networks," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 2261-2269, doi: 10.1109/CVPR.2017.243.

[74] G. Huang, Z. Liu, G. Pleiss, L. Van Der Maaten and K. Weinberger, "Convolutional Networks with Dense Connectivity," in IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2019.2918284.

[75] Sermanet, Pierre, et al. "Overfeat: Integrated recognition, localization and detection using convolutional networks. 2nd international conference on learning representations, iclr 2014." 2nd International Conference on Learning Representations, ICLR 2014 (2014).

[76] Simonyan, Karen, and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In ICLR 2015.

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

[78] Sergey Ioffe. 2017. Batch renormalization: towards reducing minibatch dependence in batch-normalized models. In Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17). Curran Associates Inc., Red Hook, NY, USA, 1942–1950.

[79] Huang, L., Yang, D., Lang, B. and Deng, J., 2018. Decorrelated batch normalization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 791-800).

[80] Ioffe, Sergey, and Christian Szegedy. "Batch normalization: Accelerating deep network training by reducing internal covariate shift." International conference on machine learning. PMLR, 2015.

[81] Szegedy, Christian, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. "Rethinking the inception architecture for computer vision." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 2818-2826. 2016.

[82] Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. "Inception-v4, inception-resnet and the impact of residual connections on learning." In Proceedings of the AAAI Conference on Artificial Intelligence, vol. 31, no. 1. 2017.

[83] Ruder, S., 2016. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

[84] Xie, Saining, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. "Aggregated residual transformations for deep neural networks." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 1492-1500. 2017.

[85] Chen, Yunpeng, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, and Jiashi Feng. "Dual path networks." arXiv preprint arXiv:1707.01629 (2017).

[86] Zoph, Barret, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. "Learning transferable architectures for scalable image recognition." In Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 8697-8710. 2018.

[87] Bello, I., Zoph, B., Vasudevan, V. and Le, Q.V., 2017, July. Neural optimizer search with reinforcement learning. In International Conference on Machine Learning (pp. 459-468). PMLR.

[88] Bergstra, J. and Bengio, Y., 2012. Random search for hyper-parameter optimization. Journal of machine learning research, 13(2).

[89] Liu, Chenxi, Barret Zoph, Maxim Neumann, Jonathon Shlens, Wei Hua, Li-Jia Li, Li Fei-Fei, Alan Yuille, Jonathan Huang, and Kevin Murphy. "Progressive neural architecture search." In Proceedings of the European conference on computer vision (ECCV), pp. 19-34. 2018.

[90] Tan, M. and Le, Q., 2019, May. Efficientnet: Rethinking model scaling for convolutional neural networks. In International Conference on Machine Learning (pp. 6105-6114). PMLR.

[91] Touvron, Hugo, Andrea Vedaldi, Matthijs Douze, and Hervé Jégou. "Fixing the train-test resolution discrepancy." In NeurIPS 2019.

[92] Xie, Qizhe, Minh-Thang Luong, Eduard Hovy, and Quoc V. Le. "Self-training with noisy student improves imagenet classification." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 10687-10698. 2020.

[93] Kolesnikov A. et al. (2020) Big Transfer (BiT): General Visual Representation Learning. In: Vedaldi A., Bischof H., Brox T., Frahm JM. (eds) Computer Vision – ECCV 2020. ECCV 2020. Lecture Notes in Computer Science, vol 12350. Springer, Cham.

[94] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn et al., "An image is worth 16X16 words: Transformers for image recognition at scale,". In ICLR, 2021.