Chapter 16

An Introduction to Deep Learning

Machine learning enabled computers fed on real life raw data or examples and it tries to extract patterns from it and make better decisions by itself. Some of the machine learning algorithms are logistic regression, naive bayes, SVM etc.

The performance of these machine learning algorithms depends heavily on the representation of the data that are given. Each piece of information included in the representation is known as features and these algorithms learns how to use these features to extract patterns or to get knowledge.

But sometimes it's difficult to know what features should be extracted. For example, suppose we wants to detect cars from an image, now we might like to use the presence of wheel as a feature. But it is difficult to describe what a wheel looks like in terms of pixel values. One solution to this problem is to use machine learning to discover not only output from those features(representation) but the features itself. This approach is known as representation learning. Again it is much better if algorithm learns features by itself with minimal human intervention.

While designing these algorithms our goal is usually to separate the factors of variation. Now these factors are not always directly observed they might be unobserved factors also that can affect our algorithm. Now of course it can be difficult to extract high-level, abstract features from raw data like speaker's accent in case of voice recognition because these can be identified by only sophisticated human level understanding of the data.

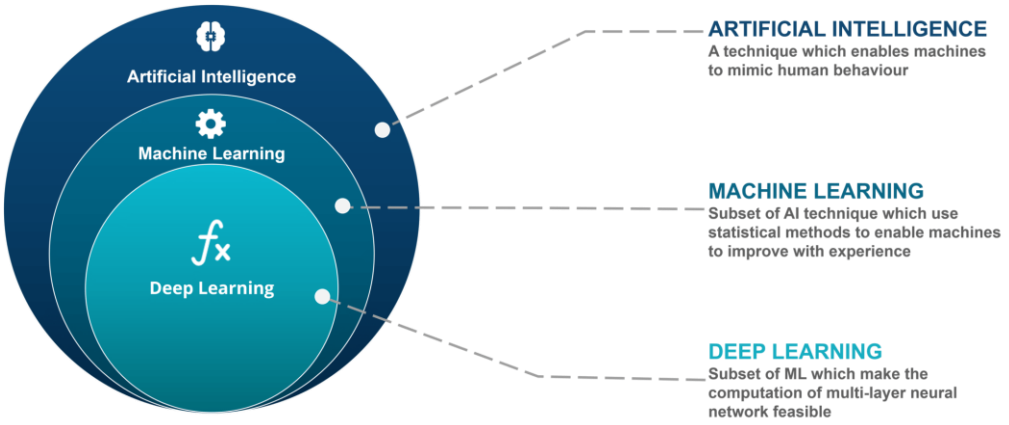
11.1 AI vs Machine Learning vs Deep Learning

Deep learning is a specific subset of Machine Learning, which is a specific subset of Artificial Intelligence. For individual definitions:

Artificial Intelligence: Artificial Intelligence is a technique which allows the machines to act like humans by replicating their behaviour and nature.

Machine Learning: Machine Learning is a subset of artificial intelligence. It allows the machines to learn and make predictions based on its experience(data).

Deep learning: Deep learning is a particular kind of machine learning that achieves great power and flexibility by learning to represent the world as nested hierarchy of concepts or abstraction.



11.2 Why is Deep Learning Important?

Deep Learning is important for one reason, and one reason only: we've been able to achieve meaningful, useful accuracy on tasks that matter. Machine Learning has been used for classification on images and text for decades, but it struggled to cross the threshold – there's a baseline accuracy that algorithms need to have to work in business settings. Deep Learning is finally enabling us to cross that line in places we weren't able to before.

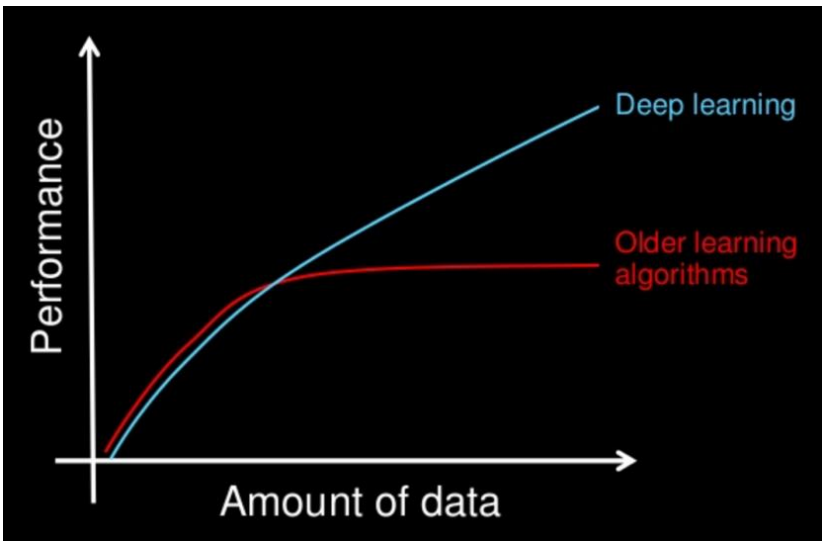
Computer vision is a great example of a task that Deep Learning has transformed into something realistic for business applications. Using Deep Learning to classify and label images isn't only better than any other traditional algorithms: it's starting to be better than actual humans.

Facebook has had great success with identifying faces in photographs by using Deep Learning. It's not just a marginal improvement, but a game changer: "Asked whether two unfamiliar photos of faces show the same person, a human being will get it right 97.53 percent of the time. New software developed by researchers at Facebook can score 97.25 percent on the same challenge, regardless of variations in lighting or whether the person in the picture is directly facing the camera."

Speech recognition is an another area that has felt Deep Learning's impact. Spoken languages are so vast and ambiguous. Baidu – one of the leading search engines of China – has developed a voice recognition system that is faster and more accurate than humans at producing text on a mobile phone; in both English and Mandarin.

Google is now using deep learning to manage the energy at the company's data centers. They've cut their energy needs for cooling by 40%. That translates to about a 15% improvement in power usage efficiency for the company and hundreds of millions of dollars in savings.

Deep Learning is important because it finally makes these tasks accessible – it brings previously irrelevant workloads into the purview of Machine Learning. We're just at the cusp of developers and business leaders understanding how they can use Machine Learning to drive business outcomes, and having more available tasks at your fingertips because of Deep Learning is going to transform the economy for decades to come.

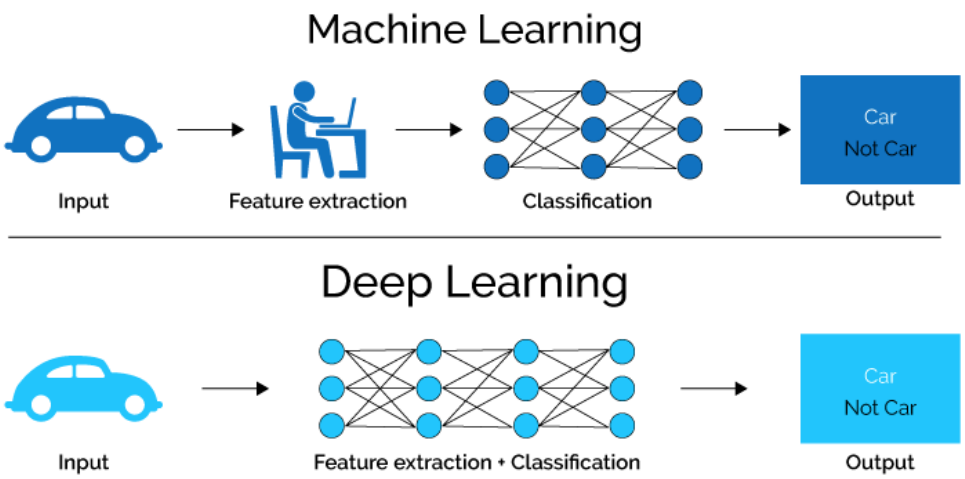


11.1 Deep Learning

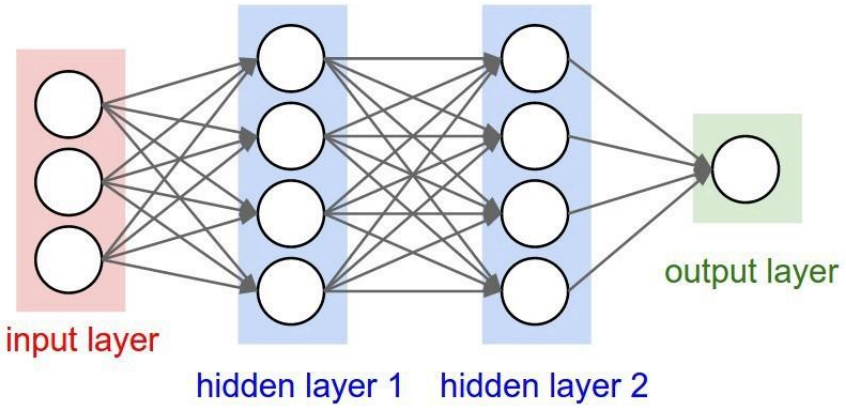
Deep learning is a sub-field of machine learning dealing with algorithms inspired by the structure and function of the brain called artificial neural networks. In other words, it mirrors the functioning of our brains. Deep learning algorithms are similar to how the nervous system structured where each neuron connected each other and passing information.

Deep learning models work in layers and a typical model at least have three layers. Each layer accepts the information from previous and pass it on to the next one.

Deep Learning can solve problem in representation learning by introducing representation that are expressed in terms of other, simpler representation.



How Do Deep Learning algorithms "learn"?
Deep Learning Algorithms use neural network to find associations between a set of inputs and outputs. The basic structure is seen below:



What kind of problems does deep learning solve?
deep neural networks excel at making predictions based on largely unstructured data. That means they deliver best in class performance in areas such as speech and image recognition, where they work with messy data such as recorded speech and photographs.

Should you use always deep learning instead of machine learning?
No, because deep learning can be very expensive from a computational point of view. For non-trivial tasks, training a deep-neural network will often require processing large amounts of data using clusters of high-end GPUs for many, many hours.

Given top-of-the-range GPUs can cost thousands of dollars to buy, or up to $5 per hour to rent in the cloud, it's unwise to jump straight to deep learning.

If the problem can be solved using a simpler machine-learning algorithm such as Bayesian inference or linear regression, one that doesn't require the system to grapple with a complex combination of hierarchical features in the data, then these far less computational demanding options will be the better choice.

Deep learning may also not be the best choice for making a prediction based on data. For example, if the dataset is small then sometimes simple linear machine-learning models may yield more accurate results—although some machine-learning specialists argue a properly trained deep-learning neural network can still perform well with small amounts of data.

What deep learning techniques exist?

There are various types of deep neural network, with structures suited to different types of tasks. For example, Convolutional Neural Networks (CNNs) are typically used for computer vision tasks, while Recurrent Neural Networks (RNNs) are commonly used for processing language. Each has its own specializations, in CNNs the initial layers are specialized for extracting distinct features from the image, which are then fed into a more conventional neural network to allow the image to be classified. Meanwhile, RNNs differ from a traditional feed-forward neural network in that they don't just feed data from one neural layer to the next but also have built-in feedback loops, where data output from one layer is passed back to the layer preceding it—lending the network a form of memory. There is a more specialized form of RNN that includes what is called a memory cell and that is tailored to processing data with lags between inputs.

There are a large number of different types of deep neural networks. No one network is inherently better than the other, they just are better suited to learning particular types of tasks.

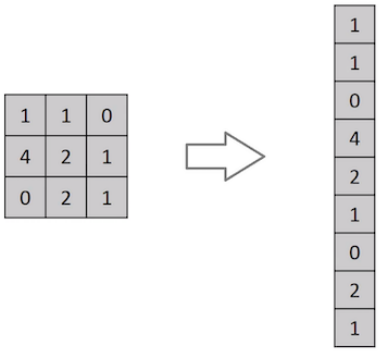
Convolutional Neural Network (ConvNet/CNN)

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.

Why ConvNets over Feed-Forward Neural Nets?

An image is nothing but a matrix of pixel values. Why not just flatten the image (e.g. 3x3 image matrix into a 9x1 vector) and feed it to a Multi-Level Perceptron for classification purposes?

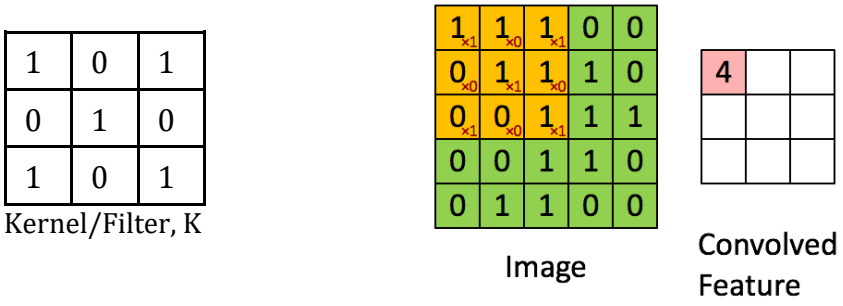


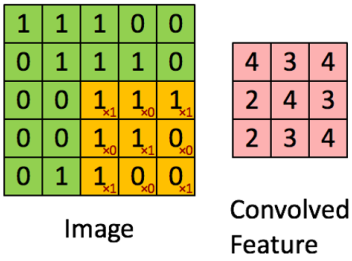
In cases of extremely basic binary images, the method might show an average precision score while performing prediction of classes but would have little to no accuracy when it comes to complex images having pixel dependencies throughout.

A ConvNet is able to successfully capture the Spatial and Temporal dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better.

Convolution Layer — The Kernel

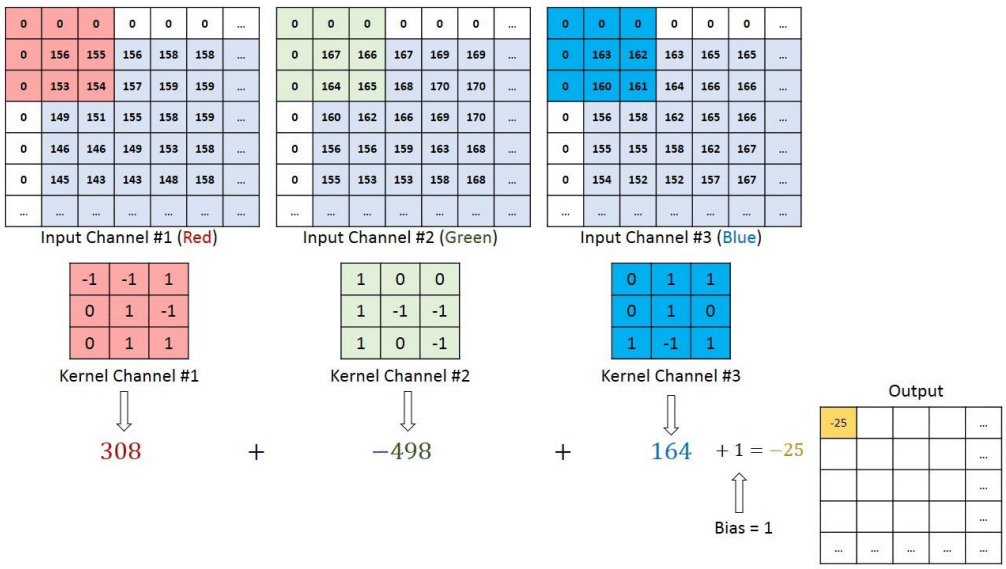
Consider a 5x5x1 input image, I. The element involved in carrying out the convolution operation in the first part of a Convolutional Layer is called the Kernel/Filter, K, represented in the color yellow. We have selected K as a 3x3x1 matrix.





The Kernel shifts 9 times because of Stride Length = 1 (Non-Strided), every time performing a matrix multiplication operation between K and the portion P of the image over which the kernel is hovering.

The filter moves to the right with a certain Stride Value till it parses the complete width. Moving on, it hops down to the beginning (left) of the image with the same Stride Value and repeats the process until the entire image is traversed.



Convolution operation on a MxNx3 image matrix with a 3x3x3 Kernel

In the case of images with multiple channels (e.g. RGB), the Kernel has the same depth as that of the input image. Matrix Multiplication is performed between Kn and In stack ([K1, I1]; [K2, I2]; [K3, I3]) and all the results are summed with the bias to give us a squashed one-depth channel Convoluted Feature Output.

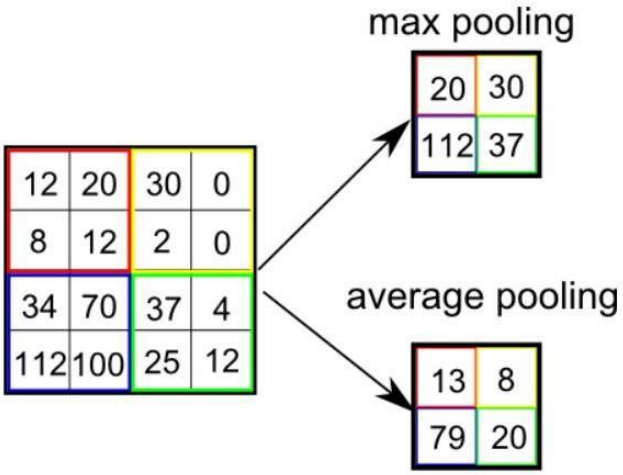
The objective of the Convolution Operation is to extract the high-level features such as edges, from the input image. ConvNets need not be limited to only one Convolutional Layer. Conventionally, the first ConvLayer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the

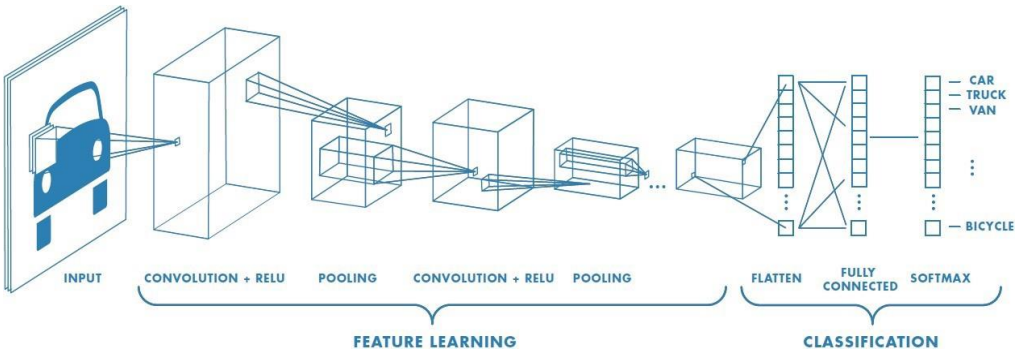
High-Level features as well, giving us a network which has the wholesome understanding of images in the dataset, similar to how we would.

Pooling Layer

Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are two types of Pooling: *Max Pooling* and *Average Pooling*. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel.
Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling.





The Convolutional Layer and the Pooling Layer, together form the i-th layer of a Convolutional Neural Network. Depending on the complexities in the images, the number of such layers may be increased for capturing low-levels details even further, but at the cost of more computational power. After going through the above process, we have successfully enabled the model to understand the features. Moving on, we are going to flatten the final output and feed it to a regular Neural Network for classification purposes.

Classification — Fully Connected Layer (FC Layer)
Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space.

Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify them using the Softmax Classification technique.

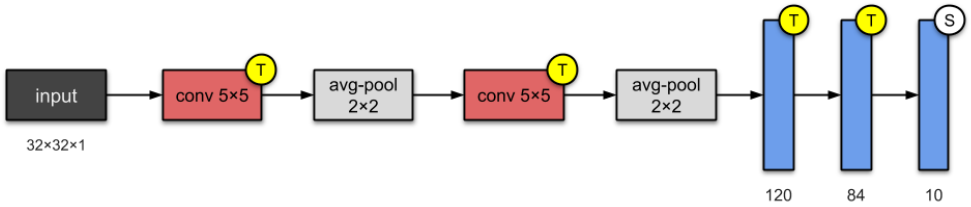
CNNs Architecture

There are various architectures of CNNs available which have been key in building algorithms which power and shall power AI as a whole in the foreseeable future. Some of them have been listed below:

- LeNet
- AlexNet
- VGGNet
- GoogLeNet
- ResNet
- ZFNet

LeNet-5 CNN Architecture (1998)
LeNet-5 is one of the simplest architectures. It has 2 convolutional and 3 fully-connected layers (hence “5” — it is very common for the names of neural networks to be derived from the number of convolutional and fully connected layers that they have). The average-pooling layer as we know it now was called a sub-sampling layer and it had trainable weights (which isn’t the current practice of designing CNNs nowadays). This architecture has about 60,000 parameters.

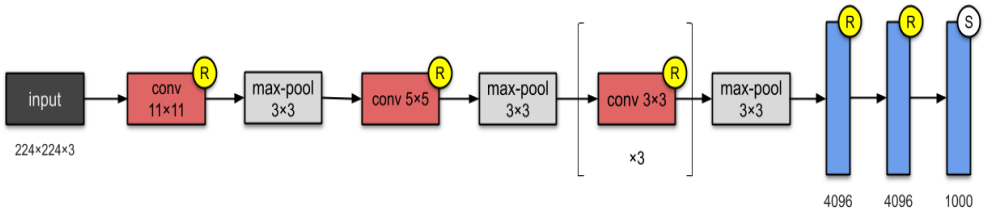
This architecture has become the standard 'template': stacking convolutions and pooling layers, and ending the network with one or more fully-connected layers.



AlexNet CNN Architecture (2012)

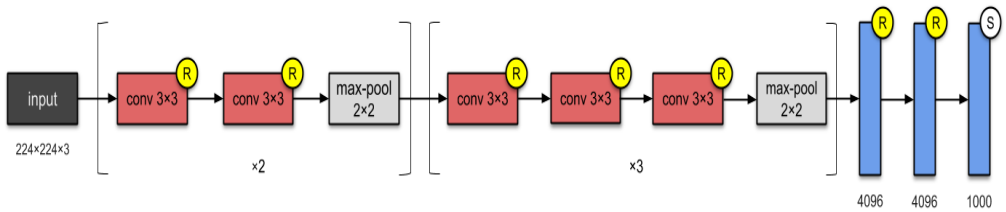
AlexNet has 8 layers — 5 convolutional and 3 fully-connected. AlexNet just stacked a few more layers onto LeNet-5. At the point of publication, the authors pointed out that their architecture was "one of the largest convolutional neural networks to date on the subsets of ImageNet."

They were the first to implement Rectified Linear Units (ReLUs) as activation functions.



VGG-16 CNN Architecture (2014)

CNNs were starting to get deeper and deeper. This is because the most straightforward way of improving performance of deep neural networks is by increasing their size (Szegedy et. al). The folks at Visual Geometry Group (VGG) invented the VGG-16 which has 13 convolutional and 3 fully-connected layers, carrying with them the ReLU tradition from AlexNet. This network stacks more layers onto AlexNet, and use smaller size filters (2×2 and 3×3). It consists of 138M parameters and takes up about 500MB of storage space. They also designed a deeper variant, VGG-19.



Bibliography

https://towardsdatascience.com/