Deep Learning for Computer Vision

The Abridged Guide

GUIDE CATEGORIES

Deep Learning for Computer Vision

Computer vision (CV) is the scientific field which defines how machines interpret the meaning of images and videos. Computer vision algorithms analyze certain

ns to predictive or

RELATED ARTICLES

Deep Learning for Computer Vision

Deep Convolutional Neural Networks

PyTorch CNN

PyTorch GAN: Understanding GAN and Coding it in PyTorch

PyTorch ResNet

TensorFlow CNN

Today, deep learning techniques are most commonly used for computer vision. This article explores different ways you can use deep learning for computer vision. In particular, you will learn about the advantages of using convolutional neural networks (CNNs), which provide a multi-layered architecture that allows neural networks to focus on the most relevant features in the image.

This is part of an extensive series of guides about AI Technology.

In this article, you will learn:

What is Computer Vision (CV)?

Computer vision is an area of machine learning dedicated to interpreting and understanding images and video. It is used to help teach computers to “see” and to use visual information to perform visual tasks that humans can.

Computer vision models are designed to translate visual data based on features and contextual information identified during training. This enables models to interpret images and video and apply those interpretations to predictive or decision making tasks.

Although both related to visual data, image processing is not the same as computer vision. Image processing involves modifying or enhancing images to produce a new result. It can include optimizing brightness or contrast, increasing resolution, blurring sensitive information, or cropping. The difference between image processing and computer vision is that the former doesn’t necessarily

Convolutional Neural Networks: The Foundation of Modern Computer Vision

Modern computer vision algorithms are based on convolutional neural networks (CNNs), which provide a dramatic improvement in performance compared to traditional image processing algorithms.

CNNs are neural networks with a multi-layered architecture that is used to gradually reduce data and calculations to the most relevant set. This set is then compared against known data to identify or classify the data input.

CNNs are typically used for computer vision tasks although text analytics and audio analytics can also be performed. One of the first CNN architectures was AlexNet (described below), which won the ImageNet visual recognition challenge in 2012.

How CNNs work

When an image is processed by a CNN, each base color used in the image (e.g. red, green, blue) is represented as a matrix of values. These values are evaluated and condensed into 3D tensors (in the case of color images), which are collections of stacks of feature maps tied to a section of the image.

These tensors are created by passing the image through a series of convolutional and pooling layers, which are used to extract the most relevant data from an image segment and condense it into a smaller, representative matrix. This process is repeated numerous times (depending on the number of convolutional

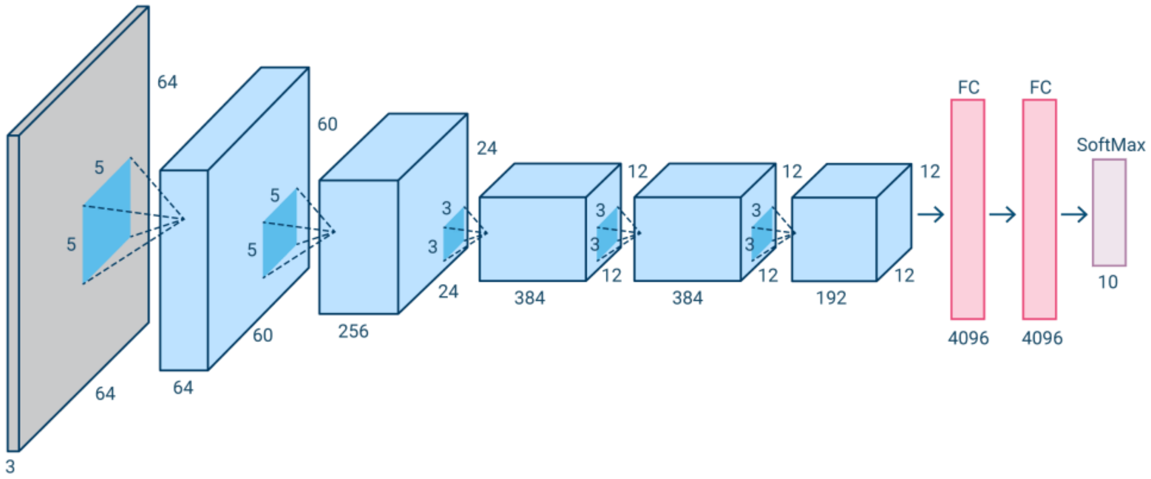
the convolutional

es predictions.

Deep Learning Architectures for Computer Vision

The performance and efficiency of a CNN is determined by its architecture. This includes the structure of layers, how elements are designed, and which elements are present in each layer. Many CNNs have been created, but the following are some of the most effective designs.

AlexNet (2012)



Alexnet

AlexNet is an architecture based on the earlier LeNet architecture. It includes five convolutional layers and three fully connected layers. AlexNet uses a dual pipeline structure to accommodate the use of two GPUs during training.

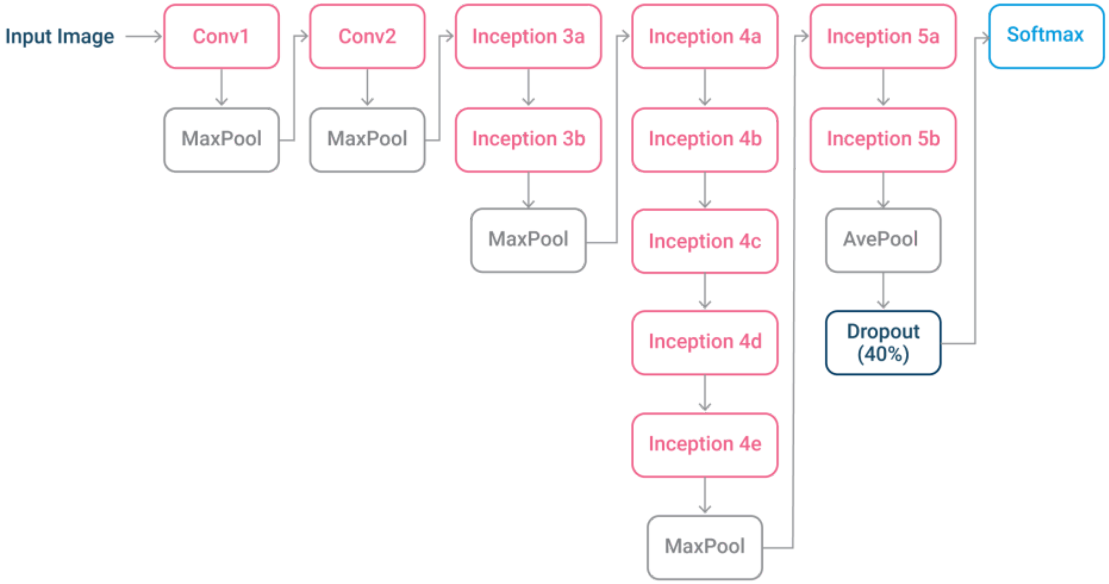
Learn more in our GPU guide, which reviews the best GPUs for deep

tectures is its use of

tivation functions which

were used in traditional neural networks. ReLU is simpler and faster to compute, enabling AlexNet to train models faster.

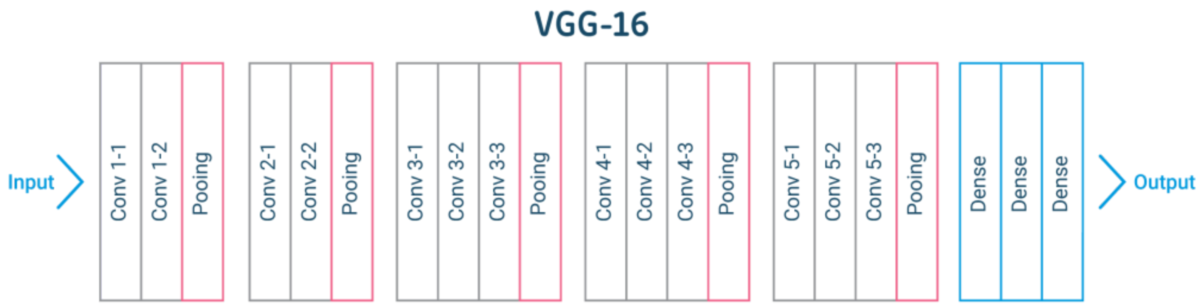
GoogleNet (2014)



Googlenet

GoogleNet, also known as Inception V1, is based on the LeNet architecture. It is made up of 22 layers made up of small groups of convolutions, called "inception modules". These inception modules use batch normalization and RMSprop to reduce the number of parameters GoogleNet needs to process. RMSprop is an algorithm that uses adaptive learning rate methods.

VGGNet (2014)

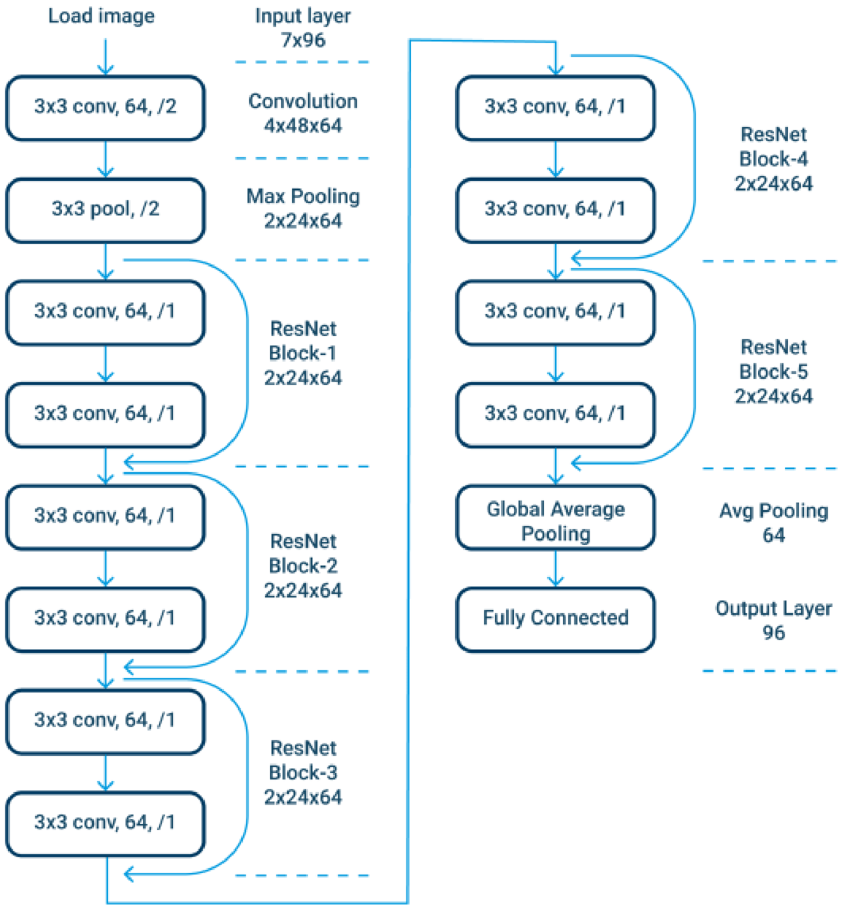


VGGNet

VGG 16 is a 16 layer architecture (some variants had 19 layers). VGGNet has convolutional layers, a pooling layer, a few more convolutional layers, a pooling layer, several more conv layers and so on.

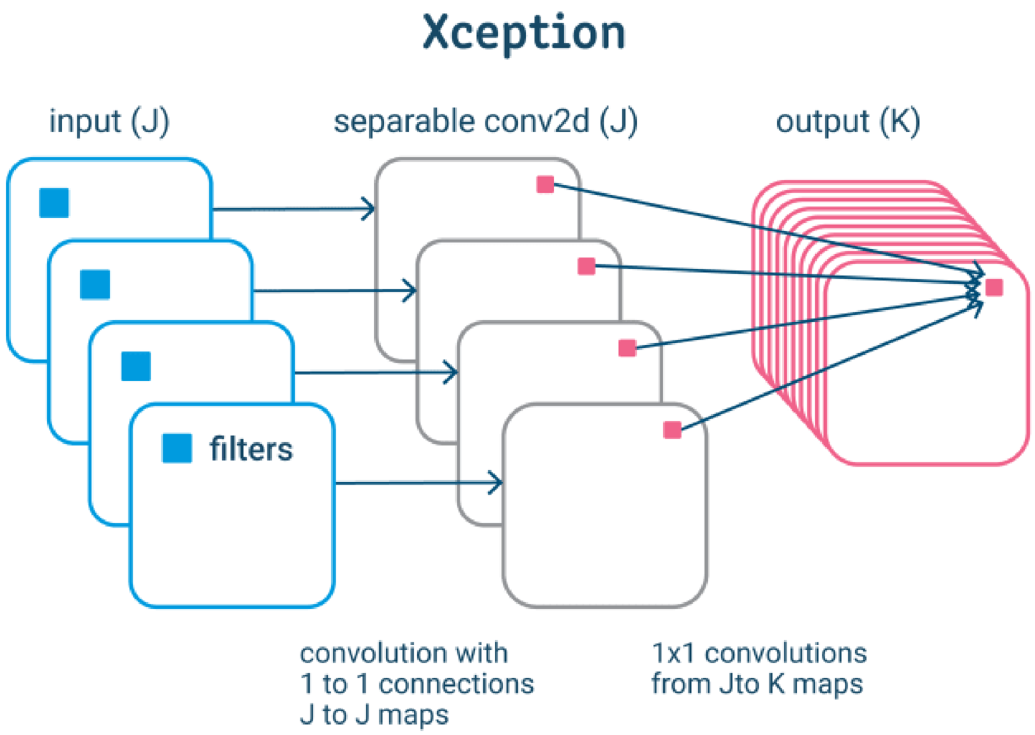
VGG is based on the notion of a much deeper network with smaller filters – it uses 3×3 convolutions all the way, which is the smallest conv filter size that only looks at some of the neighbouring pixels. It uses small filters because of fewer parameters, making it possible to add more layers. It has the same effective receptive field as if you have one 7×7 convolutional layer.

ResNet (2015)



ResNet

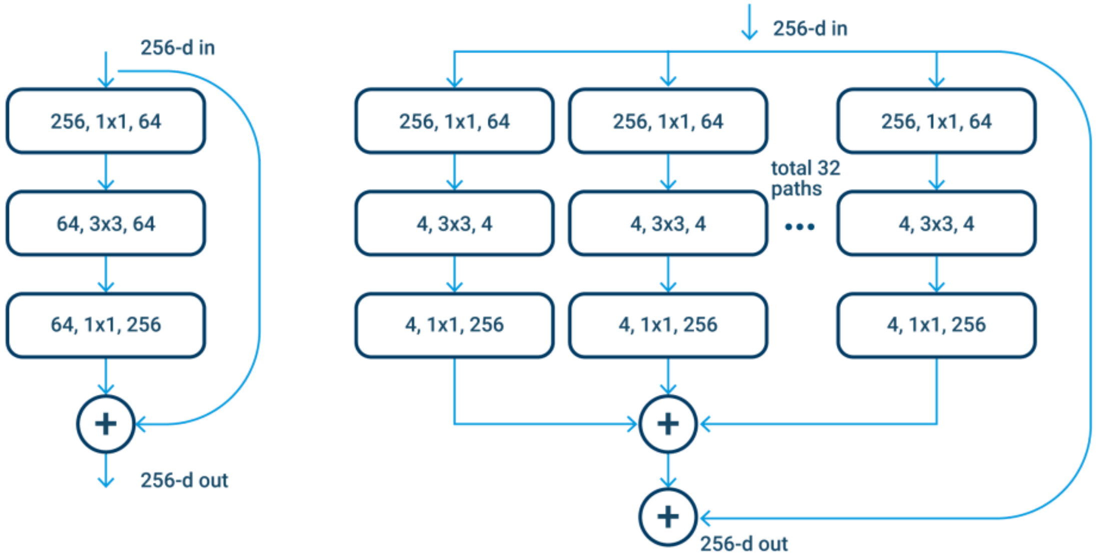
ResNet, short for Residual Neural Network, is an architecture designed to have a large number of layers – typically used architectures range from ResNet-18 (with 18 layers) to ResNet-1202 (with 1202 layers).These layers are set up with gated units or "skip connections" which enable it to pass information to later convolutional layers. ResNet also employs batch normalization to improve the stability of the network.



Xception

Xception is an architecture based on Inception, that replaces the inception modules with depthwise separable convolutions (depthwise convolution followed by pointwise convolutions). It works by first capturing cross-feature map correlations and then spatial correlations. This enables more efficient use of model parameters.

ResNeXt-50 (2017)



ResNeXt-50

ResNeXt-50 is an architecture based on modules with 32 parallel paths. It uses cardinality to decrease validation errors and represents a simplification of the inception modules used in other architectures.

Uses of Deep Learning in Computer Vision

The development of deep learning technologies has enabled the creation of more accurate and complex computer vision models. As these technologies increase, the incorporation of computer vision applications is becoming more useful. Below are a few ways deep learning is being used to improve computer vision.

ned via computer vision

- **Two-step object detection** – the first step requires a Region Proposal Network (RPN), providing a number of candidate regions that may contain important objects. The second step is passing region proposals to a neural classification architecture, commonly an RCNN-based hierarchical grouping algorithm, or region of interest (ROI) pooling in Fast RCNN. These approaches are quite accurate, but can very slow.
- **One-step object detection** – with the need for real time object detection, one-step object detection architectures have emerged, such as YOLO, SSD, and RetinaNet. These combine the detection and classification step, by regressing bounding box predictions. Every bounding box is represented with just a few coordinates, making it easier to combine the detection and classification step and speed up processing.

Localization and object detection

Image localization is used to determine where objects are located in an image. Once identified, objects are marked with a bounding box. Object detection extends on this and classifies the objects that are identified. This process is based on CNNs such as AlexNet, Fast RCNN, and Faster RCNN.

Localization and object detection can be used to identify multiple objects in complex scenes. This can then be applied to functionalities such as interpreting diagnostic images in medicine.

Semantic segmentation

Semantic segmentation, also known as object segmentation, is similar to object detection except it is based on the specific pixels related to an object. This

does not require

ned using fully

One popular use for semantic segmentation is for training autonomous vehicles. With this method, researchers can use images of streets or throughways with accurately defined boundaries for objects.

Pose estimation

Pose estimation is a method that is used to determine where joints are in a picture of a person or an object and what the placement of those joints indicates. It can be used with both 2D and 3D images. The primary architecture used for pose estimation is PoseNet, which is based on CNNs.

Pose estimation is used to determine where parts of the body may show up in an image and can be used to generate realistic stances or motion of human figures. Often, this functionality is used for augmented reality, mirroring movements with robotics, or gait analysis.

Deep Learning for Computer Vision at Large Scale With Run:ai

Computer vision algorithms are highly compute-intensive, and may require multiple GPUs to run at production scale. Run:ai automates resource management and workload orchestration for machine learning infrastructure. With Run:ai, you can automatically run as many compute intensive experiments as needed.

Here are some of the capabilities you gain when using Run:ai:

resource sharing by

- **No more bottlenecks**—you can set up guaranteed quotas of GPU resources, to avoid bottlenecks and optimize billing.
- **A higher level of control**—Run:ai enables you to dynamically change resource allocation, ensuring each job gets the resources it needs at any given time.

Run:ai simplifies machine learning infrastructure pipelines, helping data scientists accelerate their productivity and the quality of their deep learning models.

Learn more about the Run:ai GPU virtualization platform.

Learn More About Deep Learning for Computer Vision

TensorFlow CNN: Building Your First CNN with Tensorflow

Convolutional Neural Networks (CNN), a key technique in deep learning for computer vision, are little-known to the wider public but are the driving force behind major innovations, from unlocking your phone with face recognition to safe driverless vehicles.

Get a quick review of TensorFlow CNN concepts, and follow a quick tutorial to create your first CNN on TensorFlow, using the MNIST-Fashion dataset.

Read more: TensorFlow CNN: Building Your First CNN with Tensorflow

PyTorch ResNet: The Basics and a Quick Tutorial

ResNets are a common neural network architecture used for deep learning computer vision applications like object detection and image segmentation.

itecture, and see how to
code examples.

Read more: PyTorch ResNet: The Basics and a Quick Tutorial

Understanding Deep Convolutional Neural Networks

Deep learning is a machine learning technique used to build artificial intelligence (AI) systems. It is based on the idea of artificial neural networks (ANN), designed to perform complex analysis of large amounts of data by passing it through multiple layers of neurons.

Understand what deep convolutional neural networks (CNN or DCNN) are, what types exist, and what business applications the networks are best suited for.

Read more: Understanding Deep Convolutional Neural Networks

PyTorch CNN: The Basics and a Quick Tutorial

PyTorch is a Python framework for deep learning that makes it easy to perform research projects, leveraging CPU or GPU hardware. The basic logical unit in PyTorch is a tensor, a multidimensional array. PyTorch combines large numbers of tensors into computational graphs, and uses them to construct, train and run neural network architectures.

Learn about PyTorch, how convolutional neural networks work, and follow a quick tutorial to build a simple CNN in PyTorch, train it and evaluate results.

Read more: PyTorch CNN: The Basics and a Quick Tutorial

PyTorch GAN: Understanding GAN and Coding it in PyTorch

A generative adversarial network (GAN) uses two neural networks, called a generator and discriminator, to generate synthetic data that can convincingly mimic real data. For example, GAN architectures can generate fake, photorealistic pictures of animals or people.

earn how to code a

Read more: PyTorch GAN: Understanding GAN and Coding it in PyTorch

Distributed Training: What It Is and How It Can Be Valuable

Training deep learning models takes time. Deep neural networks often consist of millions or billions of parameters that are trained over huge datasets. As deep learning models become more complex, computation time can become unwieldy. Training a model on a single GPU can take weeks.

Take a deep dive into Distributed Training and how it can speed up the process of training deep learning models on GPUs.

Read more: Distributed Training: What It Is and How It Can Be Valuable

See Our Additional Guides on Key AI Technology Topics

Together with our content partners, we have authored in-depth guides on several other topics that can also be useful as you explore the world of AI Technology.

CUDA NVIDIA

- CUDA Programming: An In-Depth Look
- CUDA vs OpenCL
- NVIDIA cuDNN: Fine-Tuning GPU Performance for Neural Networks

Deep Learning GPU

- Best GPU for Deep Learning: Critical Considerations for Large-Scale AI

nd in the Cloud

Machine Learning Engineer

- Machine Learning Infrastructure: Components of Effective Pipelines
- Machine Learning Automation: Speeding Up the Data Science Pipeline
- Machine Learning Workflow: Streamlining Your ML Pipeline

MLOps

- Apache Airflow: Use Cases, Architecture, and Best Practices
- Edge AI: Benefits, Use Cases & Deployment Models
- JupyterHub: A Practical Guide

Multi GPU

- Keras Multi GPU: A Practical Guide
- PyTorch Multi GPU: Four Techniques Explained
- Tensorflow with Multiple GPUs: 5

NVIDA A100

- NVIDIA Deep Learning GPU: AI & Machine Learning Guide
- NVIDIA DGX: Under the Hood of DGX-1, DGX-2 and A100
- NVIDIA NGC: Features, Popular Containers & Quick Tutorial

HPC Clusters

- How Does HPC Leverage GPUs?

run:ai

Tel Aviv, Israel
New York, NY

AICPA Service Organization Control Reports
AICPA
SOC 2
Formerly SAS 70 Reports

Solutions

Building AI

Centralizing AI

Scaling AI

Data Scientist

Platform

How Run:ai Works

Applications

Operating System

AI Workload Scheduler

GPU Abstraction Layer

Control Plane

Infrastructure Resources

Integrations

Resources

Guides

Blog

News

White Papers

Video & Webinars

Podcast

Case Studies

About

Join Us

Partners

Meet Our Partners

Contact

Customers

Documentation

Status

Get Help

Terms of Service Privacy Policy