How Does Deep Learning Work? Deep learning models are based on neural network architectures. Inspired by the human brain, a neural network consists of interconnected nodes or neurons in a layered structure that relate the inputs to the desired outputs. The neurons between the input and output layers of a neural network are referred to as hidden layers. The term “deep” usually refers to the number of hidden layers in the neural network. Deep learning models can have hundreds or even thousands of hidden layers.

Viewing a typical neural network architecture.

Deep learning models are trained by using large sets of labeled data and can often learn features directly from the data without the need for manual feature extraction. While the first artificial neural network was theorized in 1958, deep learning requires substantial computing power that was not available until the 2000s. Now, researchers have access to computing resources that make it possible to build and train networks with hundreds of connections and neurons.

High-performance GPUs have a parallel architecture that is efficient for deep learning. When combined with clusters or cloud computing, this enables development teams to reduce training time for a deep learning network from weeks to hours or less.

What Is Deep Learning? | Introduction to Deep Learning 3:33 Video length is 3:33 What Is Deep Learning? | Introduction to Deep Learning

Top 5 Reasons to Use MATLAB for Deep Learning 2:16 Video length is 2:16 Top 5 Reasons to Use MATLAB for Deep Learning

AI for Engineers: Building an AI System 3:40 Video length is 3:40 AI for Engineers: Building an AI System

Types of Deep Learning Models Three types of deep learning models are convolutional neural networks (CNNs), recurrent neural networks (RNNs), and transformer models.

CNNs: A CNN convolves learned features with input data, and uses 2D convolutional layers, making this architecture well suited for processing 2D data, such as images. The CNN works by extracting features directly from images. The relevant features are learned while the network trains on a collection of images. This automated feature extraction makes deep learning models highly accurate for image classification tasks. CNNs can also be used for classifying other types of data, such as time series and text.

Visualizing an example of a convolutional neural network.

RNNs: A recurrent neural network (RNN) is a network architecture for deep learning that predicts on time-series or sequential data. RNNs are particularly effective for working with sequential data that varies in length and solving problems such as natural signal classification,

language processing, and video analysis. The long short-term memory (LSTM) network is a special type of RNN that is better in learning longer term dependencies than simple RNNs.

Transformers: Transformers are designed to track relationships in sequential data. They rely on a self-attention mechanism to capture global dependencies between input and output. They are often used for natural language processing and they are the basis for large language models (LLMs) such as BERT and ChatGPT™.

Keep Exploring This Topic What Are Convolutional Neural Networks? (4:44) Create Simple Sequence Classification Network Using Deep Network Designer Train BERT Document Classifier How to Create Deep Learning Models You can create a deep learning model from scratch or start with a pretrained deep learning model, which you can apply or adapt to your task.

Training from Scratch: To train a deep learning model from scratch, you gather a large, labeled data set and design a network architecture that will learn the features and model. This is a good approach for new or specific applications, or more generally, applications for which preexisting models do not exist. The main disadvantage of this approach is that it requires a large dataset (with the ground truth annotated) and the training time can take from hours to weeks, depending on your task and computing resources.

Transfer Learning: In deep learning applications such as image classification, computer vision, audio processing, and natural language processing, the transfer learning approach is commonly used. It involves fine-tuning a pretrained deep learning model. You start with an existing model, such as SqueezeNet or GoogLeNet for image classification, and feed in new data containing previously unseen classes. After making some tweaks to the network, you can now perform a new task, such as categorizing only dogs or cats instead of 1000 different objects. This also has the advantage of needing much less data, so the training time drops significantly.

A pretrained deep learning model can also be used as a feature extractor. You can use the layer activations as features to train another machine learning model (such as a support vector machine (SVM)). Or you can use the pretrained model as a building block for another deep learning model. For example, you can use an image classification CNN as the feature extractor for an object detector.

Keep Exploring This Topic Deep Learning: Deep Learning in 11 Lines of MATLAB Code (2:38) Deep Learning with MATLAB: Transfer Learning in 10 Lines of Code (4:41) Interoperability Between Deep Learning Toolbox, TensorFlow, PyTorch, and ONNX Deep Learning vs Machine Learning Deep learning is a specialized form of machine learning, and both are part of the artificial intelligence (AI) field. Machine learning offers a variety of techniques and models you can choose based on your application, the size of data you are processing, and the type of problem you want to solve.

Comparing the choice between deep learning or machine learning algorithms for your artificial intelligence application depends on your system’s goals and requirements.

Why choose deep learning over machine learning? In one word, accuracy. Deep learning generally achieves higher accuracy and offers more automation of the extended workflow than machine learning. The main shortcomings of deep learning models are that they are more complex and require larger training datasets, thus taking longer to train. Methods exist to overcome, or at least diminish the effect of, these shortcomings.

Feature Engineering Automation A machine learning workflow starts with relevant features being manually extracted from the data. The features are then used to create a model that can predict on new data. With a deep learning workflow, in applications such as image recognition and computer vision, relevant features are automatically extracted from the image data. When dealing with signals or text data, deep learning can reduce the required data preprocessing.

Comparing a machine learning approach to categorize vehicle (left) with deep learning (right).

Training Data and Time Deep learning accuracy scales with data. That is, deep learning performance continues to improve as the size of your training data increases. Typically, deep learning requires a very large amount of data (for example, thousands of images for image classification) to train the model. Access to high-performance GPUs, can significantly reduce training time. As an alternative, modifying and retraining a pretrained network with transfer learning is usually much faster and requires less labeled data than training a network from scratch.

If there is not enough training data available, you can complement your existing data with synthetic data. You can generate synthetic data by using generative adversarial networks (GANs) or by creating and simulating a model of the physical system.

Model Size and Complexity Deep learning models, compared to machine learning models, are far more complex and larger as they are built with hundreds of interconnected layers. As deep learning technology continues to advance, the complexity of deep learning network architectures continues to increase. Their complexity and size contribute to the accuracy deep learning can achieve.

Because of their complexity, deep learning models are often considered as “black-boxes” that lack interpretability. An emerging field, known as Explainable AI, offers techniques that aim to explain the behavior of deep learning models in human terms. For example, you can use Grad-CAM and LIME to explain predictions by deep learning models for image classification tasks.

Deep learning models do not just live on the desktop anymore. Deploying increasingly large and complex deep learning models onto resource-constrained devices is a growing challenge that many deep learning practitioners face. There are numerous techniques for compressing deep learning models, which can be used to reduce the deep learning models’ size on disk, runtime memory, and inference times, while retaining high accuracy.

Projecting a fully connected layer. Projection is a method for network compression.

Keep Exploring This Topic Deep Learning and Traditional Machine Learning: Choosing the Right Approach Compressing Neural Networks Using Network Projection Deploy Deep Learning Applications to the Edge Why Deep Learning Is Important Deep learning is a key technology behind driverless cars, enabling them to recognize a stop sign, or to distinguish a pedestrian from a lamppost. It is the key to voice control in consumer devices like phones, tablets, TVs, and hands-free speakers. Deep learning is getting lots of attention lately and for good reason. With deep learning, computers and systems can perform complex tasks with increasing accuracy and automation.

Deep Learning Applications Deep learning is applied in computer vision, image processing, automated driving, signal processing, and many more areas. Each deep learning application area can encompass several sub-application areas. For example, image classification, object detection, and semantic segmentation are sub-applications of computer vision. As new deep learning methods and technologies are developed, deep learning applications will continue to expand and new sub-applications where deep learning can improve accuracy will be uncovered.

Computer Vision

Signal Processing

Robotics

Natural Language Processing

Automated Driving

Predictive Maintenance Examples of Deep Learning at Work Visual inspection is the image-based inspection of parts where a camera scans the part under test for failures and quality defects. By using deep learning and computer vision techniques, visual inspection can be automated for detecting manufacturing flaws in many industries such as biotech, automotive, and semiconductors.

Detecting defects on printed circuit boards using YOLOX network.

Electroencephalography (EEG) signals are the most accessible and not surprisingly, the most investigated brain signals. You can use deep learning to automatically diagnose epilepsy and predict epileptic seizures in EEG signals.

Classifying EEG data with a time-frequency convolutional network.

Virtual sensors can be used in any system where real-time monitoring and control are required, and where the use of physical sensors might be impractical or costly. You can combine deep learning with model-based-design to design virtual sensors.

Designing virtual sensors with AI and Model-Based Design.

More on Deep Learning Applications Practical Deep Learning Examples with MATLAB Deep Learning Applications Apply AI to Your Domain Deep Learning Applied to Cancer Research Read how Moffit Cancer Center used deep learning to accelerate scientific discovery and personalize treatment plans.

MATLAB® provided all the necessary tools to successfully apply AI in imaging and radiomics. MATLAB was used for tasks such as data preprocessing, denoising, and image segmentation.

Deep Learning with MATLAB Using MATLAB with Deep Learning Toolbox™ enables you to design, analyze, and simulate deep learning networks. You can extend deep learning workflows to many applications (such as computer vision, signal processing, and natural language processing) by using additional toolboxes (such as Computer Vision Toolbox™, Signal Processing Toolbox™, and Text Analytics Toolbox™).

Visualizing a deep learning workflow from data preparation to deployment.

Data for Deep Learning Deep learning requires large amounts of good-quality data. You can use datastores to conveniently manage collections of data that are too large to fit in memory at one time. You can use low-code apps and built-in functions to improve the data quality and automatically label the ground truth.

Designing Deep Learning Models

Networks from Scratch With a few lines of code, you can create deep learning networks such as CNNs, LSTMs, GANs, and transformers. Speed up training using multiple GPUs, the cloud, or clusters. In training deep learning models, MATLAB uses GPUs (when available) without requiring you to explicitly program the GPUs.

Creating LSTM network programmatically.

Pretrained Networks Apply a pretrained model directly to your task, adapt it by performing transfer learning, or use it as a featured extractor. Explore the MATLAB Deep Learning Model Hub to access the latest models by category and load pretrained models at the command line. Convert TensorFlow™, PyTorch®, and ONNX™ models to MATLAB networks by using an import function. Export trained and untrained networks from Deep Learning Toolbox to TensorFlow and ONNX. Perform transfer learning with ease with built-in functions for replacing layers and adjusting learning rates for retraining.

Getting pretrained models to adapt for your deep learning task.

Low Code Apps With the Deep Network Designer app, you can design, analyze, and modify networks interactively. You can also import pretrained networks from Deep Learning Toolbox, TensorFlow, and PyTorch. The Experiment Manager app helps you manage multiple deep learning experiments, keep track of training parameters, analyze results, and compare code from different experiments.

Designing, analyzing, and modifying deep learning networks graphically with Deep Network Designer.

Deep Learning for Engineers Explainable and Robust Deep Learning With MATLAB, you can visualize and verify deep learning models and their predictions.

Monitor the training progress of deep learning models and plot performance metrics. Visualize the outputs of deep learning models by applying explainability techniques, such as Grad-CAM, occlusion sensitivity, LIME, and deep dream. This helps you understand how deep learning models make predictions. Train robust deep learning models and verify the models' robustness. You can verify the robustness properties of a network, compute network output bounds, find adversarial examples, detect out-of-distribution data, and verify compliance with industry standards. System-Level Simulation Use Deep Learning Toolbox blocks to integrate trained networks with Simulink® systems. This allows you to test the integration of deep learning models with other parts of the system.

Deploy To Target You can deploy deep learning models to edge devices, embedded systems, or the cloud. Prior to deployment, you can compress your models by performing quantization, projection, or pruning.