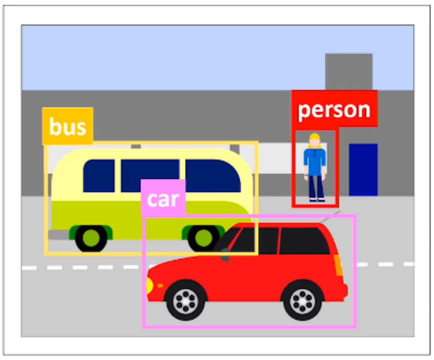
Applications: Object recognition, Computer vision, natural language processing. Introduction to Deep Learning Tools: TensorFlow, Torch.

Object Recognition:

Object recognition, also known as object detection or object classification, is a computer vision task that involves identifying and classifying objects or entities within images or videos. The goal is to teach a computer system to understand and interpret visual data, recognize specific objects, and assign appropriate labels or categories to them.

Object recognition plays a significant role in deep learning, which is a subfield of artificial intelligence that focuses on training neural networks with multiple layers (deep neural networks) to learn hierarchical representations of data. Deep learning has revolutionized object recognition by enabling the development of highly accurate and efficient models.



The role of deep learning in object recognition can be summarized as follows:

- **Feature Learning:** Deep learning models, such as convolutional neural networks (CNNs), are designed to automatically learn hierarchical representations or features from raw visual data. These models learn to extract low-level features (edges, textures) and progressively build more abstract and high-level representations that capture object-specific characteristics. This feature learning capability of deep learning networks greatly aids object recognition.

- **End-to-End Learning:** Deep learning models can be trained end-to-end, meaning that they learn to recognize objects directly from raw pixel data without relying on handcrafted features or pre-processing steps. This is in contrast to traditional computer vision approaches that require manual feature engineering. End-to-end learning simplifies the development process and allows for more robust and scalable object recognition systems.

- **Object Localization:** Deep learning-based object recognition models can also perform object localization, which involves not only detecting objects but also localizing them within an image or video. This is achieved by predicting bounding boxes or regions of interest around the objects of interest. The ability to perform both object detection and localization within a single model is a significant advantage of deep learning in object recognition.

- **Flexibility and Generalization:** Deep learning models are highly flexible and can generalize well across different object categories, appearances, and variations. They can learn complex patterns and capture intricate details, making them suitable for a wide range of object recognition tasks. Additionally, transfer learning techniques enable pre-trained deep learning models to be fine-tuned on specific object recognition tasks with limited labeled data, further enhancing their performance.

- **Scalability:** Deep learning models can scale to handle large and complex datasets. With the availability of powerful computational resources and the ability to parallelize computations, deep learning networks can process vast amounts of visual data efficiently. This scalability is crucial for real-world applications that deal with extensive image or video collections.

Here are some applications of object recognition:

- **Autonomous Vehicles:** Object recognition plays a critical role in autonomous vehicles for tasks such as detecting pedestrians, vehicles, traffic signs, traffic lights, and other objects on the road. It helps the vehicle understand its surroundings and make informed decisions to navigate safely.
- **Surveillance and Security:** Object recognition is widely used in surveillance systems to detect and track suspicious activities or objects in public spaces, airports, banks, and other security-sensitive areas. It can identify unauthorized individuals, detect suspicious objects, and alert security personnel in real-time.

- **Robotics:** Object recognition is crucial for robots to interact with the physical world. It enables robots to perceive and recognize objects, grasp them, and manipulate them accordingly. This is essential in industrial automation, manufacturing, and household robot applications.
- **Augmented Reality (AR):** Object recognition is used in AR applications to recognize real-world objects and overlay virtual content on top of them. It enables interactive and immersive experiences by detecting and tracking objects such as markers, images, or 3D objects.
- **E-commerce:** Object recognition is employed in e-commerce applications to enhance user experiences. By recognizing and tagging objects within images, it allows for visual search, where users can find similar products by uploading or capturing images. It also enables automated product categorization and recommendation systems.
- **Medical Imaging:** Object recognition assists in medical imaging tasks such as tumor detection, organ segmentation, and identifying anatomical structures in radiological scans. It aids in diagnosis, treatment planning, and monitoring of various diseases.
- **Quality Control and Inspection:** Object recognition is used in manufacturing industries for quality control and inspection purposes. It can detect defects, measure dimensions, verify product integrity, and ensure adherence to specifications in real-time, reducing manual inspection efforts.
- **Social Media and Content Moderation**: Object recognition techniques are utilized by social media platforms to automatically detect and moderate inappropriate or sensitive content, such as nudity, violence, or hate speech. It helps in filtering and enforcing content policies.
- **Accessibility:** Object recognition can be leveraged to assist people with visual impairments. By recognizing and providing audio descriptions of objects in their surroundings, it helps them navigate their environment more independently.
- **Cultural Heritage Preservation:** Object recognition techniques can aid in the preservation and analysis of cultural artifacts. They can identify and classify objects in historical images or artworks, assisting in cataloging, restoration, and understanding cultural heritage.

Computer Vision:

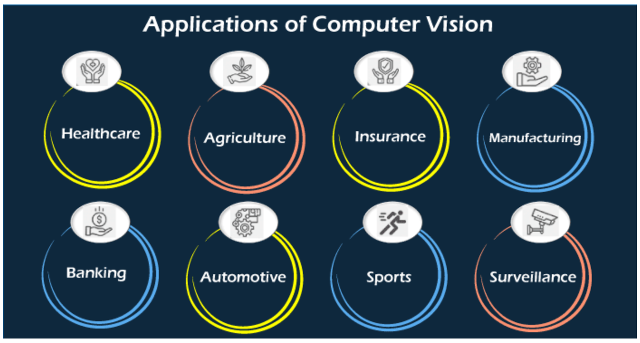
Computer vision is a field within artificial intelligence and computer science that focuses on enabling computers to understand and interpret visual data, such as images and videos. It aims to replicate human vision by developing algorithms and systems that can extract meaningful information, recognize objects, understand scenes, and perform various visual tasks.

Computer vision involves various tasks, including:

- **Image Classification:** Assigning a label or category to an image based on its content. For example, classifying images as "cat" or "dog."
- **Object Detection:** Locating and identifying specific objects within an image or video. This involves drawing bounding boxes around objects of interest, such as people, cars, or buildings.
- **Semantic Segmentation:** Assigning a class label to each pixel in an image, creating a detailed pixel-level understanding of the scene. It enables fine-grained segmentation of objects within an image.
- **Image Recognition:** Recognizing and understanding the content and context of an image. This involves identifying objects, scenes, or specific features within an image.
- **Image Generation:** Generating new images using computer algorithms. This can involve generating realistic images based on learned patterns or transforming images in creative ways.
- **Object Tracking:** Tracking the movement and position of objects over a sequence of images or video frames. It involves maintaining the identity and location of objects as they move.
- **Image Restoration:** Enhancing or restoring the quality of images by reducing noise, removing artifacts, or improving resolution.
- **Pose Estimation:** Estimating the positions and orientations of human or object poses in images or videos. It has applications in gesture recognition, action recognition, and robotics.
- **Image Captioning:** Automatically generating textual descriptions or captions for images. It involves understanding the content of the image and generating coherent and descriptive sentences.
- **3D Reconstruction:** Reconstructing three-dimensional models of objects or scenes from two-dimensional images. It involves extracting depth and spatial information from multiple images or using depth sensors.

Here are some common applications of computer vision:

- **Object Detection and Recognition:** Computer vision is extensively used for detecting and recognizing objects within images or videos. This has applications in autonomous vehicles, surveillance systems, robotics, quality control, and e-commerce.
- **Facial Recognition:** Facial recognition technology employs computer vision techniques to identify and verify individuals based on their facial features. It finds applications in security systems, access control, biometrics, and law enforcement.
- **Augmented Reality (AR):** Computer vision is essential in AR applications to overlay virtual content on real-world scenes. It enables interactive and immersive experiences by recognizing and tracking objects, markers, or environments.
- **Medical Imaging and Healthcare:** Computer vision is used in medical imaging for tasks such as tumor detection, organ segmentation, disease diagnosis, and surgical planning. It aids in improving the accuracy and efficiency of medical diagnostics and treatments.
- **Industrial Automation:** Computer vision is employed in industrial automation for tasks like object tracking, defect detection, quality control, and process monitoring. It helps streamline manufacturing processes and ensures product quality.



- **Autonomous Vehicles:** Computer vision plays a crucial role in autonomous vehicles for tasks such as object detection, lane detection, traffic sign recognition, and pedestrian tracking. It helps vehicles perceive and understand their environment for safe navigation.
- **Retail and E-commerce:** Computer vision is used in retail and e-commerce for applications like inventory management, shelf monitoring, product recognition, and visual search. It enhances customer experiences and enables efficient retail operations.
- **Security and Surveillance:** Computer vision is extensively utilized in security and surveillance systems for activities such as people counting, anomaly detection, object tracking, and threat identification. It aids in maintaining public safety and enhancing security measures.
- **Agriculture:** Computer vision is applied in agriculture for crop monitoring, plant disease detection, yield estimation, and automated harvesting. It assists in optimizing farming practices and improving crop productivity.
- **Robotics:** Computer vision plays a vital role in robotics for object recognition, localization, and grasping. It enables robots to interact with the physical world, perform complex tasks, and collaborate with humans in various settings.
- **Document Analysis:** Computer vision techniques are used for tasks such as text recognition, document classification, and form processing. It facilitates automated document processing and digitization.
- **Sports Analytics:** Computer vision is employed in sports analytics to track player movements, analyze game strategies, and provide real-time insights for performance evaluation and training.

Natural Language Processing:

Natural language processing is a sub-field of linguistics, computer science and AI concerned with the interactions between computers and human language.

The goal of NLP: Computers and machines are great at working with tabular, structured data. Much of the information humans generate has complex structure and not tabular, making it very difficult for computers to interpret. NLP makes computers understand complex language structure and retrieve meaningful pieces of information from it.

Modern challenges in NLP frequently involve

o speech recognition,
o natural language understanding and
o natural language generation.

Applications of NLP

- **Text Classification:** This is the task of bucketing the text into a known set of categories based on its content. It is the most popular task in NLP and is used in a variety of tools, from email spam identification to sentiment analysis.
- **Language Modelling:** This is the task of predicting what the next word in a sentence will be based on the history of previous words. The goal of this task is to learn the probability of a sequence of words appearing in each language. Language modelling is useful for building solutions for a wide variety of problems, such as speech recognition, optical character recognition, handwriting recognition, machine translation, and spelling correction.
- **Information Extraction:** As the name indicates, this is the task of extracting relevant information from text, such as calendar events from emails or the names of people mentioned in a social media post. Some of its common sub-tasks are Named Entity Recognition, Co-reference Resolution.
- **Information Retrieval:** This is the task of finding documents relevant to a user query from a large collection. Applications like Google Search are well-known use cases of information retrieval.
- **Conversational Agent:** This is the task of building dialogue systems that can converse in human languages. Alexa, Siri, etc., are some common applications of this task.
- **Text Summarization:** This task aims to create summaries of longer documents while retaining the core content and preserving the overall meaning of the text.
- **Question Answering:** This is the task of building a system that can automatically answer questions posed in natural language.
- **Machine Translation:** This is the task of converting a piece of text from one language to another. Tools like Google Translate are common applications of this task.
- **Topic Modelling:** This is the task of uncovering the topical structure of a large collection of documents. Topic modelling is a common text-mining tool and is used in a wide range of domains.

NLP Subproblems

- Text Categorization
- Machine Translation
- Text Summarization
- Entity Recognition
- Temporal even recognition
- Text Generation
- Natural Language Interface
- Speech Recognition
- Text to speech

Deep learning tools:

Deep learning tools are software libraries and frameworks that provide a set of functionalities and abstractions to facilitate the development, training, and deployment of deep neural networks. These tools are designed to simplify the implementation of complex deep learning models, optimize performance, and support efficient deployment on various platforms.

Deep learning tools play a significant role in the development and deployment of deep learning models. Here are some key significance of deep learning tools:

- **Abstraction and Ease of Use:** Deep learning tools provide high-level abstractions and APIs that simplify the process of building, training, and deploying deep neural networks. They offer intuitive interfaces and functionalities that make it easier for both beginners and experienced practitioners to work with complex deep learning models.
- **Efficiency and Performance:** Deep learning tools are designed to optimize the performance of deep neural networks. They provide efficient implementations of key operations and leverage hardware acceleration (such as GPUs or TPUs) to speed up computation. These tools often incorporate optimizations like automatic differentiation, parallel processing, and memory management techniques to improve efficiency.
- **Model Architecture Flexibility:** Deep learning tools offer flexibility in designing and customizing the architecture of deep neural networks. They provide a wide range of pre-defined layers and modules that can be combined and configured to create complex architectures tailored to specific tasks. This flexibility allows researchers and

practitioners to experiment with different network architectures and components to achieve optimal performance.

- **Availability of Pre-trained Models:** Many deep learning tools provide access to pre-trained models and model zoos, which are networks that have been trained on large datasets for specific tasks. These pre-trained models can be used as a starting point or as a basis for transfer learning, enabling faster development and better performance in domains with limited labeled data.
- **Support for Diverse Data Formats:** Deep learning tools offer support for various data formats commonly used in deep learning, such as images, audio, text, and time-series data. They provide data preprocessing and transformation utilities to handle different data formats, including image resizing, normalization, text tokenization, and sequence padding.
- **Community and Ecosystem:** Deep learning tools have active developer communities and ecosystems that provide resources, tutorials, and documentation. They foster collaboration, knowledge sharing, and the development of open-source libraries and frameworks. These communities also contribute to the advancement of deep learning techniques and offer support for troubleshooting and problem-solving.
- **Deployment and Integration:** Deep learning tools provide functionalities for deploying trained models into production environments. They offer APIs and integration capabilities to deploy models on different platforms, including cloud services, edge devices, and mobile devices. Some tools also provide optimizations for model size, latency, and energy efficiency, allowing for efficient deployment on resource-constrained devices.

Here are some popular deep learning tools:

TensorFlow: TensorFlow, developed by Google, is one of the most widely used deep learning frameworks. It provides a comprehensive ecosystem for building and deploying deep learning models with support for a wide range of platforms and devices. TensorFlow offers high-level APIs for ease of use and flexibility, as well as lower-level APIs for advanced customization.

PyTorch: PyTorch, developed by Facebook's AI Research lab, is another popular deep learning framework. It emphasizes flexibility and dynamic computation, making it easy to prototype and experiment with new ideas. PyTorch provides a dynamic computational graph and extensive support for GPU acceleration, along with a rich set of libraries for tasks like computer vision, natural language processing, and reinforcement learning.

Keras: Keras is a user-friendly deep learning library that acts as a high-level API on top of TensorFlow, Theano, or Microsoft Cognitive Toolkit (CNTK). It provides a simple and intuitive interface for building and training deep neural networks. Keras supports both sequential and functional model architectures and offers a wide range of predefined layers and utilities.

Caffe: Caffe is a deep learning framework developed by Berkeley Vision and Learning Center (BVLC) primarily focused on computer vision tasks. It is known for its efficiency and speed, making it suitable for large-scale image and video processing. Caffe comes with a model zoo containing pre-trained models for various vision tasks.

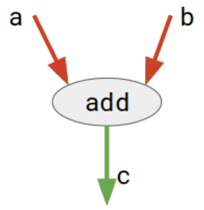
MXNet: MXNet is a deep learning framework known for its scalability and efficiency across multiple devices and distributed systems. It offers both imperative and symbolic programming interfaces and supports multiple programming languages. MXNet provides a rich set of pre-built neural network layers and supports hybridization for optimizing model performance.

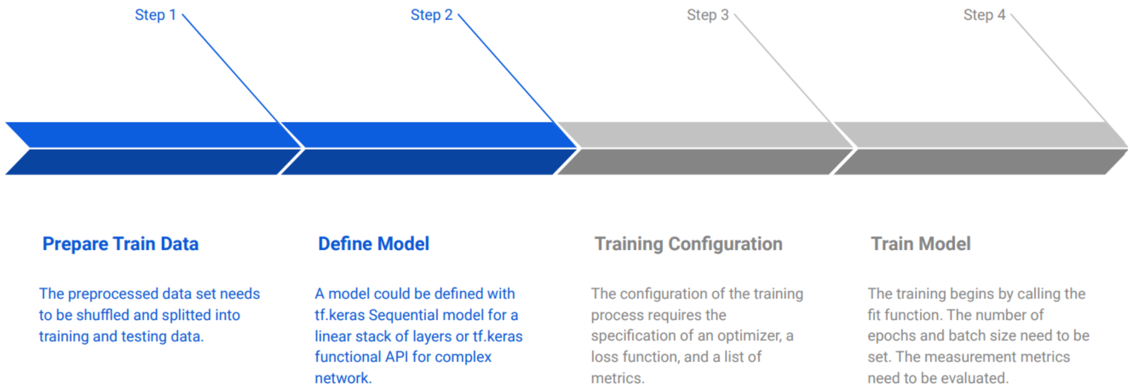
Theano: Theano is a deep learning library that allows users to define and optimize mathematical expressions, especially for building and training deep neural networks. While it is no longer actively developed, it served as the foundation for subsequent frameworks like TensorFlow and PyTorch.

Microsoft Cognitive Toolkit (CNTK): CNTK is a deep learning toolkit developed by Microsoft. It provides efficient implementations of deep learning algorithms and supports distributed training across multiple machines. CNTK offers a Python API as well as interfaces for C++, C#, and other programming languages.

TensorFlow:

- It is one of the most popular open-source platforms by Google.
- This framework is based on JavaScript, and this framework is equipped with different tools and community resources, which helps in easy deployment to the machine learning models.
- The core tool allows your deployment for the browsers, and the TensorFlow Lite is used to deploy models on mobile.
- TensorFlow uses dataflow graphs to process data. Nodes in the graph represent mathematical operations, while the graph edges represent the multidimensional data arrays (tensors) that flow between them.
- This flexible architecture allows you to deploy computation to one or more CPUs or GPUs in a desktop, server, or mobile device with a single API!
- If you are planning to learn the TensorFlow framework, then you need to have expertise in Julia, Go, and Python. You would be required to do lengthy coding, and it works with a static computation graph.
- Problem Learning flow





Prepare Train Data

The preprocessed data set needs to be shuffled and splitted into training and testing data.

Define Model

A model could be defined with tf.keras Sequential model for a linear stack of layers or tf.keras functional API for complex network.

Training Configuration

The configuration of the training process requires the specification of an optimizer, a loss function, and a list of metrics.

Train Model

The training begins by calling the fit function. The number of epochs and batch size need to be set. The measurement metrics need to be evaluated.

- TensorFlow uses a tensor data structure to represent all data. A TensorFlow tensor as an n-dimensional array or list. A tensor has a static type, a rank, and a shape.

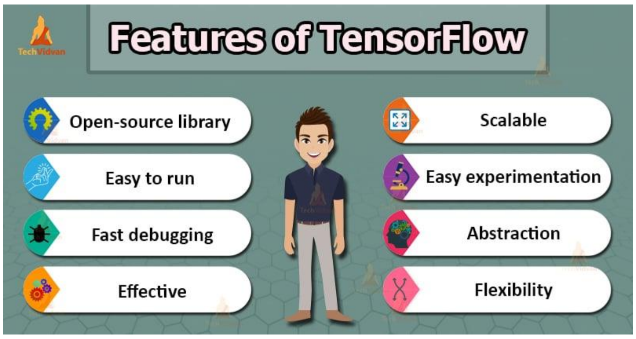
Name	Rank	Tensor
Scalar	0	[5]
Vector	1	[1 2 3]
Matrix	2	[[1 2 3 4], [5 6 7 8]]
Tensor	3	...

- *Benefits*
  - It is good for developing ML models
  - Used for data integration function, SQL Tables and images together

TensorFlow, as a popular deep learning framework, possesses several key characteristics that contribute to its widespread adoption and success:

1. **Flexibility:** TensorFlow offers a flexible and modular design, allowing users to construct and customize complex deep learning models. It provides a wide range of pre-built operations and neural network layers that can be combined to create custom architectures. TensorFlow's flexibility enables researchers and practitioners to experiment with various model configurations and optimize them for specific tasks.
2. **Scalability:** TensorFlow is designed to scale from running on a single CPU or GPU to distributed computing across multiple machines. It provides support for distributed training, allowing users to train large-scale models efficiently. TensorFlow's distributed computing capabilities enable the training of models on massive datasets and the deployment of models in production environments with high performance.
3. **Platform and Device Support:** TensorFlow supports a variety of platforms and devices, including CPUs, GPUs, and even specialized hardware like TPUs (Tensor Processing Units). It allows seamless execution of deep learning models across different hardware configurations, enabling users to leverage the computing power of various devices for model training and inference.
4. **High Performance:** TensorFlow incorporates optimization techniques to maximize computational efficiency. It takes advantage of hardware acceleration, such as GPU parallelism, to speed up computations. TensorFlow's underlying computational graph execution engine optimizes operations and memory allocation to minimize overhead and improve performance.
5. **Visualization Tools:** TensorFlow provides built-in tools for visualizing and debugging deep learning models. It offers a visualization dashboard called TensorBoard, which enables users to monitor training progress, visualize model architectures, and analyze performance metrics. TensorBoard helps in gaining insights into model behaviour, identifying bottlenecks, and facilitating model optimization.
6. **Ecosystem and Community:** TensorFlow benefits from a vibrant and active community of developers and researchers. The TensorFlow ecosystem includes a rich collection of libraries, tools, and resources developed by the community. These resources cover various domains, including computer vision, natural language processing, reinforcement learning, and more. The community support and extensive documentation make it easier for users to get started, seek help, and share knowledge.
7. **Deployment and Production Ready:** TensorFlow provides tools and functionalities for deploying deep learning models in production environments. It offers options for model serving, allowing users to expose trained models through APIs for inference. TensorFlow Serving and TensorFlow Lite are specialized components that facilitate the deployment of models on servers or resource-constrained devices, respectively.
8. **Integration and Interoperability:** TensorFlow integrates well with other popular libraries and frameworks, making it compatible with the broader machine learning ecosystem. It supports interoperability with Python and other programming languages, allowing users to leverage their existing code and libraries. TensorFlow can seamlessly work with other deep learning tools and frameworks, enabling users to combine the strengths of multiple tools in their projects.

Features of tensor flow:



1. **Computational Graph:** TensorFlow uses a dataflow computational graph paradigm, where operations are represented as nodes in the graph and data flows through the edges. This allows for efficient parallel execution and automatic differentiation for gradient computation during backpropagation.
2. **Ease of Use:** TensorFlow provides high-level APIs (such as Keras) that simplify the process of building and training deep learning models. These APIs offer user-friendly abstractions and allow for rapid prototyping. TensorFlow also provides a low-level API that offers more flexibility and control for advanced users.
3. **Neural Network Layers:** TensorFlow includes a rich collection of pre-built neural network layers. These layers, such as convolutional, recurrent, and dense layers, allow users to construct complex network architectures easily. TensorFlow's layers API provides a high-level interface for building neural networks quickly.

4. **Model Saving and Loading:** TensorFlow provides functionalities for saving and loading trained models. This allows users to save model parameters, weights, and biases after training, and reload them later for inference or further training. Saved models can be deployed across different platforms and used for transfer learning.
5. **Distributed Computing:** TensorFlow supports distributed computing across multiple devices and machines. It enables distributed training of deep learning models, allowing users to scale their models to larger datasets and take advantage of multiple GPUs or even specialized hardware like TPUs. TensorFlow's distributed computing capabilities are designed to provide efficient and scalable performance.
6. **GPU and TPU Support:** TensorFlow has built-in support for GPU acceleration, allowing users to leverage the computational power of GPUs for training and inference. TensorFlow also provides specific support for Google's Tensor Processing Units (TPUs), which are specialized hardware accelerators for deep learning. TPUs offer even faster performance for TensorFlow models.
7. **TensorBoard:** TensorBoard is a visualization tool provided by TensorFlow. It allows users to visualize and monitor various aspects of their deep learning models. With TensorBoard, users can visualize the model architecture, track training progress, analyze performance metrics, and debug models. It provides interactive and informative visualizations that aid in understanding and optimizing models.
8. **TensorFlow Hub:** TensorFlow Hub is a repository of pre-trained models, model components, and embeddings. It provides a convenient way to access and reuse pre-trained models for specific tasks, enabling faster development and improved performance. TensorFlow Hub also encourages model sharing and collaboration within the TensorFlow community.
9. **TensorFlow Serving:** TensorFlow Serving is a dedicated component for serving trained TensorFlow models in production environments. It allows users to expose trained models as a service through a scalable and efficient API. TensorFlow Serving supports flexible deployment options and enables seamless integration with other production systems.
10. **TensorFlow Lite:** TensorFlow Lite is a framework for deploying TensorFlow models on resource-constrained devices, such as mobile devices or embedded systems. It offers efficient model execution with optimized performance and memory usage. TensorFlow Lite supports on-device inference, enabling deep learning applications in edge computing scenarios.

PyTorch :

- Another popular framework for machine learning is PyTorch or Torch. Facebook developed it.
- This framework is based on the Torch library and was designed to advance the entire process from research prototyping to production deployment.
- It has a C frontend atop and a Python interface.
- Benefits
    - You can use standard debuggers like PyCharm or PDB
    - It works dynamically with an updated graph
    - It is widely used for natural language Processing
- Here are some key aspects or characteristics of PyTorch:
    - **Dynamic Computational Graph:** PyTorch adopts a dynamic computational graph, allowing for the construction of the graph on-the-fly during runtime. This dynamic nature enables greater flexibility and intuitive debugging, as users can change the network structure and modify the computation graph dynamically. It facilitates an imperative programming style, making PyTorch feel more like standard Python programming.
    - **Pythonic and Intuitive API:** PyTorch offers a Pythonic and intuitive API, making it easy to learn and use, especially for users familiar with Python. The API design emphasizes simplicity and ease of understanding, allowing for efficient prototyping and experimentation. PyTorch's API provides a straightforward way to define, train, and deploy deep learning models.
    - **Eager Execution:** PyTorch follows a "define-by-run" approach, where computations are executed immediately as they are defined. This eager execution mode enables users to interactively experiment with their models, inspect intermediate results, and debug efficiently. It simplifies the development process and provides a more intuitive programming experience.
    - **Neural Network Modules:** PyTorch offers a rich library of pre-defined modules for building deep neural networks. These modules include various types of layers, activation functions, loss functions,

and optimization algorithms. PyTorch's modular design allows users to assemble and customize their models using these building blocks, enabling flexibility in model construction.

- **Automatic Differentiation:** PyTorch features automatic differentiation, a key component for training deep learning models via backpropagation. It automatically tracks operations performed on tensors and computes the gradients with respect to the input variables. Automatic differentiation simplifies the implementation of gradient-based optimization algorithms, making it easier to train and fine-tune models.
- **GPU Acceleration:** PyTorch seamlessly integrates with CUDA, a parallel computing platform, allowing users to leverage GPU acceleration. This enables faster computations and improved performance for deep learning tasks. PyTorch's GPU acceleration capabilities make it well-suited for handling large-scale datasets and computationally demanding models.
- **Strong Community Support:** PyTorch benefits from a vibrant and active community of developers and researchers. The community contributes to the continuous improvement of PyTorch, providing resources, tutorials, and sharing knowledge. The strong community support ensures a wealth of additional libraries, tools, and frameworks that extend PyTorch's functionality and address various use cases.
- **Seamless Integration with Python Ecosystem:** PyTorch integrates seamlessly with the broader Python ecosystem, making it compatible with other popular libraries and frameworks. It can be easily combined with libraries like NumPy, SciPy, and scikit-learn for data preprocessing, visualization, and evaluation tasks. PyTorch's compatibility and interoperability with other tools allow users to leverage the strengths of different libraries in their projects.
- **Deployment and Production Readiness:** PyTorch provides tools and functionalities for deploying trained models into production environments. It offers mechanisms such as TorchServe and TorchScript to serve models as APIs and convert models into optimized representations suitable for deployment on various platforms and devices. PyTorch's deployment capabilities enable the practical use of trained models in real-world applications.

Difference between Pytorch and TensorFlow

Features	PyTorch	TensorFlow
Development Wizards	Facebook developed PyTorch using the Torch library.	TensorFlow was developed by Google and is based on Theano (Python library)
Computation Graph	PyTorch uses a dynamic computational graph where the computations are done line by line at the time when code is interpreted.	Tensorflow uses the static computation graph i.e we have to define the computations that we want to do in a sequence and then run the Machine Learning Model.
Debugging	The dynamic approach of Pytorch where the computation graph is defined at the run time makes it eligible to use python debugging tools.	The static computation graph of the TensorFlow makes it hard to debug. To debug the code in TensorFlow, we can use a tool called tfdgb, which can test and 'evaluate the TensorFlow expression at the run time.
Production	PyTorch is easier to learn and lighter to work with, and hence, is relatively better for passion	There are a lot of advantages that TensorFlow enjoys over PyTorch when it comes to production. Better performance due to the static computation graphs,

Features	PyTorch	TensorFlow
	projects and building rapid prototypes.	packages/tools which help in fast development over platforms like browser, mobile, and cloud.
Data Visualization	PyTorch doesn't have such a tool. Although we have tools like MatPlot library using which we can compare different training runs.	Tensorflow comes up with a brilliant tool called TensorBoard which helps the user to visualize the Machine Learning Model, debug and compare different training runs i.e training a model and the tuning hyperparameters and then training again.TensorBoard can show both the runs at the same time to show the differences between them.