# Deep Learning Study Guide

This guide provides an overview of key concepts in Machine Learning and Deep Learning, with a focus on their applications in Computer Vision.

1. Introduction to Machine Learning (ML) and Deep Learning (DL)

**Machine Learning (ML)** is a field of computer science that investigates how to learn from data with the goal of making predictions. Instead of explicitly programming an algorithm, ML involves "learning an algorithm" or "synthesizing an algorithm" from data.

- o **Advantages of ML-based approaches**: They can develop complex models that connect inputs and outputs even when an explicit algorithmic approach is difficult to create. They can also incorporate a wide variety of conditions that might be hard to describe explicitly.
- o **Requirements for ML-based approaches**: They need a (large) amount of data that is unbiased, uncorrelated, balanced, and representative. They also require control over the training process and a descriptive loss function.

**Deep Learning (DL)** is a subfield of ML that is very effective in learning patterns.

- o **Key characteristic**: Data is represented by means of a hierarchy of multiple layers, leading to multiple levels of representation.
- o **Advantages of DL**: Unlike handcrafted features, which can be incomplete and require significant human effort to design and test, learned features in DL are adapted to the input data. DL allows for data representation and classification to be integrated into one single network, enabling "end-to-end learning".
- o **Requirements for DL**: DL typically needs a huge amount of data to be trained.

2. Deep Neural Networks (DNNs) Basics

- A **Deep Neural Network (DNN)** is a composition of several simple functions, called **layers**. Each layer consists of a set of **neurons**.
- **Non-linear elements**: DNNs include non-linear elements, typically through **activation functions**. If all layers were linear, the network would only be able to learn linear relationships.
- **Hierarchical abstraction**: The hierarchical topology of DNNs generates a hierarchical abstraction of the data.
- **Feedforward Network**: Inputs to the neurons of one layer are the outputs of the neurons in the preceding layer, making it a "feedforward network". There are no loops.
- **Neuron Output**: The output of a neuron is the activation function applied to a weighted average of the inputs plus a bias. Specifically, `a_i^(l) = h(sum(w_ij^(l) * a_j^(l-1)) + b_i^(l))` where `h` is the activation function, `w` are weights, and `b` are biases.
- **Output Layer**: The design of the output layer depends on the classification problem. For 2 classes, it can be one value or two values (scores for each class); this generalizes to N classes. The output can be a category (for classification) or a value (for regression).
- **Parameters**: A typical network structure involves weights and biases as learnable parameters. For example, a network with 4+2=6 neurons (not counting inputs) might have 20 weights and 6 biases, totaling 26 learnable parameters.

## 3. Convolutional Neural Networks (CNNs) for Computer Vision

- DL applied to Computer Vision (CV) specifically focuses on the image as a 2D matrix, where **spatial neighborhood is important**. CNNs are suitable architectures for this because they are designed to **catch the geometric neighborhood**.
- **Key features of CNNs**:
  - **Local connectivity**: Network connections merge together neighboring pixels. This also leads to a lower number of weights to train compared to fully connected layers.
  - **Shared weights (or tied weights)**: The same processing (filter) is applied across the entire image. This means that a single set of weights detects a specific feature everywhere in the image.
  - **Multiple feature maps**: The same process is repeated with different weights, allowing multiple ways of processing the same data (multiple filters) and learning multiple feature maps. Different units (neurons) in a layer can compute different functions while operating on the same input data windows.
- **Pooling Layers**: These are crucial in CNNs for subsampling and data reduction, which forces high-level representations.
  - **Types**: Max pooling and average pooling are common. Max pooling is usually preferred due to its efficiency and strong results.
  - **Function**: Pooling reduces the resolution of the feature map, allowing the next convolutional layer to operate at a larger scale. It was originally introduced to reduce computational complexity but also adds some deformation invariance.
  - **Process**: Pooling is applied over a "pooling area" with a "pooling stride".
- **CNN Architecture**: A typical CNN combines elements in the shape of: Input layer -> (Convolutional + Pooling) x N -> Vectorization -> (Fully connected) x M -> Output layer.

**Examples**:
- **MNIST classification problem**: This is considered the "hello world" for DL in computer vision. It involves classifying handwritten digits from a database of 60,000 training and 10,000 testing samples (28x28 grayscale images). A CNN structure for MNIST might include convolutional layers with specific filter sizes (e.g., 5x5) and pooling layers (e.g., 2x2).
- **CIFAR-10 dataset**: This involves color images (3 input channels). It has 60,000 images (50k training, 10k testing), 32x32 pixels, across 10 classes, with 6k images per class.

## 4. Transfer Learning

- **Concept**: Transfer learning involves adapting a pre-trained network to solve a different problem than what it was originally trained for. It transfers knowledge from a source problem/domain to a target problem/domain, requiring some degree of overlap between the source and target data.
- **Motivation**: It exploits already trained networks, eliminating the need to train from scratch, which saves time and allows for using smaller datasets. As Andrej Karpathy reportedly said, "Don't try to be an hero".
- **Why it works**: Some parts of a network are more general and can be more easily adapted to other problems.
- **Generic vs. Specific Features**: Layers close to the input often learn **low-level features** (more generic), while layers closer to the output learn **mid-level and problem-specific features** relevant to classification.
- **Adaptation**: The last layer's shape is connected to the problem (e.g., number of output neurons) and often needs to be reshaped with randomly initialized weights. Other 1-2 layers

close to the output might also need to be reset. Layers close to the input typically remain as they are, especially if the new dataset is correlated or if the source dataset was large and general, forcing the network to learn a wide range of patterns.

- **Simplest Pattern**: Take a pre-trained network, reset the last layer(s), freeze the deeper layers, and then train the network.

- **Fine-tuning**: This focuses on adapting an existing network to a new dataset by resuming training, involving also the deep layers. However, resuming training of deeper layers might cause the network to lose recognition of some previously learned patterns.
  - **Common Pattern (Transfer Learning + Fine-tuning)**: Rework the last 1-2 layers, freeze other layers, train the new layers only, then un-freeze the previous layers, and fine-tune the whole network.
  - **Note**: The definitions of "transfer learning" and "fine-tuning" can vary among different authors and sources.
- **Transfer Learning Patterns**: The variety of patterns is wide. Examples include exploiting the output of pre-trained networks as feature vectors or using features provided by inner layers (which might require dimensionality reduction).

5. Dataset Considerations

- **Dataset Size**: The required dataset size depends on factors like how different the classes are, how spread the data samples are, how aggressively data augmentation can be applied, and the availability of pre-trained weights. Datasets are difficult, expensive, and slow to create, so taking advantage of available, weakly related datasets is beneficial.
- **Unbalanced Datasets**: Real-world datasets are often unbalanced, leading to an unbalanced cost of misclassification.
  - **Rebalancing techniques**: Over-/under-sampling (changing data presentation frequency), negative mining (focusing on easily misclassified samples), weighting the loss, changing labeling, or ignoring data points.
- **Data Augmentation**: This involves expanding your dataset through transformations. These are often trivial transformations, such as flipping, cropping, changing illumination and colors, or mixing multiple methods. Geometric transformations are commonly used. Data augmentation is one way to change dataset dimension and data distribution.

6. Deep Learning for Object Detection: YOLO

- **YOLO (You Only Look Once)** is a popular family of object detectors. The name refers to its "one-stage" or "proposal-free" nature, meaning bounding box (Bbox) and category are found in a single pass.
- **Unified Detection**: YOLO achieves object detection by solving both a regression problem (determining bounding box location and dimension) and a classification problem (determining the class of the object) within a single network.
- **Working Principle**:
  - **Grid Cells**: An image is divided into a grid. If the center of an object falls into a specific grid cell, that cell is responsible for detecting the object.
  - **Bounding Box Prediction**: Each grid cell predicts `B` bounding boxes and confidence scores for those boxes. The confidence value indicates how confident the model is that the box contains an object and how accurate the box is. A bounding box is defined by `{x, y, w, h, con}` (box location, dimension, and confidence).

    - **Object Classification**: Each cell predicts `C` conditional class probabilities, one set unrelated to the number of bounding boxes. At test time, the class probability is multiplied by the bounding box confidence.
- **Output Layer**: The shape of the output layer depends on the number of bounding boxes (`B`) and the number of categories (`C`) relevant to the dataset.
- **Training**: YOLO is typically pre-trained on large datasets like ImageNet (first 20 convolutional layers at reduced resolution), and then the whole network is trained on datasets like Pascal VOC.
- **Performance Measurement**: Measured using metrics like mAP (mean Average Precision) and speed.

7. Deep Learning for Segmentation: U-Net

- **U-Net** is a fully convolutional network originally designed for biomedical image segmentation, particularly effective with small datasets. Its name comes from its distinctive 'U' shape.
- **Working Principle**: U-Net operates by contracting and expanding data.
    - **Contraction Path**: This path involves pooling operations (downsampling). Each downsampling step doubles the number of feature channels.
    - **Expansion Path**: This path uses upsampling operators. Each upsampling step reduces the number of feature channels by half and upsamples the feature map.
    - **Context Propagation**: High-resolution features from the contracting path are combined (concatenated) with the upsampled output of the expansion path. This allows the upsampling part, with its large number of feature channels, to propagate context information to higher resolution layers.
- **Architecture Details**: The last layer is typically a 1x1 convolution for mapping into the number of categories.
- **Experiments**: Original U-Net was tested on microscopic images, often with very small datasets (e.g., 30 512x512 images). This necessitates strong **data augmentation**, including shifts, rotations, smooth deformations, and interpolation.

8. Vision Transformers (ViTs)

- **Definition**: Vision Transformers (ViTs) are deep learning models that apply the **Transformer architecture**—originally developed for Natural Language Processing (NLP)—to image data. The foundational paper is "An Image is Worth 16x16 Words" by Dosovitskiy et al. (2020).
- **Advantages Over CNNs**:
    - **Better scalability** with large datasets.
    - Ability to **capture global image context** via self-attention mechanisms.
    - **Simpler architecture** without explicit convolutions.

- **How ViTs Work (Overview)**:
    1. The input image is split into fixed-size **patches** (e.g., 16x16 pixels).
    2. Each patch is **flattened** into a 1D vector and linearly projected into a D-dimensional **embedding space**.
    3. **Positional encoding** is added to these patch embeddings to retain spatial information, as Transformers are inherently permutation-invariant.

4. These embeddings, along with a special learnable **[CLS] token**, are passed through multiple **Transformer encoder layers**.

5. For classification, the output of the **[CLS] token** from the final encoder layer is passed to a Multilayer Perceptron (MLP) for classification.

- **Detailed Architecture**:
  - **Patch Embedding**: An input image (e.g., 224x224) is divided into patches (e.g., 16x16), resulting in a sequence of patches (e.g., 196 patches). Each patch is flattened (e.g., 16x16x3 = 768 values for RGB) and projected to an embedding space.
  - **Positional Encoding**: Added element-wise to patch embeddings to encode both content and position. Can be fixed (sinusoidal) or learnable.
  - **Transformer Encoder**: Consists of multiple layers (e.g., 12, 24). Each layer has:
    - **Multi-head Self-Attention (MHSA)**: Allows each token (patch) to attend to all other tokens, capturing diverse relationships.
    - **Layer Normalization and Residual Connections (Add & Norm)**: Applied after MHSA and after the Feed-Forward Network to ensure stability and better gradients.
    - **Feed-Forward Network (MLP)**: Fully connected layers applied independently to each token.

- **Training and Challenges**:
  - **Challenges**: ViTs typically require **large datasets** and have **high computational costs**.
  - **Solutions**: **Pretraining on large datasets** is a common solution. The development of **hybrid or efficient models** (e.g., Swin Transformer, DeiT, MobileViT) addresses these challenges.
- **Applications**: ViTs are applied in various computer vision tasks, including **image classification, object detection, semantic segmentation, medical imaging, and video understanding**.

ViTs are rapidly evolving, with many new variants continuously being developed.