NEW HORIZON
COLLEGE OF ENGINEERING

Module 5

Course Code 20AIM61A

NEW HORIZON
COLLEGE OF ENGINEERING

Image Segmentation

Application of Deep Learning

What is Image Segmentation?

Image segmentation is the process of partitioning a digital image into multiple segments.

It is the process of dividing an image into its constituent parts, such as objects, regions, or pixels.

The goal is to simplify and/or change the representation of an image into something more meaningful and easier to analyze.

It is a fundamental task in computer vision with a wide range of applications, such as object detection, image classification, and medical image analysis.

NEW HORIZON
COLLEGE OF ENGINEERING

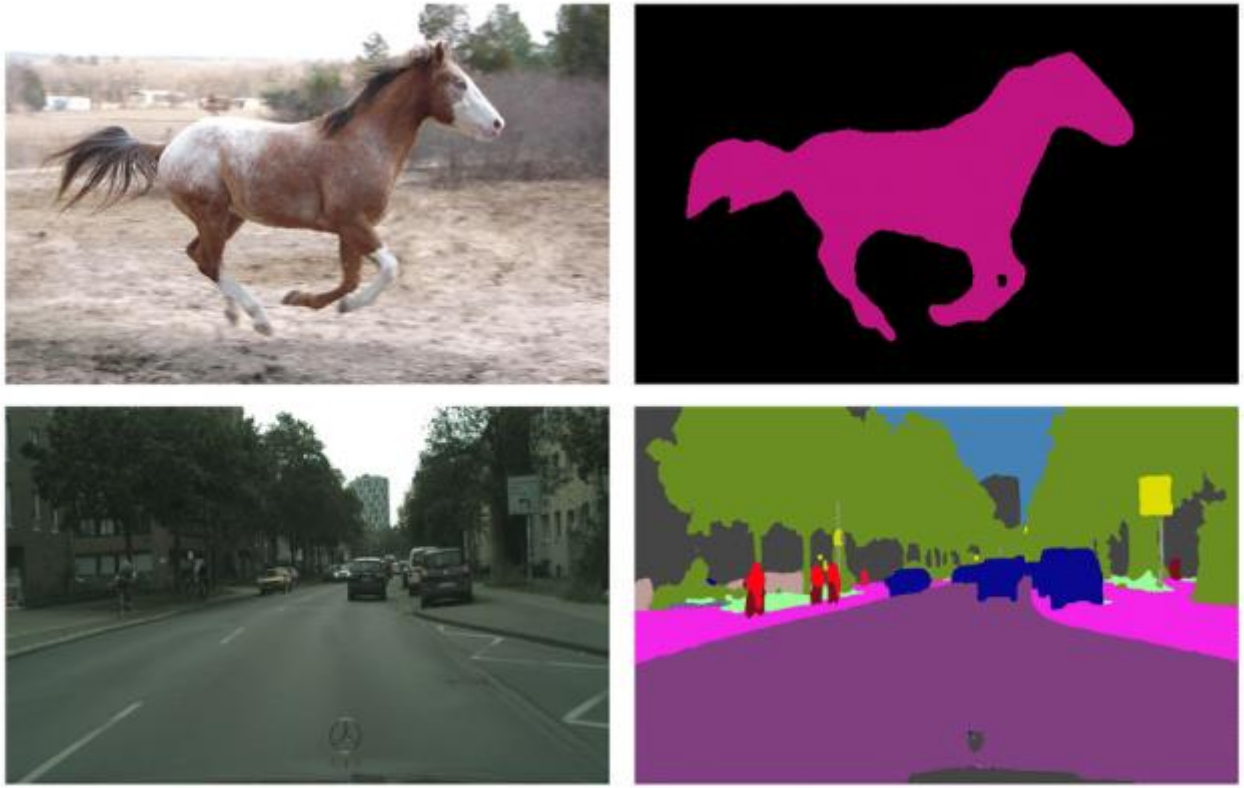


Image Segmentation

Traditional Image Segmentation Methods

Traditional image segmentation methods can be divided into two main categories:

1. **Region-based methods**
2. **Edge-based methods**

Region-based Image segmentation: It divides an image into regions that are similar according to a set of predefined criteria.

Edge-based Image segmentation: It divides an image based on abrupt changes in the intensity of pixels.

Region and Edge Based Image Segmentation

<u>Region-based methods</u> identify regions pixels that have similar properties, such as color, texture, or brightness.

They are typically more robust to noise and variations in illumination, but they can be slow and computationally expensive. (Region Growing & Splitting)

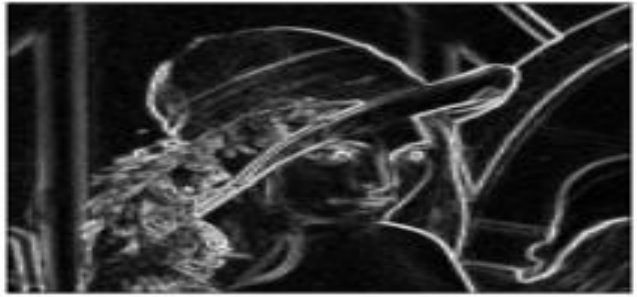
<u>Edge-based methods</u> identify the boundaries between regions of pixels that have different properties.

They are typically faster and more efficient, but they can be less robust to noise and variations in illumination.

Region-based Image Segmentation

NEW HORIZON COLLEGE OF ENGINEERING

Edge-based Image Segmentation



Deep Learning in Image Segmentation

Deep Learning is a subset of machine learning th uses neural networks with many layers to progressively extract higher-level features from raw input.

It allows computational models composed of multiple processing layers to learn representations of data with multiple levels of abstraction.

Deep learning has revolutionized image segmentation by providing highly accurate and efficient solutions.

DL models have been successfully applied to image segmentation tasks due to their ability to learn complex features from images and accurately classify each pixel.

Deep Learning in Image Segmentation

Deep learning models are trained on lar datasets of images that have been manu segmented.

The model learns to identify the features that distinguish different regions of pixels, and then uses this information to segment new images.

Over the years, researchers have developed several architectures that have significantly improved the performance of image segmentation tasks.

NEW HORIZON
COLLEGE OF ENGINEERING

DL based Image Segmentation Methods

There are a number of different deep learning models that can be used for image segmentation.

Some of the most popular models include:

1. **U-Net**
2. **FCN**
3. **Mask R-CNN**
4. **DeepLab**

U-Net

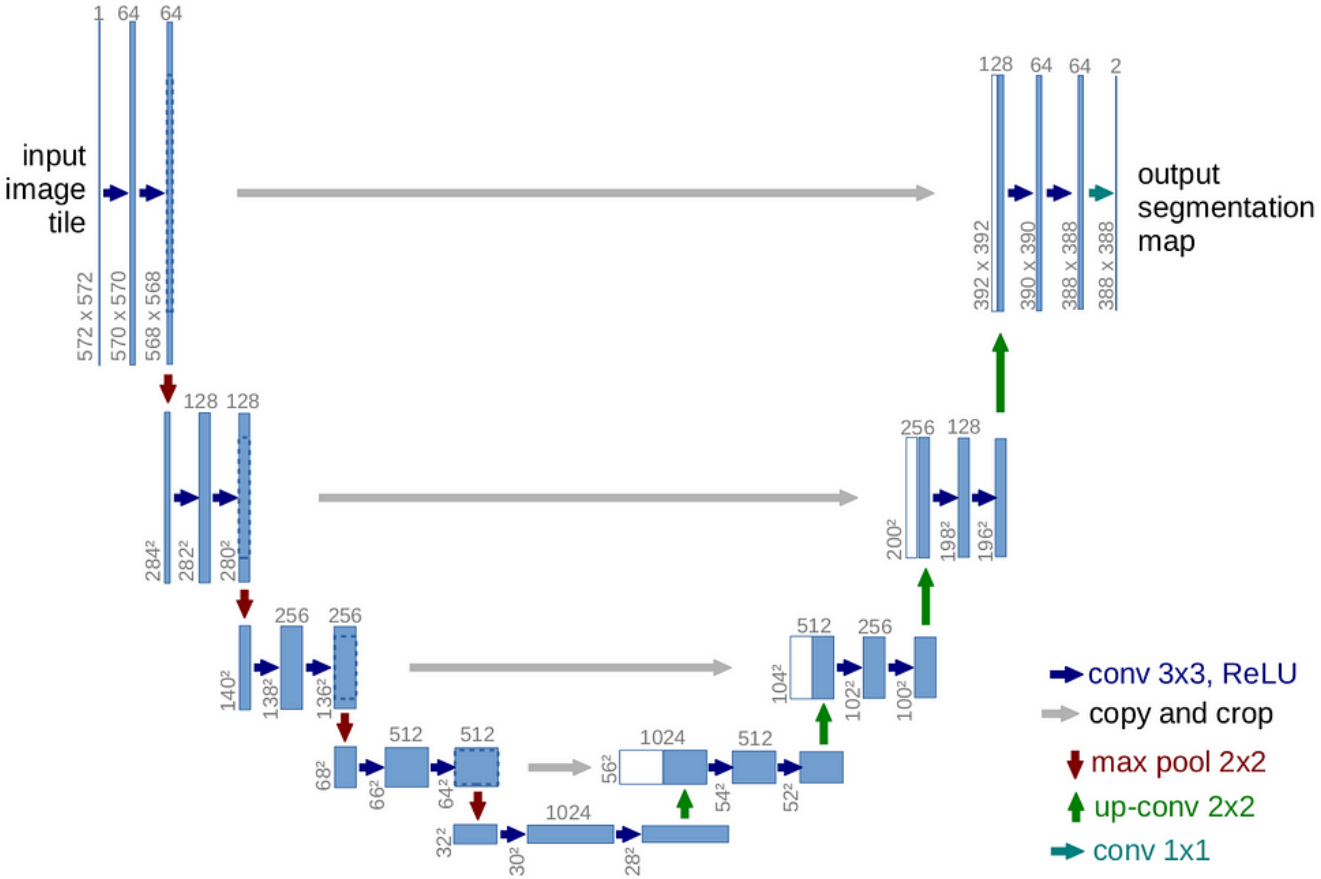
U-Net is an architecture for semantic segmentation, which expands a typical convolutional network with an up-sampling path to create a full-resolution output.

It is a fully convolutional neural network that has been shown to be very effective for image segmentation.

It is a symmetric architecture with two main parts: an encoder and a decoder.

The encoder extracts features from the image, while the decoder reconstructs the image with the segmented regions.

NEW HORIZON
COLLEGE OF ENGINEERING



U-Net Model – Architecture

Fully Convolutional Network (FCN)

FCN is a fully convolutional network that can be used for both semantic segmentation and instance segmentation.

It is a simple architecture that consists of a series of convolutional layers followed by a fully connected layer.

FCNs are trained end-to-end, pixels-to-pixels, for semantic segmentation.

- FCN based Image Segmentation

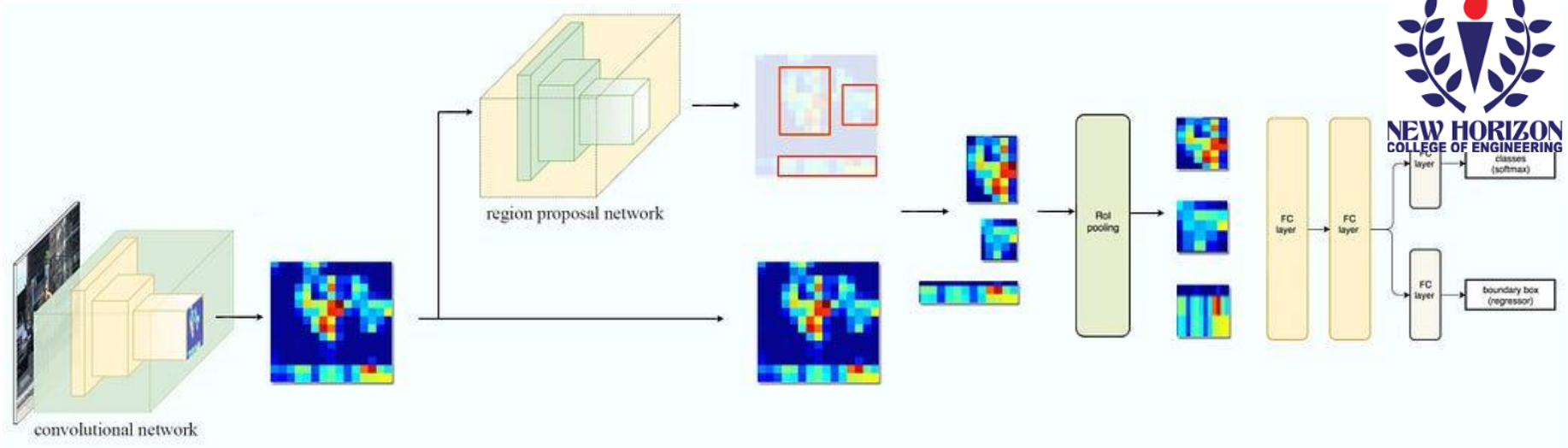
NEW HORIZON
COLLEGE OF ENGINEERING

Mask R-CNN

Mask R-CNN is a two-stage framework:

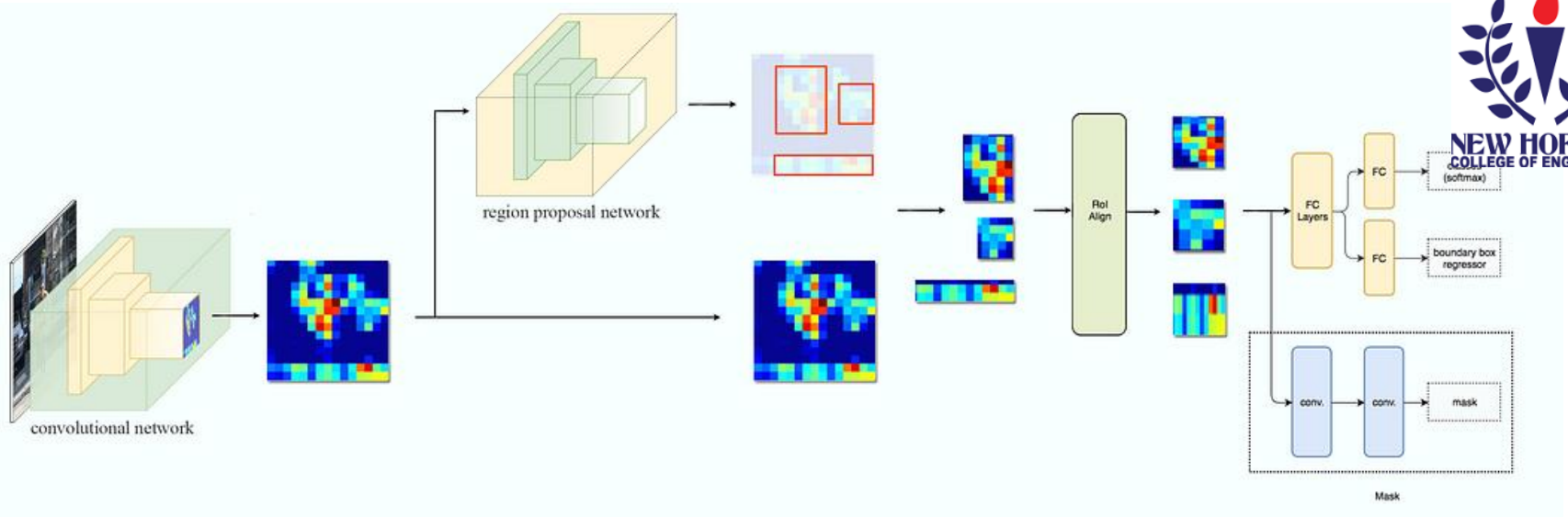
The first stage scans the image and generates proposals (areas likely to contain an object).

The second stage classifies the proposals and generates bounding boxes and masks.



NEW HORIZON COLLEGE OF ENGINEERING

Fast R-CNN – Architecture



Mask R-CNN – Architecture

NEW HORIZON COLLEGE OF ENGINEERING

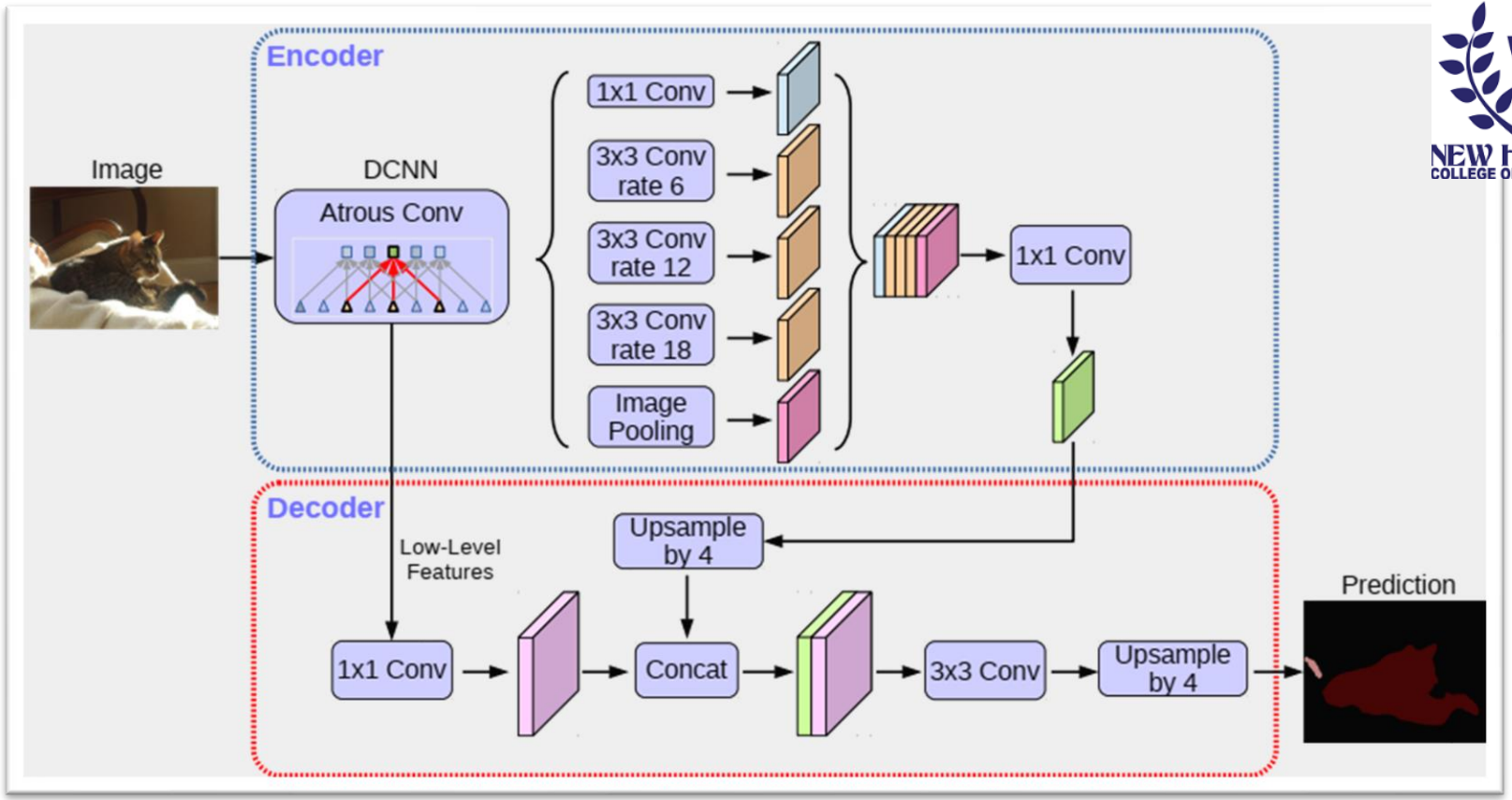
NEW HORIZON
COLLEGE OF ENGINEERING

DeepLab

DeepLab uses atrous convolution to explicitly control the resolution at which feature responses are computed in CNNs.

It is a deep learning model that has been specifically designed for semantic segmentation.

It is a hierarchical architecture that consists of a series of convolutional layers followed by a spatial pyramid pooling layer.



NEW HORIZON COLLEGE OF ENGINEERING

DeepLab v3 - Architecture

Applications of DL based Image Segmentation

Deep learning image segmentation has a wide range of applications, including:

1. **Object detection**
2. **Image classification**
3. **Medical image analysis**
4. **Autonomous driving**

Challenges and Future Directions

Challenges: Some challenges in image segmentation include the complexity and diversity of images, the requirement of large amounts of labeled data for training, and the computational resources needed.

Future Directions: Advancements in unsupervised learning, transfer learning, and real-time processing could improve the accuracy and efficiency of image segmentation tasks.

NEW HORIZON
COLLEGE OF ENGINEERING

Image Captioning

Generating Description for Images

Introduction

NEW HORIZON
COLLEGE OF ENGINEERING

- Image captioning is the task of generating a textual description for an image.
- It combines computer vision, which involves understanding and analyzing visual data, with natural language processing, which focuses on language generation.
- Image captioning plays a crucial role in various applications, revolutionizing the way we interact with images.
- In this presentation, we will explore the concept of image captioning, discuss its importance in different domains, and provide an overview of the presentation structure.

Image Captioning: Generating Textual Descriptions for Images

NEW HORIZON
COLLEGE OF ENGINEERING

- Image captioning refers to the process of generating a descriptive and coherent textual description for an image automatically.
- It involves combining computer vision techniques to understand the visual content of an image and natural language processing to generate a human-like description.

Combining Computer Vision and Natural Language Processing

- Image captioning integrates computer vision, which focuses on analyzing and interpreting visual data, and natural language processing, which involves language generation and understanding.
- Computer vision techniques extract meaningful features and representations from the image, providing context and visual understanding.
- Natural language processing techniques generate textual descriptions based on the extracted visual features, creating a coherent and meaningful caption.

Example:

Image captioning bridges the gap between visual perception and language generation, enabling machines to understand and describe visual content accurately and effectively.

NEW HORIZON
COLLEGE OF ENGINEERING

Steps Involved in Image Captioning

Image Preprocessing:

NEW HORIZON
COLLEGE OF ENGINEERING

- The first step in image captioning is image preprocessing, which involves preparing the image for further analysis and feature extraction.
- Common preprocessing steps include resizing the image to a standard size, normalizing pixel values, and applying transformations like cropping or rotation.
- The goal of preprocessing is to enhance the image quality and ensure consistency in the input data.

Feature Extraction using Convolutional Neural Networks (CNNs):

NEW HORIZON
COLLEGE OF ENGINEERING

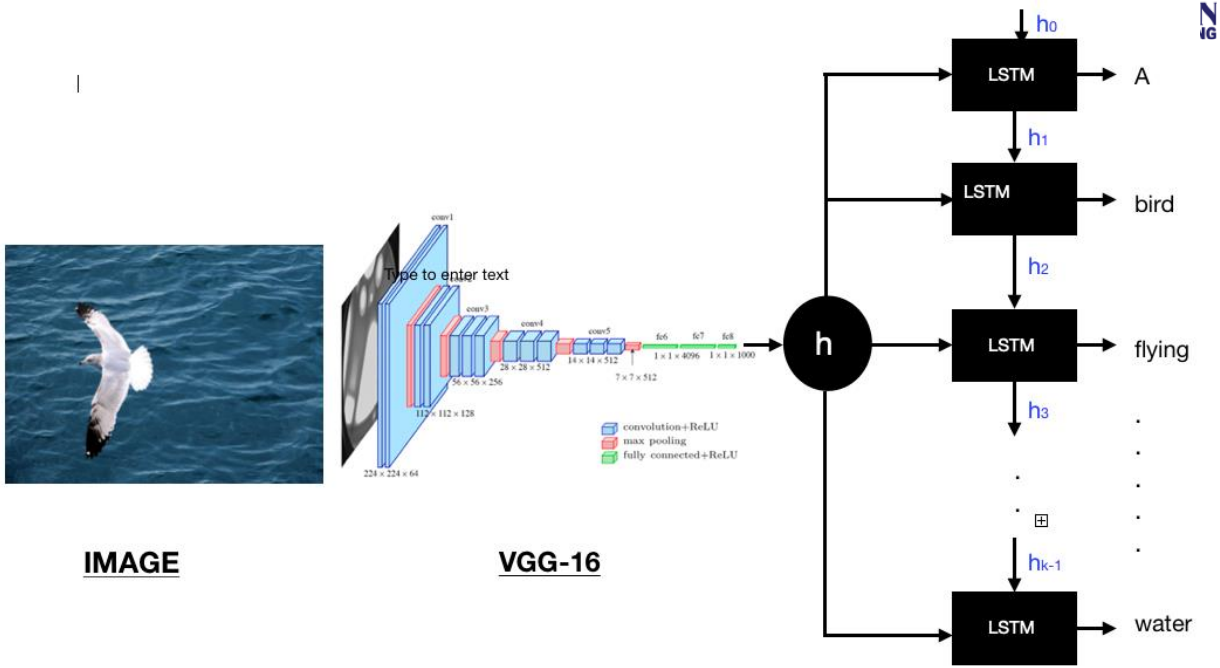
- After preprocessing, Convolutional Neural Networks (CNNs) are employed to extract relevant features from the image.
- CNNs are deep learning models specifically designed for visual data analysis, capable of capturing hierarchical and meaningful representations.
- CNNs process the image through multiple convolutional and pooling layers, learning to recognize patterns, edges, and textures.
- The output of the CNN is a feature vector that encodes the salient visual information of the image.

Text Generation using Recurrent Neural Networks (RNNs)

- Once the image features are obtained, they are combined with the textual context to generate captions.
- Recurrent Neural Networks (RNNs), such as Long Short-Term Memory (LSTM) or Gated Recurrent Units (GRUs), are commonly used for sequential data generation, including language generation.
- RNNs process the image features along with previously generated words to predict the next word in the caption.
- Alternatively, Transformer models, known for their self-attention mechanism, can be used to generate captions. Transformers capture global dependencies and context, improving the coherence and quality of generated captions.

Architecture

By following these steps, image captioning systems effectively utilize both visual information and textual context to generate accurate and meaningful descriptions for images.

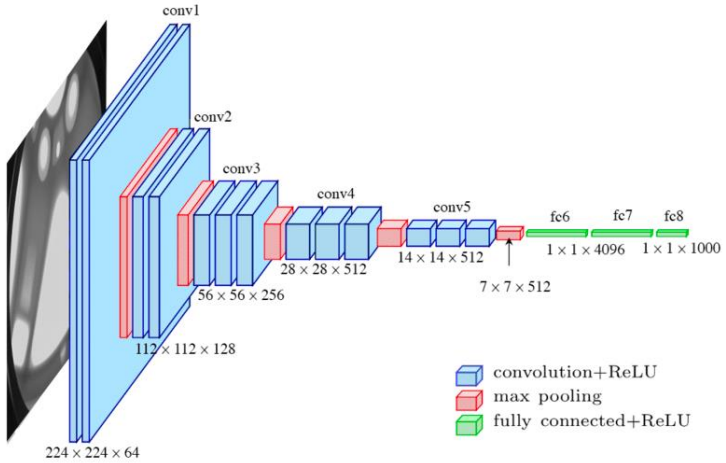


NEW HORIZON
COLLEGE OF ENGINEERING

Convolutional Neural Networks (CNNs) for Image Feature Extraction

CNN

- Convolutional Neural Networks (CNNs) are a popular and effective architecture for extracting features from images.
- Some popular CNN models include VGG16, ResNet (Residual Network), and Inception.
- These models are pre-trained on large-scale image datasets, such as ImageNet, and have learned to recognize a wide range of visual patterns and concepts.



Hierarchical Feature Learning:

NEW HORIZON
COLLEGE OF ENGINEERING

- One of the key strengths of CNNs is their ability to learn hierarchical features from images.
- By employing convolutional and pooling layers, CNNs capture low-level features like edges and textures in early layers and gradually learn high-level semantic features in deeper layers.
- This hierarchical representation allows CNNs to capture both low-level details and high-level semantic concepts, enabling them to understand complex visual scenes.

With their ability to automatically learn meaningful features from images, CNNs serve as a vital component in image captioning systems by providing rich visual representations for generating accurate and descriptive captions.

NEW HORIZON
COLLEGE OF ENGINEERING

Recurrent Neural Networks (RNNs) for Caption Generation

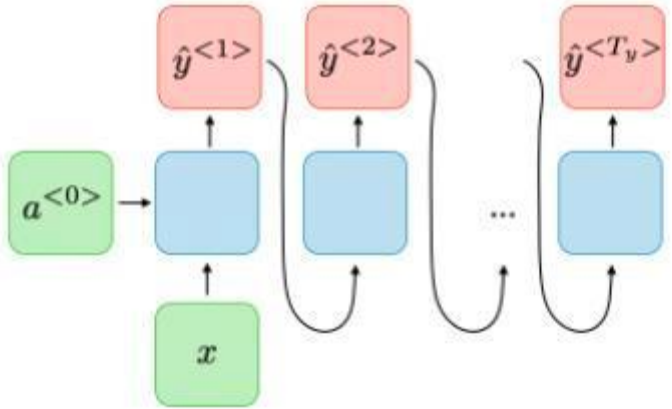
RNNs:

NEW HORIZON
COLLEGE OF ENGINEERING

- Recurrent Neural Networks (RNNs) are sequential models well-suited for generating captions or sequences of words.
- Unlike feed-forward neural networks, RNNs have feedback connections that allow information to be looped back and processed iteratively.
- This looping mechanism enables RNNs to maintain an internal memory or hidden state, which helps them capture dependencies and context across sequential data.
- https://www.kaggle.com/code/vivekgediya/image-caption-vgg16-lstm

RNN in Text Generation

- RNNs process sequences one element at a time, making them ideal for language generation tasks.
- In image captioning, RNNs take image features as input and generate words one at a time, taking into account the previous context.
- At each step, the RNN produces a probability distribution over the vocabulary, and the most likely word is chosen as the output.
- This sequential nature allows RNNs to generate captions that are coherent and contextually relevant.



Popular RNN Variants:

NEW HORIZON
COLLEGE OF ENGINEERING

- Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) are popular variants of RNNs frequently used in image captioning.
- LSTM and GRU architectures address the vanishing gradient problem in traditional RNNs, allowing them to capture long-term dependencies more effectively.
- They achieve this by introducing gating mechanisms that control the flow of information through the hidden state, enabling better gradient flow and memory retention.

RNNs, especially LSTM and GRU variants, are widely used in image captioning due to their sequential processing capabilities, allowing them to generate coherent and contextually relevant captions for images.

Image Generative Adversarial Network

INTRODUCTION

NEW HORIZON
COLLEGE OF ENGINEERING

- ***Machines are generating perfect images these days and it's becoming more and more difficult to distinguish the machine-generated images from the originals.***

EXAMPLES

Machine Generated Digits using MNIST

Images Generated by Nvidia's StyleGAN

HOW TO GENERATE?

NEW HORIZON
COLLEGE OF ENGINEERING

- To generate -well basically- anything with machine learning, we have to use a **generative algorithm** and at least for now, one of the best performing generative algorithms for image generation is Generative Adversarial Networks (or GANs).
- **Generative Algorithms try to model "how to populate the dataset."** Sampling the model gives generated, synthetic data points.
- Generative Adversarial Networks (GANs)
- Gaussian Mixture Model (GMM)
- Hidden Markov Model (HMM)
- Probabilistic context-free grammar (PCFG

STRUCTURE OF THE GANS

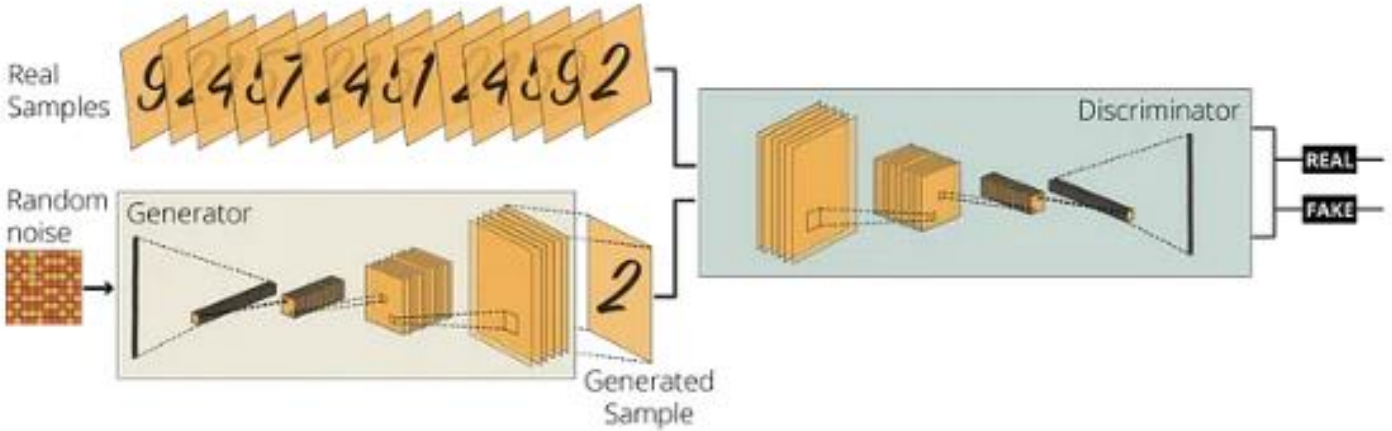


Figure 4. Generative Adversarial Networks (GANs) utilizing CNNs | (Graph by author)

HOW DOES GAN MODEL OPERATE?

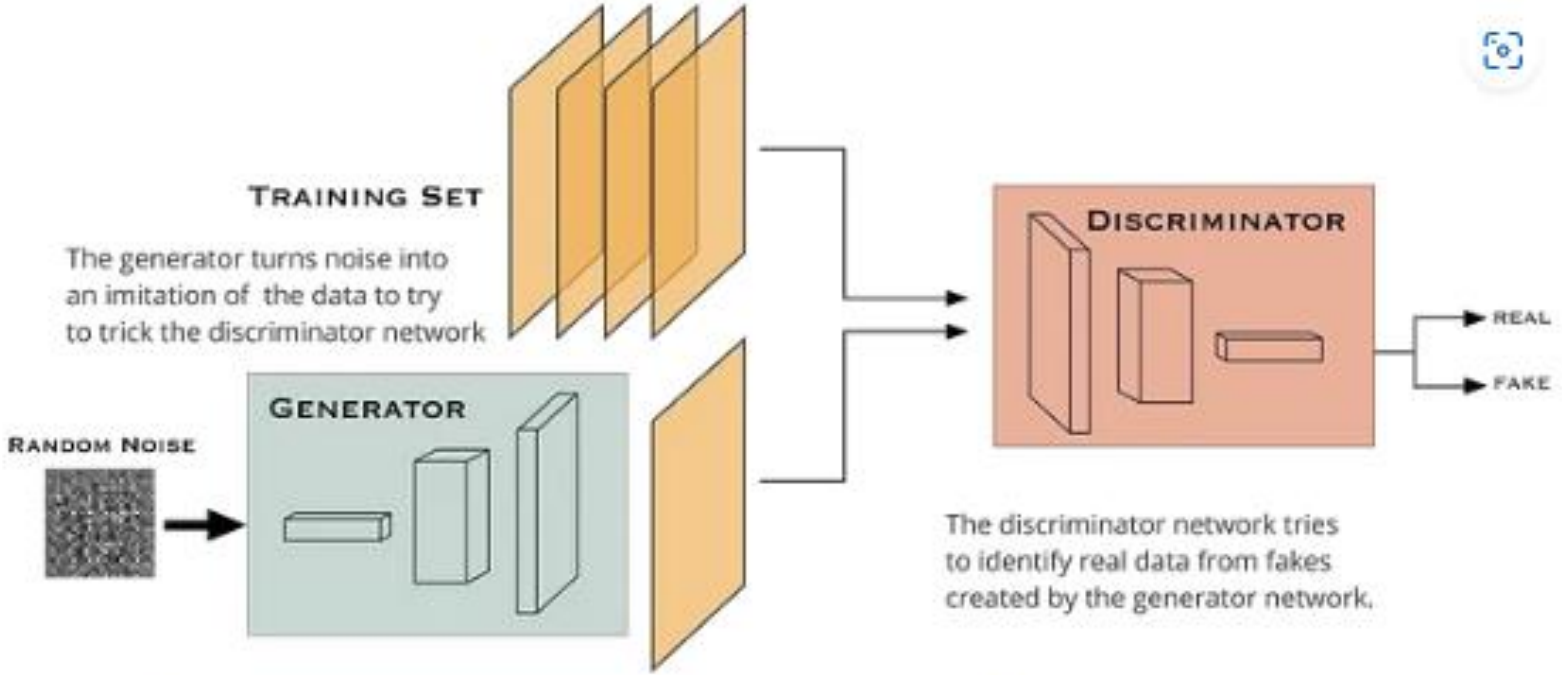


Figure 5. Generator and Discriminator Relationship in a GAN Network | (Graph by author)

HOW DOES GAN MODEL OPERATE?

NEW HORIZON
COLLEGE OF ENGINEERING

- In a nutshell, we will ask the generator to generate handwritten digits without giving it any additional data.
- Simultaneously, we will fetch the existing handwritten digits to the discriminator and ask it to decide whether the images generated by the Generator are genuine or not.
- At first, the Generator will generate lousy images that will immediately be labeled as fake by the Discriminator.
- After getting enough feedback from the Discriminator, the Generator will learn to trick the Discriminator as a result of the decreased variation from the genuine images.
- Consequently, we will obtain a very good generative model which can give us very realistic outputs

- GAN must have at least one generator and one discriminator.
- Since we are dealing with image data, we need to benefit from **Convolution and Transposed Convolution** (Inverse Convolution) layers in these networks.
- Transposed convolutions are usually used in **auto-encoders and GANs, or generally any network that must reconstruct an image**.

- **Generator Network**
- Our generator network is responsible for generating 28x28 pixels grayscale fake images from random noise.
- Therefore, it needs to accept 1-dimensional arrays and output 28x28 pixels images. For this task, we need Transposed Convolution layers after reshaping our 1-dimensional array to a 2-dimensional array.

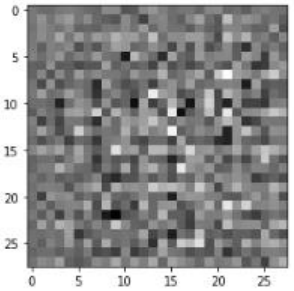


Figure 8. A Sample Image Generated by Non-Trained Generator Network | (Image by author)

Discriminator Network

- For our discriminator network, we need to follow the **inverse version of our generator network**. It takes the 28x28 pixels image data and outputs a single value, representing the possibility of authenticity. So, our discriminator can review whether a sample image generated by the generator is fake.
- Finally, we can check what our non-trained discriminator says about the sample generated by the non-trained generator
- tf.Tensor([[**-0.00108097**]], shape=(1, 1), dtype=float32)
- A negative value shows that our non-trained discriminator concludes that the image sample in is fake.

TRAIN THE MODEL

NEW HORIZON
COLLEGE OF ENGINEERING

- GANs require custom training loops and steps
- **During the Training**:
- Start recording time spent at the beginning of each epoch;
- Produce GIF images and display them,
- Save the model every five epochs as a checkpoint,
- Print out the completed epoch time; and
- Generate a final image in the end after the training is completed.

Figure 10. The Digits Generated by Our GAN after 60 Epochs. Note that we are seeing 16 samples because we configured our output this way. | (Image by author)

CONCLUSION

NEW HORIZON
COLLEGE OF ENGINEERING

- Once you can build and train this network, you can generate much more complex images,
- by working with a larger dataset with colored images in high definition;
- by creating a more sophisticated discriminator and generator network;
- by increasing the number of epochs;
- by working on a GPU-enabled powerful hardware

NEW HORIZON
COLLEGE OF ENGINEERING

- Attention Mechanism in Deep Learning- Scaler Topics

Attention Mechanism in Deep Learning

NEW HORIZON
COLLEGE OF ENGINEERING

- Attention is the cognitive process of selectively concentrating on one or a few things while ignoring others.
- A neural network is considered to be an effort to mimic human brain actions in a simplified manner.
- Attention Mechanism is also an attempt to implement the same action of selectively concentrating on a few relevant things, while ignoring others in deep neural networks.

Example

NEW HORIZON
COLLEGE OF ENGINEERING

- Let's say you are seeing a group photo of your first school. Typically, there will be a group of children sitting across several rows, and the teacher will sit somewhere in between.
- Now, if anyone asks the question, "How many people are there?", how will you answer it?
- Simply by counting heads, right? You don't need to consider any other things in the photo.
- Now, if anyone asks a different question, "Who is the teacher in the photo?", your brain knows exactly what to do. It will simply start looking for the features of an adult in the photo. The rest of the features will simply be ignored.
- **This is the 'Attention' which our brain is very adept at implementing.**

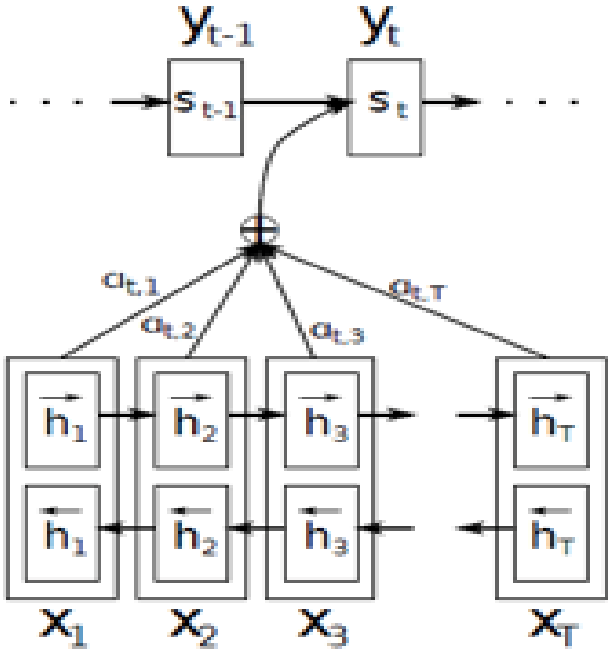
- The attention mechanism emerged as an improvement over the encoder decoder-based neural machine translation system in natural language processing (NLP).
- Later, this mechanism, or its variants, was used in other applications, including computer vision, speech processing, etc.
- first Attention model in 2015, neural machine translation was based on encoder-decoder RNNs/LSTMs.
- Both encoder and decoder are stacks of LSTM/RNN units. It works in the two following steps:

1. **The encoder LSTM is used to process the entire input sentence and encode it into a context vector**, which is the last hidden state of the LSTM/RNN. This is expected to be a good summary of the input sentence. All the intermediate states of the encoder are ignored, and the final state id supposed to be the initial hidden state of the decoder
2. **The decoder LSTM or RNN units produce the words in a sentence one after another**

- In short, there are two RNNs/LSTMs. One we call the encoder – this reads the input sentence and tries to make sense of it, before summarizing it.
- It passes the summary (context vector) to the decoder which translates the input sentence by just seeing it.
- The main drawback of this approach is evident. If the encoder makes a bad summary, the translation will also be bad. It is called the **long-range dependency problem of RNN/LSTMs.**

- RNNs cannot remember longer sentences and sequences due to the vanishing/exploding gradient problem.
- It can remember the parts which it has just seen. **The performance of the encoder-decoder network degrades rapidly as the length of the input sentence increases.**
- Although an LSTM is supposed to capture the long-range dependency better than the RNN, it tends to become forgetful in specific cases.
- Another problem is that there is no way to give more importance to some of the input words compared to others while translating the sentence.

Attention Mechanism



- The Bidirectional LSTM used here generates a sequence of annotat*ions (h1, h2,....., hTx)* for each input sentence.
- All the vectors h1,h2.., etc., used in their work are basically the concatenation of forward and backward hidden states in the encoder.

$$h_j = \left[\overrightarrow{h}_j^\top ; \overleftarrow{h}_j^\top\right]^\top$$

- the weights are also learned by a feed-forward neural network .
- The context vector ci for the output word yi is generated using the weighted sum of the annotations:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j .$$

- The weights αij are computed by a softmax function

- These attention weights αi are used to build the context vector to input decoder.
- It enables the decoder to access the entire decoded input vector to generate the output.
- Each token of this context vector is a weighted sum of all candidate states of the encoder and their respective weights.