Ministry of Science and Higher Education of the Republic of Kazakhstan
L.N. Gumilyov Eurasian National University

Faculty of Information Technology
Department of Information Systems

YOLO & ITS ALTERNATIVES

Done by: Zhumabek A.R
Check: Zhukabaeva T.K

WHAT IS YOLO

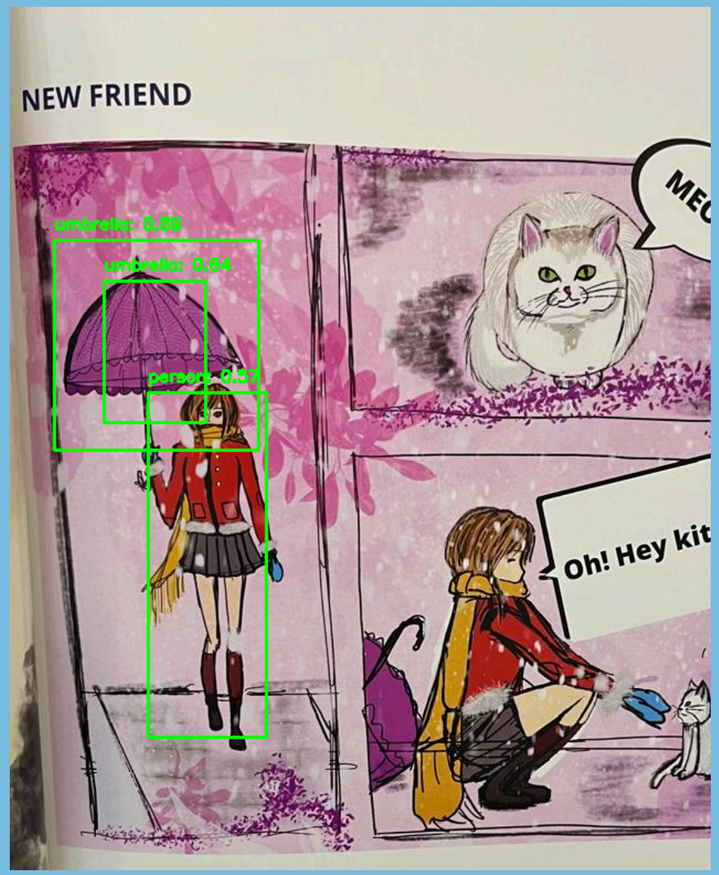
Y.O.L.O.

YOU ONLY LIVE ONCE

YOLO (You Only Look Once) is a real-time object detection algorithm that uses a single neural network for both classification and predicting bounding boxes, making it faster than traditional multi-step processes. YOLOv1, the first version, can detect objects at 45 frames per second, making it ideal for real-time applications. It analyzes the entire image at once, reducing false positives compared to methods like R-CNN, which look at parts of the image separately. YOLO is also good at generalizing across different environments.

How works?

YOLO divides the image into a grid and for each cell it predicts class probabilities and coordinates of bounding boxes. It then filters out weak predictions, merges intersecting frames (NMS), and outputs the final objects. All this happens in a single pass of the neural network, making YOLO fast and efficient.



YOLO & ITS ALTERNATIVES

Today, YOLO remains one of the most popular algorithms for object detection due to its speed and efficiency. However, depending on the task, YOLO may not always be the best choice. For example, if we need maximum accuracy under difficult conditions, if we are working with very small objects, or if we need detection on mobile devices, it is worth considering other options.

There are many analogs and alternatives to YOLO that offer different approaches to solving the problem of object detection. Some of them develop the ideas of YOLO, making it even faster and more accurate. Others use a completely different principle of operation, providing higher accuracy or better adaptation to complex scenes.

Let's take a look at the main alternatives and their features.

Alternatives (other detection methods)

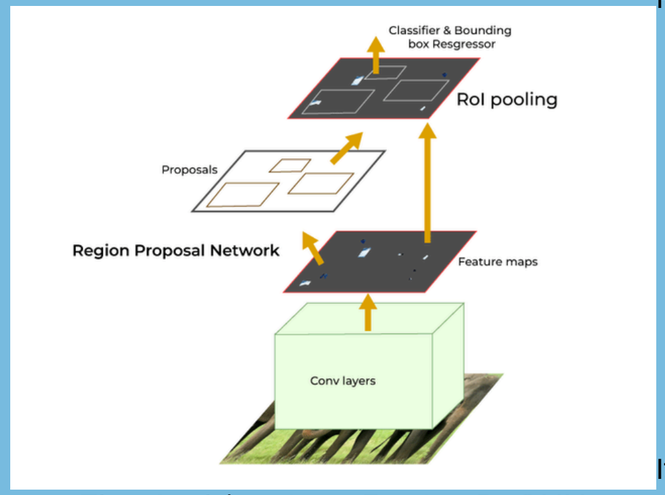
- Faster R-CNN - two-stage model, slower but more accurate for complex objects.
- RetinaNet - uses Focal Loss, works well on small objects.
- DETR (DEtection TRansformer) - uses transformers instead of CNNs, works well on complex scenes.

FASTER R-CNN

Faster R-CNN is one of the best object detection algorithms, and it uses a regional proposal network to generate object bounding boxes. It is highly accurate when region proposals and object classification are used.

Features

- Faster R-CNN will generate region proposals by checking the anchor boxes at different scales and aspect ratios.
- It generates a two-stage detection pipeline where, in the first stage, it creates regional proposals, and in the second phase, it filters and classifies the proposals.
- Extract fixed-size feature maps from the complex feature maps depending on the region proposals.
- Handles the regions of different sizes and locations, making it invariant at various locations and scales.



The Architecture Facter R-CNN

Faster R-CNN works like this:

1. Conv Layers – The image goes through convolutional layers to extract important features.
2. Region Proposal Network (RPN) – Suggests possible object locations (proposals) based on feature maps.
3. RoI Pooling – Extracts fixed-size regions from proposals for classification.
4. Classifier & Bounding Box Regressor – Determines the object category and refines its bounding box.

It's accurate but slower compared to YOLO due to the extra proposal step.

Pros	Cons
Generates region proposals for different types of objects embedded in an image.	Takes a lot of time to detect objects.
Has the ability to detect over 80 classes.	The architecture used by this tool has some aspects that could be improved.

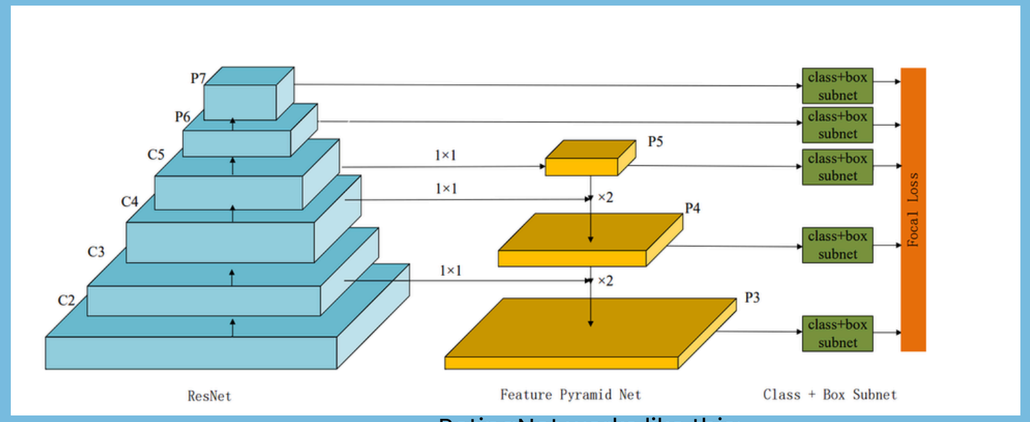
RETINANET

RetinaNet is one of the best object detection models and an alternative to YOLO, which uses a pyramid network and focal loss function. It has developed new techniques to address the critical challenges encountered in object detection.

Features

- They are designed to handle various object detection tasks of multiple sizes.
- It contains a multi-scale feature to let the model detect different objects of different scales.
- Detects small to large objects in the same image, resulting in model accuracy.
- The focal loss function assigns weight to hard-to-classify objects for easy classification.

RetinaNet Architecture



RetinaNet works like this:

1. ResNet (C2-C5) - Used as a basis for extracting features from an image.
2. Feature Pyramid Network (P3-P7) - Creates multi-level representations of objects of different sizes.
3. Class + Box Subnet - Separate subnets predict object classes and their coordinates.
4. Focal Loss - Allows better recognition of complex and small objects, reducing the influence of easy-to-classify examples.

High accuracy, especially for detection of small objects, but requires more computational resources.

Pros	Cons
Able to handle class imbalance, resulting in accuracy.	Encounters difficulties in detecting small objects from images.
Designed to run efficiently and is easy to deploy on low-end devices.	Requires diverse dataset training to attain optimal performance.

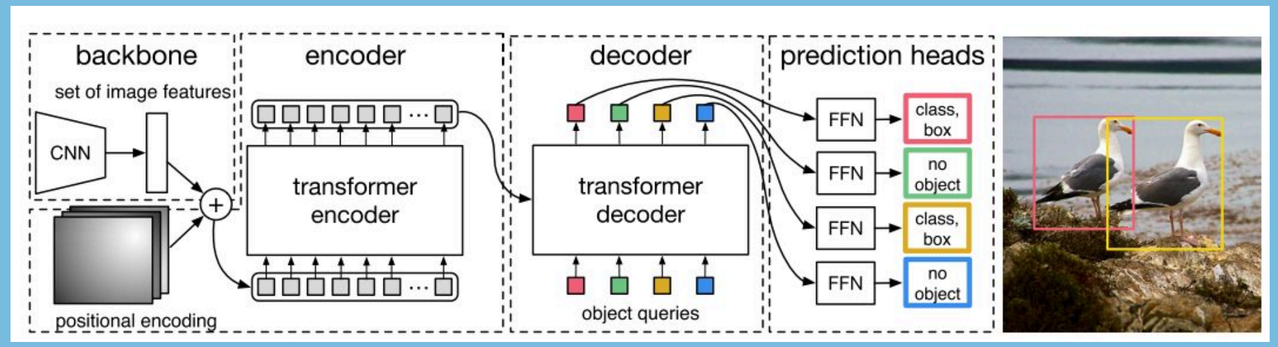
DETR (DETECTION TRANSFORMER)

DETR is the best object detection deep learning algorithm that plays a crucial role in computer vision. It uses transformers' power to predict object classes and bounding boxes.

Features

- Apply transformers to detect objects seamlessly.
- It uses self-attention to process the images holistically.
- Predicts the position of objects and corresponding classes from the input image.
- Eliminate the need for anchor boxes to streamline the detection process.

DETR Architecture



DETR (DEtection TRansformer) works like this:

1. Backbone (CNN) - Extracts image features.
2. Encoder (Transformer) - Adds position coding and processes the features.
3. Decoder (Transformer) - Uses object queries for object prediction.
4. Prediction Heads - Determines the class and coordinates of objects (or lack of objects).

DETR efficiently handles complex scenes and overlapping objects, but requires more data and computational resources.

Pros	Cons
Uses transformer-based architecture to detect objects accurately.	Computationally expensive to attain optimal performance.
Has the capability to handle overlapping objects easily.	Large datasets are required; smaller datasets lead to suboptimal performance.

COMPARISON YOLO WITH ALTERNATIVES

Model	Type	Speed ⚡	Accuracy 🎯	Strengths ✅	Weaknesses ❌
YOLO (You Only Look Once)	One-stage	🚀 Fastest	🎯 Good	Real-time detection, great for embedded systems	Struggles with small objects
Faster R-CNN	Two-stage	🐢 Slow	🎯🎯 High	Best for complex objects and high-accuracy tasks	High computational cost
RetinaNet	One-stage	⚡ Fast	🎯🎯 High	Focal Loss handles class imbalance, good for detecting small objects	Slower than YOLO
DETR (DEtection TRansformer)	Transformer-based	🐢 Slowest	🎯🎯🎯 Very High	Handles overlapping objects, works well in complex scenes	Needs a large dataset, computationally expensive

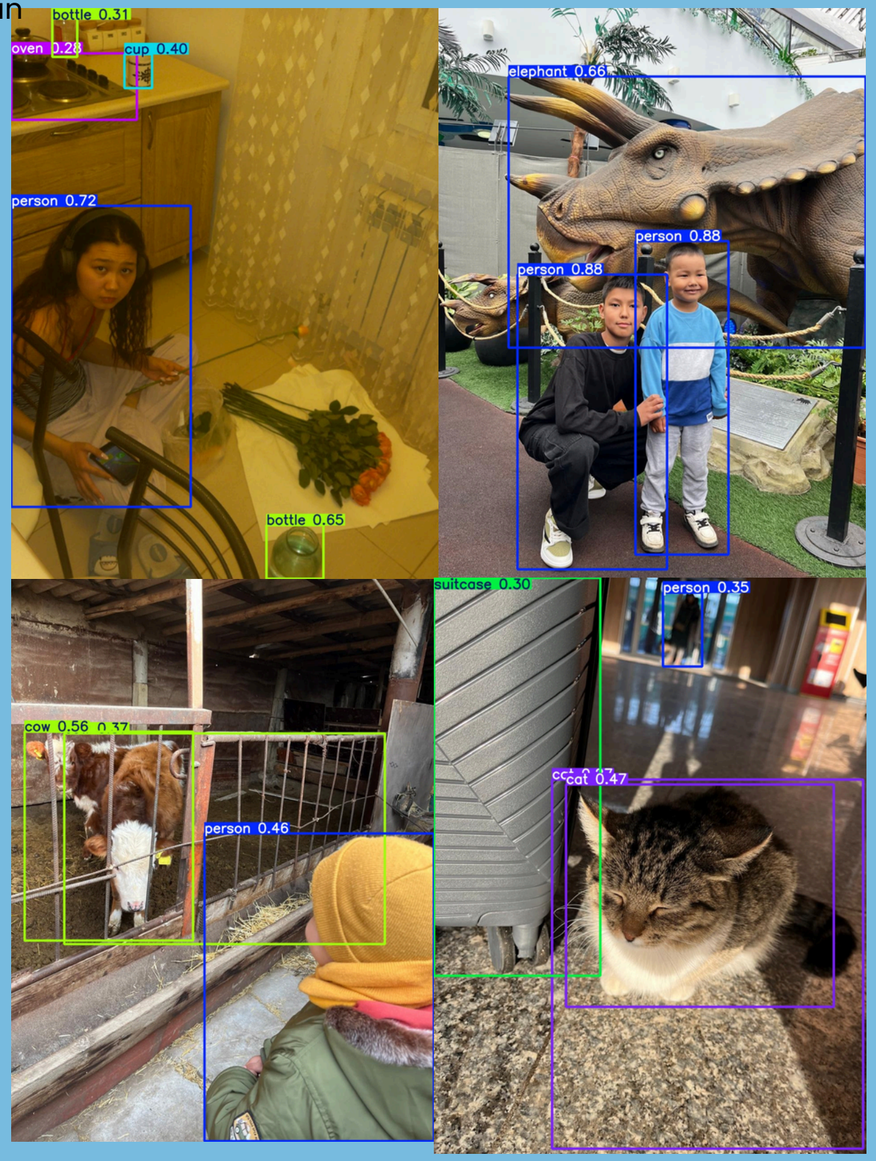
If we need maximum speed - YOLO.
If we prioritize accuracy and complex scenes - Faster R-CNN or DETR.
If we are looking for balance - RetinaNet.

CONCLUSION

As to me the most convenient yolo, with it:
Fast - will work even on a weak computer.
Simple - easy to start, lots of tutorials.
One-step - immediately searches for objects without extra steps.
But if we have powerful hardware and need maximum accuracy for complex scenes, for example, if objects overlap, we can try DETR or Faster R-CNN. And if there is a balance between accuracy and speed - RetinaNet.
Many people say that YOLO is the easiest way to start!

This code recognizes objects using the YOLOv8 model and stores the results in an image.

```python
# install YOLOv8
!pip install ultralytics

# import libraries
from ultralytics import YOLO
import cv2
import torch
from google.colab.patches import cv2_imshow

device = "cuda" if torch.cuda.is_available() else "cpu"
print(f"use {device}")

# model yolo8
model = YOLO("yolov8n.pt").to(device)

results = model("/content/1.jpg")
results[0].save(filename="/content/1.jpg")

img = cv2.imread("/content/1.jpg")
cv2_imshow(img)
```



\What makes YOLOv8 better?
✅ Faster and more accurate than previous versions.
✅ Can be used for different tasks (object detection, segmentation, classification).
✅ Works even on a regular computer (but faster with GPU).

Where is it used?
🚗 Video surveillance (detection of people, cars).
🤖 Robotics (recognizing objects).
📱 Mobile applications (photo and video analysis).

THANK YOU

References

1. https://medium.com/analytics-vidhya/yolo-explained-5b6f4564f31
2. https://www.geeksforgeeks.org/yolo-you-only-look-once-alternatives/#detr
3. https://viso.ai/deep-learning/detr-end-to-end-object-detection-with-transformers/
4. https://viso.ai/deep-learning/retinanet/
5. https://www.geeksforgeeks.org/faster-r-cnn-ml/