Localization in Deep Learning: Mechanisms of Detection, Matching, and Learning in Visual Recognition Systems

I. Introduction to Localization in Deep Learning Models

A. The Imperative of Localization in Computer Vision

Localization, the capacity to determine the spatial or temporal extent of specific entities within input data such as images or videos, represents a cornerstone of sophisticated visual understanding systems.[1] This capability transcends simple image-level classification (i.e., identifying *what* objects are present) by additionally addressing the crucial question of *where* these objects are situated.[3] The ability to precisely localize objects or events is not merely an academic pursuit; it forms the bedrock for a multitude of real-world applications. These span diverse fields, including autonomous navigation, where vehicles must identify and locate other road users and infrastructure [4]; medical image analysis, for tasks like tumor detection and organ segmentation [3]; robotics, enabling machines to interact with their environment [4]; and security/surveillance systems for monitoring and event detection.[4] The output of localization processes is inherently structural, commonly taking the form of bounding boxes in object detection or defined temporal segments in tasks like video moment retrieval.

The evolution of localization techniques within deep learning mirrors a broader trajectory in the field: a persistent movement away from fragmented, heuristic-driven multi-stage pipelines towards more cohesive, end-to-end learnable architectures. Early methodologies in object detection, for instance, often depended on distinct, non-learnable algorithms for critical sub-tasks, such as employing Selective Search for generating region proposals in the original R-CNN model.[6] Subsequent advancements focused on integrating these disparate components. Fast R-CNN introduced shared feature computation [6], and Faster R-CNN further incorporated a learnable Region Proposal Network (RPN).[6] Ultimately, models like YOLO [7] and DETR [9] championed unified architectures that perform localization in a single, integrated process. This architectural consolidation often translates to enhancements in both performance and computational efficiency, reflecting a fundamental principle in deep learning where learnable modules progressively supersede manually engineered elements.

B. Defining Core Concepts: Detection Box Prediction, Matching, and Learning

Understanding how models acquire localization capabilities necessitates a clear definition of three interconnected processes:

1. **Detection Box Prediction:** This is the mechanism through which a model generates hypotheses regarding the location and spatial extent of objects or temporal segments. These hypotheses are typically parameterized; for instance, a 2D bounding box is commonly represented by its center coordinates, width, and height (x,y,w,h). Models may predict these parameters directly or, more commonly, as adjustments (offsets) relative to a set of predefined reference structures, such as anchor boxes [6] or default boxes [10], or even per-pixel reference points in anchor-free approaches.[11]
2. **Detection Box Matching (Label Assignment):** During the training phase, the numerous predicted detection boxes must be systematically associated with the ground truth (GT) annotations. This matching process is pivotal as it categorizes each prediction as either a "positive" sample (a correct detection of a GT object) or a "negative" sample (a false positive or background detection). The outcome of this assignment directly informs the computation of the training loss. Matching strategies vary in complexity, from straightforward Intersection over Union (IoU) thresholding, where a prediction is deemed positive if its IoU with a GT box surpasses a certain value [10], to more sophisticated methods like bipartite matching employed in DETR [9] or optimal transport formulations seen in OTA.[13]
3. **Detection Box Learning:** This encompasses the iterative process by which the model refines its ability to predict accurate box coordinates or temporal boundaries. It is almost universally driven by the definition of a loss function that quantifies the discrepancy between the parameters of matched predicted boxes and their corresponding GT boxes. The gradients derived from this localization loss are then backpropagated through the network to update its learnable parameters, thereby guiding the model towards improved prediction accuracy.[10]

The continuous refinement of matching strategies and the development of more sophisticated loss functions have been instrumental in advancing localization accuracy. Rudimentary loss functions may fail to capture all the desired geometric properties of an accurate bounding box, and naive matching schemes can introduce ambiguity or provide suboptimal supervisory signals during training. For example, the inherent limitations of basic L1 or L2 losses for direct bounding box coordinate regression [14] catalyzed the adoption of Smooth L1 loss [16], which offers greater stability. Similarly, the well-documented issues with the standard IoU loss, such as vanishing gradients for non-overlapping boxes [14], spurred the innovation of more advanced IoU-based losses like GIoU, DIoU, CIoU, and EIoU.[17] Each of these was designed to address specific geometric nuances and provide more informative gradients, illustrating a direct causal relationship: an identified deficiency in an existing loss or matching technique prompts the development of a more robust solution.

C. Scope of this Report

This report aims to provide a comprehensive examination of the mechanisms underpinning localization in deep learning models, with a primary focus on two prominent tasks: Object Detection (OD) in images and Text-based Video Moment Retrieval (MR). The investigation will particularly emphasize *how* different algorithmic families generate detection boxes or temporal spans, *how* these predictions are matched against ground truth annotations during the training process, and *which specific loss functions* are employed to facilitate the learning of precise localization. The analysis will trace the evolution of these techniques, from seminal multi-stage methodologies to contemporary end-to-end deep learning systems, providing the reader with a thorough understanding of the current state-of-the-art and the foundational principles driving it.

II. Paradigms for Detection Box Prediction and Matching in Object Detection

The dominant paradigm in object detection has long been to "generate more detection boxes than ground truth boxes, then match," followed by refinement and filtering. This section dissects this paradigm by examining its evolution through various influential families of algorithms.

A. Two-Stage Detectors: The R-CNN Family – Pioneering Region-Based Detection

The R-CNN lineage marked a significant departure from earlier sliding-window techniques by introducing a region-based approach, which selectively processes promising areas of an image. This family of models demonstrates a clear evolutionary path towards greater integration and efficiency.

1. **R-CNN (Regions with CNNs):**
   - **Box Prediction:** R-CNN initiated its process by employing an external, traditional computer vision algorithm, typically Selective Search, to generate approximately 2000 region proposals, or Regions of Interest (RoIs), per image.[6] These RoIs served as the initial candidate detection boxes.
   - **Learning:** Each of these RoIs was independently warped to a fixed size and then fed through a Convolutional Neural Network (CNN) to extract features. Subsequently, a set of Support Vector Machines (SVMs) were trained to classify these features into object categories or background. A separate linear regression model was trained for each class to refine the coordinates of the bounding boxes for positively classified RoIs.[6]

    - **Matching:** The matching was implicit in the training of the components. For the SVMs, RoIs with sufficient overlap (e.g., IoU > 0.5) with a ground truth box were considered positive examples for that object's class. For the bounding box regressors, positive samples were RoIs that had a high IoU with a ground truth box, and the regression targets were the transformations needed to morph the RoI into the GT box.
2. **Fast R-CNN: Accelerating Training and Inference:**
    - **Box Prediction:** Fast R-CNN still relied on external region proposal methods like Selective Search, similar to its predecessor.[6]
    - **Key Innovation (RoI Processing & Learning):** The principal innovation of Fast R-CNN was to process the entire image through the CNN *once* to generate a global feature map.[6] The RoIs (still generated externally) were then projected onto this feature map. A novel RoI Pooling layer was introduced to extract a fixed-length feature vector for each RoI from this shared feature map.[6] This sharing of convolutional computations across all RoIs dramatically reduced processing time compared to R-CNN.
    - **Matching and Learning:** Fast R-CNN introduced a multi-task loss function, enabling the joint optimization of classification and bounding-box regression in a single training stage, unlike the separate stages in R-CNN.[12]
        - **RoI Sampling for Training:** During training, Stochastic Gradient Descent (SGD) mini-batches were constructed hierarchically. First, N images were sampled, and then R/N RoIs were sampled from each image. A common strategy was to select 25% of RoIs from object proposals having an IoU overlap with a ground-truth bounding box of at least 0.5; these were labeled with a foreground object class ($u\geq1$). The remaining RoIs were sampled from proposals with a maximum IoU with ground truth in the interval [0.1, 0.5), considered background examples and labeled $u=0$.[12] This threshold-based IoU matching was fundamental.
        - **Loss Function:** The multi-task loss L for each labeled RoI was defined as $L(p,u,tu,v)=Lcls(p,u)+\lambda[u\geq1]Lloc(tu,v)$. Here, $Lcls(p,u)=-logpu$ is the log loss for the true class u. Lloc is the Smooth L1 loss between the predicted bounding-box regression offsets $tu=(txu,tyu,twu,thu)$ for class u, and the ground-truth regression targets $v=(vx,vy,vw,vh)$. The Iverson bracket $[u\geq1]$ ensures that Lloc is only active for foreground RoIs (i.e., when $u\geq1$). The hyperparameter λ (typically set to 1) balances the two task losses. The ground-truth regression targets vi were normalized to have zero mean and unit variance.[12]
3. **Faster R-CNN: Integrating Region Proposal:**
    - **Key Innovation (Region Proposal Network - RPN):** The most significant

advancement of Faster R-CNN was the introduction of the Region Proposal Network (RPN). The RPN replaced the slow, CPU-intensive Selective Search algorithm with a fully convolutional network that shares convolutional layers with the main detection network.[6] This integration made the region proposal step learnable and significantly faster by leveraging GPU computation.

  - **Anchor Boxes for Prediction:** The RPN operates by sliding a small neural network over the convolutional feature map produced by the shared backbone. At each sliding-window location, the RPN simultaneously predicts objectness scores (whether an object is present) and refines coordinates for multiple "anchor boxes".[6] These anchor boxes are a set of predefined reference boxes with varying scales and aspect ratios, centered at the sliding window location. They serve as initial guesses or priors for potential object locations and sizes.
  - **RPN Training (Matching & Learning):** Anchors are assigned labels based on their IoU with ground-truth boxes. An anchor is typically labeled positive if its IoU with any GT box is > 0.7, or if it has the highest IoU for a particular GT box. It's labeled negative if its IoU with all GT boxes is < 0.3. Anchors that fall between these thresholds or cross image boundaries are usually ignored during RPN training. The RPN is trained with a multi-task loss: a binary classification loss for objectness and a regression loss (e.g., Smooth L1) for the coordinates of positive anchors.
  - **Detection Network Training:** After the RPN generates proposals, the rest of the network (the detector component) is trained similarly to Fast R-CNN, using these RPN-generated proposals as input RoIs.

The progression from R-CNN to Faster R-CNN illustrates a clear trend: from disjointed, computationally expensive components towards increasingly integrated and learnable architectures. R-CNN's independent processing of thousands of RoIs created a significant bottleneck.[6] Fast R-CNN's introduction of shared convolutional features and RoI pooling was a direct response to this, dramatically improving speed.[6] However, the region proposal mechanism (Selective Search) remained an external, slow element. Faster R-CNN addressed this by making the proposal generation itself a learnable neural network (the RPN), enabling end-to-end training and GPU acceleration.[6] This iterative integration of pipeline stages into the neural network is a characteristic theme in the advancement of deep learning systems.

The introduction of anchor boxes in Faster R-CNN was a pivotal development. Unlike the heuristic-based Selective Search, anchors provided a systematic and dense set of prior boxes across various scales and aspect ratios, directly linked to locations on the feature map.[6] The RPN then learns to select and refine these predefined anchors. This discretization combined with learnable refinement proved more amenable to deep learning optimization than relying on external, non-learnable proposal methods.

Throughout this evolution, IoU-based matching strategies remained fundamental. They define which proposals or anchors are considered "positive" (targets for learning) and "negative" (background or incorrect). For example, the multi-task loss in Fast R-CNN explicitly uses the class label u (derived from IoU matching) to gate the localization loss Lloc via the [u≥1] term.[12] If an RoI does not significantly overlap with a ground truth object, it is labeled as background (u=0), and no localization learning occurs for it. This demonstrates the direct causal link: the matching strategy defines the positive samples, and only these positive samples are used to train the bounding box regressor.

B. One-Stage Detectors: Efficiency through Unified Prediction

Driven by the need for higher inference speeds for real-time applications, one-stage detectors emerged as an alternative to the multi-stage R-CNN family. These models aim to perform object detection in a single pass through the network.

1. **YOLO (You Only Look Once) - especially YOLOv1:**
   - **Unified Detection (Box Prediction & Learning):** YOLO reframed object detection as a single regression problem, directly mapping from input image pixels to bounding box coordinates and class probabilities.[3] This represented a significant paradigm shift.
   - **Grid System:** The input image is divided into an S×S grid (e.g., 7×7 in YOLOv1).[7] Each grid cell is designated as responsible for detecting objects whose center points fall within that cell's boundaries.
   - **Output Tensor Structure:** For each of the S×S grid cells, the network predicts:
     - B bounding boxes (e.g., B=2). Each bounding box prediction consists of 5 values: (x,y,w,h) and a confidence score C.[7]
       - The coordinates (x,y) represent the center of the box relative to the bounds of the grid cell. The width w and height h are predicted relative to the dimensions of the whole image.
       - The **confidence score** C is intended to reflect Pr(Object)×IoU(pred, GT). If no object is present in the grid cell, the confidence score should ideally be zero. Otherwise, it should approximate the IoU between the predicted box and the ground truth box.[8]
     - K class probabilities, Pr(Classk∣Object), conditional on an object being present in the grid cell.[8] Importantly, YOLOv1 predicts only one set of class

    probabilities per grid cell, irrespective of the number of bounding boxes B predicted by that cell.
  - The final output of the network is an S×S×(B×5+K) tensor.[20]
- **Matching during Training:** A key aspect of YOLOv1's training is its assignment strategy. For each ground truth object in an image, only one grid cell is deemed "responsible" for its detection: the grid cell that contains the center of that ground truth object.[20] Within this responsible grid cell, among the B bounding box predictors, the one whose current prediction has the highest IoU with the ground truth box is designated as "responsible" for predicting that specific object. All other predictors in that cell (and all predictors in other cells not containing the object's center) are not responsible for that particular ground truth object.
- **Loss Function (Multi-part Sum-Squared Error):** YOLOv1 employs a multi-part loss function, primarily based on sum-squared errors, to penalize different types of prediction inaccuracies [20]:
  - **Localization Loss (Coordinate Error):** This component penalizes errors in the predicted center coordinates (x,y) and dimensions (w,h). It is calculated as the sum of squared errors for x,y and for w,h. This loss is applied only for the *responsible* bounding box predictor in grid cells that *contain* an object (indicated by Iijobj where i is the grid cell and j is the predictor). The use of square roots for width and height, w and h, is intended to make the loss less sensitive to absolute errors in large boxes compared to small boxes. This part of the loss is typically weighted more heavily using a hyperparameter λcoord (e.g., 5).
    Lcoord=λcoordΣi=0S2Σj=0BIijobj[(xi−x^i)2+(yi−y^i)2]
    +λcoordΣi=0S2Σj=0BIijobj[(wi−w^i)2+(hi−h^i)2]
  - **Confidence Loss (Objectness Error):** This loss has two parts:
    - For the *responsible* predictor in grid cells *containing* an object: Σi=0S2Σj=0BIijobj(Ci−C^i)2. Here, the target confidence C^i is the IoU between the predicted box and the ground truth.
    - For *all* predictors in grid cells *not containing* an object (indicated by Iijnoobj): λnoobjΣi=0S2Σj=0BIijnoobj(Ci−C^i)2. Here, the target confidence C^i is 0. The hyperparameter λnoobj (e.g., 0.5) down-weights this term because most grid cells do not contain objects, preventing these negative samples from overwhelming the gradient.
  - **Classification Loss (Class Error):** This is the sum of squared errors for the class probabilities, calculated only for grid cells that *contain* an object (indicated by Iiobj). Lclass=Σi=0S2IiobjΣk∈classes(pi(k)−p^i(k))2

   - **Limitations of YOLOv1:** While fast, YOLOv1 faced challenges in accurately localizing objects, especially small objects that appeared in groups (due to each grid cell predicting only a limited number of boxes and a single class distribution).[5] Its localization accuracy generally trailed that of two-stage detectors.
2. **SSD (Single Shot MultiBox Detector):**
   - **Box Prediction:** SSD also operates as a single-shot detector, eliminating explicit region proposal generation and subsequent feature resampling stages.[10] It utilizes a standard backbone network (e.g., VGG16) truncated before the classification layers, followed by a series of auxiliary convolutional layers that progressively decrease in size.[10]
   - **Multi-scale Feature Maps & Default Boxes:** A key feature of SSD is its use of multiple feature maps from different layers in the network for detection.[10] Predictions are made from several of these feature maps, each at a different resolution. At each location on these selected feature maps, SSD associates a set of "default boxes" (conceptually similar to anchor boxes in Faster R-CNN) with varying aspect ratios and scales. The network then predicts:
     - Offsets from these default box coordinates to better match the ground truth object.
     - Per-class scores indicating the probability of each class being present in each default box.
   - **Matching Strategy during Training:** SSD's matching strategy during training is designed to simplify the learning process [10]:
     1. Each ground truth box is first matched to the default box that has the best Jaccard overlap (IoU).
     2. After this initial matching, SSD then matches default boxes to *any* ground truth box for which the Jaccard overlap is higher than a specific threshold (e.g., 0.5). This strategy allows the network to learn to predict high scores for multiple overlapping default boxes that correspond to the same ground truth object, rather than forcing it to select only the single default box with the absolute maximum overlap.
   - **Loss Function (MultiBox Objective):** The overall training objective for SSD is a weighted sum of a localization loss (Lloc) and a confidence loss (Lconf) [10]: $L(x,c,l,g)=N1(Lconf(x,c)+\alpha Lloc(x,l,g))$ Where N is the number of matched default boxes. If $N=0$, the loss is set to 0.
     - **Localization Loss (Lloc):** This is a Smooth L1 loss computed between the predicted box parameters (l) and the ground truth box parameters (g). Similar to Faster R-CNN, SSD regresses to offsets for the center (cx,cy) of the default bounding box (d) and for its width (w) and height (h). The

    specific parameterization for the ground truth targets (g^) relative to a default box di=(dicx,dicy,diw,dih) and a ground truth box gj=(gjcx,gjcy,gjw,gjh) is: g^jcx=(gjcx−dicx)/diw g^jcy=(gjcy−dicy)/dih g^jw=log(gjw/diw) g^jh=log(gjh/dih) The loss is then Σi∈PosNΣm∈{cx,cy,w,h}xijksmoothL1(lim−g^jm).
    - **Confidence Loss (Lconf):** This is a softmax loss over multiple class confidences (c). It is computed as: Lconf(x,c)=−Σi∈PosNxijplog(c^ip)−Σi∈Neglog(c^i0), where c^ip=Σpexp(cip)exp(cip) and xijp={1,0} is an indicator for matching default box i to GT box j of category p. Hard negative mining is typically employed for the negative samples. The weight term α is often set to 1 by cross-validation.

One-stage detectors like YOLO and SSD were fundamentally motivated by the demand for greater computational efficiency and faster inference speeds compared to the R-CNN family.[3] They achieve this by performing detection in a single, unified network pass. YOLO's core concept was to treat detection as a regression problem directly from the image [8], while SSD also encapsulates all computation within one network, predicting from multiple feature maps.[10]

A notable difference lies in how they predict box parameters. YOLOv1 performs direct regression of box coordinates (though relative to grid cell dimensions or overall image size) [7], whereas SSD, much like the RPN in Faster R-CNN, regresses *offsets* from a set of predefined default boxes (anchors).[10] The use of these priors (anchors/default boxes) often aids in stabilizing the training process and can improve localization accuracy for objects of diverse shapes and scales, as they provide better initial "guesses" that the network learns to refine. YOLOv1's attempt to address scale sensitivity with w and h in its loss function [20] was an early acknowledgment of this challenge. SSD's strategy of using default boxes at multiple scales across different feature maps provides a more structured way to handle scale variation. This highlights a recurring principle: well-chosen priors can simplify the learning task for the network.

The complexity of the matching strategy also influences learning. YOLOv1 employs a relatively simple matching rule: one specific predictor within one specific grid cell is responsible for each ground truth object.[20] This is a "hard" assignment. In contrast, SSD's matching is more permissive: multiple default boxes can be matched to a single ground truth object if their IoU exceeds a threshold (0.5), in addition to the best-IoU match.[10] The SSD paper suggests this "simplifies the learning problem".[10] This difference underscores how the design of the matching strategy directly affects the supervisory signals the network receives and, consequently, what and how easily it learns. A denser supervision signal, where multiple "good enough" predictions are considered positive for the same object, might facilitate learning compared to a sparse signal from only one "best" predictor.

C. Anchor-Free Detectors: Simplifying Prediction by Removing Anchors

A significant trend in object detection has been the development of anchor-free methods. Predefined anchor boxes, while effective in models like Faster R-CNN, SSD, and many YOLO versions, introduce several hyperparameters (e.g., scales, aspect ratios, number of anchors) that often require careful tuning for optimal performance on different datasets. Anchor-free detectors aim to eliminate this dependency, thereby simplifying the detection pipeline and potentially improving generalization.[11]

1. **FCOS (Fully Convolutional One-Stage Object Detection):**
   - **Box Prediction (Per-Pixel):** FCOS approaches object detection in a per-pixel prediction manner, drawing an analogy to semantic segmentation.[11] Instead of relying on anchor boxes, FCOS directly predicts bounding box information from each spatial location (pixel) on the feature maps generated by a Feature Pyramid Network (FPN).
   - For each foreground location (x,y) on a feature map, FCOS predicts a 4D vector (l,t,r,b). These values represent the distances from the location (x,y) to the left, top, right, and bottom sides of the bounding box that encloses the object associated with that location.[11]
   - A location (x,y) is considered a positive sample if it falls within any ground-truth bounding box. The class label for this positive location is then set to the class label of that ground-truth box. If a location falls within multiple ground-truth boxes (an ambiguous sample), FCOS assigns it to the ground-truth box with the minimal area for regression targeting.[11] Multi-level prediction with FPN helps to reduce such ambiguities, as objects of significantly different sizes are typically detected on different FPN levels.
   - **Center-ness Score:** A key challenge for per-pixel prediction is the potential generation of many low-quality bounding boxes from locations far from the center of an object. To mitigate this, FCOS introduces an additional single-layer branch, parallel to the classification branch, which predicts a "center-ness" score for each location.[11]
     - The center-ness target for a location is defined as: $\text{centerness}{*}=\max(l{*},r{*})\min(l{*},r{*})\times\max(t{*},b{*})\min(t{*},b{*})$ where $l{*},t{*},r{*},b{*}$ are the regression targets (distances to box sides) for that location. The square root is used to slow down the decay of center-ness.
     - This score ranges from 0 to 1, being 1 for locations at the center of the

    - object and decreasing as the location moves towards the object's boundary. It is trained using Binary Cross-Entropy (BCE) loss.
    - During inference, the predicted center-ness score is multiplied by the corresponding classification score. This multiplication effectively down-weights the scores of bounding boxes predicted from off-center locations, making them less likely to survive the final Non-Maximum Suppression (NMS) process, thereby significantly improving detection performance.[11]
  - **Matching and Learning:** Matching is implicit: locations within GT boxes are positive. The overall loss includes a classification loss (e.g., Focal Loss), a regression loss for the (l,t,r,b) predictions (e.g., GIoU loss), and the center-ness loss.

2. **CenterNet (Objects as Keypoints):**
   - **Box Prediction (Keypoint Triplets or Center Point):** CenterNet proposes a different anchor-free approach by modeling objects as keypoints. There are a couple of prominent variations:
     - One version [22] detects each object as a triplet of keypoints: a top-left corner, a bottom-right corner, and a center keypoint. This is considered a bottom-up approach.
     - Another influential version of CenterNet (Objects as Points) detects object centers as keypoints and then regresses to other object properties like size (width, height) and local offsets to refine the center location due to quantization from CNN downsampling.[22]
   - **Heatmap Prediction:** The network typically outputs several heatmaps:
     - For the keypoint triplet version: two corner heatmaps (one for top-left, one for bottom-right) and one center keypoint heatmap. These are often multi-class heatmaps.[22]
     - For the center-point version: a keypoint heatmap where peaks indicate object centers. The number of channels in this heatmap corresponds to the number of object classes.
   - **Regression for Other Properties:**
     - For the center-point version: In addition to the center heatmap, the network predicts an offset map (to correct for discretization errors in locating the center) and a size map (to regress the width and height of the bounding box at each detected center).
     - For the keypoint triplet version (CenterNet++): Offsets are predicted for center keypoints. Bounding boxes are initially formed by grouping detected corners (often using learned embeddings to associate pairs). These initial boxes are then filtered and confirmed by checking for the

presence of a center keypoint of the same class within a defined central region of the box.[22] The confidence of the final box can be an average of the corner and center keypoint scores.

- **Matching and Learning:**
  - For center-based CenterNet: During training, ground truth object centers are rendered onto the target heatmap, typically using a Gaussian kernel to create a "soft" target around the precise center location. The regression targets (object size, center offset) are learned only at these positive center locations. The loss function usually includes a heatmap loss (e.g., a variant of Focal Loss to handle the sparsity of centers) and L1 loss for the size and offset regressions.
  - For CenterNet++ (keypoint triplet): Training involves heatmap losses for the corner and center keypoints. Grouping of corners might involve learning embeddings such that corners from the same object have similar embeddings. The filtering step (checking for a center keypoint in the central region of a corner-defined box) is a crucial part of its inference logic.

The motivation behind anchor-free detectors like FCOS and CenterNet is primarily the simplification of detector design by eliminating anchor-related hyperparameters and the associated complex computations (e.g., IoU calculations between anchors and GTs, matching heuristics).[11] Anchor boxes introduce numerous parameters (number, scales, aspect ratios) that are often dataset-specific and require meticulous tuning. FCOS achieves this by directly predicting distances from a point to the four sides of a box [11], while CenterNet focuses on detecting semantic keypoints (like object centers or corners).[22] This represents a fundamental shift in how potential object locations are defined, moving away from a dense grid of predefined anchor shapes.

These anchor-free approaches also redefine what constitutes a "positive sample" for training. In FCOS, any point (pixel on the feature map) that falls inside a ground-truth bounding box is initially considered a positive sample for regression.[11] This can lead to a large number of positive predictions for a single object. The "center-ness" score is then introduced as a mechanism to prioritize predictions originating from locations near the object's true center, effectively learning to down-weight predictions from less "central" points. In contrast, keypoint-based CenterNet variants define positive samples based on the presence of object keypoints (e.g., the object center or corners) on a predicted heatmap. This distinction in positive sample definition is a key differentiator from anchor-based methods, which assign anchors as positive or negative based on IoU with GT boxes.

While two-stage detectors rely on explicit region proposals and anchor-based one-stage methods use explicit anchors, anchor-free methods often employ more implicit ways of defining potential object locations. For example, FCOS effectively treats every foreground pixel on the feature map as a potential object center from which to regress a box.[11] CenterNet identifies potential objects by looking for peaks in the predicted keypoint heatmaps.[22] This is distinct from the approach of refining a fixed set of anchor boxes; here, the "proposal" is, in effect, any location that the network predicts with high confidence as being an object part (like a center or corner) or an interior point suitable for regression.

D. Transformer-Based Detectors: End-to-End Set Prediction with Attention

The introduction of Transformers, initially successful in natural language processing, has led to innovative architectures in computer vision, including object detection. DETR (DEtection TRansformer) marked a significant paradigm shift.

1. **DETR (DEtection TRansformer):**
    - **Core Idea:** DETR reconceptualizes object detection as a direct set prediction problem. This approach aims to streamline the detection pipeline by eliminating many hand-designed components common in previous detectors, such as Non-Maximum Suppression (NMS) or explicit anchor generation mechanisms.[9]
    - **Architecture:** The DETR architecture comprises several key components [9]:
        - **CNN Backbone:** A standard CNN (e.g., ResNet) is used to extract a rich feature representation from the input image. This backbone produces a 2D feature map.
        - **Transformer Encoder:** The 2D feature map from the CNN is flattened into a sequence and augmented with fixed positional encodings. This sequence is then processed by a Transformer encoder, which uses self-attention mechanisms to model global context and relationships between different parts of the image.
        - **Transformer Decoder:** The decoder takes two sets of inputs: the processed image features from the encoder and a small, fixed number (N) of learned positional embeddings called "object queries." These object queries are learnable parameters that act as placeholders or "slots" for potential objects. The decoder uses self-attention among the object queries and cross-attention between object queries and encoder outputs to refine these queries into N output embeddings.
        - **Prediction Heads (FFNs):** Each of the N output embeddings from the decoder is independently passed through a Feed-Forward Network (FFN).

    Each FFN predicts the class label (including a special "no object" class, ∅) and the bounding box coordinates (normalized center coordinates x,y, height h, and width w) for one potential object.

  - **Box Prediction:** DETR directly outputs a fixed-size set of N predictions (class label and box coordinates). The number N is chosen to be larger than the maximum number of objects typically expected in an image.
  - **Matching (Bipartite Matching with Hungarian Algorithm):** A cornerstone of DETR is its matching strategy during training.[9] Since the model predicts a set of N objects and the ground truth is also a set of objects (padded with ∅ to size N), DETR finds an optimal one-to-one matching between the predicted and ground truth boxes. This is achieved by searching for a permutation σ of the N predictions that minimizes a total matching cost: σ^=argminσΣiLmatch(yi,y^σ(i)) The pairwise matching cost Lmatch(yi,y^σ(i)) considers both the class prediction probability for the true class ci and the similarity of the predicted box b^σ(i) to the ground truth box bi (typically a combination of L1 loss and GIoU loss). This optimal assignment is computed efficiently using the Hungarian algorithm. This unique one-to-one assignment is crucial as it forces each object query to specialize and make a unique prediction.
  - **Learning (Set-Based Hungarian Loss):** After the optimal matching σ^ is found, the Hungarian loss is computed for all matched pairs [9]: LHungarian(y,y^)=Σi=1N[−logp^σ^(i)(ci)+1{ci偃=∅}Lbox(bi,b^σ^(i))] The term 1{ci偃=∅} ensures the box loss Lbox (a linear combination of L1 loss and GIoU loss) is applied only to matched, non-∅ objects. The log-probability for the ∅ class is typically down-weighted to handle class imbalance.
  - **NMS-Free:** Due to the set prediction formulation and the one-to-one bipartite matching, DETR inherently avoids producing multiple redundant detections for the same object, thus eliminating the need for NMS post-processing.[9]

2. **Deformable DETR: Addressing DETR's Limitations:**
  - **Problem:** The original DETR model, while innovative, suffered from slow convergence during training and high computational complexity when dealing with high-resolution feature maps. This was primarily due to the global nature of the attention mechanism in the Transformer, where every element attends to every other element, leading to quadratic complexity with respect to the number of feature map pixels.[27] This limited its ability to effectively process multi-scale features, impacting performance on small objects.
  - **Innovation (Deformable Attention Module):** Deformable DETR introduced a more efficient attention mechanism called deformable attention.[27] Instead of

attending to all spatial locations in the feature map, the deformable attention module learns to attend to a small, fixed number of key sampling points around a reference point. The locations of these sampling points are themselves learned as offsets from the reference point, allowing the model to adaptively focus on relevant image regions.

- This sparse attention mechanism significantly reduces the computational complexity from quadratic to linear with respect to the spatial size of the feature maps. This efficiency gain allows Deformable DETR to effectively process multi-scale feature maps (by applying deformable attention across different scales), leading to improved performance, especially on small objects, and much faster convergence (e.g., requiring 10 times fewer training epochs than the original DETR).[27]

DETR signifies a fundamental departure from the traditional object detection pipeline. Instead of predicting an overcomplete set of bounding boxes that are subsequently filtered by NMS [4], DETR aims to directly predict a fixed-size set of unique object detections.[9] This is achieved through the novel combination of learnable object queries and a bipartite matching loss that enforces unique assignments during training. This architectural design inherently avoids the redundancy that NMS is designed to resolve.

The learned object queries in DETR are a distinctive feature. Unlike fixed priors such as anchor boxes, these object queries are learnable embeddings that act as "slots" or "placeholders".[9] The Transformer decoder uses these queries to probe the image features and reason about the presence and properties of objects. Their learnable nature allows them to adapt and potentially specialize for detecting certain types of objects, or objects in particular locations or scales, based on the training data.

The Hungarian algorithm, used for bipartite matching, is central to DETR's NMS-free property and its ability to learn one-to-one correspondences between predictions and ground truth objects.[9] This algorithm efficiently finds the optimal (lowest cost) one-to-one assignment. The design of the matching cost function, Lmatch, which typically incorporates both classification accuracy and bounding box similarity (e.g., L1 and GIoU loss), is critical for guiding this matching process towards semantically meaningful assignments.

Deformable DETR demonstrates that the powerful attention mechanisms of Transformers can be made more computationally tractable for vision tasks by incorporating spatial priors. Standard Transformer attention is global, meaning every pixel attends to every other pixel, which is prohibitively expensive for large feature maps.[9] Deformable convolution had already shown the effectiveness of sparse, learned sampling patterns. Deformable attention [27] applies a similar concept to the attention mechanism itself, restricting each query's attention to a small set of learnable sampled points in the key/value features. This directly reduces computational complexity, enabling the use of multi-scale features crucial for detecting objects of varying sizes, and significantly accelerates training convergence.

E. Advanced Label Assignment Strategies: Refining Positive/Negative Definition

Label assignment, the process of determining which predicted boxes (or anchors/points) should be considered positive or negative samples for each ground-truth object, is a critical step in training object detectors. Beyond simple IoU thresholds (as in Fast R-CNN [12]) or the unique one-to-one matching in DETR [9], more sophisticated and dynamic strategies have been developed to provide better supervisory signals.

1. **Optimal Transport Assignment (OTA):**
   - **Core Idea:** OTA reformulates the label assignment problem from a global perspective by casting it as an Optimal Transport (OT) problem.[13] In this formulation, ground-truth objects are considered "suppliers" of positive labels, and anchors (or anchor-free prediction locations) are "demanders" requiring labels. The goal is to find the globally optimal "transport plan" (i.e., assignment) that minimizes the total "transportation cost" of assigning labels from suppliers to demanders. This contrasts with strategies that assign labels independently for each ground-truth object or for each anchor.
   - **Cost Function:** The unit transportation cost between an anchor aj and a ground-truth object gti is typically defined as a weighted sum of their pair-wise classification loss (e.g., Focal Loss between the predicted class for aj and the class of gti) and regression loss (e.g., IoU loss or GIoU loss between the box predicted from aj and gti).[13]
     cij=λcls·Lcls(Pjcls(θ),Gicls)+λreg·Lreg(Pjbox(θ),Gibox)
   - **Solving:** The optimal transport plan, which represents the optimal assignment of anchors to ground truths (and to a background class), is found by solving the OT problem, often using an iterative algorithm like the Sinkhorn-Knopp algorithm.[13]
   - **Benefit:** OTA can more effectively resolve ambiguous assignments, where an anchor might be a plausible candidate for multiple ground-truth objects. By considering the global cost, it makes assignments that are optimal for the overall set of predictions and ground truths, which has been shown to improve performance, particularly in scenes with crowded objects.[13]

2. **SimOTA (Simplified Optimal Transport Assignment - e.g., in YOLOX):**
   - **Core Idea:** SimOTA is a more practical and computationally efficient approximation of OTA, prominently used in the YOLOX detector.[32] It retains the core idea of dynamic, cost-aware label assignment but simplifies the optimization process.
   - **Mechanism:**
     1. **Cost Matrix Calculation:** Similar to OTA, a cost matrix is computed. The cost cij of assigning prediction j (e.g., an anchor-free prediction from a specific FPN level and grid cell) to ground truth i is a weighted sum of the classification cost (e.g., BCE loss between predicted class scores and GT class) and the regression cost (e.g., 1−IoUij).[36] costij=Lcls_ij+λ·Lreg_ij
     2. **Candidate Selection (Center Prior & Top-k):** For each ground truth gti:
        - A "center prior" is often used: only predictions whose centers fall within a small radius [36] around the gti's center are considered as initial candidates. This significantly reduces the number of prediction-GT pairs to evaluate.
        - Among these candidates, the top k0 [36] predictions with the lowest IoU cost (i.e., highest IoU) with gti are selected.
     3. **Dynamic k Estimation:** The number of positive samples to assign to gti, denoted as kdyn, is dynamically determined based on the sum of IoUs of these top k0 candidates. For example, kdyn might be the integer part of the sum of these IoUs. This means more anchors will be assigned to a GT if many anchors overlap well with it.
     4. **Final Assignment:** The kdyn predictions with the lowest overall cost cij (from step 1) among the candidates (from step 2) are assigned as positive samples for gti.
     5. **Ambiguity Resolution:** If a single prediction is assigned as positive to multiple ground truths, it is ultimately assigned only to the ground truth for which it has the minimum cost.[36]
   - **Simplification over OTA:** Full OTA often involves iterative solutions to the OT problem (like Sinkhorn-Knopp). SimOTA avoids these complex iterations by using a greedy, multi-step approach involving center priors, top-k candidate selection based on IoU, dynamic determination of the number of positives per GT, and final assignment based on the combined cost. This makes SimOTA computationally cheaper and more suitable for fast training of high-performance detectors.
   - **YOLOX Context:** YOLOX is an anchor-free detector that features a decoupled head for classification and regression.[32] SimOTA is a crucial component that contributes significantly to YOLOX's strong performance by providing

high-quality label assignments.

Advanced label assignment strategies like OTA and SimOTA represent a shift from using fixed heuristics (like static IoU thresholds [10]) to more adaptive, learning-dependent assignment processes. By incorporating the model's current prediction quality (via classification and regression losses) into the assignment cost [13], these methods make the definition of positive and negative samples dynamic. If a prediction is good for classification but poor for regression with respect to a particular ground truth, its assignment cost will reflect this, leading to more nuanced assignments than simple IoU overlap criteria.

OTA's emphasis on a *global* optimum for assignment [13], considering all predictions and ground truths simultaneously, is a departure from local strategies that make decisions independently for each ground truth or each anchor. This global perspective can be particularly beneficial in complex scenes with many overlapping objects and predictions, where local greedy choices might lead to suboptimal overall assignments. SimOTA, with its dynamic-k estimation and cost-based selection [36], offers a pragmatic approximation of this global consideration, balancing effectiveness with computational efficiency. This makes it suitable for high-throughput detectors like YOLOX, where training speed is also a concern.

F. Non-Maximum Suppression (NMS) and NMS-Free Detection

Non-Maximum Suppression (NMS) has long been an indispensable post-processing step in the vast majority of object detection pipelines that inherently produce multiple, often overlapping, bounding box predictions for the same object instance.[3]

- **Role of NMS:** The primary function of NMS is to refine the raw output from detection models. These models frequently identify numerous potential bounding boxes around a single object. NMS intelligently filters these redundant detections to ensure that each distinct object in the image or video frame is ultimately represented by a single, optimal bounding box.[4] This significantly enhances the clarity and precision of the final detection results, making them more reliable for downstream tasks.[4]
- **How NMS Works:** The standard NMS algorithm operates iteratively based on confidence scores and spatial overlap (IoU) [4]:
    1. All proposed bounding boxes are sorted according to their confidence scores, typically in descending order.
    2. The bounding box with the highest confidence score is selected as a definitive detection and is added to the list of final detections.
    3. The IoU of this selected box is calculated with all other remaining (not yet

selected or suppressed) bounding boxes.

4. All other bounding boxes that have an IoU with the selected box greater than a predefined iou_threshold (e.g., 0.5) are suppressed (removed from further consideration).
5. The process is repeated from step 2, selecting the next highest-scoring box from the remaining unsuppressed boxes, until no boxes remain.

- **NMS-Free Detection:**
  - Recognizing that NMS is a heuristic, often computationally intensive, and not always seamlessly integrated into end-to-end training, some modern detector architectures have been designed to be NMS-free. Prominent examples include DETR [9] and certain versions of YOLO, such as YOLOv10.[4]
  - DETR achieves its NMS-free nature through its unique set prediction formulation coupled with bipartite matching during training. This inherently encourages the model to produce a set of unique, non-overlapping detections.[9]
  - YOLOv10, for instance, employs techniques like "consistent dual assignments" during its training phase to actively discourage the prediction of redundant boxes, thereby aiming for an NMS-free inference pipeline.[4]
  - **Motivation:** The drive towards NMS-free detection stems from several factors: NMS can be a bottleneck in terms of processing speed, especially with a large number of initial predictions; its iou_threshold is a sensitive hyperparameter that may require careful tuning for different datasets or object types; and it is typically a non-differentiable component, making truly end-to-end optimization challenging for most detectors. NMS-free methods strive for simpler, more elegant, and fully learnable detection pipelines.

While highly effective and widely used, NMS is fundamentally a heuristic algorithm applied *after* the main network has made its predictions.[4] Its typically non-differentiable nature means it doesn't participate in the end-to-end gradient-based optimization of most detection models. Furthermore, its performance is sensitive to the choice of the IoU threshold [37], which often needs empirical tuning. The desire for fully end-to-end differentiable and learnable systems has therefore motivated the development of NMS-free approaches.

The emergence of NMS-free detectors like DETR [9] and specialized YOLO versions [4] signals a trend towards designing architectures that inherently prevent or minimize redundant predictions, rather than relying on a separate filtering step. This often involves more sophisticated matching or assignment strategies during the training phase. For example, DETR's bipartite matching mechanism is specifically designed to enforce one-to-one assignments between predictions and ground truth objects, naturally avoiding the scenario of multiple high-scoring boxes for a single object that NMS is built to handle.[9] Similarly, strategies like YOLOv10's "consistent dual assignments" [4] are training-time innovations aimed at teaching the model to avoid redundancy from the outset. This reflects a philosophical shift: build redundancy avoidance directly into the learning process itself, rather than treating it as a post-hoc clean-up task.

Table 1: Comparative Overview of Major Object Detection Paradigms

Paradigm	Key Model Example(s)	Box Prediction Strategy	Primary Matching Strategy	Typical Localization Loss Components	NMS Required?
Two-Stage (Region-Based)	R-CNN	Selective Search + CNN features per RoI	IoU threshold for SVMs/regressors	L2 regression on box parameters	Yes
	Fast R-CNN	Selective Search + Shared CNN features + RoI Pooling	IoU thresholds (e.g., ≥0.5 for FG, While it has a smooth gradient that is zero at the minimum, the large gradients produced by outliers can lead to exploding gradients during training if learning rates are not carefully		

			managed.[16] L2 loss was used in the regression stage of the original R-CNN.[16]

*   **Limitations for Bounding Box Regression:** When applied to bounding box parameters (e.g., <span class="math-inline">x, y, w, h</span>) independently, these losses may not optimally reflect the visual quality of the predicted box. For instance, the same L2 norm (distance) between predicted and ground truth box parameters can correspond to vastly different Intersection over Union (IoU) values, making it a potentially misleading proxy for localization quality.[14] Furthermore, treating the four box parameters as independent variables for regression, as is common with <span class="math-inline">L_n</span>-norm losses, can be suboptimal because these parameters are inherently coupled in defining the box's geometry.[15]

2. **Smooth L1 Loss (Huber Loss):**
   - **Rationale:** Smooth L1 loss was introduced as a robust compromise between L1 and L2 losses. It aims to combine the L2 loss's smoothness around zero (ensuring stable gradients for small errors) with the L1 loss's reduced sensitivity to outliers (preventing large errors from dominating the loss and causing exploding gradients).[12]
   - **Formulation:** The Smooth L1 loss is typically defined as [12]: $$ \text{smooth}_{L1}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < \beta \ |x| - 0.5\beta & \text{otherwise} \end{cases} $$ Often, the transition point β is set to 1. In this case, the loss is quadratic for errors smaller than 1 and linear for errors larger than or equal to 1.
   - **Widespread Adoption:** Due to its favorable properties, Smooth L1 loss became the de facto standard for bounding box regression in many influential object detectors, including Fast R-CNN [12], Faster R-CNN, and SSD.[10]
   - **Self-Adjusting Smooth L1 Loss:** To address the heuristic nature of choosing the hyperparameter β, a Self-Adjusting Smooth L1 Loss has been proposed. This variant attempts to dynamically set the transition point by recording the running mean and variance of the absolute errors during training, thereby

adapting the loss function's behavior to the data and training dynamics.[39]

The evolution from L2 loss to Smooth L1 loss was largely driven by practical challenges encountered during the training of early object detectors. The sensitivity of L2 loss to outliers and its propensity to cause exploding gradients when regressing unbounded box parameters necessitated more robust alternatives.[12] The Fast R-CNN paper explicitly highlights Smooth L1's advantages in this regard.[12] Smooth L1 effectively balances the trade-off between smoothness (desirable for stable convergence when errors are small) and robustness to large errors. Its piecewise definition, behaving quadratically like L2 for small errors and linearly like L1 for large errors [12], directly addresses the respective weaknesses of using pure L1 or L2 loss for bounding box regression.

B. IoU-based Loss Functions: Aligning Loss with Evaluation Metrics

While traditional regression losses operate on the box parameters (x,y,w,h) individually, the primary evaluation metric for object detection accuracy is Intersection over Union (IoU), which holistically measures the overlap between the predicted box and the ground truth box. This disconnect motivated the development of loss functions that directly incorporate IoU or related geometric properties, aiming to align the training objective more closely with the evaluation metric.[14] Optimizing such losses can lead to a more direct improvement in the final perceived localization quality.

1. **IoU Loss:**
    - **Formulation:** The most straightforward IoU-based loss is LIoU=1−IoU.
    - **Problem Addressed:** Directly optimizes the IoU metric, which is desirable.
    - **Limitations:** The IoU loss suffers from two critical limitations [14]:
        - **Vanishing Gradient for Non-Overlapping Boxes:** If the predicted bounding box and the ground truth box do not overlap, their IoU is 0. Consequently, LIoU=1, and the gradient of the IoU with respect to the box parameters becomes zero. This means the loss function provides no learning signal to guide the predicted box towards the ground truth box when they are disjoint.
        - **Inability to Distinguish Alignment Quality for Same IoU:** The IoU metric itself (and thus IoU loss) cannot distinguish between different configurations of predicted and ground truth boxes if they yield the same IoU value. For example, it doesn't reflect how far apart two non-overlapping boxes are.
2. **Generalized IoU (GIoU) Loss:**
    - **Motivation:** GIoU loss was proposed to address the vanishing gradient problem of IoU loss for non-overlapping boxes.[14]

- **Formulation:** LGIoU=1−GIoU, where the GIoU is defined as: GIoU=IoU−∣C∣∣C∖(B∪Bgt)∣ Here, B is the predicted box, Bgt is the ground truth box, and C is the smallest convex bounding box that encloses both B and Bgt.[17] The term ∣C∣∣C∖(B∪Bgt)∣ acts as a penalty that increases as the predicted box moves further away from the ground truth, even when they do not overlap. This penalty term is 0 when B and Bgt have maximum overlap relative to C.
- **Problem Addressed:** GIoU provides a non-zero gradient even for non-overlapping boxes, encouraging the predicted box to move towards the ground truth box and increase their eventual overlap.
- **Limitations:** While GIoU addresses the non-overlapping issue, it can still suffer from slow convergence, particularly when one box is contained within the other or when the boxes are aligned but have different aspect ratios.[17] In cases where the predicted box is inside the ground truth box (or vice versa) and their union is equal to the larger box, the penalty term becomes zero, and GIoU effectively degrades to IoU, potentially leading to slow refinement if the centers are misaligned but containment exists.[15]

3. **Distance IoU (DIoU) Loss:**
    - **Motivation:** DIoU loss was introduced to achieve faster convergence and more accurate regression than GIoU by directly penalizing the normalized distance between the center points of the predicted and ground truth boxes, in addition to their overlap.[17]
    - **Formulation:** LDIoU=1−DIoU, where DIoU is defined as: DIoU=IoU−c2ρ2(b,bgt) In this formula, b and bgt are the center points of the predicted box B and the ground truth box Bgt, respectively. ρ(⋅,⋅) denotes the Euclidean distance. c is the diagonal length of the smallest enclosing convex box C that covers both B and Bgt.[17] The term c2ρ2(b,bgt) represents the normalized squared distance between the centers.
    - **Problem Addressed:** DIoU provides a penalty related to the center distance, which leads to faster convergence compared to GIoU, as it directly encourages the alignment of box centers. It also maintains a non-zero gradient for non-overlapping boxes.
    - **Limitations:** DIoU loss does not explicitly consider the consistency of aspect ratios between the predicted and ground truth boxes. Two boxes can have perfectly aligned centers and the same IoU value but still have very different shapes (aspect ratios), which DIoU would not penalize further.[17] Some analyses also point to a "gradients inconsistency problem" where minimizing the DIoU loss might not always correspond to the most direct path for the predicted box center to reach the ground truth box center.[15]

4. **Complete IoU (CIoU) Loss:**
   - **Motivation:** CIoU loss builds upon DIoU by incorporating an additional penalty term for inconsistencies in the aspect ratio between the predicted and ground truth boxes, aiming for a more comprehensive geometric alignment.[17]
   - **Formulation:** LCIoU=1−CIoU, where CIoU is defined as: CIoU=DIoU−αv The term αv penalizes differences in aspect ratio. Here, v=π24(arctanhgtwgt−arctanhw)2 measures the consistency of aspect ratios, where (w,h) and (wgt,hgt) are the width and height of the predicted and ground truth boxes, respectively. α=(1−IoU)+vv is a positive trade-off parameter that gives higher priority to the aspect ratio consistency when overlap is poor (low IoU).[40]
   - **Problem Addressed:** CIoU considers three key geometric factors: overlap area (via IoU), center point distance (via the DIoU term), and aspect ratio consistency (via αv). This provides a more holistic measure of bounding box similarity.
   - **Limitations:** While CIoU incorporates aspect ratio, the gradient of the aspect ratio term v might sometimes conflict with the optimization of IoU, particularly for small boxes where small changes in w or h can lead to large changes in aspect ratio. The term focuses on the *ratio* w/h rather than directly penalizing differences in the absolute values of w and h.
5. **Efficient IoU (EIoU) Loss:**
   - **Motivation:** EIoU loss aims to improve upon CIoU by directly minimizing the differences in the width and height of the predicted and ground truth boxes, rather than just their aspect ratio. It also decouples the aspect ratio loss into separate width and height difference penalties, which can lead to faster convergence and better localization accuracy.[17]
   - **Formulation:** EIoU loss is typically composed of three parts: the IoU loss, the distance loss (same as in DIoU), and an aspect loss that directly penalizes width and height differences: LEIoU=LIoU+Ldist+Lasp LEIoU=(1−IoU)+c2ρ2(b,bgt)+Cw2ρ2(w,wgt)+Ch2ρ2(h,hgt) Here, Cw and Ch are the width and height of the smallest enclosing box C that covers both the predicted and ground truth boxes.[40]
   - **Problem Addressed:** EIoU seeks to accelerate convergence and improve localization precision by directly penalizing discrepancies in individual dimensions (width and height), leading to better overall shape alignment. Some formulations of EIoU also incorporate a "FocalL1 loss" component [17], which applies the focusing principle (similar to Focal Loss for classification) to the regression loss. This aims to make high-quality anchor samples (those

with small regression errors) contribute more significantly to the training gradient, thereby prioritizing the refinement of already good predictions.

The entire family of IoU-based losses emerged from the fundamental realization that optimizing a loss function more directly aligned with the final evaluation metric (IoU) should yield better practical performance than relying on traditional Ln-norm losses on coordinate parameters.[14] A small change in a coordinate like width can have a vastly different impact on the IoU depending on the box's overall size and its current overlap with the ground truth. IoU-based losses inherently account for this holistic overlap.

The progression from IoU loss to GIoU, DIoU, CIoU, and finally EIoU demonstrates a clear pattern of incremental problem-solving. Each new loss function was specifically designed to address observed limitations in its predecessors, leading to increasingly comprehensive geometric penalty terms. IoU loss fails for non-overlapping boxes; GIoU introduces a penalty using the enclosing box.[17] GIoU exhibits slow convergence; DIoU adds a center distance penalty to speed it up.[17] DIoU doesn't account for aspect ratio differences; CIoU incorporates an aspect ratio consistency term.[40] CIoU's aspect ratio term is indirect; EIoU introduces direct penalties on width and height differences.[40] This iterative refinement showcases a methodical, scientific approach to improving loss functions for better bounding box regression.

This evolution also reflects a trade-off: as the loss functions become more sophisticated by incorporating more geometric considerations (e.g., EIoU considers overlap, center distance, and individual side length differences), their mathematical formulations become more complex. However, this increased complexity is generally justified by reported improvements in convergence speed and final mean Average Precision (mAP), as these more elaborate losses provide richer and more targeted gradient information to guide the bounding box regression process effectively.

C. Addressing Training Challenges: Focal Loss for Classification

While Focal Loss is primarily a classification loss, its role is pivotal in the success of many modern object detectors, especially one-stage dense detectors like RetinaNet [43] and FCOS. In these detectors, an extreme imbalance between the number of foreground (object) and background samples can severely hinder training if not properly addressed.[43] Since accurate classification is a prerequisite for meaningful localization (i.e., localizing correctly identified objects), improvements in classification often translate to better overall detection performance.

- **Problem:** Dense object detectors evaluate an enormous number of potential object locations (e.g., anchors at every spatial position across multiple scales, or every pixel in anchor-free approaches). The vast majority of these locations correspond to easily classifiable background (easy negatives). If a standard cross-entropy loss is used, the accumulated loss from these numerous easy negatives can dominate the loss from the relatively few, but crucial, positive (foreground) examples, leading to a model that is poorly optimized for detecting actual objects.[43]

- **Focal Loss Formulation:** Focal Loss addresses this class imbalance by dynamically reshaping the standard cross-entropy loss to down-weight the contribution of well-classified examples. The formulation is [44]: $FL(pt)=-\alpha t(1-pt)\gamma \log(pt)$ Where:
  - pt is the model's estimated probability for the ground truth class (for a binary case, $pt=p$ if $y=1$ and $pt=1-p$ if $y=-1$).
  - $\gamma\geq 0$ is the tunable *focusing parameter*. When an example is well-classified ($pt\rightarrow 1$), the modulating factor $(1-pt)\gamma$ becomes very small, significantly reducing that example's contribution to the loss. Conversely, for hard, misclassified examples ($pt\rightarrow 0$), the modulating factor $(1-pt)\gamma$ is close to 1, and the loss is largely unaffected. This focuses the training on hard negatives and positives. A common value for γ is 2.
  - αt is a weighting factor (analogous to α in balanced cross-entropy) that can be used to balance the importance of positive and negative examples. For instance, it might be α for the positive class and $1-\alpha$ for the negative class.
- **Impact:** By reducing the loss assigned to easy examples, Focal Loss prevents the vast number of easy negatives from overwhelming the detector during training. This allows the model to learn more effectively from the sparse set of hard examples, leading to significant improvements in the accuracy of one-stage detectors.[43]
- **FocalL1 Loss (in EIoU context):** The principle of focusing on certain samples has also been extended to regression. The EIoU paper [17] introduces a "FocalL1 loss." This loss aims to increase the gradient contribution of high-quality anchor samples (those with small regression errors) to the training process, effectively making the model pay more attention to refining already good predictions rather than being overly influenced by outliers with large errors.

The success of Focal Loss underscores the critical interplay between classification and localization in object detection. Although Focal Loss directly targets the classification imbalance, its effectiveness is vital for the overall detector's performance. A poorly trained classifier, overwhelmed by background samples, will fail to identify relevant object regions accurately, thereby providing poor inputs to the localization regressor. By enabling robust classification in the face of extreme imbalance [43], Focal Loss ensures that the localization component can operate on more reliably identified candidate objects.

The core idea underpinning Focal Loss—dynamically down-weighting the influence of "easy" examples to concentrate learning on "hard" ones—is a powerful and generalizable concept. Its adaptation in the EIoU paper as FocalL1 loss for regression [17] demonstrates this. If a regression sample is already of high quality (i.e., its predicted box is very close to the ground truth), its contribution to the loss can be modulated to allow the model to focus on more challenging regression targets. This suggests a broader applicability of the "focal" principle in designing loss functions for various deep learning tasks where imbalances or varying sample difficulties exist.

D. Multi-Task Loss Formulations in Detectors

Object detection is inherently a multi-task learning problem, requiring simultaneous object classification (determining if an object is present and its category) and localization (determining its spatial extent via bounding box regression). Consequently, the overall loss function in an object detector is typically a weighted combination of a classification loss and a localization/regression loss.[10]

- **Example (Fast R-CNN):** The loss function is $L(p,u,tu,v)=Lcls(p,u)+\lambda[u\geq1]Lloc(tu,v)$.[12]
    - $Lcls$ is the classification loss (log loss).
    - $Lloc$ is the bounding box regression loss (Smooth L1 loss).
    - λ is a hyperparameter that balances the two losses (e.g., $\lambda=1$).
    - The Iverson bracket $[u\geq1]$ ensures that the localization loss $Lloc$ is only applied to foreground objects (i.e., when the true class u is not background).
- **Example (SSD):** The loss is $L(x,c,l,g)=N1(Lconf(x,c)+\alpha Lloc(x,l,g))$.[10]
    - $Lconf$ is the confidence loss, which includes both objectness (object vs. background) and classification scores (softmax loss).
    - $Lloc$ is the bounding box regression loss (Smooth L1 loss).
    - N is the number of matched default boxes, and α is a hyperparameter to balance the localization loss against the confidence loss.
- **Example (YOLOv1):** The loss is a sum of squared errors for coordinates, confidence (objectness), and class probabilities, with specific weighting factors $\lambda coord$ and $\lambda noobj$ to give different importance to localization error, object confidence error for boxes containing objects, and object confidence error for boxes not containing objects.[20]
- **Challenges:** A significant challenge in multi-task learning is balancing the different loss components. The individual losses might have different scales or

convergence rates. The relative weighting of these losses (e.g., λ in Fast R-CNN, α in SSD, λcoord and λnoobj in YOLOv1) is crucial and often requires careful empirical tuning. An improper balance can lead the network to prioritize one task (e.g., classification) at the expense of the other (e.g., localization), resulting in suboptimal overall detection performance. For instance, [20] notes for YOLOv1 that localization errors are weighted more heavily (λcoord=5) because localization is generally considered a more challenging task than classification for the network to learn.

- **Conditional Learning:** A common and critical practice is the conditional application of the localization loss. Typically, the regression loss is computed only for "positive" samples—those predicted boxes or anchors that have been successfully matched to a ground truth object. The network is not trained to regress bounding boxes for regions identified as background. This selective application is fundamental to how localization is learned, as there is no ground truth box to regress towards for a background region. The [u≥1] term in Fast R-CNN's loss function [12] is a clear example of this conditional learning principle.

The careful design and balancing of these multi-task loss components are essential for training effective object detectors. The weighting parameters are critical hyperparameters that often need to be tuned based on the specific architecture, dataset, and other training configurations.

Table 2: Evolution and Characteristics of Bounding Box Regression Loss Functions

Loss Function	Core Idea / Mathematical Formulation Snippet	Key Problem Addressed (vs. previous / Ln losses)	Advantages	Limitations / Disadvantages
L1 Loss	$L_1 = \$	y - \hat{y}\		
L2 Loss	L2=(y−y^)2	Original regression loss.	Smooth gradient.	Very sensitive to outliers; can lead to exploding gradients.
Smooth L1 Loss	$0.5x^2 \text{ if } \$	x\	<1, \text{ else } \	x\
IoU Loss	LIoU=1−IoU	Ln-norm losses	Directly	Zero gradient

		don't directly correlate with IoU evaluation metric.	optimizes for overlap metric. Scale-invariant.	for non-overlapping boxes; cannot distinguish different non-overlap qualities.
GIoU Loss	$L_{GIoU} = 1 - (IoU - \frac{\$	C \setminus (B \cup B_{gt})\	}{\	C\
DIoU Loss	LDIoU=1−(IoU−c 2ρ2(b,bgt))	Slow convergence of GIoU; GIoU doesn't directly minimize center point distance.	Faster convergence by directly penalizing center point distance. Robust to non-overlapping cases.	Does not consider aspect ratio consistency. Potential gradient inconsistency.
CIoU Loss	LCIoU=1−(DIoU−αv), where v penalizes aspect ratio difference.	DIoU does not consider aspect ratio.	Considers overlap, center distance, and aspect ratio consistency.	Aspect ratio term can sometimes conflict with IoU optimization; w,h definition indirect.
EIoU Loss	LEIoU=(1−IoU)+c 2ρ2(b,bgt)+Cw2 ρ2(w,wgt)+Ch2ρ 2(h,hgt)	CIoU's aspect ratio term is indirect; EIoU directly penalizes width/height differences.	Accelerates convergence and improves accuracy by directly penalizing width/height differences. Can incorporate FocalL1 for focusing on high-quality samples.	More complex formulation.

This table summarizes the progression of loss functions specifically designed for bounding box regression. It highlights how each new loss function attempted to address the shortcomings of its predecessors or the limitations of general-purpose regression losses when applied to the specific geometric task of localizing objects with boxes. The trend is towards incorporating more comprehensive geometric properties (overlap, center distance, aspect ratio, side lengths) to provide richer and more effective learning signals.

IV. Localization in Text-based Video Moment Retrieval (MR)

Text-based Video Moment Retrieval (MR) aims to localize a specific temporal segment (a "moment" or "span") within an untrimmed video that corresponds to a given natural language text query.[26] This task requires understanding both the visual content of the video and the semantics of the text query, and then aligning them in the temporal domain. Many successful approaches in MR have adapted or drawn inspiration from paradigms originally developed for object detection in images.

A. Adapting Object Detection Paradigms to Temporal Domain

The core challenge in MR is to predict a start time and an end time for the relevant video segment, which can be conceptualized as a 1D "bounding box" or "span" along the time axis.[26] The DETR (DEtection TRansformer) architecture, with its success in object detection, has been notably influential in MR.

- **Moment-DETR:**
  - **Adapting DETR for MR:** Moment-DETR was one of the pioneering works to adapt the DETR framework for concurrently addressing Video Moment Retrieval and Highlight Detection tasks.[26] It employs a DETR-like encoder-decoder architecture.
  - **Inputs:** The model processes features extracted from the input video (e.g., using a CNN or a video-specific Transformer for frame/clip-level features) and features from the input text query (e.g., using a pre-trained language model like BERT or a Transformer encoder).
  - **Span Anchors & Queries:** In DETR-based MR models, the "object queries" from the original DETR are re-conceptualized as "moment queries." These queries typically consist of two components: a "span anchor" that provides positional guidance for the temporal segment (e.g., initial guesses for start/end times or duration) and a "content embedding" that carries semantic information derived from the query or learned by the model.[26]
  - **Temporal Localization:** The Transformer decoder, conditioned on the moment queries and the encoded video/text features, predicts a set of temporal spans (start time, end time) along with relevance scores indicating

how well each span matches the query. Similar to DETR, bipartite matching (using the Hungarian algorithm) is employed during training to establish a one-to-one assignment between predicted spans and ground truth moment spans.
    - **Limitations Addressed by Successors:** While foundational, early adaptations like Moment-DETR might have used learnable parameters for initializing span anchors. Subsequent research, such as in SA-DETR [26], argued that this approach is suboptimal for MR because span anchors in this context are often more tightly coupled to the specifics of the video-text pair rather than being generic.
- **SA-DETR (Span Aware Detection Transformer):**
    - **Improvement over Moment-DETR:** SA-DETR explicitly emphasizes the critical role of *instance-related* span anchors, meaning anchors that are more directly informed by the specific video and text query being processed.[26]
    - **Instance-Related Query Initialization:** Instead of initializing span anchors purely as learnable parameters, SA-DETR generates them based on the video-text pair itself. These instance-aware span anchors are then directly supervised using ground truth labels via Hungarian matching, leading to an "initialization moment loss".[26] Furthermore, SA-DETR incorporates denoise learning: it creates noisy query groups by perturbing the boundaries of ground truth spans. Training the model to denoise these perturbed spans enhances its robustness to varying initial anchor qualities and can accelerate convergence.
    - **Span Aware Refine Decoder:** The decoder in SA-DETR is designed to be "span aware." It includes a "Span Based Enhance Block" that refines each content embedding using features from video clips corresponding to its associated span anchor, guided by textual memory. This helps mitigate semantic mismatches. Additionally, Gaussian masks derived from the span anchors are used to modulate the cross-attention mechanism between content embeddings and fused video-text features, directly leveraging the correspondence between span anchors and video clips to guide the refinement process more effectively.[26]
    - **Prediction, Matching, Learning:** SA-DETR predicts temporal spans. Matching relies on the Hungarian algorithm. The learning process involves losses on the predicted span coordinates and their relevance to the query, including the aforementioned initialization moment loss.

B. Specialized Approaches for Temporal Localization

Beyond direct adaptations of object detectors, some methods have been developed with architectures more specifically tailored to the temporal nature of video data.

- **2D-TAN (Learning 2D Temporal Adjacent Networks):**
  - **Core Idea:** 2D-TAN introduces a unique way to model temporal relations between video moments by constructing a 2D map. In this map, one dimension represents the start time of a potential moment, and the other dimension represents its end time.[51] Each cell (i,j) in this map corresponds to a candidate moment spanning from time iτ to (j+1)τ. This 2D representation inherently covers moments of diverse lengths and explicitly captures their adjacent temporal relationships.
  - **Prediction:** 2D-TAN operates as a single-shot framework. The input video is first segmented into clips, and features are extracted for each clip. These clip features are then used to construct a 2D temporal feature map FM, where FM[a,b,:] stores the feature representation for the moment spanning from clip a to clip b. To manage computational cost, a sparse sampling strategy is used to select candidate moments rather than enumerating all possible start-end pairs.[51]
  - **Fusion & Context Modeling:** The 2D temporal feature map FM is fused with the encoded sentence feature (e.g., via Hadamard product after projection into a common space). A Temporal Adjacent Network, consisting of several 2D convolutional layers, is then applied to this fused 2D map. These convolutions allow the model to gradually perceive larger temporal contexts by aggregating information from adjacent moment candidates, thereby learning temporal dependencies and differences between them.[51]
  - **Matching/Scoring:** The output of the Temporal Adjacent Network is passed through a fully connected layer and a sigmoid function to produce a 2D score map. Each valid score pi on this map represents the matching confidence between a candidate moment and the input query sentence. The moment candidate corresponding to the maximum score is selected as the retrieved moment.[51]
  - **Learning:** During training, 2D-TAN uses a scaled Intersection over Union (tIoU - temporal IoU) value between each candidate moment and the ground truth moment as the supervision signal, rather than a hard binary label. The network is trained using a binary cross-entropy loss based on these scaled tIoU scores.[51]

C. Box/Span Prediction, Matching, and Learning in MR

Regardless of the specific architecture, the core components of localization—prediction, matching, and learning—are present in MR models:

- **Prediction:** Models predict (start_time, end_time) pairs, which are analogous to (x1,x2) coordinates in a 1D space. These predictions can be made directly or as offsets from proposed temporal anchors or spans.
- **Matching:**
  - **DETR-based methods (Moment-DETR, SA-DETR):** Employ the Hungarian algorithm for bipartite matching between the set of predicted temporal spans and the set of ground truth spans.[26] This enforces a one-to-one assignment.
  - **2D-TAN:** The matching is more implicit. Each cell (a,b) in the 2D score map corresponds to a candidate moment. The supervisory signal, which is the scaled tIoU with the ground truth, effectively defines how "positive" or "negative" each candidate is for training purposes.[51]
- **Learning:**
  - The learning process typically involves a regression loss for the predicted span coordinates (e.g., L1 loss, Smooth L1 loss, or an IoU-like loss adapted for 1D temporal spans, such as tIoU loss) and a classification-style loss for the relevance or confidence score of the span.
  - For instance, SA-DETR includes an "initialization moment loss" applied to its instance-related initialized queries, in addition to refinement losses.[26]

The adaptation of successful object detection paradigms, particularly DETR, to the domain of video moment retrieval underscores the generality and power of these architectural principles.[9] The core concepts of learnable queries, Transformer-based attention mechanisms for context modeling, and bipartite matching for set prediction have proven flexible enough to be translated from 2D spatial localization to 1D temporal localization. This suggests that the underlying mechanisms for identifying and parameterizing "objects" (whether spatial or temporal) share fundamental similarities.

However, the MR domain presents unique challenges that necessitate domain-specific adaptations. One such challenge relates to the nature of "span anchors." Unlike anchors in object detection, which aim to cover a wide variety of object sizes and aspect ratios within a single image, span anchors in MR are often more tightly coupled to the specific semantics of the video-text query pair.[26] A given text query usually refers to an event or action with a characteristic temporal signature. SA-DETR's introduction of instance-related span anchors, initialized based on the video-text pair rather than being purely learnable generic parameters, is a direct response to this domain specificity.[26]

Temporal context is another critical aspect of MR. The meaning and boundaries of a moment are often defined by the events that precede and follow it. 2D-TAN explicitly addresses this by constructing a 2D map of start and end times and applying 2D convolutions over this map to capture "adjacent temporal relations".[51] Transformer-based models, through their self-attention and cross-attention mechanisms operating on sequences of video clip features, can also implicitly learn these temporal dependencies and contextual relationships.

The matching process remains crucial in MR, just as in OD. DETR-based MR methods naturally extend the bipartite matching strategy to find unique assignments between predicted and ground truth temporal spans.[26] 2D-TAN's approach of using a scaled tIoU as a continuous target score for every candidate cell in its 2D map can be seen as a form of "soft" matching, where each candidate receives a graded supervisory signal indicating its quality relative to the ground truth.[51] Both approaches aim to provide effective learning signals for refining temporal boundary predictions.

V. Exploring Alternative Localization Paradigms (Beyond Generate-then-Match)

While the "generate more detection boxes than ground truth boxes, then match and filter" paradigm (and its variants like direct set prediction in DETR) has been dominant, research continues to explore alternative ways for models to acquire localization capabilities. This section delves into some of these alternative or emerging paradigms.

A. Keypoint-Based Localization (Revisited as an Alternative Component)

Keypoint-based approaches, such as CenterNet discussed in Section II.C.2, primarily use detected keypoints (e.g., object centers, corners) to *form* bounding boxes. In this context, they still align with the generate-and-match framework if the final output is a bounding box. However, the underlying principle of localizing semantic keypoints can be viewed as a more direct form of localization if these keypoints *themselves* constitute the desired structural output. For instance, in tasks like human pose estimation (see Section VI.C), the locations of anatomical joints are the primary goal.

The paradigm here shifts to:

1. Detect a sparse set of semantic keypoints.
2. Optionally, group or connect these keypoints to infer a larger structure (e.g., an object's extent or a human skeleton).

If the primary output is the set of keypoints, this method differs from generating a dense field of box proposals. The "matching" during training involves associating predicted keypoint locations (often represented on heatmaps) with ground truth keypoint locations, and the "learning" involves minimizing the error in these keypoint predictions.

B. Direct Set Prediction (Revisited as an Alternative Paradigm)

DETR and its variants (Section II.D.1) introduce the concept of direct set prediction, where the model outputs a fixed-size set of (bounding box, class label) pairs.[9] While DETR generates N predictions, where N is typically larger than the number of ground truth objects, the crucial Hungarian matching step enforces a one-to-one mapping between these predictions and the ground truth objects (or "no object" slots).

This "direct set prediction with unique matching" can be considered an alternative to the more traditional approach of over-generating candidate boxes and then heavily filtering them using Non-Maximum Suppression (NMS). The traditional "generate-then-match" often implies many-to-one or many-to-many initial matches that are subsequently resolved (e.g., an anchor might overlap with multiple GTs, or multiple anchors might overlap with one GT, requiring heuristics or NMS). DETR, by its architectural design and loss function, aims for unique, one-to-one assignments from the outset. This reduces reliance on post-processing heuristics and integrates the instance disambiguation problem more directly into the learning framework.

C. Emerging Paradigms: Diffusion Models for Object Detection

A more radical departure from traditional discriminative approaches is the application of generative diffusion models to object detection and localization tasks.[52] Diffusion models are a class of generative models that learn a data distribution by reversing a gradual noising process.

- **Core Idea for Object Detection:** Instead of predicting bounding boxes in a single forward pass or by refining predefined anchors, diffusion-based object detectors typically start with a set of random boxes (or other object representations like heatmaps) initialized from a noise distribution. They then iteratively refine these random initializations over a series of "denoising" steps, conditioned on the input image features, to produce the final set of accurate object detections.[53]
- **DiffusionDet (Noise-to-Box):** This was an early approach to adapt diffusion models for object detection.[53] It operates as a two-stage model where bounding boxes themselves are treated as the targets of the generative process.
  - **Training:** The "forward diffusion" process involves gradually adding Gaussian noise to ground truth bounding box coordinates until they resemble pure noise. The detection model (specifically its decoder) then learns to reverse this process, i.e., to denoise noisy boxes back to the original ground truth boxes, conditioned on image features.

    - **Inference:** Detection begins with a set of randomly sampled Gaussian boxes (pure noise). The learned denoising network then iteratively refines these boxes over multiple steps, also predicting class labels, to arrive at the final detections. DiffusionDet often requires a heuristic "box-renewal" process during inference, where poorly predicted boxes might be re-initialized from noise to improve exploration.
- **DiffusionPoint (Noise-to-Heatmap):** Proposed as a one-stage alternative to DiffusionDet, DiffusionPoint applies the diffusion process to object heatmaps rather than directly to bounding box coordinates.[53]
    - **Training:** This approach is often built upon anchor-free detection concepts like CenterNet. Ground truth object centers are used to create initial heatmaps. The forward diffusion process gradually adds noise to these heatmaps until they become random. The model learns to reverse this, denoising random heatmaps back into clean heatmaps representing object center likelihoods.
    - **Inference:** Starts with a random noise map and iteratively denoises it to produce a refined heatmap. Peaks in this final heatmap indicate object centers, from which bounding box dimensions and offsets can then be regressed, similar to CenterNet.
- **ODGEN (Object Detection GENeration):** While not a detection paradigm itself, ODGEN demonstrates another way diffusion models interact with localization.[55] It is a method to generate high-quality synthetic training images *conditioned on specified bounding box layouts*. By fine-tuning a pre-trained diffusion model on domain-specific object crops and full images, ODGEN can synthesize new images with objects placed according to input bounding boxes and class labels. This technique can be used for data augmentation to improve the performance of standard object detectors, especially in data-scarce domains.
- **How this constitutes an "Alternative Paradigm":**
    - **Iterative Refinement from Noise:** The core mechanism is fundamentally different. Instead of proposing many candidates and then selecting or refining a few (as in R-CNNs or anchor-based methods), or directly regressing parameters (as in YOLOv1 or FCOS), diffusion models begin with random noise (representing random boxes or heatmaps) and progressively "sculpt" these into meaningful detections through a learned, iterative denoising process. This is a generative, step-wise refinement rather than a discriminative proposal-filter or direct regression approach.
    - **Probabilistic Nature:** Diffusion models are inherently probabilistic, as they learn to model the data distribution $p(\text{boxes}|\text{image})$. This can offer a more nuanced way to handle uncertainty in localization compared to deterministic

regression.

- **Different "Matching" and "Learning" Dynamics:**
    - In training DiffusionDet [53], for example, the noisy box inputs to the denoising network are directly derived from ground truth boxes by adding noise. The "matching" is thus explicit: a specific noisy version of a ground truth box is expected to be denoised back towards that same ground truth box.
    - The "learning" objective is typically to predict the noise that was added at each step of the forward diffusion process, or equivalently, to predict the less noisy (or original uncorrupted) data state. This contrasts with IoU-based matching of many proposals against ground truths and minimizing a geometric loss on the best matches.

The application of diffusion models to object detection represents a conceptual shift from purely discriminative models to generative ones. Instead of learning a direct mapping from image features to box parameters and class labels, these models learn to reverse a stochastic noising process applied to the target outputs (boxes or heatmaps).[53] This "denoising to detect" philosophy is fundamentally different from traditional approaches. The iterative refinement inherent in the reverse diffusion process [53] potentially offers greater flexibility in handling complex scenes or uncertainties, as predictions evolve gradually over multiple steps. This could, for example, allow the model to explore a wider solution space or to adjust predictions more subtly than a single-shot regression or a fixed number of refinement stages.

The nature of "matching" and "learning" also changes. In a typical diffusion-based detector training, the correspondence between a noisy input sample (e.g., a GT box with added noise) and its clean target (the original GT box) is explicitly known.[53] The loss function then usually penalizes the difference between the network's prediction of the noise component and the actual noise added, or the difference between the predicted denoised state and the true clean state. This is distinct from the IoU-based matching prevalent in many discriminative detectors, where numerous proposals are compared against ground truths to find suitable positive samples for regression and classification losses.

VI. Localization in Other Computer Vision Tasks

Beyond object detection (OD) and video moment retrieval (MR), many other computer vision tasks require models to output structural information for localization within the input data. These tasks often build upon or adapt core localization principles developed in OD.

A. Instance Segmentation

- **Task Definition:** Instance segmentation extends object detection by not only localizing individual objects with bounding boxes but also providing a precise pixel-level mask for each distinct object instance.[56] It differentiates between separate instances of the same class (e.g., labeling each individual car in a scene with a unique mask).
- **Localization Technique:** A dominant approach for instance segmentation is "detection-based" or "detect-then-segment." These methods typically first employ an object detection framework to generate bounding box proposals for potential objects. Subsequently, a segmentation branch predicts a binary mask indicating which pixels within each proposal belong to the object.[56]
  - **Mask R-CNN** is a seminal example of this approach.[56] It extends Faster R-CNN by adding a parallel branch that predicts a segmentation mask for each Region of Interest (RoI) identified by the detection component. A key innovation in Mask R-CNN is **RoIAlign**, which replaces RoIPooling. RoIAlign uses bilinear interpolation to compute the exact values of the input features at floating-point locations within an RoI, avoiding the quantization errors of RoIPooling and leading to more accurate mask prediction [[556]].
- **Output Structure:** The output for each detected instance includes its class label, a bounding box, and a pixel-level binary mask.
- **Relation to Object Detection:** Instance segmentation directly builds upon object detection. The initial localization often starts with predicting bounding boxes, which are then refined with pixel-level masks.

B. Panoptic Segmentation

- **Task Definition:** Panoptic segmentation aims to provide a complete and coherent parsing of an image by unifying instance segmentation (for "thing" classes like cars, people, animals) and semantic segmentation (for "stuff" classes like road, sky, grass).[56] The goal is to assign each pixel in the image both a semantic label and an instance ID. For "stuff" classes, the instance ID is typically ignored or set to a default value, as individual instances are not distinguished.[60] Critically, the predicted segments must be non-overlapping.
- **Localization Technique:** Panoptic segmentation systems often combine the outputs of separate instance segmentation and semantic segmentation heads or pathways. A significant challenge is to resolve conflicts and ensure consistency between the "thing" predictions (which need instance IDs and non-overlapping masks) and "stuff" predictions (which cover background regions). Some methods

achieve this by first performing instance segmentation and then filling in the remaining pixels with semantic segmentation predictions, or by using heuristics and post-processing steps to merge outputs from independent instance and semantic segmenters.[60] More advanced approaches aim for end-to-end learning of panoptic outputs. For example, UPLAM uses particle filters that leverage panoptic maps for robot localization, where the likelihood of a particle (robot pose hypothesis) is updated based on the IoU between a locally generated panoptic map and a global panoptic map.[61]

- **Output Structure:** A single output map where every pixel is assigned a class label and an instance ID (the instance ID is meaningful only for "thing" classes).

C. Human Pose Estimation (HPE)

- **Task Definition:** Human Pose Estimation involves detecting and localizing anatomical keypoints (such as joints like elbows, wrists, knees, ankles) of one or more persons within an image or video.[62] The set of localized keypoints forms a skeletal representation of the person's pose. This can be done in 2D (pixel coordinates) or 3D (world coordinates).
- **Localization Techniques:**
  - **Heatmap Regression:** This is the dominant and most successful approach for keypoint localization in HPE.[65] For each type of keypoint (e.g., left elbow, right knee), the network is trained to predict a 2D heatmap. The intensity of each pixel in this heatmap represents the likelihood or confidence that the specific keypoint is located at that pixel. The final keypoint coordinate is typically derived by finding the location of the maximum activation (peak) in the predicted heatmap. During training, ground truth heatmaps are often generated by placing a 2D Gaussian kernel centered at the annotated keypoint location.
  - **Direct Coordinate Regression:** Earlier HPE methods attempted to directly regress the (x,y) (and z for 3D) coordinates of each keypoint using fully connected layers or convolutional regression heads.[65] While conceptually simpler, this approach generally yields lower accuracy compared to heatmap regression, especially for precise localization, as it can be harder to train and more sensitive to image variations.
  - **Top-Down vs. Bottom-Up Approaches:** [64]
    - *Top-Down:* These methods first detect individual persons in the image (typically obtaining bounding boxes using an object detector like YOLO or Faster R-CNN). Then, for each detected person (within their bounding box), a single-person pose estimation model is applied to localize their keypoints. Mask R-CNN, for example, can be adapted for top-down pose

        estimation.
        - *Bottom-Up:* These methods first detect all potential keypoints of a certain type (e.g., all elbows, all wrists) across the entire image, irrespective of which person they belong to. Subsequently, a grouping or parsing algorithm is used to associate these detected keypoints into individual person skeletons. OpenPose, which uses Part Affinity Fields (PAFs) to encode the association between pairs of body parts, and DeepCut are examples of bottom-up approaches.
    - **NerPE (Neural Radiance Pose Estimation / Continuous Heatmap Regression):** A more recent development, NerPE, aims to address the quantization errors inherent in traditional heatmap-based methods that discretize continuous keypoint locations onto a pixel grid.[66] NerPE uses an implicit neural representation (a coordinate-input MLP) to regress the confidence score for a body joint at *any* continuous 2D coordinate within the image range, conditioned on surrounding image features. This allows for continuous heatmap regression and can achieve sub-pixel localization precision without needing to generate very high-resolution heatmaps during inference.
- **Output Structure:** A set of 2D or 3D coordinates for each predefined keypoint, for each detected person in the input.

D. Visual Grounding (Referring Expression Comprehension)

- **Task Definition:** Visual Grounding, also known as Referring Expression Comprehension, involves localizing a specific object or region within an image that is described by a given natural language expression (text query).[68] This task requires a deep understanding of both visual content and language, as well as the ability to establish fine-grained correspondences between them.
- **Localization Techniques:**
    - **Two-Stage Methods:** These approaches typically first generate a set of object proposals (candidate regions) using a pre-trained object detector. Then, features are extracted for each proposal and for the text query. A matching or scoring mechanism is used to determine which proposal best corresponds to the textual description.[68] The performance of these methods can be limited by the quality and coverage of the initial proposals.
    - **One-Stage Methods:** To avoid reliance on a separate proposal generation stage, one-stage methods aim to directly predict the bounding box of the referred object. These methods often fuse visual features (from a CNN) and language features (from a language model) early in the pipeline and then perform dense regression of bounding box coordinates, potentially using

    anchor-based or anchor-free mechanisms similar to those in object detection.[68]
    - **DETR-based Methods:** More recently, Transformer-based architectures, particularly those inspired by DETR, have become prominent in visual grounding.[68] These models leverage the attention mechanisms of Transformers to effectively capture intra-modal (within vision or language) and inter-modal (between vision and language) relationships. They typically use object/region queries that are conditioned on the textual query features to directly predict the bounding box of the target object in an end-to-end manner. Multi-modal fusion within the Transformer encoder and/or decoder is a key component. For example, RefFormer [68] enhances DETR-like visual grounding by introducing a Query Adaptation (QA) module that generates "referential queries" by adaptively learning target-related information from multi-level image features, aiming to provide better prior context to the decoder.
- **Output Structure:** The typical output is a bounding box that localizes the object or region referred to by the textual description. Some methods might also output attention maps highlighting the relevant image regions.[69]

These diverse localization tasks showcase a spectrum of requirements for structural output, ranging from coarse bounding boxes in OD and Visual Grounding, to pixel-perfect masks for individual objects in Instance Segmentation, to a complete pixel-wise labeling of the entire scene with instance differentiation in Panoptic Segmentation, and finally to a sparse set of semantic keypoints in Human Pose Estimation.

A common thread across many of these tasks is the reuse and adaptation of techniques initially pioneered in object detection. For instance, Mask R-CNN for instance segmentation directly extends the Faster R-CNN object detection framework.[56] Top-down human pose estimation often begins with an object detection step to locate individuals.[64] Similarly, visual grounding methods frequently employ object proposal mechanisms from detectors or adapt entire detector architectures like DETR for their specific multi-modal needs.[68] This demonstrates that the core principles developed for proposing or predicting regions and learning their locations in object detection are foundational and transferable to a broader range of visual understanding problems.

While the underlying localization principles (e.g., feature extraction, regression heads, classification heads) might share commonalities, the specific structure of the output is meticulously tailored to the unique demands of each task. Object detection outputs bounding box coordinates and class labels.[6] Instance segmentation adds pixel-level masks to these.[56] Human Pose Estimation outputs sets of keypoint coordinates or generates heatmaps from which these coordinates are derived.[65] This specialization of the output representation is essential for meeting the distinct objectives of each task. For example, a heatmap provides a natural and effective way to represent the probability distribution of a keypoint's location [65], which is a different kind of structural information than the extent of a bounding box.

The "structural information" that models output also varies in its explicitness. Bounding boxes in OD explicitly define a simple rectangular region. Pixel masks in segmentation tasks explicitly delineate object boundaries, which can be arbitrarily complex. In Human Pose Estimation, the localized keypoints are discrete points; the "structure" (e.g., a human skeleton) is often inferred by connecting these keypoints based on a predefined topology or learned affinities, such as the Part Affinity Fields used in OpenPose.[64] This highlights the diverse ways in which "localization within the input" can be manifested and utilized.

VII. Synthesis and Future Directions

The exploration of localization mechanisms in deep learning models for tasks like Object Detection (OD) and Text-based Video Moment Retrieval (MR) reveals a dynamic field characterized by continuous innovation and adaptation. Several key trends have emerged, shaping the current landscape and pointing towards future research avenues.

A. Recap of Key Trends in Localization Learning

1. **Progression Towards End-to-End Learning:** A dominant trend has been the move from complex, multi-component pipelines (e.g., early R-CNN with separate proposal, feature extraction, classification, and regression stages) towards unified, end-to-end trainable networks (e.g., YOLO, SSD, FCOS, DETR). This integration generally leads to improved performance, faster inference, and simplified training.
2. **Sophistication in Matching/Label Assignment:** The strategies for matching predicted detections to ground truth annotations have evolved significantly. Initial methods relied on simple IoU thresholds. More advanced techniques include the one-to-one bipartite matching in DETR, and learnable, cost-based assignment strategies like Optimal Transport Assignment (OTA) and its simplification, SimOTA. These sophisticated methods aim to provide more accurate and robust supervisory signals during training.
3. **Alignment of Loss Functions with Evaluation Metrics:** There has been a clear

shift from using generic regression losses (L1, L2, Smooth L1) for bounding box coordinates to developing specialized IoU-based losses (GIoU, DIoU, CIoU, EIoU). These losses are designed to more directly optimize the geometric properties relevant to the primary evaluation metric (IoU), leading to better localization performance.

4. **Rise of Anchor-Free Detection:** To simplify detector design and reduce reliance on dataset-specific anchor hyperparameters, anchor-free methods like FCOS and CenterNet have gained prominence. These approaches predict object locations and extents directly from feature map locations or by detecting keypoints, without predefined anchor box priors.
5. **Impact of Attention Mechanisms and Transformers:** The introduction of Transformer architectures, particularly DETR and its derivatives like Deformable DETR, has provided new ways to model global image context, handle object relationships, and perform direct set prediction for objects. Deformable attention has specifically addressed the computational challenges of applying Transformers to high-resolution visual features.
6. **Cross-Task Adaptation and Generalization of Principles:** Core localization concepts and architectural components initially developed for object detection (e.g., FPNs, anchor mechanisms, DETR framework) are being successfully adapted and extended to other localization-centric tasks, including instance segmentation, panoptic segmentation, human pose estimation, video moment retrieval, and visual grounding. This demonstrates the underlying commonality of the localization problem across different visual domains.

B. Open Challenges and Unresolved Questions

Despite significant progress, several challenges persist in the field of localization:

1. **Robustness and Generalization:** Achieving consistently high localization accuracy in challenging real-world scenarios—such as those involving heavy occlusion, dense clutter, unusual object poses or scales, novel object categories, or adverse environmental conditions (e.g., poor illumination, bad weather)—remains an ongoing research focus.[73]
2. **Efficiency for Real-Time Applications:** The trade-off between detection accuracy and computational efficiency (latency, memory footprint) is a critical concern, especially for deployment on resource-constrained platforms like mobile devices or edge AI systems.[5]
3. **Small Object Detection:** Accurately detecting and localizing small objects continues to be a significant difficulty for many detection architectures, often due to loss of resolution in deep CNNs or insufficient feature representation.[18]

4. **Interpretability of Localization Decisions:** Current deep learning models for localization operate largely as "black boxes." Understanding *why* a model makes a particular localization decision (e.g., why it places a bounding box in a specific location or assigns a certain mask) is still an open and important research area for building trust and enabling debugging.
5. **Beyond 2D Bounding Boxes:** While 2D bounding box localization is relatively mature, extending robust and efficient localization to more complex outputs, such as precise 3D object detection and pose estimation [41], articulated object tracking, and fine-grained part localization, presents ongoing challenges.
6. **Unified Multi-Task Localization Frameworks:** Developing truly unified models that can seamlessly and efficiently perform a wide range of localization tasks (e.g., object detection, instance segmentation, keypoint estimation) within a single, coherent framework is an ambitious goal. Panoptic segmentation represents a step in this direction by unifying semantic and instance segmentation.

C. Potential Avenues for Future Research

The challenges outlined above also point to promising directions for future investigation:

1. **Novel Loss Functions:** Continued research into loss functions that can better capture perceptual similarity, handle extreme scale variations more effectively, demonstrate greater robustness to noisy or incomplete annotations, or incorporate richer geometric priors beyond simple box properties.
2. **Advanced and Learnable Label Assignment:** The development of even more sophisticated, computationally efficient, and perhaps fully learnable label assignment strategies that can adapt dynamically to different datasets, object types, and stages of training.
3. **Neuro-Symbolic Approaches:** Exploring the integration of deep learning's pattern recognition capabilities with symbolic reasoning or structured knowledge representations. This could lead to more interpretable, robust, and generalizable localization models that can leverage common sense or domain-specific constraints.
4. **Self-Supervised and Unsupervised Localization:** Reducing the heavy reliance on large-scale, meticulously annotated datasets for training localization models. Advances in self-supervised representation learning and unsupervised or weakly-supervised object discovery and localization are crucial.
5. **Continual and Lifelong Learning for Localization:** Enabling models to incrementally learn to localize new object categories or adapt to new

environments over time, without catastrophically forgetting previously learned knowledge.

6. **Diffusion Models as a Core Paradigm:** Further exploration and refinement of generative diffusion models as a primary paradigm for object detection and other localization tasks.[52] Their iterative refinement process and ability to model distributions may offer new ways to handle uncertainty, generate diverse hypotheses, and achieve high-fidelity localization. The use of diffusion models for generating synthetic training data with precise localization information (e.g., ODGEN [55]) is also a promising avenue for improving detector robustness and performance in data-limited scenarios.

Many of the "alternative" paradigms discussed, such as keypoint-based detection (which influenced models like CenterNet [23]) or direct set prediction (pioneered by DETR [9]), are increasingly becoming integrated into mainstream approaches or are inspiring new hybrid variants. This cross-pollination of ideas signifies a maturing field where powerful concepts are being combined and refined.

While the evolution from Smooth L1 loss to sophisticated IoU-based losses like EIoU (Section III.B) has greatly improved bounding box regression, there is likely still considerable scope for innovation, particularly in how models learn the precise extent of non-rigid, articulated, or heavily occluded objects. Current IoU-based losses primarily focus on aligning axis-aligned bounding boxes and may not fully capture the nuances required for these more complex scenarios. New loss concepts that incorporate richer shape priors or are more attuned to perceptual quality beyond simple overlap could be beneficial.

Finally, the pervasive need for large, accurately annotated datasets remains a significant bottleneck in supervised deep learning. Annotation for localization tasks—whether bounding boxes, pixel-level masks, or keypoints—is notoriously expensive and time-consuming. Consequently, research into methods that can learn robust localization with less supervision (e.g., few-shot learning, weakly-supervised learning, self-supervised learning) or generative approaches like ODGEN [55] that can synthesize high-quality, diverse, and correctly labeled training data will continue to be of paramount importance. These approaches hold the key to democratizing access to high-performance localization models and extending their applicability to a wider range of domains and data modalities.

Table 3: Key Label Assignment Strategies in Object Detection

Strategy Name	Mechanism Summary	Key Characteristics/Benefits	Example Model(s)
IoU Thresholds	Predictions are positive if IoU with any GT > threshold (e.g., 0.5), negative if IoU < lower_threshold (e.g., 0.1-0.4).	Simple, widely used heuristic. One-to-many possible (one GT can match multiple predictions).	Fast R-CNN [12], many early anchor-based detectors.
Max IoU (for RPN Anchors)	Anchor assigned to GT with highest IoU, or if its IoU with any GT > threshold (e.g., 0.7). Negative if max IoU < threshold (e.g., 0.3).	Prioritizes best anchor for each GT. Still can be one-to-many.	Faster R-CNN (RPN).[6]
SSD Default Box Matching	Each GT matched to best IoU default box. Then, default boxes matched to any GT with IoU > 0.5.	More generous assignment; multiple default boxes can be positive for one GT. Simplifies learning.	SSD.[10]
FCOS Point Assignment	Any feature map location (point) inside a GT box is a positive sample for that GT. Ambiguity (point in multiple GTs) resolved by minimal area GT.	Dense positive samples per GT. Center-ness branch used to weigh quality.	FCOS.[11]
CenterNet Keypoint Assignment	Locations on heatmap corresponding to GT keypoints (e.g., object centers) are positive, typically by rendering a Gaussian at GT location.	Focuses on precise keypoint locations. Heatmap loss (e.g., Focal variant) handles sparsity.	CenterNet (Objects as Points) [[22] (related)].
Hungarian Matching	Finds optimal	Enforces unique	DETR [9], Deformable

(DETR)	one-to-one bipartite matching between N predictions and N (padded) GTs by minimizing a global cost (class prob + box similarity).	prediction per GT. Enables NMS-free detection. Computationally more complex than local heuristics.	DETR.[27]
Optimal Transport Assignment (OTA)	Formulates assignment as an OT problem. Assigns labels from GTs (suppliers) to anchors/predictions (demanders) to minimize global transport cost (weighted cls+reg loss).	Global optimum. Dynamic assignment based on current model predictions. Can resolve ambiguity well. Iterative solution (e.g., Sinkhorn-Knopp).	Various modern detectors; original OTA paper.[13]
SimOTA	Simplified OTA. Uses cost matrix (cls+reg loss). Dynamic-k positives per GT based on IoU sums of top candidates. Greedy assignment based on cost.	Computationally cheaper than full OTA. Dynamic and cost-aware. Effective for high-performance detectors.	YOLOX.[32]

This table offers a comparative glance at the diverse strategies employed for label assignment in object detection. It underscores the evolution from simpler, static IoU-based heuristics to more complex, dynamic, and globally-aware optimization procedures. The choice of matching strategy profoundly impacts the supervisory signals provided to the model during training, influencing what features it learns and how effectively it localizes objects.

Works cited

1. [2412.05252] From classical techniques to convolution-based models: A review of object detection algorithms - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2412.05252
2. Object Detection with Deep Learning: A Review - arXiv, accessed June 7, 2025, http://arxiv.org/pdf/1807.05511
3. A Decade of You Only Look Once (YOLO) for Object Detection - arXiv, accessed June 7, 2025, https://arxiv.org/html/2504.18586v1

4. Non-Maximum Suppression (NMS) Explained - Ultralytics, accessed June 7, 2025, https://www.ultralytics.com/glossary/non-maximum-suppression-nms
5. A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS - arXiv, accessed June 7, 2025, https://arxiv.org/html/2304.00501v6
6. Faster R-CNN Object Detector | ArcGIS API for Python - Esri Developer, accessed June 7, 2025, https://developers.arcgis.com/python/latest/guide/faster-rcnn-object-detector/
7. YOLOv1 to YOLOv10: The fastest and most accurate real-time object detection systems - arXiv, accessed June 7, 2025, https://arxiv.org/html/2408.09332v1
8. You Only Look Once - Wikipedia, accessed June 7, 2025, https://en.wikipedia.org/wiki/You_Only_Look_Once
9. End-to-End Object Detection with Transformers, accessed June 7, 2025, https://arxiv.org/abs/2005.12872
10. [1512.02325] SSD: Single Shot MultiBox Detector - ar5iv - arXiv, accessed June 7, 2025, https://ar5iv.labs.arxiv.org/html/1512.02325
11. FCOS: Fully Convolutional One-Stage Object Detection, accessed June 7, 2025, http://arxiv.org/abs/1904.01355
12. Fast R-CNN, accessed June 7, 2025, https://arxiv.org/abs/1504.08083
13. OTA: Optimal Transport Assignment For Object Detection | PDF ..., accessed June 7, 2025, https://www.scribd.com/document/554744311/2103-14259
14. Bounding Box Regression Loss | CloudFactory Computer Vision Wiki, accessed June 7, 2025, https://wiki.cloudfactory.com/docs/mp-wiki/loss/bounding-box-regression-loss
15. Optimized Loss Functions for Object detection: A Case Study ... - arXiv, accessed June 7, 2025, https://arxiv.org/pdf/2011.05523/1000
16. Smooth L1 Loss | Mohit Jain, accessed June 7, 2025, https://mohitjain.me/wp-content/uploads/2018/03/smoothl1loss.pdf
17. arxiv.org, accessed June 7, 2025, https://arxiv.org/html/2401.10525v1
18. You Only Look Once: Unified, Real-Time Object Detection | Request ..., accessed June 7, 2025, https://www.researchgate.net/publication/278049038_You_Only_Look_Once_Unified_Real-Time_Object_Detection
19. YOLO Explained: From v1 to Present - viso.ai, accessed June 7, 2025, https://viso.ai/computer-vision/yolo-explained/
20. YOLO 1 through 5: A complete and detailed overview - Kaggle, accessed June 7, 2025, https://www.kaggle.com/code/vikramsandu/yolo-1-through-5-a-complete-and-detailed-overview
21. "Nano-YOLO" - insights on the multi-part loss function of a simplified YOLO v1, accessed June 7, 2025, https://towardsdatascience.com/nano-yolo-insights-on-the-multi-part-loss-function-of-a-simplified-yolo-v1-5104bdee7ff1/
22. CenterNet++ for Object Detection - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2204.08394

23. CenterNet++ for Object Detection - arXiv, accessed June 7, 2025, https://arxiv.org/pdf/2204.08394
24. [2212.06137] NMS Strikes Back - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2212.06137
25. Hausdorff Distance Matching with Adaptive Query Denoising for Rotated Detection Transformer - arXiv, accessed June 7, 2025, https://arxiv.org/html/2305.07598v5
26. SA-DETR:Span Aware Detection Transformer for Moment Retrieval - ACL Anthology, accessed June 7, 2025, https://aclanthology.org/2025.coling-main.510.pdf
27. [2010.04159] Deformable DETR: Deformable Transformers for End ..., accessed June 7, 2025, https://ar5iv.labs.arxiv.org/html/2010.04159
28. [2503.18287] Solar Radio Burst Detection Based on Deformable DETR - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2503.18287
29. [2409.05200] Lung-DETR: Deformable Detection Transformer for Sparse Lung Nodule Anomaly Detection - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2409.05200
30. Attention Deficit is Ordered! Fooling Deformable Vision Transformers with Collaborative Adversarial Patches - arXiv, accessed June 7, 2025, https://arxiv.org/html/2311.12914v2
31. A Deep Dive Into Non-Maximum Suppression (NMS) | Built In, accessed June 7, 2025, https://builtin.com/machine-learning/non-maximum-suppression
32. YOLOX vs YOLOv6-3.0: A Detailed Technical Comparison - Ultralytics YOLO Docs, accessed June 7, 2025, https://docs.ultralytics.com/compare/yolox-vs-yolov6/
33. YOLOX vs YOLOv7: A Detailed Technical Comparison - Ultralytics YOLO, accessed June 7, 2025, https://docs.ultralytics.com/compare/yolox-vs-yolov7/
34. Model Comparison: YOLOX vs DAMO-YOLO for Object Detection - Ultralytics YOLO Docs, accessed June 7, 2025, https://docs.ultralytics.com/compare/yolox-vs-damo-yolo/
35. YOLOX: Exceeding YOLO Series in 2021 - ResearchGate, accessed June 7, 2025, https://www.researchgate.net/publication/353343997_YOLOX_Exceeding_YOLO_Series_in_2021
36. simota – cjm-yolox-pytorch - GitHub Pages, accessed June 7, 2025, https://cj-mills.github.io/cjm-yolox-pytorch/simota.html
37. Selecting the Right Bounding Box Using Non-Max Suppression, accessed June 7, 2025, https://www.analyticsvidhya.com/blog/2020/08/selecting-the-right-bounding-box-using-non-max-suppression-with-implementation/
38. arXiv:2502.12524v1 [cs.CV] 18 Feb 2025, accessed June 7, 2025, https://arxiv.org/pdf/2502.12524
39. Self-Adjusting Smooth L1 Loss - SERP AI, accessed June 7, 2025, https://serp.ai/self-adjusting-smooth-l1-loss/
40. Focaler-IoU: More Focused Intersection over Union Loss - arXiv, accessed June 7, 2025, https://arxiv.org/pdf/2401.10525

41. Using Efficient IoU loss function in PointPillars Network For Detecting 3D Object - EPU Academic Staff - Erbil Polytechnic University, accessed June 7, 2025, https://academicstaff.epu.edu.iq/directory/sazan.mohammed/research/67_research_1_20230604085937793677.pdf
42. An Improved Bounding Box Regression Loss Function Based on CIOU Loss for Multi-scale Object Detection - ResearchGate, accessed June 7, 2025, https://www.researchgate.net/publication/356457078_An_Improved_Bounding_Box_Regression_Loss_Function_Based_on_CIOU_Loss_for_Multi-scale_Object_Detection
43. (PDF) Focal Loss for Dense Object Detection (2017) | Tsung-Yi Lin | 20552 Citations, accessed June 7, 2025, https://scispace.com/papers/focal-loss-for-dense-object-detection-j0su4gk9as
44. [1708.02002] Focal Loss for Dense Object Detection - ar5iv - arXiv, accessed June 7, 2025, https://ar5iv.labs.arxiv.org/html/1708.02002
45. Context-aware Keyword Attention for Moment Retrieval and Highlight Detection, accessed June 7, 2025, https://ojs.aaai.org/index.php/AAAI/article/view/32804/34959
46. Watch Video, Catch Keyword: Context-aware Keyword Attention for Moment Retrieval and Highlight Detection - arXiv, accessed June 7, 2025, https://arxiv.org/html/2501.02504v1
47. Background-aware Moment Detection for Video Moment Retrieval - arXiv, accessed June 7, 2025, https://arxiv.org/html/2306.02728v3
48. Saliency-Guided DETR for Moment Retrieval and Highlight Detection - arXiv, accessed June 7, 2025, https://arxiv.org/html/2410.01615v1
49. Language-based Audio Moment Retrieval - arXiv, accessed June 7, 2025, https://arxiv.org/pdf/2409.15672
50. arXiv:2501.07305v2 [cs.CV] 20 Mar 2025, accessed June 7, 2025, https://arxiv.org/pdf/2501.07305
51. Learning 2D Temporal Adjacent Networks for Moment Localization ..., accessed June 7, 2025, https://arxiv.org/abs/1912.03590
52. [2309.02049] Diffusion-based 3D Object Detection with Random Boxes - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2309.02049
53. One-Stage Object Detection with Diffusion Probabilistic Models, accessed June 7, 2025, http://vip.joonseok.net/courses/mlvu_2023_1/projects/09.pdf
54. TrackDiffusion: Multi-object Tracking Data Generation via Diffusion Models - arXiv, accessed June 7, 2025, https://arxiv.org/html/2312.00651v1
55. ODGEN: Domain-specific Object Detection Data Generation with Diffusion Models - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2405.15199
56. Instance Segmentation in Computer Vision [2024 Overview] | Encord, accessed June 7, 2025, https://encord.com/blog/instance-segmentation-guide-computer-vision/
57. Differences between Instance and Panoptic segmentation - Mindkosh AI, accessed June 7, 2025, https://mindkosh.com/blog/differences-between-instance-and-panoptic-segmentation/

58. Semantic Segmentation vs Object Detection: Understanding the Differences - Keymakr, accessed June 7, 2025, https://keymakr.com/blog/semantic-segmentation-vs-object-detection-understanding-the-differences/
59. What is the difference between object detection, semantic segmentation and localization?, accessed June 7, 2025, https://cs.stackexchange.com/questions/51387/what-is-the-difference-between-object-detection-semantic-segmentation-and-local
60. Panoptic Segmentation - CVF Open Access, accessed June 7, 2025, https://openaccess.thecvf.com/content_CVPR_2019/papers/Kirillov_Panoptic_Segmentation_CVPR_2019_paper.pdf
61. [Literature Review] uPLAM: Robust Panoptic Localization and Mapping Leveraging Perception Uncertainties - Moonlight, accessed June 7, 2025, https://www.themoonlight.io/en/review/uplam-robust-panoptic-localization-and-mapping-leveraging-perception-uncertainties
62. Review of models for estimating 3D human pose using deep ... - PeerJ, accessed June 7, 2025, https://peerj.com/articles/cs-2574/
63. Review on Human Pose Estimation and Human Body Joints Localization - ResearchGate, accessed June 7, 2025, https://www.researchgate.net/publication/357160044_Review_on_Human_Pose_Estimation_and_Human_Body_Joints_Localization
64. Human Pose Estimation: Ultimate Guide [2023 edition], accessed June 7, 2025, https://kili-technology.com/data-labeling/machine-learning/human-pose-estimation-ultimate-beginners-guide-2023-edition
65. Motion-Aware Heatmap Regression for Human Pose Estimation in Videos - IJCAI, accessed June 7, 2025, https://www.ijcai.org/proceedings/2024/0138.pdf
66. Continuous Heatmap Regression for Pose Estimation via Implicit Neural Representation, accessed June 7, 2025, https://openreview.net/forum?id=GgIJeoSLjQ¬eId=SpIqaRf2Ei
67. Continuous Heatmap Regression for Pose Estimation via Implicit Neural Representation, accessed June 7, 2025, https://neurips.cc/virtual/2024/poster/95889
68. Referencing Where to Focus: Improving Visual Grounding with Referential Query - NIPS, accessed June 7, 2025, https://proceedings.neurips.cc/paper_files/paper/2024/file/54c67d3db2df24a31cf045525f9460b9-Paper-Conference.pdf
69. [Literature Review] Your Large Vision-Language Model Only Needs A Few Attention Heads For Visual Grounding - Moonlight | AI Colleague for Research Papers, accessed June 7, 2025, https://www.themoonlight.io/en/review/your-large-vision-language-model-only-needs-a-few-attention-heads-for-visual-grounding
70. SimVG: A Simple Framework for Visual Grounding with Decoupled Multi-modal Fusion - NIPS papers, accessed June 7, 2025, https://proceedings.neurips.cc/paper_files/paper/2024/file/dc6319dde4fb182b22fb902da9418566-Paper-Conference.pdf

71. Grounding Beyond Detection: Enhancing Contextual Understanding in Embodied 3D Grounding - arXiv, accessed June 7, 2025, https://arxiv.org/html/2506.05199v1
72. What is Phrase Grounding? - Roboflow Blog, accessed June 7, 2025, https://blog.roboflow.com/what-is-phrase-grounding/
73. YOLO11 to Its Genesis: A Decadal and Comprehensive Review of The You Only Look Once (YOLO) Series - arXiv, accessed June 7, 2025, https://arxiv.org/html/2406.19407v5
74. You Only Look Once: Unified, Real-Time Object Detection | Request PDF - ResearchGate, accessed June 7, 2025, https://www.researchgate.net/publication/311609522_You_Only_Look_Once_Unified_Real-Time_Object_Detection
75. [1801.05918] Extend the shallow part of Single Shot MultiBox Detector via Convolutional Neural Network - arXiv, accessed June 7, 2025, https://arxiv.org/abs/1801.05918
76. Loss curves of the four loss functions a: CIoU loss. b: EIoU loss. c - ResearchGate, accessed June 7, 2025, https://www.researchgate.net/figure/Loss-curves-of-the-four-loss-functions-a-CIoU-loss-b-EIoU-loss-c-EIoU-loss-d-SIoU_fig6_379373183
77. Fully Convolutional One-Stage 3D Object Detection on LiDAR Range Images - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2205.13764
78. Diffusion Models in 3D Vision: A Survey - arXiv, accessed June 7, 2025, https://arxiv.org/html/2410.04738v1
79. Diffusion models for 3D generation: A survey - SciOpen, accessed June 7, 2025, https://www.sciopen.com/article/10.26599/CVM.2025.9450452
80. [2501.11430] A Survey on Diffusion Models for Anomaly Detection - arXiv, accessed June 7, 2025, https://arxiv.org/abs/2501.11430