

# YOLO-World: Real-Time Open-Vocabulary Object Detection

Tianheng Cheng<sup>3,2,\*</sup>, Lin Song<sup>1,\*□</sup>, Yixiao Ge<sup>1,2,†</sup>, Wenyu Liu<sup>3</sup>, Xinggang Wang<sup>3,□</sup>, Ying Shan<sup>1,2</sup>  
 \*equal contribution    † project lead    □ corresponding author

<sup>1</sup> Tencent AI Lab <sup>2</sup> ARC Lab, Tencent PCG

<sup>3</sup> School of EIC, Huazhong University of Science & Technology

Code & Models: [YOLO-World](#)

## Abstract

The You Only Look Once (YOLO) series of detectors have established themselves as efficient and practical tools. However, their reliance on predefined and trained object categories limits their applicability in open scenarios. Addressing this limitation, we introduce YOLO-World, an innovative approach that enhances YOLO with open-vocabulary detection capabilities through vision-language modeling and pre-training on large-scale datasets. Specifically, we propose a new Re-parameterizable Vision-Language Path Aggregation Network (RepVL-PAN) and region-text contrastive loss to facilitate the interaction between visual and linguistic information. Our method excels in detecting a wide range of objects in a zero-shot manner with high efficiency. On the challenging LVIS dataset, YOLO-World achieves 35.4 AP with 52.0 FPS on V100, which outperforms many state-of-the-art methods in terms of both accuracy and speed. Furthermore, the fine-tuned YOLO-World achieves remarkable performance on several downstream tasks, including object detection and open-vocabulary instance segmentation.

## 1. Introduction

Object detection has been a long-standing and fundamental challenge in computer vision with numerous applications in image understanding, robotics, and autonomous vehicles. Tremendous works [15, 26, 40, 42] have achieved significant breakthroughs in object detection with the development of deep neural networks. Despite the success of these methods, they remain limited as they only handle object detection with a fixed vocabulary, e.g., 80 categories in the COCO [25] dataset. Once object categories are defined and labeled, trained detectors can only detect those specific categories, thus limiting the ability and applicability of open

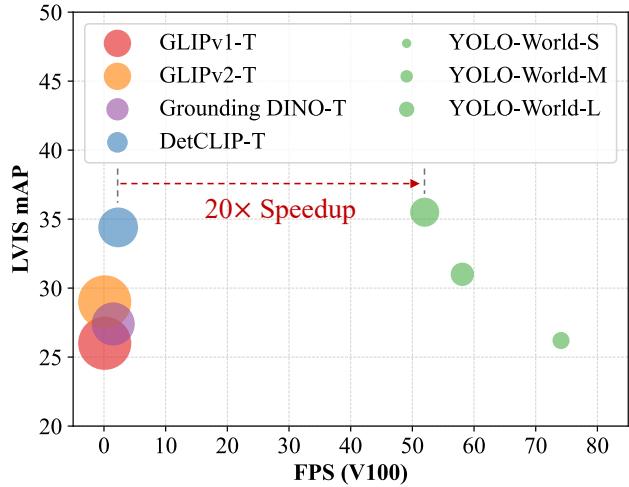


Figure 1. **Speed-and-Accuracy Curve.** We compare YOLO-World with recent open-vocabulary methods in terms of speed and accuracy. All models are evaluated on the LVIS minival and inference speeds are measured on one NVIDIA V100 w/o TensorRT. The size of the circle represents the model’s size.

scenarios.

Recent works [7, 12, 45, 50, 55] have explored the prevalent vision-language models [18, 36] to address open-vocabulary detection [55] through distilling vocabulary knowledge from language encoders, e.g., BERT [5]. However, these distillation-based methods are much limited due to the scarcity of training data with a limited diversity of vocabulary, e.g., OV-COCO [55] containing 48 base categories. Several methods [23, 29, 53, 54, 56] reformulate object detection training as region-level vision-language pre-training and train open-vocabulary object detectors at scale. However, those methods still struggle for detection in real-world scenarios, which suffer from two aspects: (1) heavy computation burden and (2) complicated deployment for edge devices. Previous works [23, 29, 53, 54, 56] have

demonstrated the promising performance of pre-training large detectors while pre-training small detectors to endow them with open recognition capabilities remains unexplored.

In this paper, we present YOLO-World, aiming for high-efficiency open-vocabulary object detection, and explore large-scale pre-training schemes to boost the traditional YOLO detectors to a new open-vocabulary world. Compared to previous methods, the proposed YOLO-World is remarkably efficient with high inference speed and easy to deploy for downstream applications. Specifically, YOLO-World follows the standard YOLO architecture [19] and leverages the pre-trained CLIP [36] text encoder to encode the input texts. We further propose the Re-parameterizable Vision-Language Path Aggregation Network (RepVL-PAN) to connect text features and image features for better visual-semantic representation. During inference, the text encoder can be removed and the text embeddings can be re-parameterized into weights of RepVL-PAN for efficient deployment. We further investigate the open-vocabulary pre-training scheme for YOLO detectors through region-text contrastive learning on large-scale datasets, which unifies detection data, grounding data, and image-text data into region-text pairs. The pre-trained YOLO-World with abundant region-text pairs demonstrates a strong capability for large vocabulary detection and training more data leads to greater improvements in open-vocabulary capability.

In addition, we explore a *prompt-then-detect* paradigm to further improve the efficiency of open-vocabulary object detection in real-world scenarios. As illustrated in Fig. 2, traditional object detectors [15, 19, 22, 38–40, 49] concentrate on the fixed-vocabulary (close-set) detection with predefined and trained categories. While previous open-vocabulary detectors [23, 29, 53, 56] encode the prompts of a user for online vocabulary with text encoders and detect objects. Notably, those methods tend to employ large detectors with heavy backbones, *e.g.*, Swin-L [31], to increase the open-vocabulary capacity. In contrast, the *prompt-then-detect* paradigm (Fig. 2 (c)) first encodes the prompts of a user to build an offline vocabulary and the vocabulary varies with different needs. Then, the efficient detector can infer the offline vocabulary on the fly without re-encoding the prompts. For practical applications, once we have trained the detector, *i.e.*, YOLO-World, we can pre-encode the prompts or categories to build an offline vocabulary and then seamlessly integrate it into the detector.

Our main contributions can be summarized into three folds:

- We introduce the YOLO-World, a cutting-edge open-vocabulary object detector with high efficiency for real-world applications.
- We propose a Re-parameterizable Vision-Language PAN

to connect vision and language features and an open-vocabulary region-text contrastive pre-training scheme for YOLO-World.

- The proposed YOLO-World pre-trained on large-scale datasets demonstrates strong zero-shot performance and achieves 35.4 AP on LVIS with 52.0 FPS. The pre-trained YOLO-World can be easily adapted to downstream tasks, *e.g.*, open-vocabulary instance segmentation and referring object detection. Moreover, the pre-trained weights and codes of YOLO-World will be open-sourced to facilitate more practical applications.

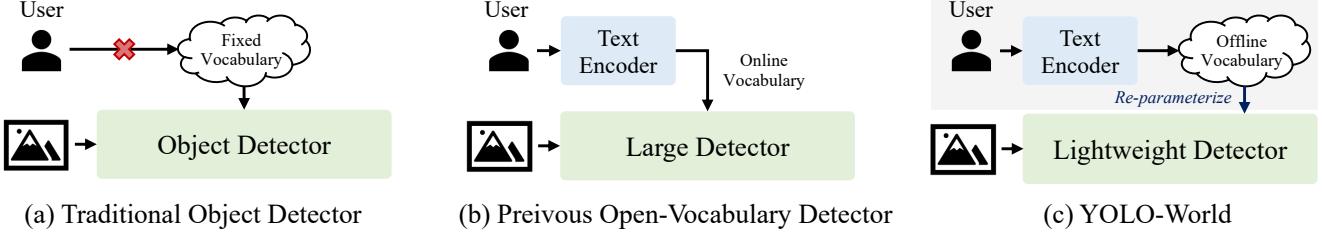
## 2. Related Works

### 2.1. Traditional Object Detection

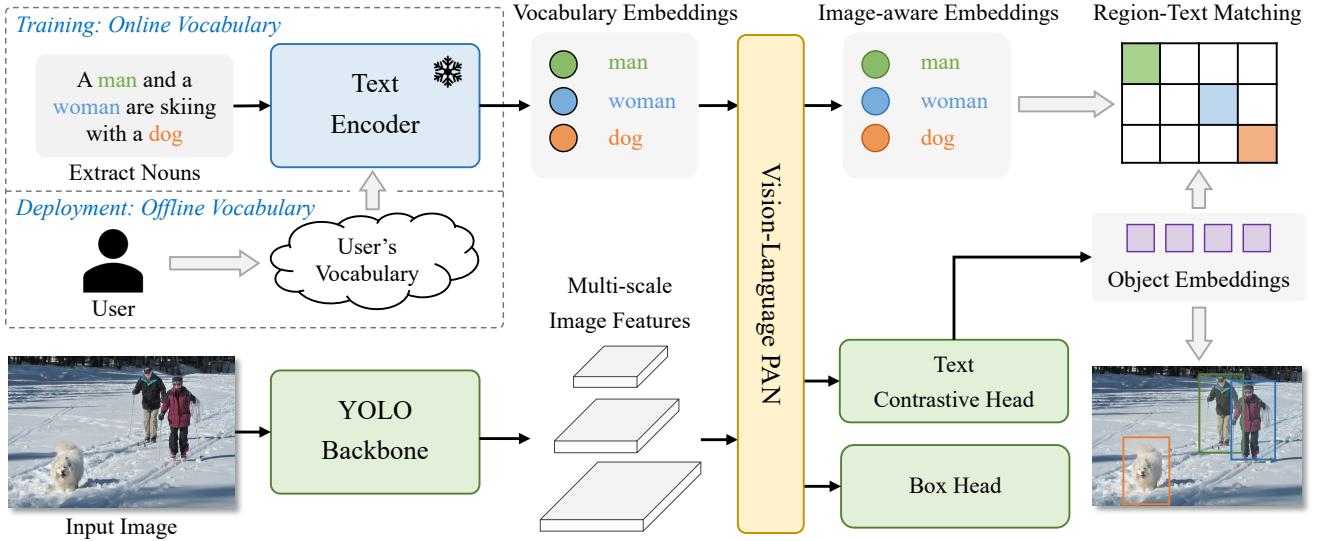
Prevalent object detection research concentrates on fixed-vocabulary (close-set) detection, in which object detectors are trained on datasets with pre-defined categories, *e.g.*, COCO dataset [25] and Objects365 dataset [43], and then detect objects within the fixed set of categories. During the past decades, the methods for traditional object detection can be simply categorized into three groups, *i.e.*, region-based methods, pixel-based methods, and query-based methods. The region-based methods [10, 11, 15, 26, 41], such as Faster R-CNN [41], adopt a two-stage framework for proposal generation [41] and RoI-wise (Region-of-Interest) classification and regression. The pixel-based methods [27, 30, 39, 46, 58] tend to be one-stage detectors, which perform classification and regression over pre-defined anchors or pixels. DETR [1] first explores object detection through transformers [47] and inspires extensive query-based methods [61]. In terms of inference speed, Redmon *et al.* presents YOLOs [37–39] which exploit simple convolutional architectures for real-time object detection. Several works [9, 22, 32, 49, 52] propose various architectures or designs for YOLO, including path aggregation networks [28], cross-stage partial networks [48], and re-parameterization [6], which further improve both speed and accuracy. In comparison to previous YOLOs, YOLO-World in this paper aims to detect objects beyond the fixed vocabulary with strong generalization ability.

### 2.2. Open-Vocabulary Object Detection

Open-vocabulary object detection (OVD) [55] has emerged as a new trend for modern object detection, which aims to detect objects beyond the predefined categories. Early works [12] follow the standard OVD setting [55] by training detectors on the base classes and evaluating the novel (unknown) classes. Nevertheless, this open-vocabulary setting can evaluate the capability of detectors to detect and recognize novel objects, it is still limited for open scenarios and lacks generalization ability to other domains due to training on the limited dataset and vocabulary.



**Figure 2. Comparison with Detection Paradigms.** (a) **Traditional Object Detector:** These object detectors can only detect objects within the fixed vocabulary pre-defined by the training datasets, e.g., 80 categories of COCO dataset [25]. The fixed vocabulary limits the extension for open scenes. (b) **Previous Open-Vocabulary Detectors:** Previous methods tend to develop large and heavy detectors for open-vocabulary detection which intuitively have strong capacity. In addition, these detectors simultaneously encode images and texts as input for prediction, which is time-consuming for practical applications. (c) **YOLO-World:** We demonstrate the strong open-vocabulary performance of lightweight detectors, e.g., YOLO detectors [19, 39], which is of great significance for real-world applications. Rather than using online vocabulary, we present a *prompt-then-detect* paradigm for efficient inference, in which the user generates a series of prompts according to the need and the prompts will be encoded into an offline vocabulary. Then it can be re-parameterized as the model weights for deployment and further acceleration.



**Figure 3. Overall Architecture of YOLO-World.** Compared to traditional YOLO detectors, YOLO-World as an open-vocabulary detector adopts text as input. The *Text Encoder* first encodes the input text into text embeddings. Then the *Image Encoder* encodes the input image into multi-scale image features and the proposed *RepVL-PAN* exploits the multi-level cross-modality fusion for both image and text features. Finally, YOLO-World predicts the regressed bounding boxes and the object embeddings for matching the categories or nouns that appeared in the input text.

Inspired by vision-language pre-training [18, 36], recent works [7, 21, 50, 59, 60] formulate open-vocabulary object detection as image-text matching and exploit large-scale image-text data to increase the training vocabulary at scale. GLIP [23] presents a pre-training framework for open-vocabulary detection based on phrase grounding and evaluates in a zero-shot setting. Grounding DINO [29] incorporates the grounded pre-training [23] into detection transformers [57] with cross-modality fusions.

Several methods [24, 53, 54, 56] unify detection datasets and image-text datasets through region-text matching and pre-train detectors with large-scale image-text pairs, achieving promising performance and generalization. However,

these methods often use heavy detectors like ATSS [58] or DINO [57] with Swin-L [31] as a backbone, leading to high computational demands and deployment challenges. In contrast, we present YOLO-World, aiming for efficient open-vocabulary object detection with real-time inference and easier downstream application deployment. Differing from ZSD-YOLO [51], which also explores open-vocabulary detection [55] with YOLO through language model alignment, YOLO-World introduces a novel YOLO framework with an effective pre-training strategy, enhancing open-vocabulary performance and generalization.

### 3. Method

#### 3.1. Pre-training Formulation: Region-Text Pairs

The traditional object detection methods, including the YOLO-series [19], are trained with instance annotations  $\Omega = \{B_i, c_i\}_{i=1}^N$ , which consist of bounding boxes  $\{B_i\}$  and category labels  $\{c_i\}$ . In this paper, we reformulate the instance annotations as region-text pairs  $\Omega = \{B_i, t_i\}_{i=1}^N$ , where  $t_i$  is the corresponding text for the region  $B_i$ . Specifically, the text  $t_i$  can be the category name, noun phrases, or object descriptions. Moreover, YOLO-World adopts both the image  $I$  and texts  $T$  (a set of nouns) as input and outputs predicted boxes  $\{\hat{B}_k\}$  and the corresponding object embeddings  $\{e_k\}$  ( $e_k \in \mathbb{R}^D$ ).

#### 3.2. Model Architecture

The overall architecture of the proposed YOLO-World is illustrated in Fig. 3, which consists of a *YOLO detector*, a *Text Encoder*, and a *Re-parameterizable Vision-Language Path Aggregation Network* (RepVL-PAN). Given the input text, the text encoder in YOLO-World encodes the text into text embeddings. The image encoder in the YOLO detector extracts the multi-scale features from the input image. Then we leverage the RepVL-PAN to enhance both text and image representation by exploiting the cross-modality fusion between image features and text embeddings.

**YOLO Detector.** YOLO-World is mainly developed based on YOLOv8 [19], which contains a Darknet backbone [19, 40] as the image encoder, a path aggregation network (PAN) for multi-scale feature pyramids, and a head for bounding box regression and object embeddings.

**Text Encoder.** Given the text  $T$ , we adopt the Transformer text encoder pre-trained by CLIP [36] to extract the corresponding text embeddings  $W = \text{TextEncoder}(T) \in \mathbb{R}^{C \times D}$ , where  $C$  is the number of nouns and  $D$  is the embedding dimension. The CLIP text encoder offers better visual-semantic capabilities for connecting visual objects with texts compared to text-only language encoders [5]. When the input text is a caption or referring expression, we adopt the simple n-gram algorithm to extract the noun phrases and then feed them into the text encoder.

**Text Contrastive Head.** Following previous works [19], we adopt the decoupled head with two  $3 \times 3$  convs to regress bounding boxes  $\{b_k\}_{k=1}^K$  and object embeddings  $\{e_k\}_{k=1}^K$ , where  $K$  denotes the number of objects. We present a text contrastive head to obtain the object-text similarity  $s_{k,j}$  by:

$$s_{k,j} = \alpha \cdot \text{L2-Norm}(e_k) \cdot \text{L2-Norm}(w_j)^\top + \beta, \quad (1)$$

where  $\text{L2-Norm}(\cdot)$  is the L2 normalization and  $w_j \in W$  is the  $j$ -th text embeddings. In addition, we add the affine

transformation with the learnable scaling factor  $\alpha$  and shifting factor  $\beta$ . Both the L2 norms and the affine transformations are important for stabilizing the region-text training.

**Training with Online Vocabulary.** During training, we construct an online vocabulary  $T$  for each mosaic sample containing 4 images. Specifically, we sample all positive nouns involved in the mosaic images and randomly sample some negative nouns from the corresponding dataset. The vocabulary for each mosaic sample contains at most  $M$  nouns, and  $M$  is set to 80 as default.

**Inference with Offline Vocabulary.** At the inference stage, we present a *prompt-then-detect* strategy with an offline vocabulary for further efficiency. As shown in Fig. 3, the user can define a series of custom prompts, which might include captions or categories. We then utilize the text encoder to encode these prompts and obtain offline vocabulary embeddings. The offline vocabulary allows for avoiding computation for each input and provides the flexibility to adjust the vocabulary as needed.

#### 3.3. Re-parameterizable Vision-Language PAN

Fig. 4 shows the structure of the proposed RepVL-PAN which follows the top-down and bottom-up paths in [19, 28] to establish the feature pyramids  $\{P_3, P_4, P_5\}$  with the multi-scale image features  $\{C_3, C_4, C_5\}$ . Furthermore, we propose the Text-guided CSPLayer (T-CSPLayer) and Image-Pooling Attention (I-Pooling Attention) to further enhance the interaction between image features and text features, which can improve the visual-semantic representation for open-vocabulary capability. During inference, the offline vocabulary embeddings can be re-parameterized into weights of convolutional or linear layers for deployment.

**Text-guided CSPLayer.** As Fig. 4 illustrates, the cross-stage partial layers (CSPLayer) are utilized after the top-down or bottom-up fusion. We extend the CSPLayer (also called C2f) of [19] by incorporating text guidance into multi-scale image features to form the Text-guided CSPLayer. Specifically, given the text embeddings  $W$  and image features  $X_l \in \mathbb{R}^{H \times W \times D}$  ( $l \in \{3, 4, 5\}$ ), we adopt the *max-sigmoid attention* after the last dark bottleneck block to aggregate text features into image features by:

$$X'_l = X_l \cdot \delta \left( \max_{j \in \{1..C\}} (X_l W_j^\top) \right)^\top, \quad (2)$$

where the updated  $X'_l$  is concatenated with the cross-stage features as output. The  $\delta$  indicates the sigmoid function.

**Image-Pooling Attention.** To enhance the text embeddings with image-aware information, we aggregate image

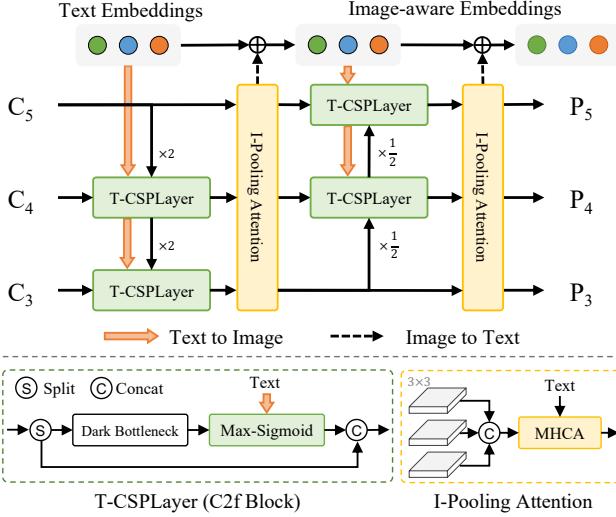


Figure 4. **Illustration of the RepVL-PAN.** The proposed RepVL-PAN adopts the *Text-guided CSPLayer* (T-CSPLayer) for injecting language information into image features and the *Image Pooling Attention* (I-Pooling Attention) for enhancing image-aware text embeddings.

features to update the text embeddings by proposing the Image-Pooling Attention. Rather than directly using cross-attention on image features, we leverage max pooling on multi-scale features to obtain  $3 \times 3$  regions, resulting in a total of 27 patch tokens  $\tilde{X} \in \mathbb{R}^{27 \times D}$ . The text embeddings are then updated by:

$$W' = W + \text{MultiHead-Attention}(W, \tilde{X}, \tilde{X}) \quad (3)$$

### 3.4. Pre-training Schemes

In this section, we present the training schemes for pre-training YOLO-World on large-scale detection, grounding, and image-text datasets.

**Learning from Region-Text Contrastive Loss.** Given the mosaic sample  $I$  and texts  $T$ , YOLO-World outputs  $K$  object predictions  $\{B_k, s_k\}_{k=1}^K$  along with annotations  $\Omega = \{B_i, t_i\}_{i=1}^N$ . We follow [19] and leverage task-aligned label assignment [8] to match the predictions with ground-truth annotations and assign each positive prediction with a text index as the classification label. Based on this vocabulary, we construct the region-text contrastive loss  $\mathcal{L}_{\text{con}}$  with region-text pairs through cross entropy between object-text (region-text) similarity and object-text assignments. In addition, we adopt IoU loss and distributed focal loss for bounding box regression and the total training loss is defined as:  $\mathcal{L}(I) = \mathcal{L}_{\text{con}} + \lambda_I \cdot (\mathcal{L}_{\text{iou}} + \mathcal{L}_{\text{df}},)$ , where  $\lambda_I$  is an indicator factor and set to 1 when input image  $I$  is from detection or grounding data and set to 0 when it is from the image-text data. Considering image-text datasets have

noisy boxes, we only calculate the regression loss for samples with accurate bounding boxes.

**Pseudo Labeling with Image-Text Data.** Rather than directly using image-text pairs for pre-training, we propose an automatic labeling approach to generate region-text pairs. Specifically, the labeling approach contains three steps: (1) *extract noun phrases*: we first utilize the n-gram algorithm to extract noun phrases from the text; (2) *pseudo labeling*: we adopt a pre-trained open-vocabulary detector, *e.g.*, GLIP [23], to generate pseudo boxes for the given noun phrases for each image, thus providing the coarse region-text pairs. (3) *filtering*: We employ the pre-trained CLIP [36] to evaluate the relevance of image-text pairs and region-text pairs, and filter the low-relevance pseudo annotations and images. We further filter redundant bounding boxes by incorporating methods such as Non-Maximum Suppression (NMS). We suggest the readers refer to the appendix for the detailed approach. With the above approach, we sample and label 246k images from CC3M [44] with 821k pseudo annotations.

## 4. Experiments

In this section, we demonstrate the effectiveness of the proposed YOLO-World by pre-training it on large-scale datasets and evaluating YOLO-World in a zero-shot manner on both LVIS benchmark and COCO benchmark (Sec. 4.2). We also evaluate the fine-tuning performance of YOLO-World on COCO, LVIS for object detection.

### 4.1. Implementation Details

The YOLO-World is developed based on the MMYOLO toolbox [3] and the MMDetection toolbox [2]. Following [19], we provide three variants of YOLO-World for different latency requirements, *e.g.*, small (S), medium (M), and large (L). We adopt the open-source CLIP [36] text encoder with pre-trained weights to encode the input text. Unless specified, we measure the inference speeds of all models on one NVIDIA V100 GPU without extra acceleration mechanisms, *e.g.*, FP16 or TensorRT.

### 4.2. Pre-training

**Experimental Setup.** At the pre-training stage, we adopt the AdamW optimizer [33] with an initial learning rate of 0.002 and weight decay of 0.05. YOLO-World is pre-trained for 100 epochs on 32 NVIDIA V100 GPUs with a total batch size of 512. During pre-training, we follow previous works [19] and adopt color augmentation, random affine, random flip, and mosaic with 4 images for data augmentation. The text encoder is frozen during pre-training.

| Dataset                | Type       | Vocab. | Images | Anno.  |
|------------------------|------------|--------|--------|--------|
| Objects365V1 [43]      | Detection  | 365    | 609k   | 9,621k |
| GQA [16]               | Grounding  | -      | 621k   | 3,681k |
| Flickr [35]            | Grounding  | -      | 149k   | 641k   |
| CC3M <sup>†</sup> [44] | Image-Text | -      | 246k   | 821k   |

Table 1. **Pre-training Data.** The specifications of the datasets used for pre-training YOLO-World.

**Pre-training Data.** For pre-training YOLO-World, we mainly adopt detection or grounding datasets including Objects365 (V1) [43], GQA [16], Flickr30k [35], as specified in Tab. 1. Following [23], we exclude the images from the COCO dataset in GolG [20] (GQA and Flickr30k). The annotations of the detection datasets used for pre-training contain both bounding boxes and categories or noun phrases. In addition, we also extend the pre-training data with image-text pairs, *i.e.*, CC3M<sup>†</sup> [44], which we have labeled 246k images through the pseudo-labeling method discussed in Sec. 3.4.

**Zero-shot Evaluation.** After pre-training, we directly evaluate the proposed YOLO-World on the LVIS dataset [13] in a zero-shot manner. The LVIS dataset contains 1203 object categories, which is much more than the categories of the pre-training detection datasets and can measure the performance on large vocabulary detection. Following previous works [20, 23, 53, 54], we mainly evaluate on LVIS minival [20] and report the *Fixed AP* [4] for comparison. The maximum number of predictions is set to 1000.

**Main Results on LVIS Object Detection.** In Tab. 2, we compare the proposed YOLO-World with recent state-of-the-art methods [20, 29, 53, 54, 56] on LVIS benchmark in a zero-shot manner. Considering the computation burden and model parameters, we mainly compare with those methods based on lighter backbones, *e.g.*, Swin-T [31]. Remarkably, YOLO-World outperforms previous state-of-the-art methods in terms of zero-shot performance and inference speed. Compared to GLIP, GLIPv2, and Grounding DINO, which incorporate more data, *e.g.*, Cap4M (CC3M+SBU [34]), YOLO-World pre-trained on O365 & GolG obtains better performance even with fewer model parameters. Compared to DetCLIP, YOLO-World achieves comparable performance (35.4 *v.s.* 34.4) while obtaining 20× increase in inference speed. *The experimental results also demonstrate that small models, e.g., YOLO-World-S with 13M parameters, can be used for vision-language pre-training and obtain strong open-vocabulary capabilities.*

### 4.3. Ablation Experiments

We provide extensive ablation studies to analyze YOLO-World from two primary aspects, *i.e.*, pre-training and architecture. Unless specified, we mainly conduct ablation experiments based on YOLO-World-L and pre-train Objects365 with zero-shot evaluation on LVIS minival.

**Pre-training Data.** In Tab. 3, we evaluate the performance of pre-training YOLO-World using different data. Compared to the baseline trained on Objects365, adding GQA can significantly improve performance with an 8.4 AP gain on LVIS. This improvement can be attributed to the richer textual information provided by the GQA dataset, which can enhance the model’s ability to recognize large vocabulary objects. Adding part of CC3M samples (8% of the full datasets) can further bring 0.5 AP gain with 1.3 AP on rare objects. Tab. 3 demonstrates that adding more data can effectively improve the detection capabilities on large-vocabulary scenarios. Furthermore, as the amount of data increases, the performance continues to improve, highlighting the benefits of leveraging larger and more diverse datasets for training.

**Ablations on RepVL-PAN.** Tab. 4 demonstrates the effectiveness of the proposed RepVL-PAN of YOLO-World, including Text-guided CSPLayers and Image Pooling Attention, for the zero-shot LVIS detection. Specifically, we adopt two settings, *i.e.*, (1) pre-training on O365 and (2) pre-training on O365 & GQA. Compared to O365 which only contains category annotations, GQA includes rich texts, particularly in the form of noun phrases. As shown in Tab. 4, the proposed RepVL-PAN improves the baseline (YOLOv8-PAN [19]) by 1.1 AP on LVIS, and the improvements are remarkable in terms of the rare categories ( $AP_r$ ) of LVIS, which are hard to detect and recognize. In addition, the improvements become more significant when YOLO-World is pre-trained with the GQA dataset and experiments indicate that the proposed RepVL-PAN works better with rich textual information.

**Text Encoders.** In Tab. 5, we compare the performance of using different text encoders, *i.e.*, BERT-base [5] and CLIP-base (ViT-base) [36]. We exploit two settings during pre-training, *i.e.*, frozen and fine-tuned, and the learning rate for fine-tuning text encoders is a  $0.01\times$  factor of the basic learning rate. As Tab. 5 shows, the CLIP text encoder obtains superior results than BERT (+10.1 AP for rare categories in LVIS), which is pre-trained with image-text pairs and has better capability for vision-centric embeddings. Fine-tuning BERT during pre-training brings significant improvements (+3.7 AP) while fine-tuning CLIP leads to a severe performance drop. We attribute the drop to that

| Method                | Backbone    | Params     | Pre-trained Data             | FPS         | AP   | AP <sub>r</sub> | AP <sub>c</sub> | AP <sub>f</sub> |
|-----------------------|-------------|------------|------------------------------|-------------|------|-----------------|-----------------|-----------------|
| MDETR [20]            | R-101 [14]  | 169M       | GoldG                        | -           | 24.2 | 20.9            | 24.3            | 24.2            |
| GLIP-T [23]           | Swin-T [31] | 232M       | O365,GoldG                   | 0.12        | 24.9 | 17.7            | 19.5            | 31.0            |
| GLIP-T [23]           | Swin-T [31] | 232M       | O365,GoldG,Cap4M             | 0.12        | 26.0 | 20.8            | 21.4            | 31.0            |
| GLIPv2-T [56]         | Swin-T [31] | 232M       | O365,GoldG                   | 0.12        | 26.9 | -               | -               | -               |
| GLIPv2-T [56]         | Swin-T [31] | 232M       | O365,GoldG,Cap4M             | 0.12        | 29.0 | -               | -               | -               |
| Grounding DINO-T [29] | Swin-T [31] | 172M       | O365,GoldG                   | 1.5         | 25.6 | 14.4            | 19.6            | 32.2            |
| Grounding DINO-T [29] | Swin-T [31] | 172M       | O365,GoldG,Cap4M             | 1.5         | 27.4 | 18.1            | 23.3            | 32.7            |
| DetCLIP-T [53]        | Swin-T [31] | 155M       | O365,GoldG                   | 2.3         | 34.4 | 26.9            | 33.9            | 36.3            |
| YOLO-World-S          | YOLOv8-S    | 13M (77M)  | O365,GoldG                   | 74.1 (19.9) | 26.2 | 19.1            | 23.6            | 29.8            |
| YOLO-World-M          | YOLOv8-M    | 29M (92M)  | O365,GoldG                   | 58.1 (18.5) | 31.0 | 23.8            | 29.2            | 33.9            |
| YOLO-World-L          | YOLOv8-L    | 48M (110M) | O365,GoldG                   | 52.0 (17.6) | 35.0 | 27.1            | 32.8            | 38.3            |
| YOLO-World-L          | YOLOv8-L    | 48M (110M) | O365,GoldG,CC3M <sup>†</sup> | 52.0 (17.6) | 35.4 | 27.6            | 34.1            | 38.0            |

Table 2. **Zero-shot Evaluation on LVIS.** We evaluate YOLO-World on LVIS minival [20] in a zero-shot manner. We report the *Fixed AP* [4] for a fair comparison with recent methods. <sup>†</sup> denotes the pseudo-labeled CC3M in our setting, which contains 246k samples. The FPS is evaluated on one NVIDIA V100 GPU w/o TensorRT. The parameters and FPS of YOLO-World are evaluated for both the re-parameterized version (w/o bracket) and the original version (w/ bracket).

| Pre-trained Data             | AP          | AP <sub>r</sub> | AP <sub>c</sub> | AP <sub>f</sub> |
|------------------------------|-------------|-----------------|-----------------|-----------------|
| O365                         | 23.5        | 16.2            | 21.1            | 27.0            |
| O365,GQA                     | 31.9        | 22.5            | 29.9            | 35.4            |
| O365,GoldG                   | 32.5        | 22.3            | 30.6            | <b>36.0</b>     |
| O365,GoldG,CC3M <sup>†</sup> | <b>33.0</b> | <b>23.6</b>     | <b>32.0</b>     | 35.5            |

Table 3. **Ablations on Pre-training Data.** We evaluate the zero-shot performance on LVIS of pre-training YOLO-World with different amounts of data.

| GQA | T→I | I→T | AP          | AP <sub>r</sub> | AP <sub>c</sub> | AP <sub>f</sub> |
|-----|-----|-----|-------------|-----------------|-----------------|-----------------|
| ✗   | ✗   | ✗   | 22.4        | 14.5            | 20.1            | 26.0            |
| ✗   | ✓   | ✗   | 23.2        | 15.2            | 20.6            | 27.0            |
| ✗   | ✓   | ✓   | 23.5        | 16.2            | 21.1            | 27.0            |
| ✓   | ✗   | ✗   | 29.7        | 21.0            | 27.1            | 33.6            |
| ✓   | ✓   | ✓   | <b>31.9</b> | <b>22.5</b>     | <b>29.9</b>     | <b>35.4</b>     |

Table 4. **Ablations on Re-parameterizable Vision-Language Path Aggregation Network.** We evaluate the zero-shot performance on LVIS of the proposed Vision-Language Path Aggregation Network. T→I and I→T denote the Text-guided CSPLayers and Image-Pooling Attention, respectively.

fine-tuning on O365 may degrade the generalization ability of the pre-trained CLIP, which contains only 365 categories and lacks abundant textual information.

#### 4.4. Fine-tuning YOLO-World

In this section, we further fine-tune YOLO-World for close-set object detection on the COCO dataset and LVIS dataset

| Text Encoder | Frozen?   | AP          | AP <sub>r</sub> | AP <sub>c</sub> | AP <sub>f</sub> |
|--------------|-----------|-------------|-----------------|-----------------|-----------------|
| BERT-base    | Frozen    | 14.6        | 3.4             | 10.7            | 20.0            |
| BERT-base    | Fine-tune | 18.3        | 6.6             | 14.6            | 23.6            |
| CLIP-base    | Frozen    | <b>22.4</b> | <b>14.5</b>     | <b>20.1</b>     | <b>26.0</b>     |
| CLIP-base    | Fine-tune | 19.3        | 8.6             | 15.7            | 24.8            |

Table 5. **Text Encoder in YOLO-World.** We ablate different text encoders in YOLO-World through the zero-shot LVIS evaluation.

to demonstrate the effectiveness of the pre-training.

**Experimental Setup.** We use the pre-trained weights to initialize YOLO-World for fine-tuning. All models are fine-tuned for 80 epochs with the AdamW optimizer and the initial learning rate is set to 0.0002. In addition, we fine-tune the CLIP text encoder with a learning factor of 0.01. For the LVIS dataset, we follow previous works [7, 12, 60] and fine-tune YOLO-World on the LVIS-base (common & frequent) and evaluate it on the LVIS-novel (rare).

**COCO Object Detection.** We compare the pre-trained YOLO-World with previous YOLO detectors [19, 22, 49] in Tab. 6. For fine-tuning YOLO-World on the COCO dataset, we remove the proposed RepVL-PAN for further acceleration considering that the vocabulary size of the COCO dataset is small. In Tab. 6, it's evident that our approach can achieve decent zero-shot performance on the COCO dataset, which indicates that YOLO-World has strong generalization ability. Moreover, YOLO-World after fine-tuning on the COCO train2017 demonstrates

| Method                           | Pre-train | AP   | AP <sub>50</sub> | AP <sub>75</sub> | FPS |
|----------------------------------|-----------|------|------------------|------------------|-----|
| <i>Training from scratch.</i>    |           |      |                  |                  |     |
| YOLOv6-S [22]                    | ✗         | 43.7 | 60.8             | 47.0             | 442 |
| YOLOv6-M [22]                    | ✗         | 48.4 | 65.7             | 52.7             | 277 |
| YOLOv6-L [22]                    | ✗         | 50.7 | 68.1             | 54.8             | 166 |
| YOLOv7-T [49]                    | ✗         | 37.5 | 55.8             | 40.2             | 404 |
| YOLOv7-L [49]                    | ✗         | 50.9 | 69.3             | 55.3             | 182 |
| YOLOv7-X [49]                    | ✗         | 52.6 | 70.6             | 57.3             | 131 |
| YOLOv8-S [19]                    | ✗         | 44.4 | 61.2             | 48.1             | 386 |
| YOLOv8-M [19]                    | ✗         | 50.5 | 67.3             | 55.0             | 238 |
| YOLOv8-L [19]                    | ✗         | 52.9 | 69.9             | 57.7             | 159 |
| <i>Zero-shot transfer.</i>       |           |      |                  |                  |     |
| YOLO-World-S                     | O+G       | 37.6 | 52.3             | 40.7             | -   |
| YOLO-World-M                     | O+G       | 42.8 | 58.3             | 46.4             | -   |
| YOLO-World-L                     | O+G       | 44.4 | 59.8             | 48.3             | -   |
| YOLO-World-L                     | O+G+C     | 45.1 | 60.7             | 48.9             | -   |
| <i>Fine-tuned w/ RepVL-PAN.</i>  |           |      |                  |                  |     |
| YOLO-World-S                     | O+G       | 45.9 | 62.3             | 50.1             | -   |
| YOLO-World-M                     | O+G       | 51.2 | 68.1             | 55.9             | -   |
| YOLO-World-L                     | O+G+C     | 53.3 | 70.1             | 58.2             | -   |
| <i>Fine-tuned w/o RepVL-PAN.</i> |           |      |                  |                  |     |
| YOLO-World-S                     | O+G       | 45.7 | 62.3             | 49.9             | 373 |
| YOLO-World-M                     | O+G       | 50.7 | 67.2             | 55.1             | 231 |
| YOLO-World-L                     | O+G+C     | 53.3 | 70.3             | 58.1             | 156 |

Table 6. **Comparison with YOLOs on COCO Object Detection.** We fine-tune the YOLO-World on COCO `train2017` and evaluate on COCO `val2017`. The results of YOLOv7 [49] and YOLOv8 [19] are obtained from MMYOLO [3]. ‘O’, ‘G’, and ‘C’ denote pertaining using Objects365, GoldG, and CC3M<sup>†</sup>, respectively. The FPS is measured on one NVIDIA V100 w/ TensorRT.

higher performance compared to previous methods trained from scratch.

**LVIS Object Detection.** In Tab. 7, we evaluate the fine-tuning performance of YOLO-World on the standard LVIS dataset. Firstly, compared to the oracle YOLOv8s [19] trained on the full LVIS datasets, YOLO-World achieves significant improvements, especially for larger models, *e.g.*, YOLO-World-L outperforms YOLOv8-L by 7.2 AP and 10.2 AP<sub>r</sub>. The improvements can demonstrate the effectiveness of the proposed pre-training strategy for large-vocabulary detection. Moreover, YOLO-World, as an efficient one-stage detector, outperforms previous state-of-the-art two-stage methods [7, 12, 21, 50, 60] on the overall performance without extra designs, *e.g.*, learnable prompts [7] or region-based alignments [12].

| Method          | AP   | AP <sub>r</sub> | AP <sub>c</sub> | AP <sub>f</sub> |
|-----------------|------|-----------------|-----------------|-----------------|
| ViLD [12]       | 27.8 | 16.7            | 26.5            | 34.2            |
| RegionCLIP [59] | 28.2 | 17.1            | -               | -               |
| Detic [60]      | 26.8 | 17.8            | -               | -               |
| FVLM [21]       | 24.2 | 18.6            | -               | -               |
| DetPro [7]      | 28.4 | 20.8            | 27.8            | 32.4            |
| BARON [50]      | 29.5 | 23.2            | 29.3            | 32.5            |
| YOLOv8-S        | 19.4 | 7.4             | 17.4            | 27.0            |
| YOLOv8-M        | 23.1 | 8.4             | 21.3            | 31.5            |
| YOLOv8-L        | 26.9 | 10.2            | 25.4            | 35.8            |
| YOLO-World-S    | 23.9 | 12.8            | 20.4            | 32.7            |
| YOLO-World-M    | 28.8 | 15.9            | 24.6            | 39.0            |
| YOLO-World-L    | 34.1 | 20.4            | 31.1            | 43.5            |

Table 7. **Comparison with Open-Vocabulary Detectors on LVIS.** We train YOLO-World on the LVIS-base (including common and frequent) report the *bbox AP*. The YOLO-v8 are trained on the full LVIS datasets (including base and novel) along with the class balanced sampling.

#### 4.5. Open-Vocabulary Instance Segmentation

In this section, we further fine-tune YOLO-World for segmenting objects under the open-vocabulary setting, which can be termed open-vocabulary instance segmentation (OVIS). Previous methods [17] have explored OVIS with pseudo-labelling on novel objects. Differently, considering that YOLO-World has strong transfer and generalization capabilities, we directly fine-tune YOLO-World on a subset of data with mask annotations and evaluate the segmentation performance under large-vocabulary settings. Specifically, we benchmark open-vocabulary instance segmentation under two settings:

- (1) *COCO to LVIS* setting, we fine-tune YOLO-World on the COCO dataset (including 80 categories) with mask annotations, under which the models need to transfer from 80 categories to 1203 categories (80 → 1203);
- (2) *LVIS-base to LVIS* setting, we fine-tune YOLO-World on the LVIS-base (including 866 categories, common & frequent) with mask annotations, under which the models need to transfer from 866 categories to 1203 categories (866 → 1203).

We evaluate the fine-tuned models on the standard LVIS `val2017` with 1203 categories, in which 337 rare categories are unseen and can be used to measure the open-vocabulary performance.

**Results.** Tab. 8 shows the experimental results of extending YOLO-World for open-vocabulary instance segmentation. Specifically, we adopt two fine-tuning strategies: (1) only fine-tuning the segmentation head and (2) fine-tuning

all modules. Under strategy (1), the fine-tuned YOLO-World still retains the zero-shot capabilities acquired from the pre-training stage, allowing it to generalize to unseen categories without additional fine-tuning. Strategy (2) enables YOLO-World fit the LVIS dataset better, but it may result in the degradation of the zero-shot capabilities.

Tab. 8 shows the comparisons of fine-tuning YOLO-World with different settings (COCO or LVIS-base) and different strategies (fine-tuning seg. head or fine-tuning all). Firstly, fine-tuning on LVIS-base obtains better performance compared to that based on COCO. However, the ratios between AP and  $AP_r$  ( $AP_r/AP$ ) are nearly unchanged, *e.g.*, the ratios of YOLO-World on COCO and LVIS-base are 76.5% and 74.3%, respectively. Considering that the detector is frozen, we attribute the performance gap to the fact that the LVIS dataset provides more detailed and denser segmentation annotations, which are beneficial for learning the segmentation head. When fine-tuning all modules, YOLO-World obtains remarkable improvements on LVIS, *e.g.*, YOLO-World-L achieves 9.6 AP gain. However, the fine-tuning might degrade the open-vocabulary performance and lead to a 0.6 box  $AP_r$  drop for YOLO-World-L.

#### 4.6. Visualizations

We provide the visualization results of pre-trained YOLO-World-L under three settings: (a) we perform zero-shot inference with LVIS categories; (b) we input the custom prompts with fine-grained categories with attributes; (c) referring detection. The visualizations also demonstrate that YOLO-World has a strong generalization ability for open-vocabulary scenarios along with referring ability.

**Zero-shot Inference on LVIS.** Fig. 5 shows the visualization results based on the LVIS categories which are generated by the pre-trained YOLO-World-L in a zero-shot manner. The pre-trained YOLO-World exhibits strong zero-shot transfer capabilities and is able to detect as many objects as possible within the image.

**Inference with User’s Vocabulary.** In Fig. 6, we explore the detection capabilities of YOLO-World with our defined categories. The visualization results demonstrate that the pre-trained YOLO-World-L also exhibits the capability for (1) fine-grained detection (*i.e.*, detect the parts of one object) and (2) fine-grained classification (*i.e.*, distinguish different sub-categories of objects.).

**Referring Object Detection.** In Fig. 7, we leverage some descriptive (discriminative) noun phrases as input, *e.g.*, the standing person, to explore whether the model can locate regions or objects in the image that match our given input. The visualization results display the phrases and their

corresponding bounding boxes, demonstrating that the pre-trained YOLO-World has the referring or grounding capability. This ability can be attributed to the proposed pre-training strategy with large-scale training data.

## 5. Conclusion

We present YOLO-World, a cutting-edge real-time open-vocabulary detector aiming to improve efficiency and open-vocabulary capability in real-world applications. In this paper, we have reshaped the prevalent YOLOs as a vision-language YOLO architecture for open-vocabulary pre-training and detection and proposed RepVL-PAN, which connects vision and language information with the network and can be re-parameterized for efficient deployment. We further present the effective pre-training schemes with detection, grounding and image-text data to endow YOLO-World with a strong capability for open-vocabulary detection. Experiments can demonstrate the superiority of YOLO-World in terms of speed and open-vocabulary performance and indicate the effectiveness of vision-language pre-training on small models, which is insightful for future research. We hope YOLO-World can serve as a new benchmark for addressing real-world open-vocabulary detection.

## References

- [1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020. [2](#)
- [2] Kai Chen, Jiaqi Wang, Jiangmiao Pang, Yuhang Cao, Yu Xiong, Xiaoxiao Li, Shuyang Sun, Wansen Feng, Ziwei Liu, Jiarui Xu, Zheng Zhang, Dazhi Cheng, Chenchen Zhu, Tian-heng Cheng, Qijie Zhao, Buyu Li, Xin Lu, Rui Zhu, Yue Wu, Jifeng Dai, Jingdong Wang, Jianping Shi, Wanli Ouyang, Chen Change Loy, and Dahua Lin. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019. [5](#)
- [3] MMYOLO Contributors. MMYOLO: OpenMMLab YOLO series toolbox and benchmark. <https://github.com/open-mmlab/mmyolo>, 2022. [5, 8](#)
- [4] Achal Dave, Piotr Dollár, Deva Ramanan, Alexander Kirillov, and Ross B. Girshick. Evaluating large-vocabulary object detectors: The devil is in the details. *CoRR*, abs/2102.01066, 2021. [6, 7](#)
- [5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, pages 4171–4186, 2019. [1, 4, 6](#)
- [6] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *CVPR*, pages 13733–13742, 2021. [2](#)
- [7] Yu Du, Fangyun Wei, Zihe Zhang, Miaojing Shi, Yue Gao, and Guoqi Li. Learning to prompt for open-vocabulary ob-

| Model        | Fine-tune Data | Fine-tune Modules | AP          | $AP_r$      | $AP_c$      | $AP_f$      | $AP^b$      | $AP_r^b$    |
|--------------|----------------|-------------------|-------------|-------------|-------------|-------------|-------------|-------------|
| YOLO-World-M | COCO           | <i>Seg Head</i>   | 12.3        | 9.1         | 10.9        | 14.6        | 22.3        | 16.2        |
| YOLO-World-L | COCO           | <i>Seg Head</i>   | 16.2        | 12.4        | 15.0        | 19.2        | 25.3        | <b>18.0</b> |
| YOLO-World-M | LVIS-base      | <i>Seg Head</i>   | 16.7        | 12.6        | 14.6        | 20.8        | 22.3        | 16.2        |
| YOLO-World-L | LVIS-base      | <i>Seg Head</i>   | 19.1        | 14.2        | 17.2        | 23.5        | 25.3        | <b>18.0</b> |
| YOLO-World-M | LVIS-base      | <i>All</i>        | 25.9        | 13.4        | 24.9        | 32.6        | 32.6        | 15.8        |
| YOLO-World-L | LVIS-base      | <i>All</i>        | <b>28.7</b> | <b>15.0</b> | <b>28.3</b> | <b>35.2</b> | <b>36.2</b> | 17.4        |

Table 8. **Open-Vocabulary Instance Segmentation.** We evaluate YOLO-World for open-vocabulary instance segmentation under the two settings. We fine-tune the segmentation head or all modules of YOLO-World and report *Mask AP* for comparison.  $AP^b$  denotes the box AP.

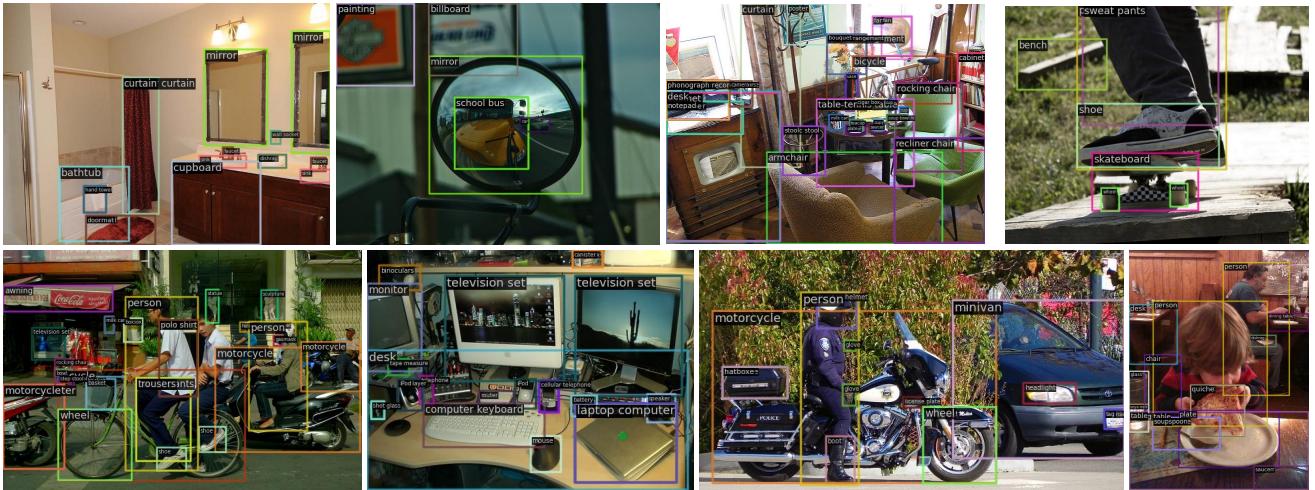


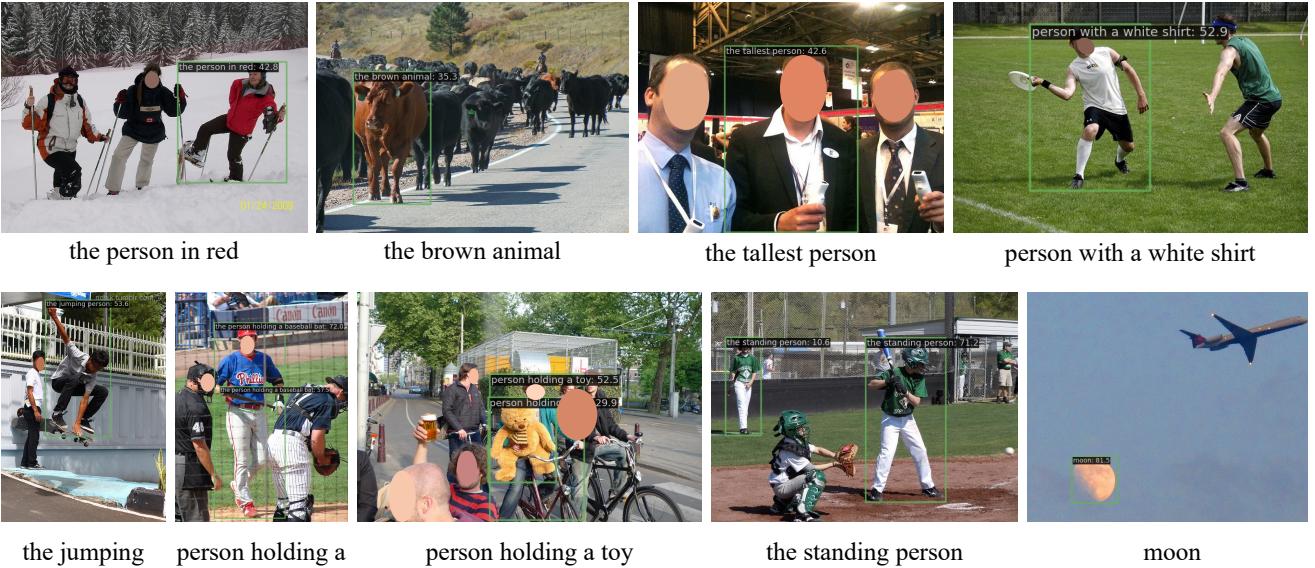
Figure 5. **Visualization Results on Zero-shot Inference on LVIS.** We adopt the pre-trained YOLO-World-L and infer with the LVIS vocabulary (containing 1203 categories) on the COCO val2017.

- ject detection with vision-language model. In *CVPR*, pages 14064–14073, 2022. [1](#), [3](#), [7](#), [8](#)
- [8] Chengjian Feng, Yujie Zhong, Yu Gao, Matthew R. Scott, and Weilin Huang. TOOD: task-aligned one-stage object detection. In *ICCV*, pages 3490–3499. IEEE, 2021. [5](#)
- [9] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. YOLOX: exceeding YOLO series in 2021. *CoRR*, abs/2107.08430, 2021. [2](#)
- [10] Ross B. Girshick. Fast R-CNN. In *ICCV*, pages 1440–1448, 2015. [2](#)
- [11] Ross B. Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. [2](#)
- [12] Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. In *ICLR*, 2022. [1](#), [2](#), [7](#), [8](#)
- [13] Agrim Gupta, Piotr Dollár, and Ross B. Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, pages 5356–5364, 2019. [6](#)
- [14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. [7](#)
- [15] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross B. Girshick. Mask R-CNN. In *ICCV*, pages 2980–2988, 2017. [1](#), [2](#)
- [16] Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*, pages 6700–6709, 2019. [6](#)
- [17] Dat Huynh, Jason Kuen, Zhe Lin, Jiuxiang Gu, and Ehsan Elhamifar. Open-vocabulary instance segmentation via robust cross-modal pseudo-labeling. In *CVPR*, pages 7010–7021, 2022. [8](#)
- [18] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc V. Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. Scaling up visual and vision-language representation learning with noisy text supervision. In *ICML*, pages 4904–4916, 2021. [1](#), [3](#)
- [19] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Ultralytics yolov8. <https://github.com/ultralytics/ultralytics>, 2023. [2](#), [3](#), [4](#), [5](#), [6](#), [7](#), [8](#)
- [20] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel



{men, women, boy, girl} {elephant, ear, leg, trunk, ivory} {golden dog, black dog, spotted dog} {grass, sky, zebra, trunk, tree}

**Figure 6. Visualization Results on User’s Vocabulary.** We define the custom vocabulary for each input image and YOLO-World can detect the accurate regions according to the vocabulary. Images are obtained from COCO val2017.



**Figure 7. Visualization Results on Referring Object Detection.** We explore the capability of the pre-trained YOLO-World to detect objects with descriptive noun phrases. Images are obtained from COCO val2017.

- Synnaeve, Ishan Misra, and Nicolas Carion. MDETR - modulated detection for end-to-end multi-modal understanding. In *ICCV*, pages 1760–1770, 2021. 6, 7
- [21] Weicheng Kuo, Yin Cui, Xiuye Gu, A. J. Piergiovanni, and Anelia Angelova. F-VLM: open-vocabulary object detection upon frozen vision and language models. *CoRR*, abs/2209.15639, 2022. 3, 8
- [22] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. Yolov6: A single-stage object detection framework for industrial applications. *CoRR*, abs/2209.02976, 2022. 2, 7, 8
- [23] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded language-image pre-training. In *CVPR*, pages 10955–10965, 2022. 1, 2, 3, 5, 6, 7, 13

- [24] Chuang Lin, Peize Sun, Yi Jiang, Ping Luo, Lizhen Qu, Ghollamreza Haffari, Zehuan Yuan, and Jianfei Cai. Learning object-language alignments for open-vocabulary object detection. In *ICLR*, 2023. 3
- [25] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755, 2014. 1, 2, 3, 13
- [26] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, pages 936–944, 2017. 1, 2
- [27] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2999–3007, 2017. 2
- [28] Shu Liu, Lu Qi, Haifang Qin, Jianping Shi, and Jiaya Jia. Path aggregation network for instance segmentation. In *CVPR*, pages 8759–8768, 2018. 2, 4

- [29] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: marrying DINO with grounded pre-training for open-set object detection. *CoRR*, abs/2303.05499, 2023. [1](#), [2](#), [3](#), [6](#), [7](#)
- [30] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *ECCV*, pages 21–37, 2016. [2](#)
- [31] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 9992–10002, 2021. [2](#), [3](#), [6](#), [7](#)
- [32] Xiang Long, Kaipeng Deng, Guanzhong Wang, Yang Zhang, Qingqing Dang, Yuan Gao, Hui Shen, Jianguo Ren, Shumin Han, Errui Ding, and Shilei Wen. PP-YOLO: an effective and efficient implementation of object detector. *CoRR*, abs/2007.12099, 2020. [2](#)
- [33] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. [5](#)
- [34] Vicente Ordonez, Girish Kulkarni, and Tamara L. Berg. Im2text: Describing images using 1 million captioned photographs. In *NeurIPS*, pages 1143–1151, 2011. [6](#)
- [35] Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. *Int. J. Comput. Vis.*, pages 74–93, 2017. [6](#)
- [36] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763, 2021. [1](#), [2](#), [3](#), [4](#), [5](#), [6](#), [13](#)
- [37] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *CVPR*, pages 6517–6525, 2017. [2](#)
- [38] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. [2](#)
- [39] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. [2](#), [3](#)
- [40] Joseph Redmon, Santosh Kumar Divvala, Ross B. Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pages 779–788, 2016. [1](#), [2](#), [4](#)
- [41] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1137–1149, 2017. [2](#)
- [42] Shaoqing Ren, Kaiming He, Ross B. Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1137–1149, 2017. [1](#)
- [43] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *ICCV*, pages 8429–8438, 2019. [2](#), [6](#)
- [44] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *ACL*, pages 2556–2565, 2018. [5](#), [6](#), [13](#)
- [45] Cheng Shi and Sibei Yang. Edadet: Open-vocabulary object detection using early dense alignment. In *ICCV*, pages 15678–15688, 2023. [1](#)
- [46] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: fully convolutional one-stage object detection. In *ICCV*, pages 9626–9635, 2019. [2](#)
- [47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017. [2](#)
- [48] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. CspNet: A new backbone that can enhance learning capability of CNN. In *CVPRW*, pages 1571–1580, 2020. [2](#)
- [49] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *CVPR*, pages 7464–7475, 2023. [2](#), [7](#), [8](#)
- [50] Size Wu, Wenwei Zhang, Sheng Jin, Wentao Liu, and Chen Change Loy. Aligning bag of regions for open-vocabulary object detection. In *CVPR*, pages 15254–15264, 2023. [1](#), [3](#), [8](#)
- [51] Johnathan Xie and Shuai Zheng. ZSD-YOLO: zero-shot YOLO detection using vision-language knowledge distillation. *CoRR*, 2021. [3](#)
- [52] Shangliang Xu, Xinxin Wang, Wenyu Lv, Qinyao Chang, Cheng Cui, Kaipeng Deng, Guanzhong Wang, Qingqing Dang, Shengyu Wei, Yuning Du, and Baohua Lai. PP-YOLOE: an evolved version of YOLO. *CoRR*, abs/2203.16250, 2022. [2](#)
- [53] Lewei Yao, Jianhua Han, Youpeng Wen, Xiaodan Liang, Dan Xu, Wei Zhang, Zhenguo Li, Chunjing Xu, and Hang Xu. Detclip: Dictionary-enriched visual-concept paralleled pre-training for open-world detection. In *NeurIPS*, 2022. [1](#), [2](#), [3](#), [6](#)
- [54] Lewei Yao, Jianhua Han, Xiaodan Liang, Dan Xu, Wei Zhang, Zhenguo Li, and Hang Xu. Detclipv2: Scalable open-vocabulary object detection pre-training via word-region alignment. In *CVPR*, pages 23497–23506, 2023. [1](#), [3](#), [6](#)
- [55] Alireza Zareian, Kevin Dela Rosa, Derek Hao Hu, and Shih-Fu Chang. Open-vocabulary object detection using captions. In *CVPR*, pages 14393–14402, 2021. [1](#), [2](#), [3](#)
- [56] Haotian Zhang, Pengchuan Zhang, Xiaowei Hu, Yen-Chun Chen, Liunian Harold Li, Xiyang Dai, Lijuan Wang, Lu Yuan, Jenq-Neng Hwang, and Jianfeng Gao. Glipv2: Unifying localization and vision-language understanding. In *NeurIPS*, 2022. [1](#), [2](#), [3](#), [6](#), [7](#)
- [57] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel M. Ni, and Heung-Yeung Shum. DINO: DETR with improved denoising anchor boxes for end-to-end object detection. In *ICLR*, 2023. [3](#)
- [58] Shifeng Zhang, Cheng Chi, Yongqiang Yao, Zhen Lei, and Stan Z. Li. Bridging the gap between anchor-based and

- anchor-free detection via adaptive training sample selection. In *CVPR*, pages 9756–9765, 2020. 2, 3
- [59] Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, and Jianfeng Gao. Regionclip: Region-based language-image pretraining. In *CVPR*, pages 16772–16782, 2022. 3, 8
- [60] Xingyi Zhou, Rohit Girdhar, Armand Joulin, Philipp Krähenbühl, and Ishan Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, pages 350–368, 2022. 3, 7, 8
- [61] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *ICLR*, 2021. 2

## A. Additional Details

### A.1. Re-parameterization for RepVL-PAN

During inference on an offline vocabulary, we adopt re-parameterization for RepVL-PAN for faster inference speed and deployment. Firstly, we pre-compute the text embeddings  $W \in \mathbb{R}^{C \times D}$  through the text encoder.

**Re-parameterize T-CSPLayer.** For each T-CSPLayer in RepVL-PAN, we can re-parameterize and simplify the process of adding text guidance by reshaping the text embeddings  $W \in \mathbb{R}^{C \times D \times 1 \times 1}$  into the weights of a  $1 \times 1$  convolution layer (or a linear layer), as follows:

$$X' = X \odot \text{Sigmoid}(\max(\text{Conv}(X, W), \text{dim}=1)), \quad (4)$$

where  $X \in \mathbb{R}^{B \times D \times H \times W}$  and  $X' \in \mathbb{R}^{B \times D \times H \times W}$  are the input and output image features.  $\odot$  is the matrix multiplication with reshape or transpose.

**Re-parameterize I-Pooling Attention.** The I-Pooling Attention can be re-parameterize or simplified by:

$$\tilde{X} = \text{cat}(\text{MP}(X_3, 3), \text{MP}(X_4, 3), \text{MP}(X_5, 3)), \quad (5)$$

where  $\text{cat}$  is the concentration and  $\text{MP}(\cdot, 3)$  denotes the max pooling for  $3 \times 3$  output features.  $\{X_3, X_4, X_5\}$  are the multi-scale features in RepVL-PAN.  $\tilde{X}$  is flattened and has the shape of  $B \times D \times 27$ . Then we can update the text embeddings by:

$$W' = W + \text{Softmax}(W \odot \tilde{X}), \text{dim}=-1 \odot W, \quad (6)$$

### A.2. Fine-tuning Details.

We remove all T-CSPLayers and Image-Pooling Attention in RepVL-PAN when transferring YOLO-World to COCO [25] object detection, which only contains 80 categories and has a relatively low dependency on visual-language interaction. During fine-tuning, we initialize

YOLO-World using pre-trained weights. The learning rate of fine-tuning is set to 0.0002 with the weight decay set to 0.05. After fine-tuning, we pre-compute the class text embeddings with given COCO categories and store the embeddings into the weights of the classification layers.

## B. Automatic Labeling on Large-scale Image-Text Data

In this section, we add details procedures for labeling region-text pairs with large-scale image-text data, *e.g.*, CC3M [44]. The overall labeling pipeline is illustrated in Fig. 8, which mainly consists of three procedures, *i.e.*, (1) extract object nouns, (2) pseudo labeling, and (3) filtering. As discussed in Sec. 3.4, we adopt the simple n-gram algorithm to extract nouns from captions.

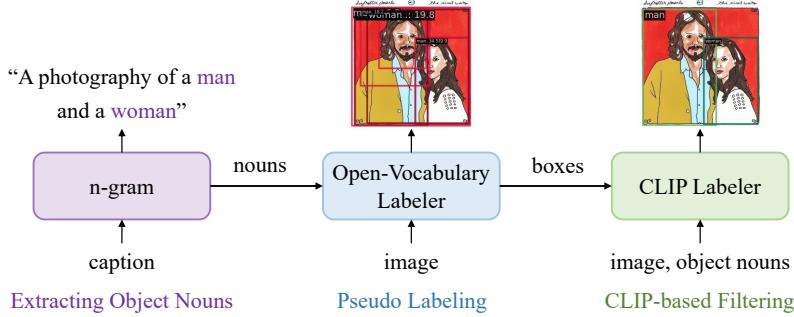
**Region-Text Proposals.** After obtaining the set of object nouns  $T = \{t_k\}^K$  from the first step, we leverage a pre-trained open-vocabulary detector, *i.e.*, GLIP-L [23], to generate pseudo boxes  $\{B_i\}$  along with confidence scores  $\{c_i\}$ :

$$\{B_i, t_i, c_i\}_{i=1}^N = \text{GLIP-Labeler}(I, T), \quad (7)$$

where  $\{B_i, t_i, c_i\}_{i=1}^N$  are the coarse region-text proposals.

**CLIP-based Re-scoring & Filtering.** Considering the region-text proposals containing much noise, we present a restoring and filtering pipeline with the pre-trained CLIP [36]. Given the input image  $I$ , caption  $T$ , and the coarse region-text proposals  $\{B_i, t_i, c_i\}_{i=1}^N$ , the specific pipeline is listed as follows:

- (1) Compute Image-Text Score: we forward the image  $I$  with its caption  $T$  into CLIP and obtain the image-text similarity score  $s^{img}$ .
- (2) Compute Region-Text Score: we crop the region images from the input image according to the region boxes  $\{B_i\}$ . Then we forward the cropped images along with their texts  $\{t_i\}$  into CLIP and obtain the region-text similarity  $S^r = \{s_i^r\}_{i=1}^N$ .
- (3) [Optional] Re-Labeling: we can forward each cropped image with all nouns and assign the noun with maximum similarity, which can help correct the texts wrongly labeled by GLIP.
- (4) Rescoring: we adopt the region-text similarity  $S^r$  to rescore the confidence scores  $\tilde{c}_i = \sqrt{c_i * s_i^r}$ .
- (5) Region-level Filtering: we first divide the region-text proposals into different groups according to the texts and then perform non-maximum suppression (NMS) to filter the duplicate predictions (the NMS threshold is set to 0.5). Then we filter out the proposals with low confidence scores (the threshold is set to 0.3).



**Figure 8. Labeling Pipeline for Image-Text Data** We first leverage the simple n-gram to extract object nouns from the captions. We adopt a pre-trained open-vocabulary detector to generate pseudo boxes given the object nouns, which forms the coarse region-text proposals. Then we use a pre-trained CLIP to rescore or relabel the boxes along with filtering.

- (6) **Image-level Filtering:** we compute the image-level region-text scores  $s^{region}$  by averaging the kept region-text scores. Then we obtain the image-level confidence score by  $s = \sqrt{s^{img} * s^{region}}$  and we keep the images with scores larger than 0.3.

The thresholds mentioned above are empirically set according to the part of labeled results and the whole pipeline is automatic without human verification. Finally, the labeled samples are used for pre-training YOLO-World. We will provide the pseudo annotations of CC3M for further research.

## C. Pre-training YOLO-World at Scale

When pre-training small models, *e.g.*, YOLO-World-S, a natural question we have is: how much capacity does a small model have, and how much training data or what kind of data does a small model need? To answer this question, we leverage different amounts of pseudo-labeled region-text pairs to pre-train YOLO-World. As shown in Tab. 9, adding more image-text samples can increase the zero-shot performance of YOLO-World-S. Tab. 9 indicates: (1) adding image-text data can improve the overall zero-shot performance of YOLO-World-S; (2) using an excessive amount of pseudo-labeled data may have some negative effects for small models (YOLO-World-S), though it can improve the on rare categories ( $AP_r$ ). However, using fine-grained annotations (GoldG) for small models can provide significant improvements, which indicates that large-scale high-quality annotated data can significantly enhance the capabilities of small models. And Tab. 3 in the main text has shown that pre-training with the combination of fine-annotated data and pseudo-annotated data can perform better. We will explore more about the data for pre-training small models or YOLO detectors in future work.

| Method       | Pre-trained Data | Samples | AP   | $AP_r$ | $AP_c$ | $AP_f$ |
|--------------|------------------|---------|------|--------|--------|--------|
| YOLO-World-S | O365             | 0.61M   | 16.3 | 9.2    | 14.1   | 20.1   |
| YOLO-World-S | O365+GoldG       | 1.38M   | 24.2 | 16.4   | 21.7   | 27.8   |
| YOLO-World-S | O365+CC3M-245k   | 0.85M   | 16.5 | 10.8   | 14.8   | 19.1   |
| YOLO-World-S | O365+CC3M-520k   | 1.13M   | 19.2 | 10.7   | 17.4   | 22.4   |
| YOLO-World-S | O365+CC3M-750k   | 1.36M   | 18.2 | 11.2   | 16.0   | 21.1   |

Table 9. **Zero-shot Evaluation on LVIS.** We evaluate the performance of pre-training YOLO-World-S with different amounts of data, the image-text data.