Streamlined Open-Vocabulary Human-Object Interaction Detection

Supplementary Material

A. Training Strategy

To ensure fair comparison with prior work, we follow the standard training protocols commonly adopted for each benchmark. As discussed in Sec. 5.2, SWiG-HOI [8] and HICO-DET [1] rely on distinct supervisory setups; therefore, rather than performing dataset-specific tuning, we adhere to the established training practices used in prior work.

Specifically, we follow THID [9] for SWiG-HOI and GEN-VLKT [5] for HICO-DET:

- **SWiG-HOI:** we adopt most of THID's default configuration. The loss weights for bounding-box L1, bounding-box GIoU, interaction classification, and confidence prediction are set to 5.0, 2.0, 5.0, and 10.0, respectively. Training is conducted for 100 epochs with a base learning rate of 1×10^{-4}, decayed by a factor of 10 at epochs 60 and 90. For data augmentation, we follow the commonly used small-scale multi-resolution strategy (224-320), which stabilizes contrastive HOI training by maintaining rich in-batch negatives. Random horizontal flipping, light color jittering, and interaction-aware random cropping are applied before resizing, followed by ImageNet [7] normalization.
- **HICO-DET:** we adopt GEN-VLKT's training scheme. In addition to interaction classification, an auxiliary object-classification loss is used, and thus the confidence loss is removed. A head-level semantic token is further introduced as an additional key-value input for the instance decoder, consistent with prior architectures. The loss weights for bounding-box L1, bounding-box GIoU, object classification, and interaction classification are set to 2.5, 1.0, 1.0, and 2.0. Training is performed for 60 epochs with a base learning rate of 1×10^{-4}, reduced by a factor of 10 at epoch 40. While GEN-VLKT employs a longer 60+30 schedule, our model converges reliably under a shorter 40+20 decay pattern. To ensure fair comparison to strong HICO-DET baselines, we adopt the large-scale multi-resolution augmentation (480-800), together with the same flipping and light color-jittering scheme as in SWiG-HOI. Interaction-aware cropping may optionally be applied to preserve human-object spatial structure.

Overall, supervision on SWiG-HOI is intrinsically stronger because it has larger label space and is more challenging, making it a more representative testbed for open-vocabulary HOI learning.

B. Clarification on Image Tokens

Image tokens in this paper contain three types: the `[CLS]` token, register tokens [2], and patch tokens, as introduced in Sec. 3. The `[CLS]` token captures global image-level semantics; register tokens serve only as auxiliary tokens during the forward pass and are not used by downstream heads; patch tokens encode local visual information and are the only tokens consumed by our instance detector.

During the semantic bootstrapping stage, we follow the input format of the vision head [4] while inserting interaction queries before the patch tokens. The token sequence is

$$\left[\texttt{CLS}, \mathrm{Reg}_1, \ldots, \mathrm{Reg}_4, Q_1, \ldots, Q_{N_q}, P_1, \ldots, P_N\right],$$

where Q_i denotes interaction queries and P_i denotes patch tokens from the backbone. In the hierarchical refinement stage, the image tokens used as keys and values in cross-attention are

$$\left[\texttt{CLS}', \overline{P'}, P'_1, \ldots, P'_N\right], \qquad \overline{P'} = \frac{1}{N}\sum_{i=1}^{N} P'_i,$$

where $'$ indicates tokens output by the vision head. Two considerations motivate this design: (1) register tokens are strictly auxiliary and are thus excluded from downstream prediction heads; (2) text embeddings are aligned with both the `[CLS]` token and the mean patch token, reinforcing semantic consistency between text features and global/region-aggregated visual representations.

C. Text-Encoder-Based Similarity Classifier

For each interaction category $\mathbf{r}_j$, we construct a descriptive sentence "a photo of a person <action+ing> a/an <object>". For example, "ride horse" is expressed as "a photo of a person riding a horse". This textual description is encoded using the `dino.txt` [4] text encoder, producing a $2D = 2048$-dimensional embedding $\mathbf{e}_t^{(j)}$. The first $D = 1024$ dimensions correspond to the `[CLS]` token, and the remaining $D = 1024$ are aligend with the mean-pooled patch tokens. To match the dimension, we project the $D = 1024$ interaction decoder embeddings $\mathbf{e}_r^{(i)}$ into the same $2D = 2048$-dimensional space, enabling direct similarity computation. Because the first and second D dimensions of the text embedding capture different semantic levels, we also explored computing similarities separately and combining them via a weighted sum. However, this alternative produced slightly inferior performance compared to the unified projection.

D. Variant Models

The architectures of the variant models are presented in Fig. A and Fig. B.

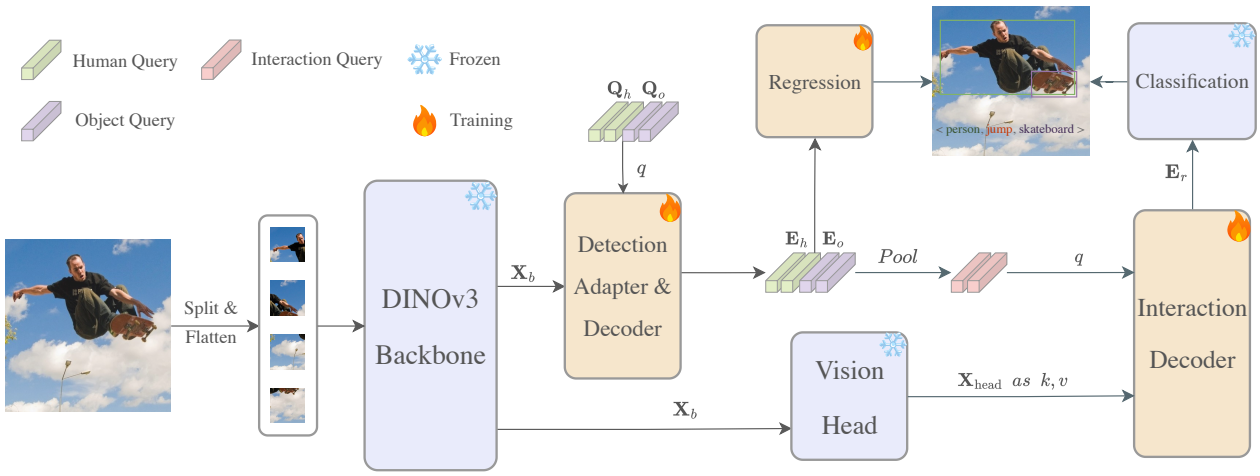


Figure A. Overall architecture of our baseline framework.

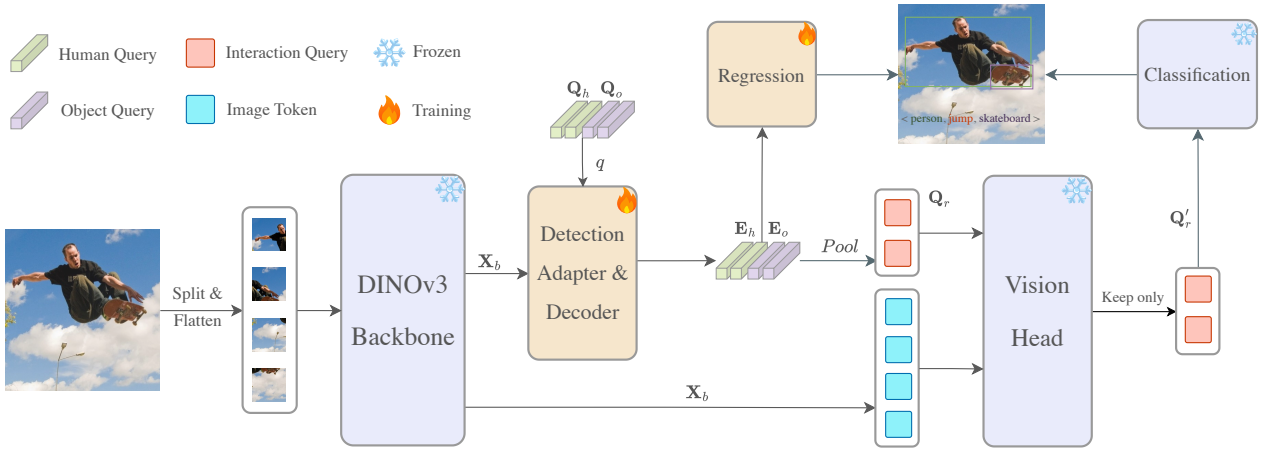


Figure B. Overall architecture of our Semantic Bootstrapping (only) framework.

D.1. Baseline Model

As illustrated in Fig. A, the interaction stage of our baseline model follows the overall structure of HOICLIP [6], employing a late fusion strategy for interaction classification. Semantic cues from the VLM are injected into the interaction queries through cross-attention. Contrary to HOICLIP, we do not rely on backbone features to assist interaction detection, as validated by the ablation results in Tab. 5 for "Late Fusion (Head only)" and "Late Fusion (Multi-Scale)".

Due to the high dimensionality of token representations, the cross-attention output is first projected down to $d = 256$. After cross-attention, the resulting interaction embeddings are then projected to match the dimension of the text embedding space.

D.2. Semantic Bootstrapping Model

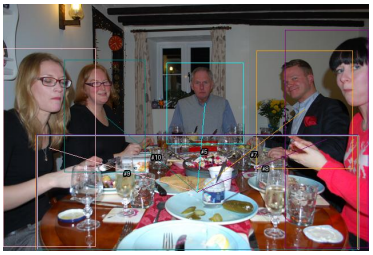
The Semantic Bootstrapping Model is conceptually simple. Compared to the final model, it removes the cross-attention block, as shown in Fig. B. Only the interaction-query outputs from the vision head are retained. Our experiments indicate that these tokens alone are already sufficient for interaction classification and exhibit stronger representational quality than the baseline model. As demonstrated in the ablation study (Tab. 4), this design yields consistent and substantial improvements across all metrics.

E. Failure Cases and Analysis

We provide a failure-case analysis to clarify where the current model still struggles.

Typical failure categories. We observe two representative failure scenarios: crowded scenes and small-object detection.

- **Crowded scenes:** multiple overlapping human–object instances increase assignment ambiguity and can cause missed detections.
- **Small-object detection:** very small targets are sensitive to slight spatial offsets, leading to localization errors in



(a) Crowded scene

(b) Small-object detection

Figure C. Representative failure cases. Left: crowded scene where the detected interactions mainly include *sitting at* and *eating at* a dining table. Right: small-object detection where the detected interactions mainly include *wearing, standing on, holding* a snowboard, and *wearing, carrying, standing on, holding, riding* a skis.

human or object boxes.

Error patterns. In crowded scenes (e.g., Fig. C(a)), while the model successfully detects primary interactions such as *sitting at* and *eating at* a dining table, it is generally robust but still shows occasional misses. For example, a small fork may not be detected even when the main interaction is correctly recognized. In small-object cases (e.g., Fig. C(b)), the model identifies complex interaction sets like *wearing, standing on,* and *holding* a snowboard/skis, but both human and object boxes can drift from the true targets. We attribute this mainly to spatial information compression in ViT downsampling, where subtle local offsets of tiny objects become less distinguishable, increasing the failure risk for precise small-object localization.

F. Training Strategy Comparison

We compare several training recipes under the same evaluation protocol on SWiG-HOI (mAP %). Beyond the default frozen strategy, we additionally evaluate partial fine-tuning and parameter-efficient adaptation (LoRA [3]).

Template for training recipes. For reproducibility, we summarize the practical configurations used in our comparison:

- **Frozen strategy (default):** keep the vision backbone and vision head frozen; train detection adapter, instance decoder, and interaction modules with base learning rate 1×10^{-4}.
- **Partial fine-tuning:** unfreeze the vision head and use a reduced learning rate of 2×10^{-5} (i.e., $1/5$ of the base learning rate) to preserve pre-trained semantics during adaptation.
- **LoRA strategy:** apply LoRA on the vision head attention *qkv* input projection and output projection layers, with rank $r = 16$ and scaling factor $\alpha = 32$; non-LoRA backbone parameters remain frozen.

As shown in Tab. A, additional training complexity does not necessarily yield meaningful gains; a simple strategy is already effective in practice.

Table A. Comparison of training recipes on the SWiG-HOI dataset (mAP %).

Training Recipe	Unseen	Rare	Seen	Full
LoRA fine-tune (all layers)	18.99	24.40	29.91	24.34
Partial fine-tune (last layer)	18.63	24.37	30.01	24.26
Ours (frozen)	19.04	24.69	30.62	24.67

G. Pseudo-Code

To illustrate the workflow of our method, we provide a simplified pseudo-code example in Fig. D, covering the core computations corresponding to the main equations in Sec. 4.

References

[1] Yu-Wei Chao, Yunfan Liu, Xieyang Liu, Huayi Zeng, and Jia Deng. Learning to detect human-object interactions. In *WACV*, 2018. 1

[2] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *ICLR*, 2024. 1

[3] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *ICLR*, 2022. 3

[4] Cijo Jose, Théo Moutakanni, Dahyun Kang, Federico Baldassarre, Timothée Darcet, Hu Xu, Daniel Li, Marc Szafraniec, Michaël Ramamonjisoa, Maxime Oquab, Oriane Siméoni, Huy V. Vo, Patrick Labatut, and Piotr Bojanowski. Dinov2 meets text: A unified framework for image- and pixel-level vision-language alignment. In *CVPR*, 2025. 1

[5] Yue Liao, Aixi Zhang, Miao Lu, Yongliang Wang, Xiaobo Li, and Si Liu. GEN-VLKT: simplify association and enhance interaction understanding for HOI detection. In *CVPR*, 2022. 1

[6] Shan Ning, Longtian Qiu, Yongfei Liu, and Xuming He. HOICLIP: efficient knowledge transfer for HOI detection with vision-language models. In *CVPR*, 2023. 2

[7] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 1

[8] Suchen Wang, Kim-Hui Yap, Henghui Ding, Jiyan Wu, Junsong Yuan, and Yap-Peng Tan. Discovering human interactions with large-vocabulary objects via query and multi-scale detection. In *ICCV*, 2021. 1

[9] Suchen Wang, Yueqi Duan, Henghui Ding, Yap-Peng Tan, Kim-Hui Yap, and Junsong Yuan. Learning transferable human-object interaction detector with natural language supervision. In *CVPR*, 2022. 1

```
import torch.nn as nn
import torch.nn.functional as F

class SL_HOI(nn.Module):
    def __init__(self, backbone, head_model, det_encoder, det_decoder, fusion_decoder):
        super().__init__()
        # Foundation models (parameters are frozen)
        self.backbone = backbone
        self.head_model = head_model

        # Learnable modules
        self.det_encoder = det_encoder
        self.det_decoder = det_decoder
        self.fusion_decoder = fusion_decoder
        self.query_h = nn.Embedding(N_q, d) # Human queries
        self.query_o = nn.Embedding(N_q, d) # Object queries

        # Projection layers for dimension matching
        self.input_proj = nn.Conv2d(D, d, kernel_size=1)
        self.query_proj = nn.Linear(d, D)
        self.cls_proj = nn.Linear(D, D_text)
        self.logit_scale = nn.Parameter(torch.tensor(2.6592))

        # Freeze pre-trained model parameters
        self.freeze_models()

    def freeze_models(self):
        for param in self.backbone.parameters():
            param.requires_grad = False
        for param in self.head_model.parameters():
            param.requires_grad = False

    def forward(self, image, text_features):
        # 1. Feature extraction from frozen backbone
        # and Interactive instance detection (Eq. 1 & Eq. 2)
        with torch.no_grad():
            X_b = self.backbone(image)
        F = self.det_encoder(self.input_proj(X_b))
        E_h, E_o = self.det_decoder(self.query_h.weight, self.query_o.weight, F)

        # 2. Semantic Bootstrapping (Eq. 4 & Eq. 5)
        Q_r = self.query_proj((E_h + E_o) / 2)
        Q_r_prime, X_head = self.head_model(Q_r, X_b)

        # 3. Hierarchical Refinement(Eq. 6)
        E_r = self.fusion_decoder(query=Q_r_prime, key_value=X_head)

        # 4. Open-vocabulary predictions (Eq. 7)
        E_r_proj = F.normalize(self.cls_proj(E_r), dim=-1)
        text_features = F.normalize(text_features, dim=-1)
        logits = self.logit_scale.exp() * E_r_proj @ text_features.t()

        return logits
```

Figure D. An oversimplified PyTorch-style pseudocode of our SL-HOI framework. It illustrates the core logic, including the two-step interaction refinement and the prediction of interaction logits, while omitting many implementation details for clarity.