Unlocking ImageNet's Multi-Object Nature: Automated Large-Scale Multilabel Annotation

Supplementary Material

We organize our supplementary material as follows:

- Supp. A details our object proposal generation pipeline, including MaskCut implementation, hyperparameter tuning, and comparison to SAM.
- Supp. B outlines the labeler training setup, relabeling statistics, and analysis of different label aggregation strategies.
- Supp. C provides experimental protocols for ImageNet training and transfer learning across architectures and input sizes.
- Supp. D presents a qualitative comparison between our labels and human-curated ReaL annotations.
- Supp. E details ambiguous class definitions in ImageNet and proposes two strategies to mitigate label noise.
- Supp. F contains additional analyses, including resolution robustness, feature entropy, soft-label training, mask filtering for CutLER, and our interactive annotation tool.

A. Object Proposal Generation

A.1. MaskCut Implementation Details

This section details MaskCut [34], which we use to extract object proposals from an image using a self-supervised vision transformer (ViT). The goal is to discover multiple salient object regions per image $I \in \mathbb{R}^{3\times h\times w}$ without any manual labels. MaskCut builds on the idea of iterative normalized cuts, generating multiple binary masks that segment distinct object-like regions. Let $\mathcal{F}$ be a pretrained ViT encoder. Given an input image I, we first extract its patch-level feature map $F = \mathcal{F}(I) \in \mathbb{R}^{h'\times w'\times d}$, which is flattened into $N = h' \times w'$ feature vectors $f_1, f_2, \ldots, f_N$ of dimension d. We then construct a fully connected graph over the N patches, where the affinity between two nodes i and j is defined by the cosine similarity of their features:

$$W_{ij} = \frac{f_i \cdot f_j}{\|f_i\|_2 \, \|f_j\|_2} \tag{1}$$

Normalized Cuts (NCut) is applied to partition the graph into foreground vs. background. NCut finds an eigenvector $x \in \mathbb{R}^N$ (the relaxed indicator of the cut) by solving the generalized eigenvalue problem

$$(D - W)x = \lambda D x \tag{2}$$

and we take the eigenvector x corresponding to the second-smallest eigenvalue λ (standard practice for NCut). We then threshold x to produce an initial binary mask M for the foreground object:

$$M(i) = \begin{cases} 1, & x(i) \geq \mu(x), \\ 0, & x(i) < \mu(x), \end{cases} \tag{3}$$

where $\mu(x)$ is the mean value of x. Here $M(i) = 1$ indicates patch i is classified as foreground. We determine which side of the cut is the object (foreground) using two criteria from MaskCut [34]: (a) the foreground mask should contain the patch corresponding to the largest magnitude in x (since the principal object tends to dominate the eigenvector), and (b) the foreground should not include more than one or two image corner patches (to avoid selecting the entire background as object). If our initial mask M does not satisfy these criteria (e.g. the largest-eigenvector patch is not in M, or M covers too many corners), we flip the assignment (set $M \leftarrow 1 - M$). This ensures M corresponds to a salient object in the image. Following MaskCut, we also threshold the affinity matrix by a hyperparameter τ^{ncut} to sharpen the segmentation by setting all $W_{ij} < \tau^{\text{ncut}}$ to $1e^{-5}$ and $W_{ij} \geq \tau^{\text{ncut}}$ to 1. A Conditional Random Field (CRF [29]) post-processing step is then applied to the patch mask to incorporate low-level pixel continuity, yielding a refined pseudo segmentation mask for the object. To discover multiple objects, we iteratively repeat the NCut process on the remaining image regions. After obtaining the first object mask P_1 (the set of patches with $M(i) = 1$), we mask out those patches by removing them from the graph. In practice, we set their feature vectors to zero or exclude them from further similarity computations. Formally, let $U_1 = P_1$ be the set of foreground patches found. For the next iteration, we update the patch similarities so that any patch $i \in U_1$ (or patch $j \in U_1$) has no affinity:

$$W^{(2)}_{ij} = \begin{cases} \dfrac{f_i \cdot f_j}{\|f_i\|_2 \, \|f_j\|_2}, & i \notin U_1 \text{ and } j \notin U_1, \\ 0, & \text{otherwise.} \end{cases} \tag{4}$$

We then rerun NCut on this updated matrix $W^{(2)}$ to find the next object mask P_2. We continue this masked NCut procedure for $t = 1, 2, \ldots, N$, each time excluding all patches from previously found masks $U_t = \bigcup_{s=1}^{t} P_s$. This yields up to N object masks $P_1, P_2, \ldots, P_N$ per image. We set the maximum iteration N as a hyperparameter, which is the maximum number of object proposals produced for one image. The process stops early if an iteration returns no significant foreground (e.g. only background remains).

A.2. Hyperparameter Selection

The performance of MaskCut depends on several key hyperparameters: the affinity threshold τ^{ncut}, the number of object proposals N, the choice of ViT backbone, and the use of CRF post-processing. A high τ^{ncut} may over-merge nearby objects (e.g., grouping multiple dogs), while a low value risks over-segmentation. Similarly, a small N may miss valid objects, whereas a large N can introduce spurious proposals. We explore various self-supervised ViT backbones: DINOv1 [6] (as used in the original MaskCut), DINOv2 [19] (ViT-S/B/L/G), and DINOv3 [27] (ViT-S+/B/L/H). For each, we sweep across:

- Input image resolutions (corresponding to patch grids of 24×24, 32×32, or 48×48),
- Feature type (last-layer patch features or last attention query/key/value),
- $\tau^{\text{ncut}} \in [0.1, 0.9]$ in steps of 0.1, followed by fine sweeps of ± 0.05 in 0.01 increments,
- Number of proposals $N \in \{3, 4, 5\}$,
- CRF post-processing (enabled or disabled).

To evaluate, we compute object recall at IoU ≥ 0.5 on 200 randomly sampled images from ImageNet-Segments [11]. For DINOv1, we adopt the best configuration from MaskCut. For DINOv2 and DINOv3, we rank the top 7 configurations by recall. We then further manually assess visual quality on a 3-point scale: (1) noisy/missing masks, (2) good with some issues, (3) all good masks. The top 4 configurations with complementary strengths are selected for the final pipeline:

- `DINOv3-B/patch_16/feat_v`, input 768, $\tau = 0.35$, $N = 4$, CRF off
- `DINOv2-G/patch_14/feat_v`, input 672, $\tau = 0.12$, $N = 3$, CRF on
- `DINOv2-L/patch_14/feat_v`, input 448, $\tau = 0.12$, $N = 3$, CRF on
- `DINOv1-B/patch_8/feat_k`, input 480, $\tau = 0.15$, $N = 3$, CRF on

We find that no single configuration is optimal for all images; using a diverse ensemble of top-performing setups improves robustness and overall coverage.

A.3. Comparison with SAM

We evaluated Segment Anything (SAMv2 [21]) as a potential object proposal generator by testing different automatic prompt configurations. As shown in Fig. 4, SAM can produce detailed masks in some cases (e.g., config.2) but struggles with consistency—often under-segmenting, over-segmenting, or missing objects entirely in others. This variability across images makes it difficult to identify a fixed set of configurations that work reliably across the diverse ImageNet distribution. In contrast, MaskCut offers more stable and interpretable object-level proposals with consistent hyperparameters, making it better suited for our large-scale relabeling pipeline.

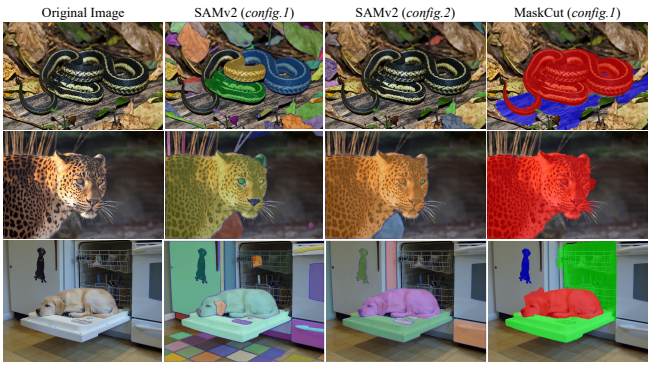

Figure 4. Comparison of mask proposals from SAMv2 [21] (two configurations) and MaskCut [34]. While SAM can produce fine-grained masks under certain settings, it often over-segments or misses key objects depending on the image, highlighting its sensitivity to hyperparameters. In contrast, MaskCut generates more consistent, instance-level masks across diverse images using a fixed configuration, making it more suitable for large-scale object proposal generation.

B. Labeler Training Details

B.1. Training Setup and Hyperparameters

We filter object proposals by requiring the ReLabel confidence on the image's original label to exceed a threshold of $\tau_{\text{sel}} = 0.75$. The classification head is trained on input images resized to 512×512 resolution for 300 epochs, using SGD with Nesterov momentum (0.9), weight decay of $1e^{-4}$, and learning rate of 0.1. We adopt a cosine learning rate schedule with 5 epochs of linear warm-up. The backbone (`DINOv3 ViT-L/16`) remains frozen throughout training. The global batch size is 512. We apply standard data augmentations including RandomResizedCrop, horizontal flip, and RandAugment [7]. All geometric augmentations are applied consistently to both the original image and its associated object proposal to preserve spatial alignment. To improve robustness, we introduce patch-level dropout: for each proposal mask P, we randomly drop 25% of active (i.e., foreground) patches prior to feature pooling. For reference, the trained labeler achieves a top-1 accuracy of 84.73% on IN-Val and 88.61% on ReaL. Note that this differs from the relabeling stage, where predictions are made by pooling over localized object regions rather than the entire patch map. These values are provided for completeness.

B.2. Train-Set Relabeling Statistics

To better illustrate the label density in our relabeled ImageNet-1K training set, we apply a confidence threshold of $\tau = 0.5$ to the softmax outputs during inference to

Table 5. Distribution of the number of unique labels ($\mathbf{k}$) predicted per image in the relabeled ImageNet training set, using a softmax confidence threshold of $\tau = 0.5$. The last column (Avg.) reports the average number of labels per image across the full training set.

$\mathbf{k=0}$	$\mathbf{k=1}$	$\mathbf{k=2}$	$\mathbf{k=3}$	$\mathbf{k\geq 4}$	**Avg.**
0.59%	78.9%	17.8%	2.5%	0.28%	1.23

filter out low-confidence predictions. Note that this filtering is not used during training with soft labels. Table 5 summarizes the distribution of predicted labels per image. Notably, 20.5% of images have two or more labels, underscoring the multi-object nature of the dataset and the limitations of its original single-label annotations.

B.3. Label Aggregation: Hard vs. Soft, Local vs. Global

We conduct a comprehensive evaluation of label aggregation strategies by training a ResNet-50 for 100 epochs on ImageNet-1K using binary cross-entropy (BCE) loss. For a fair comparison with the original single label (i.e., converted to one-hot label) setup, we first sweep key hyperparameters and stabilization techniques known to affect BCE training [8, 24, 36]. These include:

- Positive label weighting (pos_weight)
- Label smoothing which uses a relaxed (min, max) range instead of strict binary targets
- Head initialization to avoid early overconfidence (sigmoid outputs initialized near the smoothing minimum)
- Excluding the classification head from weight decay
- Optimizer: SGD vs. AdamW, with variations in learning rate and weight decay (WD)

The best configuration we identify is: *AdamW optimizer* with a learning rate of 0.001 and weight decay of 0.15 (WD=0 for the classifier head), a *label smoothing range of* $(0.0001, 0.95)$, and a *global batch size of* 1024. Training uses cosine decay with 5-epoch linear warmup, and standard augmentations of RandomResizedCrop, horizontal flip, and RandAugment. Under this setup, BCE training with the original single-label supervision slightly outperforms the best cross-entropy configuration we found (IN-Val: 77.6 vs. 77.5). We then apply this setup across the label aggregation strategies described in Sec. 5.2. For *Local-Hard*, we sweep τ from 0.3 to 0.9 and report the best-performing settings. Table 6 summarizes the performance across ImageNet benchmarks.

Several key observations emerge from the results. First, across nearly all configurations, our multi-label annotations outperform the original single-label supervision. The only exception is a marginal drop in IN-Val top-1 accuracy when no global label signal is used. Otherwise, we observe consistent improvements—up to $+0.6$ and $+1.1$ in top-1 accuracy on ReaL and INv2, and up to $+0.6$ and $+2.0$ mAP on ReaL and INv2-ML—demonstrating the effectiveness of our richer supervision. Among the aggregation strategies, combining local labels with a global signal consistently improves performance, especially on more challenging benchmarks like INv2 and INv2-ML. Soft labels also outperform hard labels across most metrics, supporting the value of preserving class confidence scores rather than applying fixed thresholds. Using the original ImageNet label as the global signal slightly outperforms our classifier's own global prediction (e.g., $+0.2$ top-1 on IN-Val and ReaL). We attribute this to the classifier being optimized for localized regions, making it less superior for global image-level predictions. Overall, the best-performing configuration, *Local-Soft + Original*, achieves the highest mAP on the challenging INv2-ML (74.7) and strong top-1 accuracy across all evaluation sets. Based on these findings, we adopt this setup as the default for all experiments in the main paper.

C. Main Experiment Protocols

C.1. Training Setup for ImageNet

For methods including *Original Label* (with or without label smoothing), *ReLabel [37]* with our object masks, and our *Multi-label* supervision—we adopt the best BCE training setup identified in Supp. B.3, extending training to 300 epochs. For *ReLabel*, we use the official released checkpoint (trained for 300 epochs with CE loss). For *Large Loss* (LL [15]), we follow their procedure: fine-tuning from the label-smoothed baseline checkpoint for 10 epochs, while sweeping key parameters: downweight ratio $\delta_{\text{rel}} \in [0.01, 0.4]$, revision mode `LL-R`(reject), `LL-Ct`(temporary correct), `LL-Cp`(permanent correct), and learning rate `lr` $\in [1e\text{-}5, 1e\text{-}4]$. The best-performing configuration is reported. For *SCL* [32], we report numbers from their original paper because code for SCL has not been released. Their method also finetunes an ImageNet-pretrained model using standard single-label supervision.

C.2. Cross-Architecture Robustness and Transfer

ResNet Training. For all ResNet experiments (original and our multi-label variants), we use the same 300-epoch BCE setup from Supp. B.3.

ViT Training. For ViTs, we adopt the DeiT-III [30] training recipe, which is well-suited for BCE loss. We adapt their implementation by replacing mixup and cutmix from `timm` (which assumes single-label) with `torchvision` versions that support true multi-label training. All other hyperparameters remain unchanged. We compare three modes: (1) single-label training (using released checkpoints), (2) fine-tuning with our multi-labels (from released checkpoints), and (3) end-to-end multi-label training from scratch. We report the best performance under each metric

Table 6. Evaluation of different relabeling configurations on multi-label benchmarks. We sweep thresholding strategies (probability vs. fixed threshold) and choice of global supervision (original vs. predicted) using a ResNet-50 trained on ImageNet for 100 epochs. The best and second best performance are highlighted by bold and underline, respectively.

Local Label	Global Label	Top-1 Acc ↑					mAP ↑		
		IN-Val	ReaL	IN-Seg	IN-v2	INv2-ML	ReaL	IN-Seg	INv2-ML
Original Label		77.6	84.1	84.1	65.7	77.7	87.1	87.6	72.7
Hard ($\tau = 0.5$)	*None*	77.3	84.6	85.0	66.0	79.2	86.9	88.1	73.4
Soft	*None*	77.4	84.3	84.5	66.2	78.4	87.2	88.1	73.0
Hard ($\tau = 0.8$)	Pred	77.7	<u>84.6</u>	84.9	66.1	78.7	87.0	88.0	72.3
Soft	Pred	77.6	84.5	85.0	66.5	<u>79.3</u>	<u>87.6</u>	88.2	<u>74.5</u>
Hard ($\tau = 0.9$)	Original	**78.0**	84.5	<u>85.0</u>	<u>66.6</u>	79.0	87.6	<u>88.3</u>	74.2
Soft	Original	<u>77.8</u>	**84.7**	**85.1**	**66.8**	**79.6**	**87.7**	**88.5**	**74.7**

with or without resize-center crop (scale ratio of $256/224$) during evaluation.

Transfer Learning. For transfer to VOC and COCO, we fine-tune each model for 100 epochs. We attach a randomly initialized linear head of shape $(1000, N_c)$, where N_c is the number of target classes (20 for VOC, 80 for COCO). This head is folded into the classifier after training, introducing no overhead at inference and empirically enabling stable and fast convergence. We use BCE loss with label smoothing $[0.001, 0.99]$, weight decay of $1e$-4, and sweep learning rates in $[5e$-$6, 3e$-$4]$. The input resolution matches the pretraining resolution of each checkpoint. No center crop is used during evaluation. Best results are reported (in Table 3, 10).

D. Comparison with Human Annotations

D.1. Qualitative Breakdown of Human Agreement

We applied our relabeling pipeline to the ImageNet validation set and compared the results against ReaL [4], the human-verified multilabel annotations. ReaL was constructed through a rigorous procedure: a diverse pool of 19 trained models was used to propose candidate labels for each image (including each model's top-1 prediction, other high-confidence predictions, and the image's original label), which was then narrowed to 6 models to ensure $> 97\%$ recall of true labels (i.e., utilizing a subset of the validation set labeled by experts as the golden standard). Human annotators reviewed on average 7–8 proposed labels per image, voting whether each label was present. Notably, if all 6 models agreed on the original ImageNet label, the image was not re-annotated by humans (this happened for $25,111$ images). The final ReaL set contains $57,553$ labels spanning $46,837$ images, leaving $3,163$ images with no label after this process. Here we assess how our pipeline's outputs compare to it.

To structure the comparison, we convert the label set into one-hot fashion by thresholding softmax score of 0.5 [2], and categorize the images into five groups based on the overlap between our pipeline's predicted labels and the ReaL labels. We then conducted a detailed human evaluation on 250 randomly sampled images (50 from each category). The reviewer is a PhD student familiar with ImageNet categories, using external references for fine-grained distinctions. The five categories are: (1) ReaL provides no label for the image; (2) Exact label set match between our method and ReaL; (3) Our labels are a superset of ReaL (we predict additional labels beyond ReaL); (4) ReaL is a superset of ours; and (5) Partial overlap (each provides some unique labels). Below we summarize the findings for each category.

ReaL Has No Label (6.3%). First, it's important to understand why ReaL might assign no label to an image. In ReaL's annotations, $3,163$ images (about 6.3% of the validation set) were discarded as having no label. This can occur if none of the expert models' proposed labels met the confidence criteria – for example, the image may not contain any object from the 1000 ImageNet classes, or it is too ambiguous (e.g. only a small part of an object is visible, making identification uncertain). Our review revealed that about 16% of the sampled images in this category clearly had no valid label (the content genuinely falls outside ImageNet's classes or is unrecognizable), and another 20% were borderline/unsure cases where a label could not be confidently assigned (as illustrated in Fig. 5 (a)). These findings are consistent with prior studies that report a non-trivial rate of label errors in ImageNet [31]. Importantly, we found that 64% of the images in this category did contain at least one object from the ImageNet classes—indicating that ReaL either overlooked them due to its conservative filtering or excluded them due to annotation errors. Among these, our pipeline correctly recovered one or more missing labels in 94% of the cases. This shows that our automated approach can effectively address many of the "missing la-

[2] While tuning the threshold could increase the exact-match ratio with ReaL labels to as high as 70%, we use a fixed threshold of 0.5 to balance precision and recall.

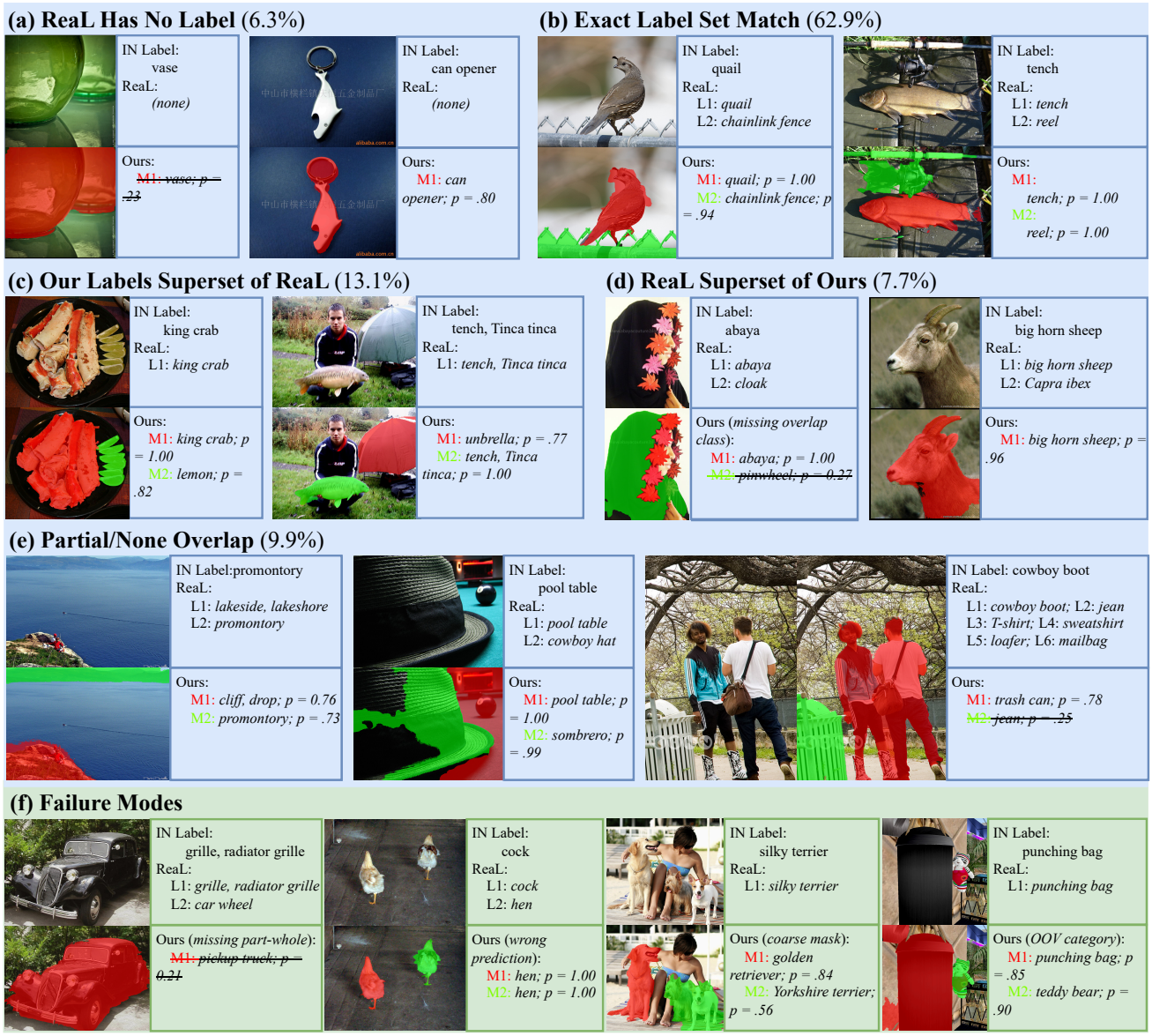

Figure 5. Additional qualitative comparisons between our multi-label annotations and those from ImageNet and ReaL [4]. Blue panels (a-e): Examples categorized by the degree of label overlap with ReaL. Labels pruned due to low confidence are indicated by strikethrough (~~deleted~~). Green panels (f): Common failure modes, including missed part–whole relations, fine-grained confusion, and proposals outside the label space (out-of-vocabulary, OOV).

bel" scenarios that ReaL leaves unannotated. However, in about 6% of these cases, while a valid object was present, our method still failed—typically due to poor segmentation (e.g., small or occluded objects), which led to missed predictions. In future work, we aim to explore strategies such as additional segmentation priors, multi-scale processing, and higher-resolution features to better handle these challenging cases.

Exact Label Set Match (62.9%). In this category, our pipeline produced an *exact match* with the ReaL multi-label annotations. This outcome indicates a strong agreement between our automatic method and the human multilabel annotation for those images. We further examined whether our method's explanations (in the form of object masks) align with the predicted labels. Reassuringly, for 94% of these images, the predicted mask(s) correspond closely to the actual object(s) of the given class, demonstrating that our model is looking at the correct regions when making the predictions. Fig. 5 (b) shows an example where the mask precisely highlights the target object, supporting the chosen label. In a small fraction (6%), however, the masks were found to be noisy or misaligned – for instance, high-

lighting unrelated patches or only part of the object. These few cases suggest that the model occasionally relies on context or struggles to precisely localize the object, even when it predicts the correct label. We hypothesize that this stems from slight overfitting or bias in the classification head: the object proposals included in its training are automatically filtered with ReLabel [37] which could include false positives (see Fig. 2 (c)). In addition, the ViT's global self-attention enables a masked patch to propagate or receive information from the entire image, which may contribute to the observed bleed-through. Nonetheless, these cases turn to be low-confidence predictions, and our pipeline's mask-based approach largely mitigates such issues (see Fig. 2 (d)).

Our Labels Superset of ReaL (13.1%). In these cases, our pipeline predicted one or more additional labels that were not present in ReaL's annotation for the image (while still predicting all the labels that ReaL did). The key question is whether our extra labels are indeed valid or are false positives. We found that 74% of the additional labels proposed by our method were correct upon human inspection – i.e. there was genuinely an instance of that class in the image, which ReaL's labels had missed (see Fig. 5 (c)). This suggests that ReaL, despite being more comprehensive than the original single-label ImageNet, still missed a non-trivial number of valid labels. One reason is the design of the ReaL annotation process: if all the models in their proposal set confidently agreed on a single label (usually the original ImageNet label), the image was not sent for multi-label human review. This likely caused many secondary objects to be overlooked. Indeed, previous analyses indicate that roughly 20–30% of ImageNet validation images contain multiple objects or multiple plausible labels [31].

A single-label model ensemble tends to pick only the dominant object, so those images could receive no additional labels in ReaL's pipeline. Our method, by contrast, explicitly searched for multiple objects via segmentation and was able to identify many of those missing labels. On the other hand, 26% of the extra labels from our pipeline in this category turned out to be incorrect upon review. The most common failure mode (about 18% out of the 26%) was that our model identified a region and assigned a related but wrong class to it – often because the true object category was not actually among the 1000 ImageNet classes (for instance, we assign `teddy bear` to stuffed toys in the image; Fig. 5 (f)). In these cases, the pipeline might detect a genuine object, but labels it as the closest known category. Such mistake roots in the fixed label space. The rest of the errors were more minor: in 4% of cases, the mask was imperfect (covering only part of the object or blending objects) which led to a misclassification, and in another 4% the mask was fine but the classifier simply made the wrong prediction for that region. Overall, however, the high percentage of correct new labels indicates that our pipeline can substantially augment ReaL by recalling additional valid labels that are missed.

Table 7. Human evaluation breakdown of agreement between our relabeling and ReaL [4] on the ImageNet validation set. Based on detailed review of 250 sampled images, we estimate the proportion of images falling into three categories: (a) ReaL provides no label (6.33%), (b) ReaL's label set exactly matches the human-determined ground truth (estimated 75.6%), and (c) ReaL misses or mislabels one or more ground-truth classes (estimated 18.1%). Each group is further subdivided by the quality of ReaL's annotations and whether our method successfully recovers missing or incorrect labels. Percentages indicate estimated proportions over the full 50,000 ImageNet validation set. Orange cells denote ReaL's labeling status and annotation breakdown; green cells indicate our method's fixes or misses. GT refers to our human-verified ground truth.

(a) ReaL Has No Label: 6.33%				
Unsure: *1.27%*	Clear No Label: 1.01%		Has Label: 4.05%	
	Ours No Label: 0.51%	Ours Has Label: 0.51%	Ours Fix: 3.80%	Not Fixed: 0.25%

(b) ReaL Exact Match to GT: 75.6%					
Ours Match: 62.9%		Ours Missed: 6.47%		Ours Wrong: 6.18%	
Good Mask: 59.1%	Noisy Mask: 3.77%	Ambig. Class: 4.31%	Other: 2.16%	Due to Mask: 3.43%	Wrong Pred.: 2.75%

(c) ReaL Not Match to GT: 18.1%					
Missing Label: 12.3%		Error Label: 2.69%		Missing and Error: 3.13%	
Ours Fix: 10.5%	Not Fixed: 1.79%	Ours Fix: 1.79%	Not Fixed: 0.90%	Ours Fix: 2.24%	Not Fixed: 0.90%

ReaL Superset of Ours (7.7%). This is the opposite scenario – here ReaL provided one or more labels that our pipeline failed to predict. The first point to note is whether ReaL's extra labels are truly correct. We found that in about 16% of such cases, the additional label from ReaL appears to be incorrect or at least very debatable. In ReaL's annotation protocol, the threshold for including a label was set to ensure high recall. This means ReaL sometimes included labels for ambiguous instances to avoid missing a possible object, even if that label might not be clearly evident. For example, an object could be labeled as two different but visually similar classes if the annotators weren't sure which it is, or a part of an object might get a separate label (e.g. labeling both "airplane" and "wing" in an image of an airplane). Some of these extra labels turn out, on closer examination, to be unnecessary or erroneous – they were essentially false positives introduced by an overly generous labeling policy. Aside from those, the remaining 84% of ReaL's additional labels were correct, meaning our pipeline genuinely missed detecting those objects or classes. We analyzed why our method missed these labels, and found a few recurring issues:

- **Ambiguity in class definitions (56%)** – In a majority of these cases, the image had an object that could plausibly belong to multiple closely related classes, or the ImageNet classes themselves overlap in scope. ReaL often handled this by assigning multiple labels to cover all

bases, whereas our pipeline typically picked just one. For instance, an image of a certain dog breed might have been given two breed labels in ReaL because it was hard to distinguish (both labels were considered valid) – our model might only predict one of them. Similarly, images with objects that fall into an "X and part-of-X" situation ("car" and "car wheel") or singular/plural duplicates ("sunglasses" vs "sunglass") can legitimately have multiple labels (as shown in Fig. 5 (d, f)). Our pipeline, due to its single-instance mask proposal, might only tag the larger object or the more obvious class. The inherent ambiguity or overlapping taxonomy of ImageNet classes led to our method missing some labels that ReaL included. Notably, recent work has argued for collapsing such overlapping classes to improve evaluation [31].

- **Segmentation limitations (18%)** – In these cases, our pipeline did not produce a separate mask for an object of interest, often because the object was very small, occluded, or touching another object. For example, if two objects were adjacent, MaskCut might generate one combined mask covering both, causing the model to predict only one class for that region (Fig. 5 (f)). Consequently, a valid label went missing simply because the object was not isolated by our proposals.
- **Low confidence pruning (6%)** – Here, our model actually did predict the correct label, but with a confidence below our threshold $\tau = 0.5$, so we filtered it out. In other words, the label was on our initial list but got discarded for not being confident enough.
- **Misclassification (4%)** – In the remaining small portion, the pipeline did propose a mask for the object and should have been able to predict the label, but it simply predicted the wrong class for that region. These are straightforward errors of the classification head on clear objects (e.g. mistaking one breed for another).

Partial Overlap (9.9%). This category includes images where our pipeline and ReaL each identified some correct labels that the other missed. In other words, the two label sets intersect but neither is a strict subset of the other. Our analysis found a mix of outcomes here: in 36% of these cases, both our labels and ReaL's labels were correct in what they included, but each method was incomplete – together they gave a more complete description of the image than either alone. This underscores how challenging it is to get a perfectly comprehensive label set, even with human annotators, and shows that our pipeline can complement the human labels by catching different subsets of objects. In 28% of partial-overlap cases, ReaL's unique labels were correct while our unique label was incorrect (so ReaL had the better coverage), whereas in 20% of the cases it was the opposite – our extra label was correct and one of ReaL's labels was actually incorrect. Finally, 16% of the cases had errors on both sides (each provided at least one wrong label that the other did not). The reasons for these misses and mistakes mirror the patterns discussed above. Ambiguities in the image or label definitions often led to one method assigning an extra label that the other omitted; for example, ReaL might include an arguably present object that our pipeline's mask proposals overlooked, or conversely our model might flag an object with a label that ReaL's annotators were overly conservative about. Likewise, some of ReaL's labels in this category turned out to be over-generalizations or slight mistakes, while some objects our model missed were due to segmentation or confidence issues as described.

This qualitative evaluation shows that our automated relabeling pipeline aligns closely with the human-curated ReaL labels while offering meaningful improvements in several areas. In the majority of cases, our method either agrees with ReaL or correctly recovers additional labels that ReaL misses, demonstrating strong recall of true object classes. In particular, our pipeline successfully resolves many "no-label" cases and augments single-label annotations with valid secondary objects—addressing gaps in ReaL's coverage due to its high-precision filtering protocol. To quantify this alignment, Table 7 presents an estimated breakdown of label agreement and disagreement categories, based on our human evaluation of 250 sampled images.

At the same time, our method is not without limitations. It cannot fully resolve inherent ambiguities in the ImageNet taxonomy—such as overlapping, fine-grained, or underspecified categories—which can challenge both automated models and human annotators. Occasional errors also arise from imperfect segmentation, conservative thresholding, or incorrect predictions for ambiguous regions. However, these failure modes are relatively rare and often complementary to those found in ReaL. Overall, our multi-label pipeline provides a robust, interpretable, and scalable alternative to manual annotation, and can help identify missing or questionable labels. In practice, combining model-driven relabeling with human verification offers a promising path toward improving the quality of large-scale datasets like ImageNet.

E. Handling Ambiguous Classes

Our method assumes one label per region, which is generally appropriate for datasets like COCO [17] or VOC [9], but can break down under ImageNet's taxonomy (see Fig. 5 (d,f)). Ambiguities arise in cases involving synonyms (e.g., `sunglass` vs. `sunglasses`), part–whole relationships (e.g., `airplane` vs. `wing`), or hierarchical categories (e.g., `wool` vs. `cardigan`). These lead to under-labeling when only one class in a semantically overlapping pair is assigned. We explore ways to mitigate this issue through targeted post-processing.

Table 8. List of identified ambiguous class pairs in ImageNet. For each pair, we report class-wise occurrence counts, as well as conditional label confidence: Conf$(A|B)$ is the proportion of times class A appears given B is present, and vice versa.

Co-occurrence	Class A	Class B	Freq(A)	Freq(B)	Conf(A|B)	Conf(B|A)
153	sunglass	sunglasses, dark glasses, shades	175	168	0.91	0.87
135	laptop, laptop computer	notebook, notebook computer	157	151	0.89	0.86
88	monitor	screen, CRT screen	227	101	0.87	0.39
87	maillot	maillot, tank suit	106	119	0.73	0.82
84	car wheel	grille, radiator grille	232	143	0.58	0.36
55	missile	projectile, missile	70	66	0.83	0.79
51	suit, suit of clothes	Windsor tie	126	83	0.61	0.40
45	space bar	typewriter keyboard	107	70	0.64	0.42
35	car wheel	convertible	232	61	0.57	0.15
34	bookcase	bookshop, bookstore, bookstall	116	77	0.44	0.29
33	bathtub, bathing tub, bath, tub	tub, vat	87	42	0.79	0.38
30	airliner	wing	79	84	0.36	0.38
26	cassette	cassette player	69	62	0.42	0.38
23	warplane, military plane	wing	57	84	0.27	0.40
20	ox	oxcart	47	65	0.31	0.43
18	analog clock	wall clock	48	86	0.21	0.38
17	car wheel	pickup, pickup truck	232	54	0.31	0.07
17	convertible	grille, radiator grille	61	143	0.12	0.28
15	grille, radiator grille	pickup, pickup truck	143	54	0.28	0.10
14	assault rifle, assault gun	rifle	85	58	0.24	0.16
14	car wheel	minivan	232	54	0.26	0.06
14	car wheel	Model T	232	60	0.23	0.06
13	grille, radiator grille	sports car, sport car	143	45	0.29	0.10
12	beach wagon, station wagon	grille, radiator grille	51	143	0.08	0.23
10	drum, membranophone, tympan	steel drum	60	76	0.13	0.17
10	flagpole, flagstaff	pole	70	85	0.12	0.14
7	barrel, cask	rain barrel	45	64	0.11	0.16
7	cardigan	wool, woolen, woollen	58	102	0.07	0.12
7	grille, radiator grille	minivan	143	54	0.13	0.05
5	abaya	cloak	55	51	0.10	0.09
5	grille, radiator grille	Model T	143	60	0.08	0.03
4	grille, radiator grille	racer, race car, racing car	143	48	0.08	0.03
4	rule, ruler	slide rule, slipstick	81	39	0.10	0.05
3	measuring cup	cup	54	140	0.02	0.06
3	espresso maker	espresso	41	57	0.05	0.07

E.1. Identifying Ambiguous Class Pairs

To systematically identify such cases, we analyze the top 1000 most frequently co-occurring class pairs in the ReaL [4] validation set (frequency ≥ 3). We manually examine each pair and verify the possible ones that contain ambiguity by sampling 50 images per class for visual comparison. This process yields 34 ambiguous class pairs, listed in Table 8.

E.2. Post-Processing with Co-occurrence Priors

We propose two complementary post-processing strategies that leverage empirical co-occurrence statistics from ReaL and model-predicted soft labels to resolve ambiguity and enrich label completeness.

Co-occurrence Prior Propagation. We construct a symmetric co-occurrence matrix $C \in [0,1]^{K\times K}$ from ReaL, where $K = 1000$ and C_{ij} denotes the normalized frequency that class j appears alongside class i among the identified ambiguous pairs. For a predicted soft label vector $\mathbf{p} \in [0,1]^K$, we apply:

$$\mathbf{p}' = \text{clip}(C \cdot \mathbf{p}, 0, 1), \tag{5}$$

where $\mathbf{p}'$ is the adjusted soft label vector and clip denotes element-wise clamping to $[0,1]$. This propagates confidence from one class to its frequent ambiguous companion.

Asymmetric Thresholding. For each ambiguous class pair (a, b), we derive class-specific thresholds τ_a and τ_b based on their conditional co-occurrence in ReaL. If class a is the top-1 predicted label and the model's predicted confidence for class b exceeds τ_b, we add b as an additional label by assigning it the same confidence score as a. This rule is applied symmetrically: a is also added if b is top-1 and a exceeds τ_a. These thresholds are designed to match the observed conditional probabilities $P(b \mid a)$ and $P(a \mid b)$, ensuring that added labels reflect the typical co-occurrence patterns in the data.

Let:

- N_a, N_b = number of images labeled with class a and b

Table 9. Effect of ambiguity resolution strategies on ImageNet multi-label training. We evaluate two approaches for handling ambiguous class pairs: (1) Coexistence Prior, which adjusts label targets using a class co-occurrence matrix derived from ReaL; and (2) Threshold Pairing, which applies asymmetric confidence thresholds to resolve semantic overlaps. Both methods improve upon the baseline IN1k-Mul supervision across all evaluation sets. Best performances are highlighted in bold.

Model	Image	Method	Top-1 Acc ↑					Multi-Label: mAP ↑		
			IN-Val	ReaL	IN-Seg	INv2	INv2-ML	ReaL	IN-Seg	INv2-ML
ResNet-50	224^2	Single-label E2E	78.2	84.1	84.4	66.1	78.2	87.0	87.7	72.3
		Multi-label E2E	78.7	85.6	85.5	67.4	81.0	88.2	88.8	76.2
		+ Threshold Pairing	78.8	**85.9**	**86.0**	**67.8**	**81.1**	88.5	89.3	**76.9**
		+ Coexistence Prior	**78.9**	85.8	85.9	67.4	80.9	**88.5**	**89.3**	76.9
ResNet-101	224^2	Single-label E2E	79.3	84.7	85.3	68.2	80.0	87.1	88.2	72.7
		Multi-label E2E	80.2	86.7	86.9	69.2	82.3	89.0	89.9	77.4
		+ Threshold Pairing	**80.6**	86.9	87.0	69.2	**83.2**	89.3	90.1	**78.8**
		+ Coexistence Prior	80.5	**87.0**	**87.1**	**69.5**	83.0	**89.5**	**90.2**	78.8
ViT-base	224^2	Single-label E2E	**83.7**	88.2	88.2	73.6	85.8	90.6	90.9	80.3
		Multi-label E2E	83.4	88.8	89.1	73.9	86.8	90.7	91.5	81.8
		+ Threshold Pairing	83.5	88.8	**89.1**	73.9	**87.5**	90.9	**91.7**	82.4
		+ Coexistence Prior	83.5	**88.9**	88.9	**74.4**	87.2	**91.1**	91.6	**82.7**
	384^2	Single-label E2E	**85.0**	88.8	88.8	74.7	87.3	91.0	91.5	81.5
		Multi-label E2E	84.6	89.4	89.5	75.2	88.0	91.2	91.9	82.9
		+ Threshold Pairing	84.6	89.6	89.5	75.3	88.6	91.4	92.0	83.4
		+ Coexistence Prior	84.6	**89.7**	**89.6**	**75.4**	**88.6**	**91.5**	**92.1**	**83.8**

respectively,

- N_{ab} = number of images labeled with both a and b,
- M_a, M_b = number of single-labeled images (with only a or only b) to be upgraded with the missing class.

We estimate M_a, M_b by solving:

$$P(b \mid a) = \frac{N_{ab} + M_b}{N_a + M_b}, \qquad P(a \mid b) = \frac{N_{ab} + M_a}{N_b + M_a}, \quad (6)$$

which gives a closed-form solution for M_a and M_b. We then cap the number of ambiguous instances added per class to ensure they do not exceed the number of images that contain class a or b individually (excluding co-occurrence):

$$M_a = \texttt{Max}(0, \texttt{Min}(M_a, N_a - N_{ab})),$$
$$M_b = \texttt{Max}(0, \texttt{Min}(M_b, N_b - N_{ab})).$$

We rank single-labeled images by their softmax score on the missing class and select the top M_a and M_b examples accordingly. The lowest selected softmax score becomes the threshold τ_b (and similarly for τ_a). These thresholds are derived based on the validation set and directly applied to the training set.

E.3. Training with Adjusted Labels

We retrain models using our ambiguity-adjusted multi-label annotations, following the setup detailed in Supp. C.2. Results are presented in Table 9. Both correction strategies—Asymmetric Thresholding and Co-occurrence Prior Propagation—consistently improve over the original multi-label supervision across all architectures, input resolutions, and evaluation sets. For example, for ResNet-50, Asymmetric Thresholding improves top-1 accuracy by $+0.27$ (ReaL) and $+0.32$ (INv2), and mAP by $+0.35$ (ReaL) and $+0.68$ (INv2-ML). Co-occurrence Prior also yields performance gains, with up to $+0.30$ (ReaL, ResNet-101) and $+0.46$ (INv2, ViT-B/224) in top-1 accuracy, and $+0.46$ (ReaL, ResNet-101) and $+0.68$ (INv2-ML, ResNet-101) in mAP. These results validate the effectiveness of our proposed remedies and suggest that class-level ambiguity, which is rooted in the structure of the ImageNet taxonomy, can be effectively mitigated using priors derived from a lightweight calibration set (e.g., ReaL, whose size is roughly 3.65% relative to the ImageNet training set).

F. Additional Analyses

F.1. Robustness to Input Resolution (384×384)

We extend the analysis in Sec. 5.4 to higher input resolution (384^2) for ViT models, following the DeiT3 training setup. Table 10 presents results for training or fine-tuning with multi-label supervision at both 224^2 and 384^2 input resolutions. Overall, the improvements from multi-label training persist across input resolutions. For fine-tuning at 384^2 resolution, the largest gains on ReaL and INv2-ML w.r.t. top-1 accuracy are $+0.79$ (ViT-Base) and $+1.45$ (ViT-Small). And, in terms of mAP, the improvements are up to $+0.48$ and $+2.55$ on ReaL and INv2-ML, respectively. End-to-end

Table 10. End-to-end training and transfer performance with our multi-label annotations. We compare single-label (Single-label E2E) training, fine-tuning with our multi-labels (+ Multi-label FT), and end-to-end multi-label training (Multi-label E2E) across various backbones and input sizes. Best results are highlighted in bold. Our multi-label supervision improves in-domain performance on ImageNet and its variants, and yields consistent gains in downstream multi-label transfer to COCO and VOC.

Model	Image	Supervision	ImageNet Evaluation								Transfer Learning	
			Top-1 Acc ↑					Multi-Label: mAP ↑			mAP ↑	
			IN-Val	ReaL	IN-Seg	INv2	INv2-ML	ReaL	IN-Seg	INv2-ML	COCO	VOC
ViT-small	224^2	Single-label E2E	81.4	87.0	87.2	70.7	83.1	89.0	89.7	75.6	79.1	91.0
		+Multi-label FT	81.5	87.7	87.8	70.8	84.1	90.1	90.9	80.2	81.0	91.9
		Multi-label E2E	**82.0**	**88.1**	**88.3**	**72.2**	**85.1**	**90.3**	**91.1**	**80.7**	**83.3**	**93.3**
	384^2	Single-label E2E	83.2	88.2	88.2	72.6	85.1	90.5	91.0	79.6	84.5	92.8
		+Multi-label FT	83.3	88.8	89.0	73.2	86.6	91.0	91.7	82.2	85.4	93.9
		Multi-label E2E	**83.5**	**88.9**	**89.1**	**74.4**	**87.0**	**91.0**	**91.8**	**82.3**	**87.3**	**94.9**
ViT-base	224^2	Single-label E2E	**83.7**	88.2	88.2	73.6	85.8	90.6	90.9	80.3	83.0	92.7
		+Multi-label FT	83.6	**88.9**	88.8	**74.0**	**86.9**	90.6	91.3	81.3	83.8	93.6
		Multi-label E2E	83.4	88.8	**89.1**	73.9	86.8	**90.7**	**91.5**	**81.8**	**84.7**	**94.5**
	384^2	Single-label E2E	**85.0**	88.8	88.8	74.7	87.3	91.0	91.5	81.5	86.7	94.3
		+Multi-label FT	84.9	**89.6**	**89.6**	75.1	**88.1**	**91.5**	**92.2**	**83.9**	87.3	94.9
		Multi-label E2E	84.6	89.4	89.5	**75.2**	88.0	91.2	91.9	82.9	**87.7**	**95.3**
ViT-large	224^2	Single-label E2E	84.6	88.6	88.5	74.7	87.1	90.8	91.2	81.4	84.8	93.4
		+Multi-label FT	**84.6**	**89.3**	**89.4**	75.1	**88.2**	**91.1**	91.9	**83.3**	85.2	94.4
		Multi-label E2E	84.3	89.3	89.4	**74.9**	87.9	91.2	**91.9**	83.0	**86.4**	**95.0**
	384^2	Single-label E2E	**85.9**	89.8	89.7	76.3	89.1	91.6	92.1	83.1	87.3	94.2
		+Multi-label FT	85.5	**90.0**	**90.1**	**76.5**	**89.5**	**91.7**	**92.5**	**84.8**	87.8	95.0
		Multi-label E2E	85.2	89.8	89.9	76.4	89.1	91.6	92.3	84.0	**89.5**	**96.1**

Table 11. Evaluation of k-NN-based entropy of penultimate-layer features (Euclidean distance). Models trained with our multi-label supervision (IN1k-Mul) consistently exhibit higher entropy across datasets, indicating greater feature diversity and reduced representation collapse compared to single-label training (IN1k-Sig). This suggests improved representational quality and helps explain the observed gains in transfer learning performance.

Model	Supervision	Entropy ↑		
		ImageNet	VOC	COCO
ResNet-50	Rand. Init.	1114	1540	1509
	IN1k-Sig	1705	2582	2310
	IN1k-Mul	**2034**	**2663**	**2426**
ViT-B	Rand. Init.	0.0	137	19
	IN1k-Sig	562	910	810
	IN1k-Mul	**791**	**1040**	**959**

Table 12. Top-1 accuracy comparison of soft-label training under different supervision strategies. We follow the ReLabel setup to generate a label map for each image and assign soft labels to random crops based on patchwise logits. All methods use ResNet-50 trained on ImageNet-1K with either 100 or 300 epochs. Our spatial label map achieves comparable or better classification accuracy, while requiring less labeled data.

Method	100 Epoch		300 Epoch	
	IN-Val	ReaL	IN-Val	ReaL
Original	77.5	83.5	78.2	84.1
ReLabel [37]	71.9	78.6	78.9	85.0
Classifier Map (Ours)	74.9	82.1	78.9	84.9

training also shows consistent trends: ViT-Small, ViT-Base, and ViT-Large yield gains of $+1.75$, $+0.48$, and $+0.12$ respectively in top-1 accuracy on INv2, and $+2.68$, $+1.44$, and $+0.85$ respectively in mAP on INv2-ML. Under transfer learning, ViT-Large benefits notably. Fine-tuning improves COCO and VOC mAP by $+0.45$ and $+0.80$ respectively, while end-to-end multi-label training yields even larger boosts of $+2.19$ and $+1.96$. These findings confirm the robustness and scalability of our multi-label supervision across model sizes and input resolutions.

F.2. Feature Diversity via k-NN Koleo Entropy

We hypothesize that multi-label training encourages less representation collapse. Prior works [12, 13] show that higher entropy correlates with lower neural collapse and vice-versa, suggesting that representations with higher entropy transfer better to out-of-distribution datasets. To evaluate transferability, following [13], we compute k-nearest-neighbor (k-NN)-based entropy ($k = 3$) for ImageNet, VOC, and COCO datasets. Results are summarized in Table 11. Across both ResNet and ViT, multi-label training consistently results in higher entropy than single-label supervision, supporting our claim that it leads to more diverse and transferable representations.

Table 13. Zero-shot segmentation performance on COCO using CutLER masks filtered by our classification head. We remove pseudo-masks whose top-1 softmax confidence falls below threshold τ. Despite excluding up to 12% of masks (at $\tau = 0.5$), segmentation performance improves across most metrics, suggesting that filtering out low-confidence proposals strengthens the quality of supervision for unsupervised segmentation. Metrics follow COCO-style evaluation: AP is the mean average precision over IoU thresholds, AP50/AP75 are AP at IoU 0.50/0.75, and APs, APm, APl represent AP for small, medium, and large objects, respectively.

Config	# Masks	AP↑	AP50↑	AP75↑	APs↑	APm↑	APl↑
Full	1.93M (100%)	8.71	17.50	7.78	1.91	7.33	22.52
$\tau = 0.3$	1.81M (94.0%)	8.81	17.80	7.90	1.87	7.49	**22.52**
$\tau = 0.5$	1.70M (88.0%)	**8.82**	**17.80**	**7.92**	**1.94**	**7.63**	22.35

F.3. Training with Soft Labels

ReLabel [37] generates a 15×15 spatial label map using a pretrained classifier without the global pooling layer. During training, a random crop is applied to both the input image and the corresponding region in the label map. The soft label for that crop is then obtained by applying ROI-align on the label map using the crop coordinates, and normalized to sum to 1 as the supervision target. We replicate this setup but replace ReLabel's ImageNet-21K–pretrained teacher network with our own classification head trained on localized region crops using only ImageNet-1K supervision. As shown in Table 12, our spatial label map achieves comparable or better classification accuracy—despite using less labeled data. Both methods outperform standard single-label training by +0.7–0.9 top-1 accuracy on IN-Val and ReaL, and our model converges faster and yields stronger performance at 100 epochs, suggesting that our spatial labels are better aligned with object regions.

F.4. Filtering CutLER Masks with Our Labeler

CutLER [34] trains an unsupervised segmentation model using MaskCut-generated object proposals from ImageNet as initial pseudo-ground-truth masks. In its first iteration, all proposals—including low-quality or background masks—are retained because there is no mechanism to assess mask reliability. A segmentation model is then trained on these raw masks and later used to refine them in subsequent iterations. To isolate the effect of proposal quality, we focus exclusively on this first-iteration training stage. Before training, we apply our region-level labeler to each released CutLER mask and compute its top-1 softmax confidence. Masks with confidence below a threshold of 0.5 are discarded, removing roughly 12% of proposals. We then train a Cascade Mask R-CNN [5] using either the original CutLER pseudo-masks or our filtered pseudo-masks (all derived from ImageNet). Evaluation is conducted on COCO in the standard zero-shot transfer setting. As shown in Table 13, filtering improves zero-shot segmentation accuracy, and even a more conservative threshold of 0.3—removing only 6% of masks—still provides a measurable gain. These results demonstrate that our region-level classifier can serve as an effective post-hoc filter for noisy pseudo-masks—enhancing the quality of the supervision signal and potentially improving downstream segmentation performance even before iterative refinement.

Figure 6. Example of interactive annotation using our classification head. Given arbitrary region proposals (highlighted in red), the model predicts ImageNet class labels with confidence scores. Even for out-of-vocabulary objects that are not explicitly covered by the label set (e.g., hiking boot), our classifier predicts semantically related categories (e.g., running shoe) with lower confidence, enabling efficient and flexible region-level labeling.

F.5. Interactive Annotation Tool

Our localized labeler can be repurposed for interactive region-level annotation. Given a user-specified bounding box, we apply ROI-Align to extract pooled features from the ViT backbone and pass them to our labeler head, which outputs a ranked list of ImageNet class predictions with confidence scores. This enables fast and flexible labeling of arbitrary regions. As illustrated in Fig. 6, our region-level labeler successfully predicts labels for multiple objects in the scene, even when the original and ReaL labels only include a single class. Notably, even when the target object is not explicitly represented in the ImageNet label space (e.g., a hiking boot), the labeler tends to predict semantically related classes (e.g., running shoe) with appropriately lower confidence. This behavior supports broad-category tagging and reduces the burden of exhaustive class coverage. This tool can accelerate human annotation for downstream datasets or for refining labels in existing benchmarks like ImageNet.