Learning to Discover and Detect Objects

Vladimir Fomenko[1*] **Ismail Elezi**[2] **Deva Ramanan**[3,4]
Laura Leal-Taixé[2,4] **Aljoša Ošep**[2,3,4]

[1]Microsoft Azure AI [2]Technical University of Munich
[3]Carnegie Mellon University [4]Argo AI

vladfomenko@microsoft.com
{ismail.elezi, aljosa.osep, leal.taixe}@tum.de deva@cs.cmu.edu

Abstract

We tackle the problem of *novel class discovery, detection, and localization (NCDL)*. In this setting, we assume a source dataset with labels for objects of commonly observed classes. Instances of other classes need to be discovered, classified, and localized automatically based on visual similarity, without human supervision. To this end, we propose a two-stage object detection network *Region-based NCDL* (*RNCDL*), that uses a region proposal network to localize object candidates and is trained to classify each candidate, either as one of the *known* classes, seen in the source dataset, or one of the extended set of *novel* classes, with a long-tail distribution constraint on the class assignments, reflecting the natural frequency of classes in the real world. By training our detection network with this objective in an end-to-end manner, it learns to classify all region proposals for a large variety of classes, including those that are not part of the labeled object class vocabulary. Our experiments conducted using COCO and LVIS datasets reveal that our method is significantly more effective compared to multi-stage pipelines that rely on traditional clustering algorithms or use pre-extracted crops. Furthermore, we demonstrate the generality of our approach by applying our method to a large-scale Visual Genome dataset, where our network successfully learns to detect various semantic classes without explicit supervision. Code is available at: `https://github.com/vlfom/RNCDL`.

1 Introduction

We tackle novel class discovery and localization in unlabeled datasets, a long-standing problem in computer vision [62, 66, 44, 45, 61]. As we are approaching the limits of well-understood and successful supervised training of object detectors [24, 23, 60, 29], this is becoming an increasingly more important problem, with the task of labeling novel and rare classes that appear in the long tail of object class distribution becoming prohibitively expensive [26].

Object class discovery is a difficult problem, as there may be many possible equally valid attributes for grouping patterns, such as object color or orientation. As shown in [32, 33], *novel class discovery (NCD)* can be well-posed and tackled in a data-driven manner by injecting prior knowledge on how we wish to group semantic classes using some degree of supervision. Existing methods [32, 33, 28, 27, 80, 7, 81, 82, 21] all assume curated datasets, where objects of interest are pre-cropped and semantic classes are fairly balanced. In contrast, we tackle joint *novel class discovery and localization (NCDL)*, a task of learning to discover and detect objects from raw, unlabeled, and uncurated data. This is a significantly more challenging problem, as images always contain a mixture of labeled and unlabeled object classes, and we need to localize as well as categorize them.

[*]work done while at Technical University of Munich.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

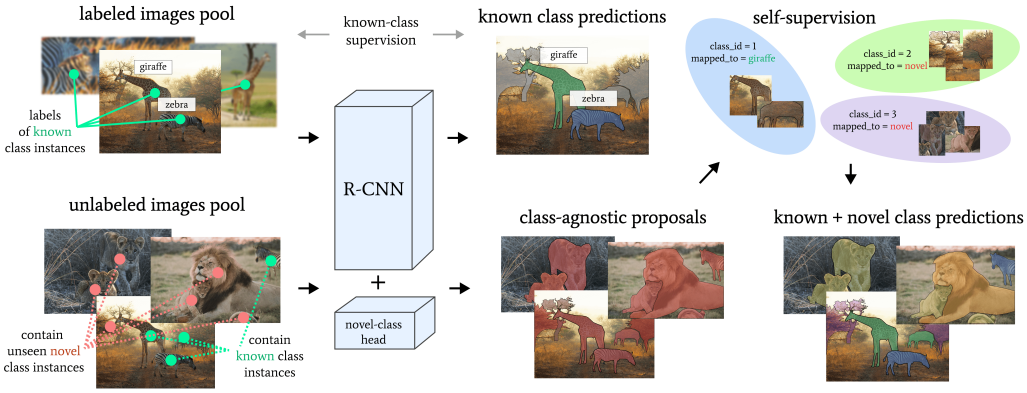


Figure 1: In *novel class discovery and localization* task we assume given a labeled images pool containing annotations for *known*, frequently observed semantic classes, and unlabeled images pool that may contain instances of *novel* classes. Our network learns to localize and recognize common semantic classes, as well as categorize novel classes, for which supervision was not given.

To study this problem, we re-purpose the COCO [48] and LVIS [26] datasets. We use 50% of COCO images together with COCO class vocabulary and labels for supervised model training. We use the remaining 50% of images without any labels to evaluate how well our network learns to detect novel classes using the long-tailed LVIS label set. This setting allows us to mimic real-world scenarios where no supervision is given for the target domain, while a rich set of labels allows us to rigorously study this problem.

We train a two-stage object detector [60, 29] in an end-to-end manner such that we can (i) correctly detect (*i.e.* localize and categorize) labeled objects, and, in parallel, (ii) detect unlabeled objects by learning feature representations suitable for categorization of objects that would otherwise be classified as the background class. We start with supervised training on the labeled, source domain, followed by the self-supervised training on the target (unlabeled) domain. We transfer knowledge from the source to the target domain by retaining weights for class-agnostic modules, such as RPN and bounding box regression, and instance segmentation heads. For classification, we reuse the primary classification head and add a new secondary head, training them jointly with an objective to categorize every region proposal consistently under multiple transformations and a constraint that the marginal distribution of class assignments follows a long-tail distribution. A non-uniform classification prior helps the network learn features that capture the diversity of objects, and also prevents the network from being biased towards the labeled or background classes.

Our end-to-end trainable method significantly outperforms prior art [75] that learns object representations using contrastive learning followed by k-means clustering, while providing categorization for any object with a single network forward pass. More importantly, with this work, we lay down the foundations for an exciting new line of research that goes beyond well-established supervised learning for object detection.

In summary, this paper makes the following **contributions**: (i) We study the problem of *novel class discovery and localization* in the wild using standard object detection datasets that do not pre-localize regions of interest manually or assume objects are roughly centered, *i.e.*, exhibit a strong photographer bias. (ii) We propose a simple yet effective and scalable method that can be trained end-to-end on the unlabeled domain. Our network is bootstrapped using a labeled dataset to learn features suitable for categorizing future instances. We train feature representations for arbitrary objects using the self-supervised objective. Finally (iii), we demonstrate that our method can discover and detect novel class instances and outperform prior work, demonstrating the generalization to datasets beyond COCO.

2 Related Work

Object discovery in image and video collections. Discovery of semantic objects that appear in unlabeled video collections is a long-standing research problem [67]. Early methods employ multiple

bottom-up segmentations of images [62, 66] to discover commonly occurring patterns using topic modeling, *e.g.*, Dirichlet allocation [6].

Lee *et al.* propose to rely on saliency [44] or to incorporate semantic knowledge about certain semantic classes, for which supervision is available [45]. These methods are evaluated on smaller datasets [25, 65, 19] that contain one or a handful of well-delineated objects. Rubinstein *et al.* [61] demonstrated that discovery can be scaled to internet photo collections.

Beyond images, Kwak *et al.* [43] tackle the discovery of objects in YouTube videos that usually contain a single object exhibiting dominant motion. They build on a two-stage approach: (i) identify the most salient region proposals in terms of appearance and motion, and (ii) co-segment consistently appearing objects across video clips. Video object discovery videos, recorded from a moving platform, were tackled in [56, 57] by first mining video-object tracks by associating region proposal network (RPN) [60] based proposals across time, followed by clustering of video proposals based on pre-trained CNN features. Recent work on open-set panoptic segmentation [35] similarly employs clustering based on RPN features to discover commonly-occurring objects, used as pseudo-labels for learning to segment unlabeled objects. The method by [75] follows the general approach of clustering RPN proposals, however, instead of relying on pre-trained CNN features, they learn representations suitable for clustering using self-supervised objective, trained via contrastive learning.

While most of the foregoing works focus on saliency criteria to significantly narrow down the number of region proposals, [69, 70] resort to combinatorial optimization that can cope with a large set of object proposals to discover objects that frequently appear in image datasets. It has been shown [71] that this approach can be scaled to large image collections, *e.g.*, OpenImages [42].

Novel class discovery. *Novel class discovery* (NCD) methods assume labeled data for some semantic classes is given. This data can be used to train a feature extractor and learn a clustering criterion to guide a notion of similarity and unlabeled data is assumed to contain novel class instances only. Hsu *et al.* [32, 33] propose a framework that injects prior knowledge on how we wish to group semantic classes from supervised learning on labeled images to the unlabeled domain via learnable similarity function. The work of Han *et al.* [28] builds on deep clustering [78] and suggests an approach for automatically discovering the number of novel classes. In follow-up work, Han *et al.* [27] build on pairwise similarity of image features [12] and propose leveraging self-supervised pretraining along with freezing the backbone to reduce bias towards the known classes. Its follow-up works [80, 37, 82, 81] further boost the performance by introducing additional self-supervised and regularization components. UNO [21] proposes a simple method based on SwAV [11] that bootstraps signal for novel class training using pseudo-labels generated under equipartitioning constraints. ORCA [7] extends the NCD problem setting and assume that at inference time, we are given images containing instances of novel as well as known objects. Motivated by [68, 12, 72], authors leverage self-supervised initialization of backbone, supervised loss along with unsupervised loss based on pairwise pseudo-labels, and novel dynamic softmax margin [72] to alleviate bias of the known classes. In contrast to our work, the foregoing methods investigate the problem in the image classification setting, assuming objects of interest are localized (pre-cropped), and operate under absence of outliers (*e.g.*, images containing no objects).

Related and complementary research directions. Methods for *generic* [49] or *large vocabulary object detection* [26] focus on detecting/segmenting a large set of a-priori known semantic classes, often by utilizing multiple datasets and (possibly weak) supervision [83, 59, 34]. A special case is *zero-shot learning* [77] and *zero-shot object detection* [3, 54, 58], where (mostly weak) supervision for unseen classes comes in the form of attributes, class names, latent features, or other forms of auxiliary data, commonly used in a conjunction with language models embeddings to learn a joint embedding space. In *few-shot learning* [39, 73, 20, 8] given is a limited supervision, where algorithms are required to learn from one or a few data samples per each class. In [20, 8] authors employ a secondary classification head that is trained using few-shot supervision and distillation [20] or margin loss [8]. In contrast to zero-shot, few-shot, and large-vocabulary detection, we assume no supervision and prior knowledge for object classes that appear in the tail.

Methods for *open-set recognition* [63, 64, 36, 5] and *detection* [54, 18] focus on calibration of per-class uncertainties to minimize the confusion between known and possibly unknown classes. *Open-world recognition* methods, as defined by [4, 50], must explicitly recognize unknown object instances that were not observed as labeled samples during the model training, and must continually

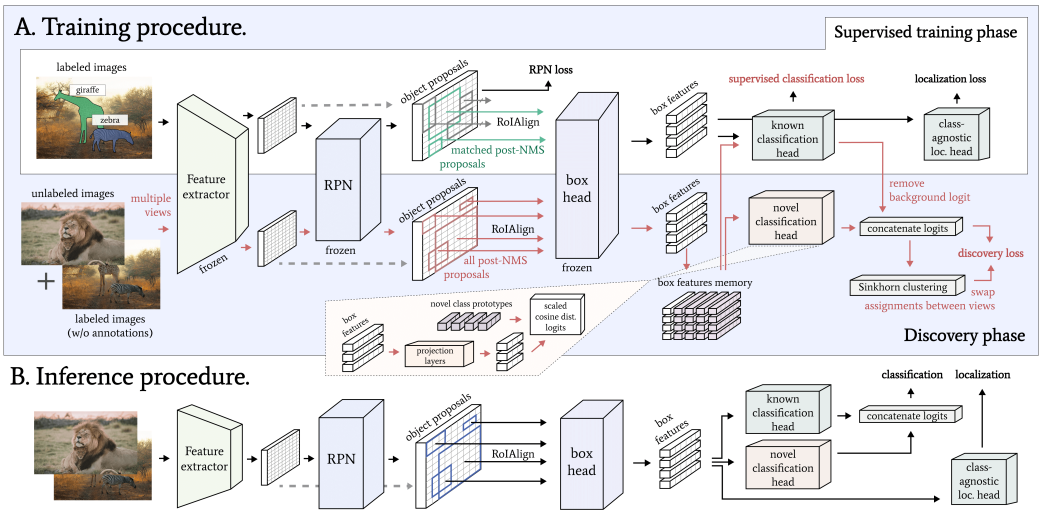


Figure 2: A high-level overview of our network. *(top)* ***During the supervised training phase***, we train our backbone and RPN networks using *labeled* data, together with classification head and a class agnostic localization head. ***During the discovery phase***, we freeze all the layers of the network apart from classification head and attach and train a novel classification head using *unlabeled* data. ***During the inference*** *(bottom)*, we perform a standard R-CNN pass, using classification heads of both known and novel categories to predict a class assignment for each proposal. This can be either one of K classes, that were presented a labeled samples during the model training, or any novel object class that appears in the training data.

update object detectors to recognize these unknown instances. Recent method for open-world detection [38] focuses on the continual learning aspect of the task: for the evaluation, labels for "novel" classes are fully provided (those are the held-out classes). Our work is orthogonal, and tackles the "missing bit" of the open-world detection pipeline: discovery and semantic grouping of novel-class objects that can appear in our training data as well, however, are unlabeled. Clusters of discovered novel classes could be verified by human annotators and used to update detection models in a continual fashion, as proposed by [4].

3 Novel Class Discovery and Localization

We first detail novel class discovery and localization (NCDL), the task of learning to discover and detect objects from raw, unlabeled, and uncurated data. Given is a source (*labeled*) object detection image dataset D^k for (known) object classes C^k ($|C^k| = K$), that provides both class and box level objects supervision: $D^k = \{(\mathbf{I}, \{(\mathbf{b}, c)\})\}$, where $\mathbf{I} \in \mathbb{R}^{h \times w \times 3}$, $\mathbf{b} \in \mathbb{R}^4$, and $c \in C^k$. In addition, given is one or more target (*unlabeled*) datasets $D^n = \{I\}$ that may contain instances of both previously seen and unseen (novel) object classes $C^k \cup C^n$ ($|C^n| = N, N > K$) that we need to learn to detect in the absence of any labels. For an arbitrary image I from the source or target domain, NCDL methods should detect all object instances present in the image: $\{(\mathbf{b}', c')\}$, *i.e.* localize them along with predicting their semantic class. For example, an instance of a known class, *e.g.*, car, needs to be detected as the `car` class, while a novel class instance with $c' \in C^n - C^k$ needs to be assigned to one of unknown, novel classes (*e.g.*, `unknown_1`).

3.1 Region-based Novel Category Discovery and Localization (RNCDL)

As the main contribution of this paper, we propose an end-to-end trainable detection and discovery network that jointly learns discriminative representations for all region proposals, their localization, and categorization alleviating the need for explicit clustering after the network training.

We base our method on a two-stage Faster or Mask R-CNN [60, 29], trained in a supervised manner on a dataset that contains labels for some semantic classes. Such detectors can reliably localize

and classify objects, for which supervision is given during the model training. Moreover, they are naturally suitable for the NCDL task, as their region proposal network (RPN) is trained in a class-agnostic manner to propose regions of interest that likely contain objects. Given object-centric feature representation for each localized candidate region one natural approach for discovering (categorizing) novel classes would be to cluster such regions in feature space using, *e.g.*, k-means, *c.f.*, [56, 35, 75]. Such feature representations learned via supervised R-CNN training are suitable for finding regions that likely contain objects and classifying known objects. However, features representing regions that do not contain objects of labeled classes are effectively collapsed and, therefore, not discriminative enough for robust categorization of novel classes. Furthermore, such an approach has a strong bias toward classifying novel instances as one of the labeled classes.

Overview. In the following, we present our Region-based Novel Category Discovery and Localization (RNCDL) method, which learns to categorize all region proposals directly on the target (unlabeled) dataset in an end-to-end manner. In a nutshell, we first train our detector on the labeled (source) domain to train class-agnostic modules (Fig. 2A, top), such as RPN, bounding box regression head, and (optionally) instance segmentation head, as well as a classifier for known classes. This way, we learn to localize and segment objects and additionally categorize instances of known classes. We proceed with self-supervised training (Fig. 2A, bottom)) on the target (unlabeled) domain to learn to additionally categorize instances of unknown classes. To this end, we add a secondary classification head, and train both classification heads jointly, with an objective to categorize every region proposal consistently under multiple transformations and a constraint that the marginal distribution of class assignments follows a long-tail distribution.

Supervised training on the source domain. We begin with a self-supervised backbone pre-training [31] on a large-scale image dataset, such as ImageNet [17], and use it to initialize the backbone for an R-CNN network. We then train the object detector on the source (labeled) domain using supervision for objects of known C^k classes, such as 80 COCO classes (Fig. 2A, top). We use the labeled data to train the shared feature extractor (backbone) together with a class-agnostic region proposal network (RPN), class-agnostic box regression head, as well as a classification head for categorization of C^k classes. Specifically, for each image I, RPN is trained to predict a set of class-agnostic proposals $\{(\mathbf{b}, o)\}$, where $\mathbf{b} \in \mathbb{R}^4$ are predicted coordinates and $o \in \mathbb{R}$ is the objectness score. We then pass the proposals' feature representations, obtained from the backbone via the RoIAlign [29] mechanism, through an MLP (box head) and pass the resulting features $\hat{\mathbf{f}} \in \mathbb{R}^{\hat{F}}$ to RoI heads for the final predictions. We train a class-agnostic bounding box regression head to refine box location for each object $\hat{\mathbf{b}} = h^{bb}(\mathbf{b}, \hat{\mathbf{f}})$, and a classification box head to categorize each object into one of the K classes or a background: $\hat{c} = h^k(\hat{\mathbf{f}}) = \mathbf{W}^k\hat{\mathbf{f}}$, where $\hat{c} \in \mathbb{R}^{K+1}$ and $\mathbf{W}^k \in \mathbb{R}^{(K+1)\times\hat{F}}$. For training we use a standard fully-supervised R-CNN loss: $L_{sup} = L_{RPN} + L_{box} + L_{cls}$, consisting of the sum of RPN, localization, and classification losses respectively [60, 29].

Discovery on the target domain. During the following self-supervised (discovery) training phase on the target domain, we attach a secondary (novel) classification head h^n on top of RoI box features $\hat{\mathbf{f}}$ (Fig. 2B). The secondary classification head consists of an MLP and a linear classification layer that we detail below. We also modify the primary (known) classification head, by removing the background class from its weight matrix such that $\hat{\mathbf{W}}^k \in \mathbb{R}^{K\times\hat{F}}$. As the background class is removed at this phase, we aim to learn to categorize (*i.e.*, cluster) proposals. To obtain classification predictions at this phase, we pass RoI features through both h^k and h^n classification heads, concatenating their logits: $\hat{\mathbf{p}} = \text{softmax}([\mathbf{l}^k, \mathbf{l}^n]) = \text{softmax}([h^k(\hat{\mathbf{f}}), h^n(\hat{\mathbf{f}})]) \in \mathbb{R}^{K+N}$.

During training, we freeze all layers except the classification heads. We train the classification heads jointly using a self-supervised classification loss and supervised classification loss: $L_{disc} = L_{ss} + \alpha \cdot L_{cls}$, where α is the supervised loss scale coefficient. We show in ablations that keeping the downscaled ($\alpha < 1$) supervised classification loss in the loop effectively alleviates forgetting the weights for the known classes, while large α may create an unwanted bias and harm learning of novel classes.

For all regions, we calculate the self-supervised classification loss, that takes form of cross-entropy loss, evaluated on top of pseudo-labels, generated online for the current batch of proposals. We detail online pseudo-label generation in the subsection below. This loss encourages the network to categorize every region proposal consistently under multiple transformations and, importantly,

under a constraint that the marginal distribution of predicted class assignments for the last batches follows a prior long-tail distribution (Fig. 2A, bottom). We calculate the supervised loss only for annotation-matched region proposals. With such a joint training objective, our network is encouraged to learn to distinguish a large variety of classes present in the regions sampled by RPN, while retaining the capability to classify known objects.

Secondary classification head architecture. As the box features $\hat{\mathbf{f}}$ are kept frozen during the discovery phase, we use several non-linear projection layers in the secondary classification head: $\hat{\mathbf{f}}^n = g^n(\hat{\mathbf{f}})$, where g^n is a multi-layer perceptron with ReLU activations. Such learned transformation is designed to better disentangle feature representation of novel classes without affecting the RoI box features. On top of the projected features $\hat{\mathbf{f}}^n$ we employ a cosine classification layer, shown to be effective in the context of few-shot learning [22, 14, 74]: $\mathbf{l}^n = \frac{\mathbf{W}^n}{||\mathbf{W}^n||} \cdot \frac{\hat{\mathbf{f}}^n}{||\hat{\mathbf{f}}^n||}$, where $\mathbf{l}^n \in \mathbb{R}^N$ are the predicted novel-class logits. We determine the number of novel classes (*i.e.* the number of novel classes, N, in the secondary head) empirically in (Sec. 4.2).

Self-supervision via constrained clustering. For self-supervised learning, we perform self-labeling via constrained clustering, where we assign a soft pseudo-label to each RPN proposal. We generate pseudo-labels $\mathbf{q}$ for all input images (labeled and unlabeled, $D^N \cup D^K$) that can span all classes, both known and novel: $\mathbf{q} \in \mathbb{R}^{(N+K)\times B}$, where B is the batch size. We use $\mathbf{q}$ to compute cross-entropy loss L_{ss} and train both classification heads. For pseudo-label generation standard clustering methods (*e.g.* k-means [52]) were shown to produce unstable results in domain of image classification [9, 10]. Asano *et al.* [2] instead propose to leverage an equipartitioning regularizer and [1] further extended the method to arbitrary prior distributions.

Following [1], we aim to minimize a clustering energy function (*i.e.*, cross-entropy loss): $E(\hat{\mathbf{p}}, \mathbf{q}) = \frac{1}{B}\sum_{i=1}^{B}\sum_{y}^{N+K} q(y|\mathbf{I}_i) \log \hat{p}(y|\mathbf{I}_i)$, where $\mathbf{q}$ are class labels (*i.e.*, pseudo-labels in our framework), and $\hat{\mathbf{p}} = \text{softmax}([\mathbf{l}^k, \mathbf{l}^n]) \in \mathbb{R}^{(N+K)\times B}$ are predicted class-probabilities. Note that both labels $\mathbf{q}$ and probabilities $\hat{\mathbf{p}}$ are subject to optimization. Similar to Asano *et al.* [2], we employ an alternating algorithm, where a constrained clustering method to generate $\mathbf{q}$ is followed by optimizing $\hat{\mathbf{p}}$, *i.e.*, during each training iteration, we first generate class pseudo-labels $\mathbf{q}$, and then use them to train the classification heads.

To generate class pseudo-labels, we first choose a target clusters' marginal probability distribution (*i.e.*, a prior distribution). Empirically, we obtained best results using a non-symmetrical lognormal distribution. This distribution is right-skewed, suitable for modeling the long-tail. We then generate such soft pseudo-labels $\mathbf{q}$ that (i) closely match the chosen marginals and (ii) minimize $E(\hat{\mathbf{p}}, \mathbf{q})$ for the given $\hat{\mathbf{p}}$. In [2, 1], it is shown that this can be posed as a constrained optimization problem and solved as an optimal transport problem using the fast online Sinkhorn-Knopp algorithm [16]. Having obtained the pseudo-labels $\mathbf{q}$, we fix them, calculate the loss $L_{ss} = E(\hat{\mathbf{p}}, \mathbf{q})$, and apply backpropagation to the classification heads' parameters.

Swapping multi-view cluster assignments. To enforce the network to output consistent classification predictions $\hat{\mathbf{p}}$, we employ supervisory signal from multiple augmentations [11]. Specifically, for each image from $D^N \cup D^K$, we first generate RPN proposals and refine their coordinates via class-agnostic classification head. We then generate two augmented views for each image (we discuss augmentations in the supplementary), and extract features for each generated proposal twice, from each of the view. Finally, we generate pseudo-labels for each view (set of features) individually, obtaining two sets of pseudo-labels $\mathbf{q}_i$, $i \in \{1, 2\}$ that correspond to the same proposals. Instead of computing the losses for each view as $E(\hat{\mathbf{p}}_i, \mathbf{q}_i)$, we swap the resulting pseudo-labels $\mathbf{q}_i$ between views, and calculate losses as $E(\hat{\mathbf{p}}_i, \mathbf{q}_j)$, where $i \neq j$, *i.e.* we use pseudo-labels obtained for one view, as labels to the other. We calculate the total loss as the average of the losses for both views: $\hat{L}_{ss} = (E(\hat{\mathbf{p}}_1, \mathbf{q}_2) + E(\hat{\mathbf{p}}_2, \mathbf{q}_1))/2$. For each view, we use a separate feature memory module.

Memory module. To generate good-quality pseudo-labels for each mini-batch, a diverse set of samples is needed, in order to capture a wide variety of classes. A small set of features per mini-batch might lead to noisy cluster assignments. As we operate in scenarios with more than a thousand classes to discover, we introduce a memory module (*c.f.*, [31, 11]) to store proposals' RoI box features from the last batches and ensure diverse set of samples when computing clustering assignments. As in [31, 11] we design the memory module as a queue, where at the end of each iteration we

replace the oldest set of stored features with the features from the current batch. During pseudo-labeling generation we pass the extended set of features to the Sinkhorn algorithm. Specifically, after extracting current-batch proposals' features, we concatenate them with the features stored in the memory module $\mathbf{M}$, compute their logits, and as an input to the Sinkhorn algorithm use: $\hat{\mathbf{l}} = [\hat{\mathbf{l}}^k, \hat{\mathbf{l}}^n] = [h^k([\hat{\mathbf{f}}, \mathbf{M}]), h^n([\hat{\mathbf{f}}, \mathbf{M}])]$, where $\mathbf{M} \in \mathbb{R}^{M_{sz} \times \hat{F}}$ and M_{sz} is the number of features stored in the memory. Such modified procedure generates $B + M_{sz}$ pseudo-labels, for both current-batch proposals and memory features. We discard pseudo-labels generated for memory features and use only those B generated for the current-batch proposals, *i.e.*, we use the memory module only to improve the quality of current-batch pseudo-labels.

Inference. During inference (Fig. 2B), for each image, we extract a set of RPN proposals $\{(\mathbf{b}, o)\}$ and pass them through RoI heads for localization refinement and class predictions. For classification, we use both heads h^k, h^n and calculate output class probabilities as $\hat{\mathbf{p}} = \text{softmax}([\mathbf{l}^k, \mathbf{l}^n])$.

Semantic class assignment for evaluation. For quantitative evaluation, as our framework provides arbitrary class predictions (*i.e.* not directly mapped to novel classes in the target domain), we have to obtain a mapping of predicted classes with ground truth labels. We follow prior art [28, 27, 21, 75] and propose a similar evaluation methodology. Having trained the network, first, we generate classification predictions for each GT annotation in the validation dataset. Then, we use the Hungarian algorithm [41] to obtain an one-to-one mapping between GT labels and validation class predictions. We ignore instances of predicted classes that were not mapped to GT classes. Then, we proceed to the standard R-CNN inference pass, where we apply the generated mapping and follow the standard R-CNN post-processing and evaluation [48].

4 Experimental Evaluation

In this section, we discuss our evaluation test-bed, including datasets, evaluation settings, and metrics (Sec. 4.1). In Sec. 4.2 we justify our main design decisions by studying NCDL performance in a well-controlled setting that closely follows a real-world scenario (training on a source *labeled* and target *unlabeled* dataset). We then compare our method's performance to several baselines (Sec. 4.3) and, finally, demonstrate that our method is applicable beyond our evaluation test-bed (Sec. 4.4).

4.1 Evaluation setting

COCO$_{half}$ + LVIS. We re-purpose COCO 2017 [48] and LVIS v1 [26] datasets for running ablations and comparisons with baselines and prior art. We use annotations for 80 COCO classes during the supervised training phase, aiming at further classifying additional $1000+$ LVIS classes. COCO dataset contains $123K$ images with modal bounding box and segmentation mask annotations for 80 classes. The LVIS dataset contains a subset of $120K$ COCO images with annotations for 1203 classes that include all classes from COCO. For our experiments, we follow LVIS training and validation splits, resulting in $100K$ training images and $20K$ validation images. To model the NCDL setup, we further split $100K$ training images in half and use only $50K$ images with 80-class annotations from COCO during the supervised training phase. We refer to this dataset as COCO$_{half}$. We treat the rest of $50K$ images as additional unlabeled data used during the discovery phase.

LVIS + Visual Genome. We perform a large-scale generalization experiment by using annotations for 1203 LVIS classes during the supervised learning (*i.e.*, treating LVIS classes as *labeled*), aiming to learn to discover extra $2700+$ classes from the Visual Genome (VG) v1.4 dataset [40] (*i.e.*, treating VG classes as *unlabeled*). We provide dataset details and splits (that ensure LVIS and VG class vocabularies do not overlap) in the supplementary. We note that annotations provided in VG are not exhaustive per class and many of its classes are abstract or semantically overlapping both within VG and when compared to LVIS. We thus mainly focus on qualitative results.

Implementation details. We base our model on Mask R-CNN [29] and FPN [47] implementations from Detectron2 [76]. We give all the implementation details in the supplementary material.

Evaluation metrics. For quantitative evaluation, we follow [28, 27, 21, 75] to match predicted cluster assignments with annotated semantic classes. We follow common practice and report mean

Table 1: Impact of supervised loss strength. We check the results of our method as function of strength of the supervised loss.

Sup. loss coeff.	mAP$_{all}$	mAP$_{known}$	mAP$_{novel}$
0	6.59	16.58	5.77
0.1	6.90	22.26	5.64
0.5	**6.92**	**25.00**	**5.42**
1.0	6.35	25.81	4.75

Table 2: Sensitivity to number of classes. We vary the number of novel classes set during the discovery phase.

# novel classes	mAP$_{all}$	mAP$_{known}$	mAP$_{novel}$
1000	5.42	23.24	3.95
3000	**6.92**	**25.00**	**5.42**
5000	6.24	24.91	4.70

average precision (mAP@[.5:.95]) [48]. For qualitative results, we include instances of classes that were matched to annotated classes only. We detail our evaluation procedure in the supplementary.

Baselines. We compare our end-to-end trainable detector with state-of-the-art approach by Weng *et al.* [75], NCD methods ORCA [7], UNO[21], and k-means [52] baseline that all operate on cropped object proposals, provided by RPN. For NCD methods we use labeled instance annotations from COCO$_{half}$ dataset as labeled images pool and cropped RPN proposals as unlabeled images pool. For proposals extraction we train R-CNN on COCO$_{half}$ under the same conditions as our base network, to ensure comparable evaluation conditions. Having trained the NCD classifiers, we use the same R-CNN to generate object proposals for the validation images, apply the classifiers on top of the proposals, and continue with the standard R-CNN post-processing steps. We provide more details on baselines in the supplementary.

4.2 Model ablations

We ablate single and multi-phase training, the impact of keeping supervised loss in the loop, the number of novel categories, pseudo-labeling prior, model architecture decisions, and backbone pretraining. In supplementary we provide further results on experiments with hyperparameters for proposals extractor and how class-agnostic heads affect the performance of the supervised baseline.

Single-phase training and unfreezing R-CNN components during discovery. As shown in Table 3 we observe that training the network with both supervised and discovery losses from the beginning, summed up without any scaling, leads to divergence. We hypothesize that this is due to noisy gradients from our self-supervised signal, with R-CNN being very sensitive to the choice of losses and hyperparameters [60]. To further support this claim, we experiment with a standard supervised phase followed by a discovery phase without freezing any of the components. We observe that such a setup also leads to divergence. Unfreezing only the RoI heads during the discovery phase, however, does not lead to divergence, yet reduces the score by 1.32 mAP. We thus keep all the components frozen during the discovery phase, apart from known and novel classification heads.

The strength of the supervised loss. We ablate the strength of the supervised classification loss that is used along with the discovery loss during the discovery phase. We vary the strength from 0 (no supervised loss) to 1 (no downscaling) and provide the results in Table 1. We observe that excluding supervised loss completely degrades the performance of the network for the known classes and benefits the performance of the novel classes, while eeping the full supervised loss introduces bias towards the known classes. This indicates that there is a tradeoff between performance on known and novel classes. We set the strength of the supervised loss to 0.5 in all experiments.

Sensitivity to the number of novel classes. In Table 2 we experiment with varying the number of novel classes we use for the novel classification head. We observe that the best number of classes for the COCO + LVIS setup is 3000, while other choices result in lower scores. We conjecture that such scores are sensitive to the semantic classes defined by the target dataset – with 3000 classes the model learns the granularity and grouping that matches closest to the LVIS classes, but that does not necessarily mean that a finer or coarser granularity categorization is worse. In supplementary we also experiment with attaching and training multiple heads jointly during the discovery phase and observe that scores slightly degrade in such setup but do not observe a significant drop in results.

Clustering prior. In Table 3 we experiment with using uniform prior distribution for class-marginals during pseudo-labels generation, as proposed in [2, 10]. We observe degradation of the score by 1.17 mAP, which confirms our hypothesis that a non-uniform lognormal prior distribution closer resembles a real-world long-tail class distribution. As an additional ablation, we experiment with

Table 3: Additional ablations. We experiment with removing individual components from the network and their impact on the overall performance.

Method	mAP_{all}	mAP_{known}	mAP_{novel}
RNCDL	**6.92**	**25.00**	**5.42**
Single-phase framework	n/a	n/a	n/a
Unfreeze all layers	n/a	n/a	n/a
Unfreeze RoI heads	5.60	20.55	4.37
Uniform class prior	5.75	24.11	4.23
k-means [52] pseudo-lab.	2.06	21.13	0.48
W/o projection MLP	5.30	26.13	3.58
W/o swapped assignments	6.74	26.02	5.14
W/o memory	2.83	24.19	1.06
W/o MoCo pretraining	4.77	16.92	3.77

Table 4: Comparison with state-of-the-art models. * as per open source code. † adapted to support known classes in target dataset. ‡ randomly initialized novel head.

Method	mAP_{all}	mAP_{known}	mAP_{novel}
Methods that operate on cropped proposals			
k-means [52]	1.33	15.61	0.14
Weng *et al.* [75]*	1.62	17.85	0.27
ORCA [7]	2.03	20.57	0.49
UNO [21]‡	2.18	21.09	0.61
Methods that operate on FPN-based features			
k-means [52]	1.55	17.77	0.20
RNCDL w/ random init.‡	1.95	23.51	0.17
RNCDL	**6.92**	**25.00**	**5.42**
Fully-supervised	18.47	39.38	16.74

using k-means for online pseudo-labels generation instead of constrained clustering and observe a significant degradation of the score by 4.86 mAP, especially notable for the novel classes.

Embedding projector and swapped assignments. In Table 3 we demonstrate the effect of adding an MLP feature projector on top of RoI box features for novel heads classification. We observe that the performance of the network improves by 1.62 mAP. In the same table, we show that swapping pseudo-labels between proposals under multiple augmentations helps boost the score by 0.18 mAP.

Feature memory module. We demonstrate the importance of memory module in Table 3. In all our experiments we used the memory size of 100 batches, unless stated otherwise. We observe that removing the memory module significantly drops the performance by 4.09 mAP, especially for the novel classes. This highlights that for an efficient pseudo-labels generation it is crucial to use a set of features representative of the full feature space that we lack in the absence of memory.

Self-supervised pre-training matters. We compare the results of our framework under different backbone initialization employed for the supervised pre-training stage. For the randomly initialized backbone we set all layers as trainable, while during the MoCo initialization the first two convolutional blocks are kept frozen (*c.f.*, [31]). In Table 3 we observe that MoCo-initialized backbone improves the score by 2.15 mAP over randomly initialized backbone mode. We only pre-train our network using ImageNet dataset [17], and not on the source or target object detection dataset.

Proposals extractor hyperparameters. We observe that when extracting the RPN proposals for self-supervised learning during the discovery phase the absence of NMS yields the best results and the optimal number of proposals per image is 50. We thus use such parameters for all the experiments.

4.3 Comparison to prior work

In Table 4 we compare our method with k-means baselines and three recent state-of-the-art methods that we adapted to our scenario. We present the results for both known (COCO) and unknown (the rest of LVIS) classes. Our method significantly outperforms the previous best NCD method UNO [21], by reaching 4.74 higher overall mAP. We outperform UNO in both the known and especially novel classes, where we reach 4.81 higher mAP. We also significantly outperform approach by Weng *et al.* [75]. Our method is the only method to reach over 1 mAP in novel classes, where we reach 5.42 mAP. We believe the main advantages of our method to be the memory module, critical for effective self-supervision, and a backbone trained in an end-to-end manner with the aid of class-agnostic losses, frozen during the discovery phase. Further, our backbone benefits from high-level semantic features via FPN that is critical for the classification of small objects [47]. We provide more comparison details in the supplementary. We note that as our method heavily relies on the generated region proposals, we are limited by the performance of region proposal network (*i.e.*, we cannot detect objects, not detected by the RPN network). We do not update the backbone feature extractor during self-supervised training. Successfully doing so could potentially further boost the performance, including the localization and segmentation components.

a) COCO → LVIS b) LVIS → VisualGenome

Figure 3: Visualization of predictions for validation images of fully-supervised model and our RNCDL framework. We color the discovered novel classes in red.

a) COCO → LVIS b) LVIS → VisualGenome

Figure 4: Visualization of the common novel classes discovered in LVIS and VisualGenome datasets.

4.4 Cross-dataset Generalization and Qualitative Results

We analyze the generalization properties of our framework by training on LVIS and evaluating on Visual Genome (VG) dataset. We present quantitative results in Table 5.

In Figure 3 we provide qualitative results with novel classes discovered. In Figure 4 we also visualize the common discovered categories. Upon manual examination of the clusters from the LVIS + VG experiment, we observe that many of the novel classes are synonyms, hyponyms, or hypernyms of known classes that appear in the LVIS dataset.

Table 5: LVIS → VisualGenome comparisons with a fully-supervised method.

Method	mAP_{all}	mAP_{known}	mAP_{novel}
RNCDL w/ random init.	2.13	7.71	0.81
RNCDL	4.46	12.55	2.56
Fully-supervised	4.52	13.72	2.35

5 Conclusions

In this paper, we introduce an end-to-end RNCDL network for the task of novel class discovery, detection, and localization. Our model is a two-stage object detection network that is able to classify both instances of the labeled, known classes and those of unlabeled, novel classes. At the core of our framework is a self-supervision guided by features of region proposals and a constraint for the class assignments to follow a long-tail distribution. In our experiments we demonstrate a significant improvement over the previous state-of-the-art. Furthermore, we demonstrate the ability to detect semantic classes without any supervision at a large scale on the Visual Genome dataset.

Broader Impact

The real world contains a vast number of object categories, but labeling them all is impractical and costly. Most deep learning models for object detection are trained on datasets that cover only a fixed and limited set of categories, and they cannot generalize to novel classes that are not seen during training. This limits their applicability in scenarios where objects of interest are unknown or rare. Therefore, there is a need for methods that can discover and detect novel classes without relying on labeled data supervision. In this work, we present a novel end-to-end method that addresses this problem and significantly improves over previous approaches. We hope our work will inspire more research on this challenging problem and lead to more general and robust object detection models.

Acknowledgments. LLT would like to acknowledge the Humboldt Foundation through the Sofja Kovalevskaja Award. DR would like to acknowledge the CMU Argo AI Center for Autonomous Vehicle Research. We thank the anonymous reviewers for their feedback.

References

[1] Yuki M. Asano, Mandela Patrick, Christian Rupprecht, and Andrea Vedaldi. Labelling unlabelled videos from scratch with multi-modal self-supervision. In *NeurIPS*, 2020.

[2] Yuki M. Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *ICLR*, 2020.

[3] Ankan Bansal, Karan Sikka, Gaurav Sharma, Rama Chellappa, and Ajay Divakaran. Zero-shot object detection. In *ECCV*, 2018.

[4] Abhijit Bendale and Terrance Boult. Towards open world recognition. In *CVPR*, 2015.

[5] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *CVPR*, 2016.

[6] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *JMLR*, 3:993–1022, 2003.

[7] Kaidi Cao, Maria Brbić, and Jure Leskovec. Open-world semi-supervised learning. In *ICLR*, 2022.

[8] Yuhang Cao, Jiaqi Wang, Ying Jin, Tong Wu, Kai Chen, Ziwei Liu, and Dahua Lin. Few-shot object detection via association and discrimination. In *NeurIPS*, 2021.

[9] Mathilde Caron, Piotr Bojanowski, Armand Joulin, and Matthijs Douze. Deep clustering for unsupervised learning of visual features. In *ECCV*, 2018.

[10] Mathilde Caron, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Unsupervised pre-training of image features on non-curated data. In *ICCV*, 2019.

[11] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *NeurIPS*, 2020.

[12] Jianlong Chang, Lingfeng Wang, Gaofeng Meng, Shiming Xiang, and Chunhong Pan. Deep adaptive image clustering. In *ICCV*, 2017.

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *ICML*, 2020.

[14] Wei-Yu Chen, Yen-Cheng Liu, Zsolt Kira, Yu-Chiang Wang, and Jia-Bin Huang. A closer look at few-shot classification. In *ICML*, 2019.

[15] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020.

[16] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *NeurIPS*, 2013.

[17] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[18] Akshay Raj Dhamija, Manuel Günther, and Terrance E Boult. Reducing network agnostophobia. In *NeurIPS*, 2018.

[19] M. Everingham, L. Van Gool, C.K.I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *IJCV*, 88(2):303–338, 2010.

[20] Zhibo Fan, Yuchen Ma, Zeming Li, and Jian Sun. Generalized few-shot object detection without forgetting. In *CVPR*, 2021.

[21] Enrico Fini, Enver Sangineto, Stéphane Lathuilière, Zhun Zhong, Moin Nabi, and Elisa Ricci. A unified objective for novel class discovery. In *ICCV*, 2021.

[22] Spyros Gidaris and Nikos Komodakis. Dynamic few-shot visual learning without forgetting. In *CVPR*, 2018.

[23] Ross Girshick. Fast r-cnn. In *ICCV*, 2015.

[24] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[25] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. Technical report, California Institute of Technology, 2007.

[26] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019.

[27] Kai Han, Sylvestre-Alvise Rebuffi, Sebastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. Automatically discovering and learning new visual categories with ranking statistics. In *ICLR*, 2020.

[28] Kai Han, Andrea Vedaldi, and Andrew Zisserman. Learning to discover novel visual categories via deep transfer clustering. In *ICCV*, 2019.

[29] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *ICCV*, 2017.

[30] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[31] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross B. Girshick. Momentum contrast for unsupervised visual representation learning. In *CVPR*, 2020.

[32] Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. Deep image category discovery using a transferred similarity function. *arXiv preprint arXiv:1612.01253*, 2016.

[33] Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. Learning to cluster in order to transfer across domains and tasks. In *ICLR*, 2018.

[34] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to Segment Every Thing. In *CVPR*, 2018.

[35] Jaedong Hwang, Seoung Wug Oh, Joon-Young Lee, and Bohyung Han. Exemplar-based open-set panoptic segmentation network. In *CVPR*, 2021.

[36] Lalit P Jain, Walter J Scheirer, and Terrance E Boult. Multi-class open set recognition using probability of inclusion. In *ECCV*, 2014.

[37] Xuhui Jia, Kai Han, Yukun Zhu, and Bradley Green. Joint representation learning and novel category discovery on single-and multi-modal data. In *ICCV*, 2021.

[38] K J Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N Balasubramanian. Towards open world object detection. In *CVPR*, 2021.

[39] Bingyi Kang, Zhuang Liu, Xin Wang, Fisher Yu, Jiashi Feng, and Trevor Darrell. Few-shot object detection via feature reweighting. In *ICCV*, 2019.

[40] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *IJCV*, 2017.

[41] H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955.

[42] Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, et al. The open images dataset v4. *IJCV*, 128(7):1956–1981, 2020.

[43] Suha Kwak, Minsu Cho, Ivan Laptev, Jean Ponce, and Cordelia Schmid. Unsupervised object discovery and tracking in video collections. In *ICCV*, 2015.

[44] Yong Jae Lee and Kristen Grauman. Object-graphs for context-aware visual category discovery. In *CVPR*, 2010.

[45] Yong Jae Lee and Kristen Grauman. Learning the easy things first: Self-paced visual category discovery. In *CVPR*, 2011.

[46] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. Pytorch distributed: Experiences on accelerating data parallel training. *arXiv preprint arXiv:2006.15704*, 2020.

[47] Tsung-Yi Lin, Piotr Dollár, Ross B. Girshick, Kaiming He, Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. In *CVPR*, 2017.

[48] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

[49] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *IJCV*, 128(2):261–318, 2020.

[50] Ziwei Liu, Zhongqi Miao, Xiaohang Zhan, Jiayun Wang, Boqing Gong, and Stella X Yu. Large-scale long-tailed recognition in an open world. In *CVPR*, 2019.

[51] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. In *ICLR*, 2017.

[52] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proc. Fifth Berkeley Symp. on Math. Statist. and Prob., Vol. 1*, 1967.

[53] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. In *ICLR*, 2018.

[54] Dimity Miller, Lachlan Nicholson, Feras Dayoub, and Niko Sünderhauf. Dropout sampling for robust object detection in open-set conditions. In *ICRA*, 2018.

[55] George A Miller. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[56] Aljoša Ošep, Paul Voigtlaender, Jonathon Luiten, Stefan Breuers, and Bastian Leibe. Large-scale object mining for object discovery from unlabeled video. In *ICRA*, 2019.

[57] Aljoša Ošep, Paul Voigtlaender, Mark Weber, Jonathon Luiten, and Bastian Leibe. 4d generic video object proposals. In *ICRA*, 2020.

[58] Shafin Rahman, Salman Hameed Khan, and Fatih Porikli. Zero-shot object detection: Learning to simultaneously recognize and localize novel concepts. *ACCV*, 2018.

[59] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *CVPR*, 2017.

[60] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[61] Michael Rubinstein, Armand Joulin, Johannes Kopf, and Ce Liu. Unsupervised joint object discovery and segmentation in internet images. In *CVPR*, 2013.

[62] Bryan C. Russell, William T. Freeman, Alexei A. Efros, Josef Sivic, and Andrew Zisserman. Using multiple segmentations to discover objects and their extent in image collections. In *CVPR*, 2006.

[63] Walter J Scheirer, Anderson de Rezende Rocha, Archana Sapkota, and Terrance E Boult. Toward open set recognition. *PAMI*, 35(7):1757–1772, 2012.

[64] Walter J Scheirer, Lalit P Jain, and Terrance E Boult. Probability models for open set recognition. *PAMI*, 36(11):2317–2324, 2014.

[65] Jamie Shotton, John M. Winn, Carsten Rother, and Antonio Criminisi. TextonBoost for image understanding: Multi-class object recognition and segmentation by jointly modeling texture, layout, and context. *IJCV*, 81(1):2–23, 2009.

[66] Josef Sivic, Bryan C. Russell, Andrew Zisserman, William T. Freeman, and Alexei A. Efros. Unsupervised discovery of visual object class hierarchies. In *CVPR*, 2008.

[67] Tinne Tuytelaars, Christoph H. Lampert, Matthew B. Blaschko, and Wray Buntine. Unsupervised object discovery: A comparison. *IJCV*, 88:284–302, 2010.

[68] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: Learning to classify images without labels. In *ECCV*, 2020.

[69] Huy V. Vo, Francis Bach, Minsu Cho, Kai Han, Yann LeCun, Patrick Pérez, and Jean Ponce. Unsupervised image matching and object discovery as optimization. In *CVPR*, 2019.

[70] Huy V Vo, Patrick Pérez, and Jean Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *ECCV*, 2020.

[71] Van Huy Vo, Elena Sizikova, Cordelia Schmid, Patrick Pérez, and Jean Ponce. Large-scale unsupervised object discovery. In *NeurIPS*, 2021.

[72] Feng Wang, Jian Cheng, Weiyang Liu, and Haijun Liu. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930, 2018.

[73] Xin Wang, Thomas E Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. *arXiv preprint arXiv:2003.06957*, 2020.

[74] Xin Wang, Thomas E. Huang, Trevor Darrell, Joseph E Gonzalez, and Fisher Yu. Frustratingly simple few-shot object detection. In *ICML*, 2020.

[75] Zhenzhen Weng, Mehmet Giray Ogut, Shai Limonchik, and Serena Yeung. Unsupervised discovery of the long-tail in instance segmentation using hierarchical self-supervision. In *CVPR*, 2021.

[76] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. `https://github.com/facebookresearch/detectron2`, 2019.

[77] Yongqin Xian, Christoph H. Lampert, Bernt Schiele, and Zeynep Akata. Zero-shot learning - a comprehensive evaluation of the good, the bad and the ugly. *PAMI*, 2018.

[78] Junyuan Xie, Ross Girshick, and Ali Farhadi. Unsupervised deep embedding for clustering analysis. In *ICML*, 2016.

[79] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, 2018.

[80] Bingchen Zhao and Kai Han. Novel visual category discovery with dual ranking statistics and mutual knowledge distillation. In *NeurIPS*, 2021.

[81] Zhun Zhong, Enrico Fini, Subhankar Roy, Zhiming Luo, Elisa Ricci, and Nicu Sebe. Neighborhood contrastive learning for novel class discovery. In *CVPR*, 2021.

[82] Zhun Zhong, Linchao Zhu, Zhiming Luo, Shaozi Li, Yi Yang, and Nicu Sebe. Openmix: Reviving known knowledge for discovering novel visual categories in an open world. In *CVPR*, 2021.

[83] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Simple multi-dataset detection. In *CVPR*, 2022.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [No]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]

A Abstract

In this supplementary, we provide implementation details (Section B), R-CNN configuration details (Section C), details on LVIS and Visual Genome datasets (Section D), results for additional ablations (Section E), details on reproducing the prior works (Section F), provide segmentation and scores of higher granularity for the best-scoring models (Section G), and additional qualitative examples (Section H).

B Implementation details

Our model is based on Mask R-CNN [29] configuration from Detectron2 [76] with ResNet50-based [30] FPN [47] backbone. Different from the standard configuration, we use AMP [53], SyncBatchNorm [79], MoCo [31] for backbone initialization, and class-agnostic localization and mask heads. In section C we provide more details and analysis of the introduced modifications.

During the supervised training phase, for all experiments, we follow Detectron2 and use a batch size of 16, train all networks using SGD optimizer for $180K$ iterations, and use random resize and random flip augmentations. We linearly increase the learning rate from 10^{-3} to 10^{-2} for the first $1K$ iterations and decrease it tenfold at iterations $120K$ and $160K$. When switching between COCO$_{half}$ + LVIS and LVIS + VG setups, architecture-wise we only change the number of target classes in the known classification head.

During the discovery training phase, we train for $15K$ iterations with a learning rate of 10^{-2} and follow a cosine decay schedule with linear ramp-up. The learning rate is linearly increased from 10^{-5} to 10^{-2} for the first $3K$ epochs and then decayed with a cosine decay schedule [51] to 10^{-3}.

For the supervised forward pass during the discovery training phase we do not change the hyperparameters of the R-CNN and use the exact same model configuration with the same batch size, base learning rate, and set of input images (re-shuffled). However, we remove the background (*unmatched*) RoI proposals from supervised pass training and keep only proposals that got matched to annotations. We also use a supervised loss scale coefficient of 0.5, which results in an effective learning rate for the supervised loss to be twice smaller than that of the discovery loss.

For the discovery forward pass, we use a separate pool of unlabeled images (that include images from the labeled pool). For RPN hyperparameters we extract 50 proposals per image and do not use NMS. We use a memory module with a size of $20K = 100 \cdot 4 \cdot 50$ features per GPU, where 100 is its size (in the number of batches stored), 4 is the per-GPU batch size, and 50 is the upper limit of the number of RPN proposals extracted per image in the batch. As the memory is empty at the beginning of training, we start training only after 150 iterations (batches) to allow it to get filled with features. We use a number of iterations slightly higher than the memory size to account for images that have less than 50 proposals available. For self-supervision prior, similarly to [1], we let marginals follow $\mathrm{Lognormal}(1, 0.5) \cdot \frac{M}{N}$, where M is the number of samples used for Sinkhorn and N is the total number of classes. In our experiments, M is calculated dynamically, based on the number of proposals in the current batch, and, at most, equals $20.2K$ (where $20K$ and 200 are the memory size and the maximum number of proposals in the batch respectively). For N we use 3080 classes in the COCO$_{half}$ + LVIS setup and $4203 = 1203 + 3000$ classes in the LVIS + VG setup. We set $\lambda = 20$ for the Sinkhorn algorithm. To obtain multi-view features for each of the proposals, we use augmentations similar to the ones from SimCLR [13]. Specifically, we use weaker versions of random color distortion (color jitter, grayscale), random Gaussian blur, and random resizing. We do not use random cropping to avoid cropping out some of the proposals only for one of the views. We use a batch size of 16 that is split across 4 GPUs, resulting in a per-GPU batch size of 4. We use $torch.distributed$ backend [46] for training, with the memory module and pseudo-labels being local to each of the 4 GPU processes.

For evaluation on LVIS and VG datasets, we follow LVIS evaluation specification [26] and use up to 300 top-scoring predictions with a per-prediction confidence cutoff of $1e^{-4}$. We use 4x NVIDIA A40 GPUs for all experiments.

C R-CNN configuration details

In Table C we provide details on modifications introduced to the Detectron2's default configuration that we used for R-CNN models. We provide scores for the fully-supervised models trained on COCO$_{half}$ and LVIS datasets. The models marked in bold were used for initialization during the discovery phases in COCO$_{half}$ + LVIS and LVIS + VG setups respectively. To the default configuration, we first introduced AMP [53] to allow for training larger models during the discovery phase and SyncBatchNorm [79] to stabilize training. Then, we replaced supervised ImageNet [17] initialization for the backbone with self-supervised initialization. We avoided using supervised initialization to ensure that no supervision was given for the novel classes that we aim to discover, as some of the ImageNet classes overlap semantically with classes from LVIS and VG. Specifically, we use MoCo v2 [15] applied on ImageNet dataset [17] and use weights from a public repository provided by the authors of the paper from: `https://github.com/facebookresearch/moco`. Finally, we switch to class-agnostic localization & mask heads.

Table C1: **Performance of fully-supervised models on COCO$_{half}$ and LVIS datasets**. In bold we highlight the configuration used for the supervised training phase and the fully-supervised baseline.

Model	COCO$_{half}$ mAP	LVIS mAP
Default Detectron Mask R-CNN FPN Res50	34.92	18.87
+ add AMP, SyncBN	34.07	18.03
+ replace ImageNet w/ MoCo init	35.78	18.85
+ use class-agnostic heads	**35.69**	**18.47**

D LVIS + Visual Genome setup

For the LVIS + Visual Genome (VG) setup, we use annotations from the LVIS v1 [26] dataset and aim to discover novel classes present in the Visual Genome (VG) v1.4 dataset [40]. The LVIS dataset contains $120K$ images with annotations for 1203 classes. Its training and validation splits contain $100K$ and $20K$ images respectively. VG dataset contains $108K$ images with annotations for more than $7K$ categories mapped to WordNet [55] synsets. $50K$ of VG images overlap with LVIS dataset images. To simplify the training and evaluation of experiments, we define the validation split of the VG dataset as a set of images from the LVIS validation split that also appear in VG. This results in $100K$ and $8K$ images for VG training and validation splits respectively. For the supervised training phase, we use $100K$ images from the LVIS train split with LVIS annotations. During the discovery phase, for the unlabeled pool, we merge LVIS and VG training images, resulting in $158K$ images total. We use $8K$ images from the VG validation split for evaluation. Out of more than $7K$ synsets from VG, we keep only those that appear in both generated training and validation VG splits, resulting in 3367 classes in total. We observe that they include 641 LVIS classes, by using exact matching of Wordnet synsets of classes, leaving 2726 novel classes to discover. We note, however, that with such exact mapping we do not account for potential overlapping word senses. Moreover, annotations provided in VG are not exhaustive per class and many of its classes are abstract. As a result, many categories of the resulting dataset may be semantically overlapping and we note that quantitative mAP results provided may not reflect the *real* model's performance.

E Additional RNCDL ablations

NMS coefficient for proposals. In Table E1a, we ablate the strength NMS used by RPN to filter out class-agnostic overlapping proposals during the discovery phase. Note that this only concerns the coefficient used for filtering the proposals generated for the discovery phase training and not the NMS used during inference. We observe that a larger (weaker) NMS yields better results and the absence of NMS yields the best results. This demonstrates that the network benefits from observing more proposals per image during training, even if they highly overlap.

Table E1: **Ablations for RPN hyperparameters, memory size, and multi-head setup.** We check the results of our method as function of a) NMS coefficient used by RPN and RoI heads to generate class-agnostic proposals during the discovery phase, b) number of RPN proposals, c) memory size in number of mini-batches stored. In d) we demonstrate that it's possible to train the model in a multi-head setup with varying number of classes without substantial loss in performance. We train the model with multiple heads of 1000, 3000, and 5000 classes and then evaluate each of the heads separately.

NMS coeff.	mAP_{all}
0.8	5.76
0.9	6.77
0.99	6.84
no NMS	**6.92**

(a)

# proposals	mAP_{all}
20	6.49
50	**6.92**
100	6.86
150	6.75

(b)

# mem. batches	mAP_{all}
0	2.83
20	4.66
50	5.83
100	**6.92**

(c)

# novel classes	mAP_{all}	mAP_{known}	mAP_{novel}
1000	5.37	25.87	3.68
3000	6.34	25.56	4.76
5000	5.14	25.25	3.48

(d)

Number of proposals per image. We experiment with extracting a different number of RPN proposals per image during the discovery phase in Table E1b. We observe that the optimal number of proposals is 50 and having more proposals per image damages its performance.

Memory module size. We experiment with different memory module sizes, expressed in number of last mini-batches stored, in Table E1c. We observe a large jump in performance of 1.83 mAP when introducing memory module of 20 batches and that the afterwards the performance tends to grow as its size increases in a roughly log-linear fashion. Due to memory and time limitations we did not experiment with memory sizes larger than 100 and expect the models' performance to further improve with more features stored.

Sensitivity to the number of novel classes for a multi-head setup. In Table E1d we experiment with attaching and training multiple heads jointly during the discovery phase, where the total discovery loss is calculated as an average loss per head and the memory module is shared across heads. We observe that scores degrade in such a setup but do not observe a substantial drop in results. This suggests that for a practitioner it is possible to train a single framework with multiple heads, each with hypothesized number of categories to discover and then to use any of the heads for the desired grouping.

FC instead of cosine classification layer. We experiment with replacing the cosine classification layer [22, 14, 74] of the novel classification head with an FC layer and observe that the training diverges.

F Details on baselines and reproducing prior work

In this section, we provide implementation details for baselines and prior works that we compare with. For all the methods we use 3000 for the number of novel categories.

For comparison with Weng *et al.* [75] work, we trained their method in our setting using the code provided by authors available at `https://github.com/ZZWENG/longtail_segmentation`. As the method starts with a similar supervised training phase, for a fair comparison we used an R-CNN trained in exactly the same manner as in RNCDL experiments. We then followed the configuration from the authors and extract the top 50 proposals per image with an NMS of 0.75 before the clustering phase. *For the clustering phase we found multiple inconsistencies with the code provided by the authors and the original manuscript. Even after correspondence with the authors, we were unable to resolve all issues and re-produce results, comparable to those reported in [75]. We thus report results that best reflect the foregoing method according to the available official implementation.*

For comparison with NCD works, which operate on images tightly cropped to the object of interest, we used localization coordinates of annotations and generated proposals to obtain a set of image crops. Specifically, for *labeled* images pool we used annotations from COCO_{half} dataset and for *unlabeled* images pool we used predicted RPN proposals. We used these annotations and proposals to crop images to the box coordinates of each instance. This resulted in more than $350K$ labeled crops and $5M$ unlabeled crops. We then padded each crop to a squared shape using the mean pixel value of all crops and scaled them to the size of 224×224 to match ImageNet images specification. To extract

Table G1: Detailed results for the RCNDL model.

Classes	mAP	mAP_{50}	mAP_{75}	mAP_s	mAP_m	mAP_l	mAP_r	mAP_c	mAP_f
known	25.00	42.41	24.93	13.25	32.41	34.56	-	-	25.00
novel	5.42	8.00	5.81	2.62	6.05	9.40	8,02	5.75	3.55
all	6.92	10.63	7.27	3.57	8.31	12.22	8.02	5.75	7.74

(a) Detection scores for the RNCDL model in COCO_{half} + LVIS setup

Classes	mAP	mAP_{50}	mAP_{75}	mAP_s	mAP_m	mAP_l	mAP_r	mAP_c	mAP_f
known	25.21	40.69	26.12	11.23	31.77	39.98	-	-	25.21
novel	5.16	7.44	5.48	2.09	6.19	9.51	8.03	5.29	3.39
all	6.69	9.97	7.06	2.91	8.39	12.92	8.03	5.29	7.65

(b) Segmentation scores for the RNCDL model in COCO_{half} + LVIS setup

Classes	mAP	mAP_{50}	mAP_{75}	mAP_s	mAP_m	mAP_l	mAP_r	mAP_c	mAP_f
known	12.55	21.43	12.48	6.24	14.59	18.25	13.33	10.24	13.91
novel	2.56	3.95	2.65	1.39	2.04	3.15	3.02	2.83	1.72
all	4.46	7.28	4.52	2.47	4.79	6.22	3.34	4.05	5.69

(c) Detection scores for the RNCDL model in LVIS + VG setup

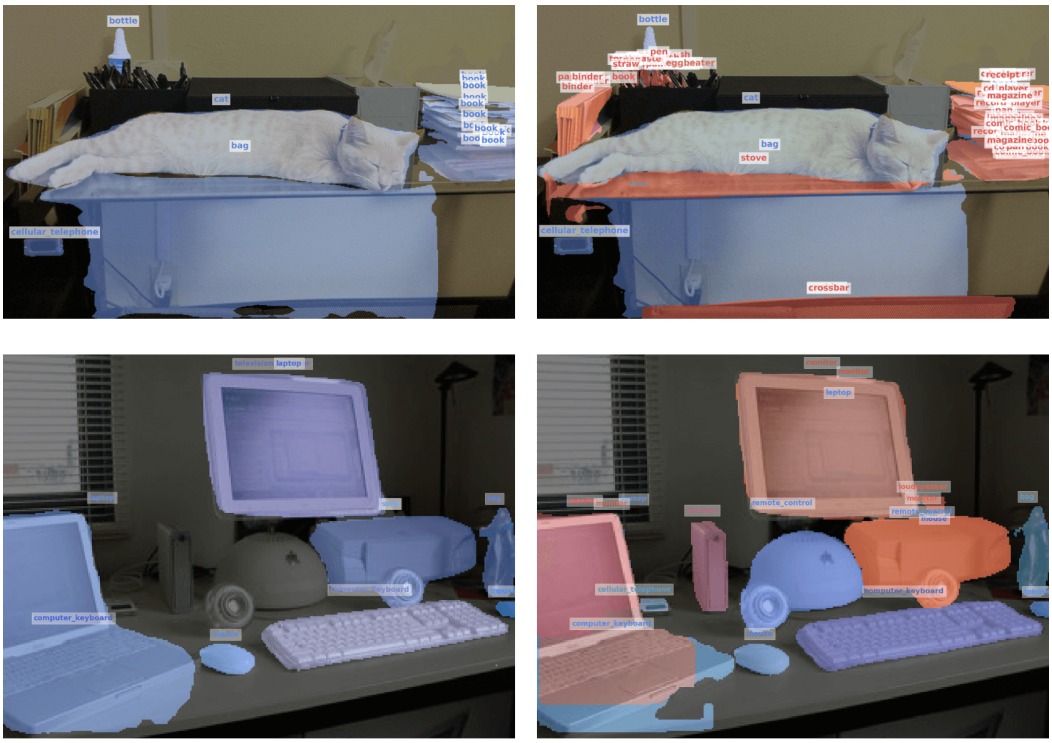
RPN proposals, we trained an R-CNN network on COCO_{half} with the exact same configuration as during the supervised training phase to ensure comparable evaluation conditions. We then applied state-of-the-art ORCA [7], UNO [21], and baseline k-means [52] method to the resulting labeled and unlabeled crops to train image classifiers. Having trained the classifiers, we followed a standard R-CNN inference loop, replacing the default FC classification head with each of the resulting classifiers. We do not alter any other R-CNN procedures, such as post-processing, NMS, etc.

ORCA [7] is directly applicable to such an image classification setup. We followed the configuration used by the authors for the ImageNet-100 dataset, but used a batch size of 1024 and trained for 10 epochs with a learning rate annealed by a factor of 10 at epochs 6 and 8. UNO [21] operates under an assumption that images from the unlabeled pool may only contain novel classes. We thus extended the method to allow for known classes to be present in the unlabeled pool as well. We implemented this by allowing pseudo-labels generated at each iteration to contain known classes. We followed the configuration used by the authors for the ImageNet dataset but used a batch size of 1024, and trained the supervised learning phase for 100 epochs and the discovery learning phase for 20 epochs. For k-means [52] we used a contrastive self-supervised method [31] to initialize a backbone and fine-tune it with images from the labeled pool. We then extracted features for all the images and learned cluster centroids. During the evaluation, as k-means does not output confidence, we classified each RPN proposal based on the closest cluster centroid and marked its confidence as 100%. In another k-means ablation, we applied the same methodology on top of RoI box head features of instances and proposals, without training a new backbone.

Finally, we also experimented with evaluating the RNCDL model right after attaching the novel head, with weights for MLP projection and cosine classification layers initialized randomly.

G Segmentation, per-area, and per-frequency scores details

In Table G1 we provide additional quantitative results for the best-scoring RNCDL models from COCO_{half} + LVIS and LVIS + VG setups. Specifically, we report mAP for `small`, `medium`, and `large` object instances, following COCO [48], and `rare`, `common`, and `frequent` object classes, as proposed in LVIS [26]. In addition, we report segmentation scores of our Mask R-CNN model for COCO_{half} + LVIS setup. We do not report segmentation scores for LVIS + VG setup as VG annotations do not contain segmentation masks.



a) COCO → LVIS

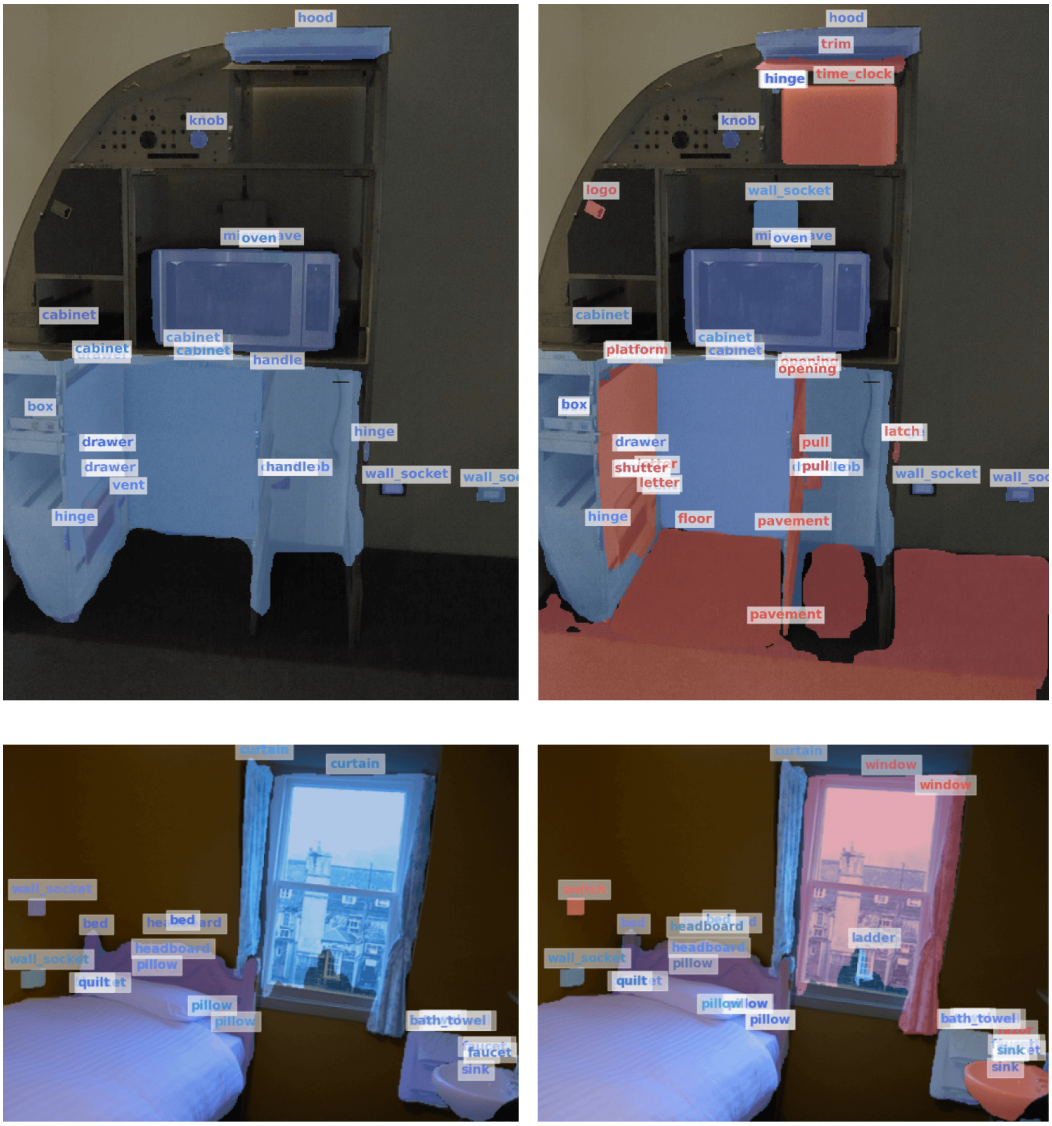
Figure H1: Visualization of predictions for validation images of fully-supervised model and our RNCDL framework in COCO$_{half}$ + LVIS setup. We color the discovered novel classes in red.

H Additional qualitative examples

We provide more qualitative results for both COCO$_{half}$ + LVIS and LVIS + VG setups. For both setups, we use larger models and train longer to obtain clusters of higher quality. Specifically, we train for $25K$ iterations, use a learning rate of 0.015, and use 8 GPUs with a batch size of 4 per GPU, making the effective batch size 32.

In Figures H1 and H2 we provide more qualitative examples for COCO$_half$ + LVIS and LVIS + VG setups respectively. We highlight novel classes, determined according to the mapping procedure, in red.

In Figure H3 we provide the top clusters ordered by confidence and size that were not mapped to one of the *known* classes by the mapping procedure. For the most confident clusters, we observe that many of them semantically overlap with the known annotated classes. This could indicate that the network splits some of the known classes into sub-clusters. For the most of the largest clusters, we observe that their content can be characterized as noisy and most images do not contain a clear object instance present. This could indicate that the network learns to group noisy proposals together.



b) LVIS → VisualGenome

Figure H2: Visualization of predictions for validation images of fully-supervised model and our RNCDL framework in LVIS + VG setup. We color the discovered novel classes in red.

a) COCO → LVIS unmatched clusters, ordered by mean confidence

b) COCO → LVIS unmatched clusters, ordered by size

c) LVIS → VG unmatched clusters, ordered by mean confidence

d) LVIS → VG unmatched clusters, ordered by size

Figure H3: The largest and the most confident clusters discovered.