Generalized Category Discovery

Appendices

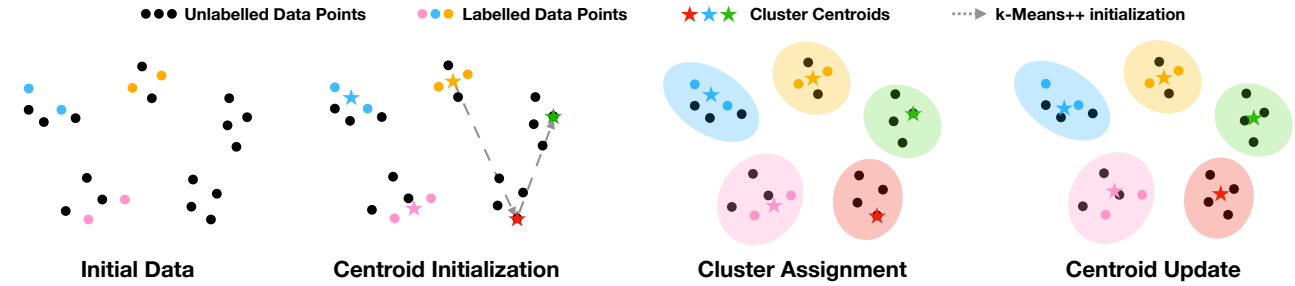

Figure 4. **Semi-supervised k-means algorithm shown for $k = 5$.** Given partially labelled data points (Initial Data), we first initialize $|\mathcal{Y}_\mathcal{L}| = 3$ centroids by the average of labelled data points in each labelled class (shown as colored dots). Starting from these centroids, we run k-means++ (dashed arrows) on the unlabelled data (black dots) to further obtain $|\mathcal{Y}_\mathcal{U} \setminus \mathcal{Y}_\mathcal{L}| = 2$ centroids (Centroid Initialization). Having obtained $k = 5$ centroids (colored stars), we assign each data point a cluster label by identifying its nearest centroid (Cluster Assignment), after which we can update the centroids by averaging all data points in each cluster (Centroid Update). We repeat the cycle of Cluster Assignment and Centroid Update iteratively, until the k-means algorithm converges. During each cycle, we force the labelled data points to follow their ground-truth label, *i.e.* all labelled points of the same class fall into the same cluster.

A. Dataset details

Here, we describe which classes constitute the 'Old' and 'New' categories in the Generalized Category Discovery setting, for each dataset used in this paper.

For all datasets, we sample 50% of the classes as 'Old' classes ($\mathcal{Y}_\mathcal{L}$) and keep the rest as 'New' ($\mathcal{Y}_\mathcal{U} \setminus \mathcal{Y}_\mathcal{L}$). The exception is CIFAR100, for which we use 80 classes as 'Old', following the novel category discovery literature. For the generic object recognition datasets, we use the first $|\mathcal{Y}_\mathcal{L}|$ classes (according to their class index) as 'Old' and use the rest as 'New'. For datasets in the Semantic Shift Benchmark suite, we use the data splits provided in [45]. For Herbarium19, to account for the long-tailed nature of the dataset, we randomly sample the 'Old' classes from the total list of classes. Further details on the splits can be found in Tab. 1 in the main paper and in the open-source code for this project.

B. Semi-supervised k-means

We elaborate on the semi-supervised k-means algorithm for GCD (from Sec. 3.1.2 of the main paper) in Fig. 4.

C. Estimating the number of classes

In Fig. 5, we provide motivation for our algorithm in Sec. 3.2 of the main paper, which is used to estimate the number of classes in the dataset ($k = |\mathcal{Y}_\mathcal{L} \cup \mathcal{Y}_\mathcal{U}|$). [1]

Specifically, we plot how the clustering accuracy on the labelled subset ($\mathcal{D}_\mathcal{L}$) changes as we run k-means with varying k on the entire dataset ($\mathcal{D}_\mathcal{L} \cup \mathcal{D}_\mathcal{U}$), for a range of datasets.

[1] Note that $|\mathcal{Y}_\mathcal{L} \cup \mathcal{Y}_\mathcal{U}| = |\mathcal{Y}_\mathcal{U}|$ as $|\mathcal{Y}_\mathcal{L} \subset \mathcal{Y}_\mathcal{U}|$. We use the former notation for clarity.

It can be seen that the ACC on the labelled subset follows an approximately bell-shaped function for all datasets. Furthermore, the function is maximized when k-means clustering is run with roughly the ground truth number of classes for each dataset, indicating that optimizing for this maximum is a reasonable way of identifying the total number of categories.

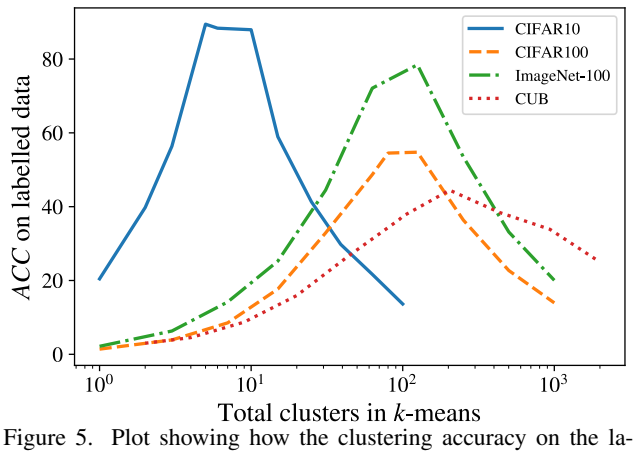

Figure 5. Plot showing how the clustering accuracy on the labelled subset ($\mathcal{D}_\mathcal{L}$) changes as we run k-means, with varying k, on the *entire* dataset ($\mathcal{D}_\mathcal{L} \cup \mathcal{D}_\mathcal{U}$). The ground-truth number of classes in CIFAR10, CIFAR100, ImageNet-100 and CUB are $[10, 100, 100, 200]$ respectively.

D. Results on FGVC-Aircraft

We run the baselines and our method on the third fine-grained dataset (FGVC-Aircraft [31]) from the Semantic

Shift Benchmark suite [45], reporting results in Tab. 6.

Table 6. Results on the FGVC-Aircraft [31] splits from the Semantic Shift Benchmark suite [45].

	FGVC-Aircraft		
Classes	All	Old	New
k-means [30]	16.0	14.4	16.8
RankStats+	26.9	36.4	22.2
UNO+	40.3	**56.4**	32.2
Ours	**45.0**	41.1	**46.9**

E. On Hungarian assignment and clustering accuracy

Here, we highlight a perhaps un-intuitive interaction between how the Hungarian assignment is performed and the reported ACC for all methods.

Background The purpose of the Hungarian algorithm [28] is to find the optimal matching between cluster indices predicted by the model and the ground truth labels. For instance, consider a toy case with two categories $\{1, 2\}$, with an 'Old' class $\mathcal{Y}_{\mathcal{L}} = \{1\}$ and a 'New' class $\mathcal{Y}_{\mathcal{U}} \setminus \mathcal{Y}_{\mathcal{L}} = \{2\}$. We further imagine the dataset to have 4 instances with ground truth labels $\{1, 1, 2, 2\}$ and a model which assigns them to clusters as $\{2, 2, 1, 1\}$. Note that 'a cluster' here could be either a cluster as referred to in the traditional sense (e.g as in our method, a cluster of points in feature space), or simply a group of instances which are predicted the same class label by a linear classifier (as with the baselines). The Hungarian algorithm ensures a reported ACC of 100% in this case (as the instances have been correctly clustered together) by solving the linear assignment between ground truth labels and cluster indices as $(1 \rightarrow 2, 2 \rightarrow 1)$.

Our evaluation protocol As stated in Sec. 4.1, we compute the Hungarian algorithm *once*, across all instances in the $\mathcal{D}_{\mathcal{U}}$. This gives us model predictions which are in the 'frame of reference' of the ground truth labels for all instances. Then, given these model predictions, we use the ground truth labels to select instances from the 'Old' and 'New' classes before computing the percentage of correct predictions (ACC) within these subsets.

Legacy NCD evaluation protocol In public implementations, we find that the novel category discovery literature [13, 18] computes ACC on these subsets differently. Specifically, the NCD literature *first* uses the ground truth labels to select which instances belong to the 'Old' and 'New' categories, before computing the Hungarian assignment on each subset *independently*. Importantly, this allows the same discovered cluster to be used *twice*. We suggest that this provides an overly optimistic view of model performance on the data subsets and does not quite reflect the true image recognition setting.

Illustration The left-hand diagram in Fig. 6 shows two discovered clusters ('Cluster 1' and 'Cluster 2') as well as the ground truth labels of their constituent instances (solid blue and yellow circles). The blue circles indicate images from an 'Old' class (e.g 'Dog') and the yellow circles indicate images from a 'New' one (e.g 'Cat'). The radii of the circles illustrate the number of instances.

On the right, we demonstrate the 'Legacy NCD' evaluation protocol for the 'Old' and 'New' data subsets, where the Hungarian assignment is computed twice, and independently on each data subset. For instance, the evaluation first looks at *only* 'Old' instances (blue circles) and the assignment algorithm allocates the old category to Cluster 1. The ACC is then computed as the number of 'Old' instances in Cluster 1 over the total number of 'Old' instances. However, subsequently, the evaluation looks *only* at the 'New' instances (yellow circles) and once again uses Cluster 1, assigning it to the new category. As such, Cluster 1 is used twice, allowing the evaluation to report high ACC on both data subsets.

In contrast, the Hungarian assignment can be computed over all instances, forcing the assignment of Cluster 2 to a ground truth category. In this way, the performance on one of the data subsets is necessarily lower. This is how we compute ACC on 'Old' and 'New' categories in this work, and show it as 'Ours' in the figure.

Finally, we show how ACC is computed over 'All' categories in the unlabelled set. This protocol is followed both in this work and in the novel category discovery literature.

F. Attention maps

Here, we expand upon the attention visualizations from Fig. 3. We first describe the process for constructing them, before providing further examples in Fig. 7.

Visualization construction The attention visualizations were constructed by considering how different attention heads, supporting the output `[CLS]` token, attended to different spatial locations. Specifically, consider the input to the final block of the ViT model, $\mathbf{X} \in \mathbb{R}^{(HW+1)\times D}$, corresponding to a feature for each of the HW patches fed to the model, plus a feature corresponding to the `[CLS]` token. Here, $HW = 14 \times 14 = 196$ patches at resolution 16×16 pixels. These features are passed to a multi-head self-attention ($MHSA$) layer which can be described as:

$$MHSA(\mathbf{X}) = [head_1, \ldots, head_h]\mathbf{W_0}, \tag{5}$$

where

$$head_j = Attention(\mathbf{X}\mathbf{W}_j^Q, \mathbf{X}\mathbf{W}_j^K, \mathbf{X}\mathbf{W}_j^V). \tag{6}$$

In other words, the layer comprises several attention heads ($h = 12$ in the ViT model) which each independently attend over the input features to the block. We refer to [44]

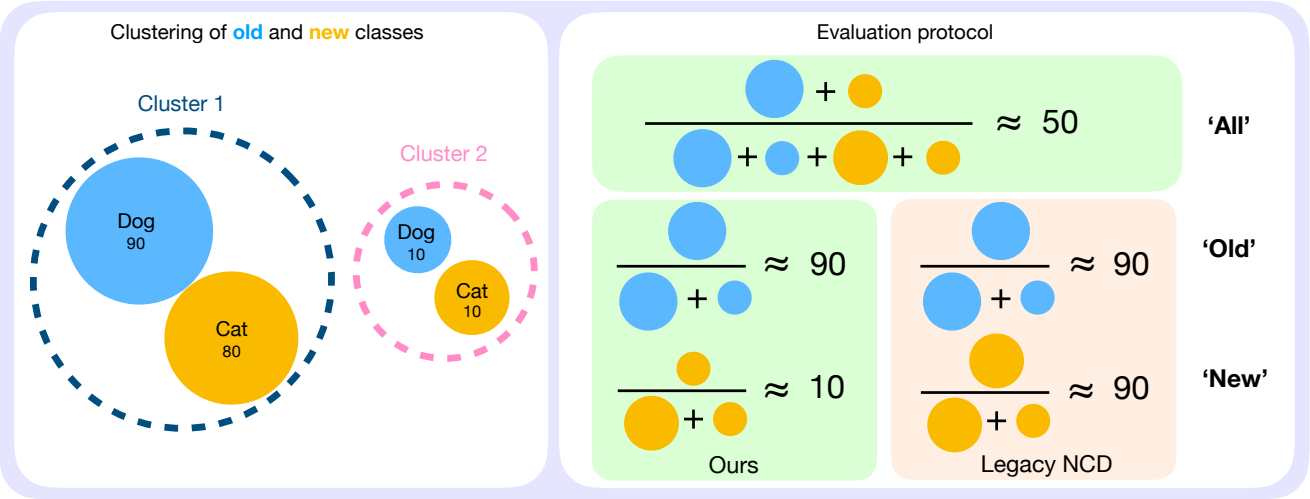

Figure 6. An illustration of how the Hungarian algorithm affects the final clustering ACC. The left-hand image shows two discovered clusters ('Cluster 1' and 'Cluster 2') and the ground truth labels of their constituent instances (solid colored circles). On the right, we show how ACC is computed if the Hungarian assignment is computed only once ('Ours') as well as how ACC is computed if the Hungarian assignment is computed independently for 'Old' and 'New' classes ('Legacy NCD').

for more details on the self-attention mechanism. We note that, for each feature $i \in \mathbf{X}$, an attention vector is generated for each head j, as $A_{ij} \in [0,1]^{HW+1}$ to describe how every head j relates each feature to every other feature.

We look at only the attention values for the `[CLS]` token, A_{0j}, and further only look at the elements which attend to spatial locations. We find that, while some heads have uninterpretable attention maps, certain heads specialize to attend to coherent semantic object parts.

Discussion on attention visualizations We provide further attention visualizations in Fig. 7. We show the *same* attention heads as shown in Fig. 3 (both for models trained with our method and for the original DINO model).

We first note that the DINO features often attend to salient object regions. For instance, 'Head 1' of the model often focuses on the wheel of the car (in the Stanford Cars examples), while 'Head 2' of the model generally attends to the heads of the birds (in the CUB examples). Overall, however, with the pre-trained DINO model, there is relatively little semantic consistency between the attention maps of a given head (*i.e.* within columns for each dataset).

In contrast, we find that the models trained with our approach specialize attention heads in semantically meaningful ways. The heads shown correspond to 'Windshield', 'Headlight' and 'Wheelhouse' for the Stanford Cars model, and 'Beak', 'Head' and 'Belly' for the CUB model. We find these maps to be relatively robust to nuisance factors such as pose and scale shift, as well as distracting objects in the image. We note a failure case (rightmost image, Row 2), as the 'Wheelhouse' attention head is forced to attend to miscellaneous regions of the car, as the car's wheelhouse is not visible in this image.

The ability of the model to identify and distinguish different semantic parts of an object is useful for the GCD task. Particularly in the fine-grained setting, the constituent set of parts of an object ('Head', 'Beak', 'Belly' *etc.* for the birds) transfer between 'Old' and 'New' classes. Thus, we suggest that the attention mechanism of the model allows it to generalize its understanding from the labelled 'Old' classes, and apply it to the unlabelled 'New' ones.

G. Broader impact and limitations

Our method assigns labels to images in an unsupervised manner, including discovering new labels. Even more than standard image classification methods, it should be used with care (e.g., by manually checking the results) in sensitive contexts, such as processing personal data.

We also note some practical limitations. First, we assume that there is no domain shift between the labelled and unlabelled subsets. For instance, we are not tackling the problem of a single model reliably classifying photographs and paintings of the same classes. Second, we do not consider the streaming setting (also known as continual learning): one would need to re-train the model from scratch as new data becomes available.

As for the data used in the experiments, we use standard third-party datasets in a manner compatible with their licenses. Some of these datasets contain Internet images that feature, often incidentally, people — see [2, 35, 47] for an in-depth discussion of the privacy implications.

Figure 7. Attention visualizations. Attention maps for the DINO model before (left) and after (right) fine-tuning with our approach on the Stanford Cars (top) and CUB (bottom) datasets. For each dataset, we show two rows of images from the 'Old' classes (solid green box) and two rows of images from the 'New' classes (dashed red box). Our model learns to specialize attention heads (shown as columns) to different semantically meaningful object parts, which can transfer between the 'Old' and 'New' categories.