Guided Cluster Aggregation: A Hierarchical Approach to Generalized Category Discovery

Jona Otholt, Christoph Meinel, Haojin Yang
Hasso Plattner Institute, University of Potsdam, Germany
{jona.otholt, christoph.meinel, haojin.yang}@hpi.de

Abstract

Despite advances in image recognition, recognizing novel categories in unlabeled data remains challenging for machine learning methods, even though humans can perform this task with ease. A recently developed setting to tackle this problem is Generalized Category Discovery (GCD), in which the task is to, given a labeled dataset, classify an unlabeled dataset, where the unlabeled dataset contains both known classes and novel classes that do not appear in the labeled data. Existing GCD methods mostly focus on learning strong image representations, on which they then apply a clustering algorithm such as k-means. Despite obtaining good performance, they do not fully exploit the potential of the learned features due to the simple nature of the clustering mechanism. To address this issue, we make use of the fact that local neighborhoods in self-supervised feature spaces are highly homogeneous. We leverage this observation to develop Guided Cluster Aggregation (GCA), a hierarchical approach that first groups the data into small clusters of high purity, then aggregates them into larger clusters. Experiments show that GCA outperforms semi-supervised k-means in most cases, especially in fine-grained classification tasks. Code available at https://github.com/J-L-O/guided-cluster-aggregation.

1. Introduction

Computer vision has made immense progress over the last ten years, mainly due to deep learning methods. While these methods may achieve human-like or even superhuman performance in some tasks, *e.g.* image classification [10], they still lack the breadth and adaptability that humans possess. For example, whereas humans can easily recognize previously unseen classes of objects, conventional supervised and semi-supervised classification methods are limited to the set of classes that are contained in the labeled data. To address this shortcoming, the problem setting of Generalized Category Discovery (GCD) has been proposed in [25]. In GCD, the task is to, given a labeled dataset containing known classes, correctly classify an unlabeled dataset, which contains both known categories, as well as novel categories that do not appear in the labeled set.

Most existing approaches decompose the task into two steps: representation learning and class assignment [22, 25, 29]. While recent improvements have mostly focused on improving the representations, achieving strong performance improvements, the class assignment has not received as much attention. Currently, many methods employ simple clustering methods such as semi-supervised k-means to perform the final class assignment [22, 25, 29], which does not take full advantage of the pretrained representations. Improving on previous methods using parametric classifiers was first explored in SimGCD [27], which jointly learns representations and a parametric classifier that outperforms k-means, showing that more research into parametric methods is worthwhile.

However, both semi-supervised k-means and SimGCD directly divide the data into the target categories, without any intermediate steps, which we believe does not make optimal use of the pretrained representations. As shown in [2], even a k-nearest neighbor classifier can achieve strong performance when working with strong pretrained image representations, which shows that the local neighborhood around a sample usually contains samples of a similar class. We thus hypothesize that it is much easier to achieve a high-purity clustering when the individual clusters are small and only cover local neighborhoods. To take advantage of this property, we propose Guided Cluster Aggregation (GCA), a hierarchical approach to GCD. We first establish many small local clusters using unsupervised clustering, then we aggregate the clusters into the target classes using the provided labels as well as neighbor relations between clusters. This way, we make effective use of the local structure while still obtaining high-level classifications.

To summarize, our contributions are as follows:

- We propose Guided Cluster Aggregation, a novel ap-

proach to Generalized Category Discovery that makes use of bottom-up cluster aggregation

- We show that our approach can be used with multiple pretrained models and outperforms the commonly used semi-supervised k-means, especially on fine-grained datasets
- Our approach not only categorizes samples into the target classes, it also yields a more fine-grained view that can be used to gain further insight into the data

2. Related Work

Previous research in the field of category discovery mostly focused on a slightly easier setting, called Novel Class Discovery (NCD). In this setting, the classes in the labeled and the unlabeled dataset are disjoint, so no unlabeled sample belongs to a known class. This setting was first formalized in [9], but some prior transfer learning approaches can also be used to solve this task [11, 12]. After these initial works, various performance improvements were introduced to NCD, including self-supervised pretraining [8], multi-view self-labeling [6], mixup augmentation [30], and meta-learning [3].

Recently, a more realistic setting called Generalized Category Discovery (GCD) was established by Han *et al*. in [25]. Here, the unlabeled data may contain both known and novel classes, making the task more challenging than NCD. In [25], this task is tackled by applying a mix of supervised and self-supervised learning to learn a strong feature extractor. The classification itself is done by applying a semi-supervised version of k-means clustering [20] on the extracted features. In contrast, OpenCon [24] employs end-to-end training where samples are assigned to the nearest class prototype. Additionally, OpenCon detects samples that are very likely to belong to novel classes and specifically improves their representations using an open-world contrastive loss. Concurrently, Cao *et al*. [1] derived the same setting by extending semi-supervised learning to also include novel classes in the unlabeled set. Their approach, called ORCA, directly learns the classifier in an end-to-end fashion by using ground-truth labels for the labeled data and pseudo-labels for the unlabeled data.

Later works in GCD mostly focus on improving the feature representation, while using semi-supervised k-means for the label assignment itself. DCCL [22] runs InfoMap [23] clustering on a sample adjacency matrix to generate conceptual labels, as well as concept prototypes. The representation of each sample is then pulled toward the corresponding concept prototype. Lastly, PromptCAL [29] uses visual prompt tuning to adapt frozen parts of the pretrained backbone to the new dataset. In addition, PromptCAL identifies pseudo-positive and pseudo-negative sample pairs that are fed into a SupCon-like loss.

The work most similar to ours is SimGCD [27], which investigates why previous works found k-means to be superior to parametric classifiers. An analysis of existing parametric methods shows that a major reason for their low performance is a bias toward the known classes, at the expense of novel class performance. They propose to mitigate this by applying mean-entropy regularization to the classifier output. Based on these findings, they demonstrate the viability of parametric classification in GCD by developing a baseline method, called SimGCD, which jointly learns both representation and classifier and outperforms k-means. While our work is also concerned with developing a parametric classification scheme for GCD, our method differs substantially from SimGCD. Whereas SimGCD uses pseudo-labels to learn the novel classes, our approach uses a bottom-up scheme that aggregates smaller clusters into bigger ones.

3. Method

In the Generalized Category Discovery (GCD) setting the task is to, given a dataset $\mathcal{D}$ consisting of a labeled dataset $\mathcal{D}_L$ and an unlabeled dataset $\mathcal{D}_U$, predict the labels of $\mathcal{D}_U$. This setup is similar to semi-supervised learning, however unlike in semi-supervised learning, in GCD the classes differ between the labeled and the unlabeled set. Concretely, the labeled samples all belong to one of the known classes C_{known}, whereas the unlabeled samples could belong either to a known class or to one of the novel classes C_{novel}. The overall set of classes is denoted as $C = C_{\text{known}} \cup C_{\text{novel}}$.

3.1. Overview

Our method consists of two phases: First we perform an initial fine-grained clustering, then we aggregate the fine-grained clusters into the coarser target classes. An overview of the method can be found in Fig. 1.

We begin with a pretrained feature extractor model f, which we use to extract image representations. Multiple prior works have investigated in depth how to improve the quality of the pretrained features, *e.g*. GCD [25], DCCL [22] or PromptCAL [29]. Our method is orthogonal to these prior works and can be initialized with any of these pretrainings.

Based on the pretrained features, we compute the k-nearest neighbor graph $\boldsymbol{G}$ defined by the adjacency matrix $\boldsymbol{A} \in \mathbb{R}^{|\mathcal{D}|\times|\mathcal{D}|}$ where $|\mathcal{D}|$ is the number of samples in the training dataset and

$$\boldsymbol{A}_{i,j} = \begin{cases} 1 & \text{if } j \in \text{kNN}(i, k_1) \\ 0 & \text{else} \end{cases}, \tag{1}$$

where $\text{kNN}(i, k_1,)$ are the k_1 samples with the closest L_2 distance to i. As shown in previous work, *e.g*. DINO [2],

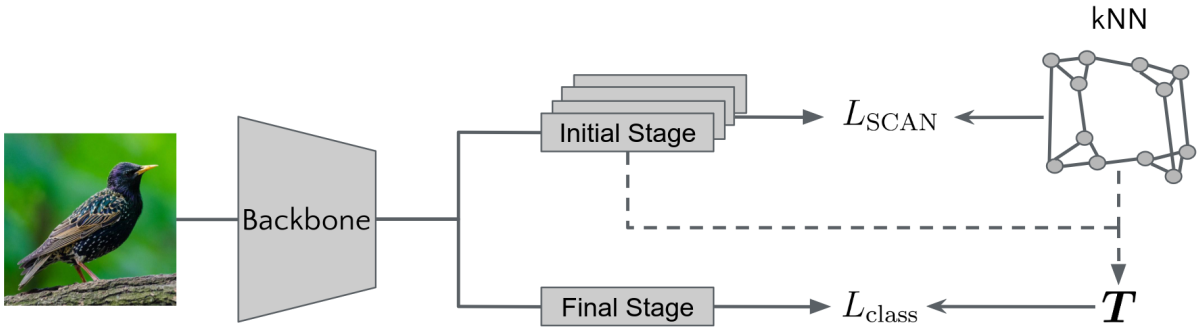

Figure 1. Overview of our method. The image is first processed by a pretrained feature extractor, then the features are used in two classification stages. First, we overcluster the data in the initial stage to a fine-grained clustering of high purity. To obtain more robust results, we train multiple such overclustering heads. The output of the overclustering stage is used to aggregate the kNN graph and derive the pairwise target matrix $\boldsymbol{T}$. Using these targets, we train the final stage, which recovers the target classes. After training, the clustering stage can be discarded, so no additional performance overhead is created compared to directly training the classification stage.

the k-nearest neighbors in the feature-space of pretrained vision models are usually from the same class as the query sample, which makes the kNN graph interesting for clustering. However, it will still contain many false positives, so directly using it for classification may introduce mistakes. To prevent this, we first overcluster the data into many small clusters, which is easier to achieve with high purity than directly classifying the data into the target classes.

3.2. Clustering

For the overclustering stage we make use of SCAN [7], which uses the k-nearest neighbors as pairwise positives to train a linear clustering head g on top of f to cluster the data into smaller clusters C_{over}. The number of clusters $c = |C_{\text{over}}|$ is set to be much higher than the amount of classes $|C|$. Given a batch $B \subseteq \mathcal{D}$ of samples SCAN minimizes the clustering loss

$$L_{\text{cluster}} = -\frac{1}{|B|} \sum_{x \in B} \sum_{k \in \text{kNN}(x)} \log g(x)^{\mathsf{T}} g(k), \quad (2)$$

where $g(x) \in \mathbb{R}^c$ is the probability distribution over the c clusters. Consequently, $g(x)^{\mathsf{T}} g(k)$ is the predicted probability of x and k being in the same cluster. This loss allows the network to learn a multi-class clustering from pairwise labels, however it does have a trivial solution, which is to simply assign every sample to the same cluster. To prevent this, SCAN employs an additional entropy loss

$$\begin{aligned} L_{\text{entropy}} &= \sum_{c \in C} g'_c \log g'_c, \\ \text{with } g'_c &= \frac{1}{|B|} \sum_{x \in B} g_c(x). \end{aligned} \quad (3)$$

The entropy term aims to avoid a collapse to a trivial solution, such as grouping every sample into the same cluster. In total, the clustering loss is

$$L_{\text{SCAN}} = L_{\text{cluster}} + \lambda L_{\text{entropy}}, \quad (4)$$

where λ is a hyperparameter governing the strength of the entropy regularization. This clustering step yields c approximately balanced clusters of high purity, which we then aggregate in the next step. We train using L_{SCAN} for T epochs until the clustering has stabilized.

3.3. Semi-supervised Aggregation

To aggregate the smaller clusters into our target classes, we take inspiration from graph pooling methods. Graph pooling is applied to obtain a more coarse-grained representation of a graph, while still maintaining its key characteristics. This can for example be done by merging similar nodes, identified by clustering, as well as aggregating their edges. We aim to achieve something very similar, in our case the clustering is given through the output of clustering head h, and we want to obtain a more coarse-grained representation of $\boldsymbol{G}$. An overview of the procedure is given in Fig. 2.

Regarding the aggregation method, we choose to follow [28], which calculates the aggregated adjacency matrix as

$$\boldsymbol{A}_{\text{agg}} = \boldsymbol{S}^{\mathsf{T}} \boldsymbol{A} \boldsymbol{S}, \quad (5)$$

where $\boldsymbol{S} \in \mathbb{R}^{|\mathcal{D}| \times c}$ is the soft cluster assignment matrix. The resulting $c \times c$ adjacency matrix now contains adjacencies not between samples, but between clusters. At this stage, we also utilize the knowledge contained in the ground truth labels that are available for $\mathcal{D}_L$ to aid the aggregation. We do so by creating an additional matrix $\boldsymbol{A}_{\text{sup}} \in \mathbb{R}^{c \times c}$ which for each pair i, j of clusters considers the set of labeled samples L_i and L_j within these clusters and counts

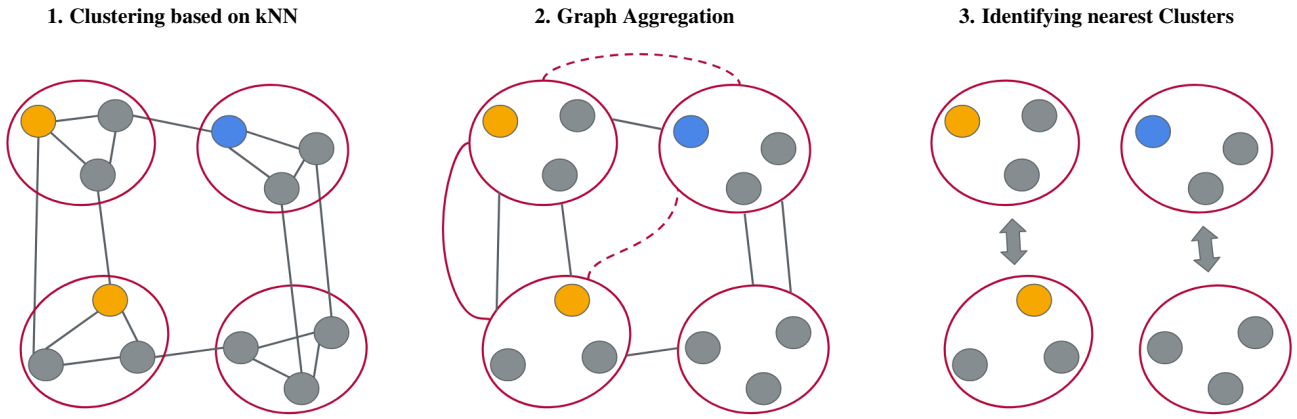

Figure 2. Our cluster aggregation procedure, divided into three steps. 1. We group the data into many small clusters based on their neighbors in the k-nearest neighbor graph. 2. We use the clustering to aggregate the graph. For this, we use both the existing kNN graph, and the given labels. In this example, the yellow and the blue samples are labeled, with the color indicating the class. This information is used to increase the link strength between clusters that contain samples of the same class, and decrease it between those with differing classes (depicted as red links between the clusters). 3. Finally, we use the aggregated graph to find clusters that are highly similar to each other.

the pairwise agreements and disagreement between the sets. Formally, we define $\boldsymbol{A}_{\text{sup}}$ as

$$\boldsymbol{A}_{sup_{i,j}} = \sum_{l \in L_i} \sum_{l' \in L_j} a(l, l')$$
$$\text{where } a(l,l') = \begin{cases} +1 & \text{if } l \text{ and } l' \text{ share the same class} \\ -1 & \text{otherwise} \end{cases}. \tag{6}$$

The two matrices are then added to obtain the combined matrix $\boldsymbol{A}_{\text{cluster}} = \boldsymbol{A}_{\text{agg}} + \boldsymbol{A}_{\text{sup}}$. Given this combined cluster adjacency matrix, we can now identify the top k_2 most similar clusters for each cluster defined as

$$\text{kNC}(i,k) = \arg\text{top}\, k(\boldsymbol{A}_{sup_{i,*}}), \tag{7}$$

where $\boldsymbol{A}_{sup_{i,*}}$ is the i^{th} row of $\boldsymbol{A}$. Similar to the k_1-nearest neighbors on the sample level, the k_2-nearest clusters are highly likely to share the same majority class. We can leverage this property to obtain a sample-level similarity matrix $\boldsymbol{T}$ with

$$\boldsymbol{T}_{i,j} = \begin{cases} 1 & \text{if } c(j) \in \text{kNC}(s(i), k_2) \\ 0 & \text{else} \end{cases}, \quad \text{with } s(i) = \arg\max \boldsymbol{S}_{i,*}. \tag{8}$$

Using the binary targets, we train the classification head h, which classifies the samples into the $|C|$ target classes. We train h using the binary cross entropy loss

$$L_{\text{class}} = -\frac{1}{|B|^2} \sum_{x \in B} \sum_{x' \in B} \boldsymbol{T}_{x,x'} \log h(x)^{\mathsf{T}} h(x') + (1 - \boldsymbol{T}_{x,x'})(1 - \log h(x)^{\mathsf{T}} h(x')). \tag{9}$$

After the training is completed, the clustering head g can be discarded, leaving the backbone f and the classification head h.

3.4. Multi-head Clustering

While the clustering is guided by the clustering loss L_{cluster}, different random initializations of the clustering head may lead to different final clusterings. To increase stability, it has often been found beneficial in image clustering to train not just one clustering head, but l independent heads [7, 13, 14]. We also make use of this approach and, following SCAN [7], train $l = 10$ independent heads for the first stage. This means that instead of one clustering head g we have multiple heads $g_0, g_1, \ldots, g_l$. Consequently, we average the clustering loss over these heads, so that

$$L_{\text{cluster}} = \frac{1}{h} \sum_{i=0}^{l} -\frac{1}{|B|} \sum_{x \in B} \sum_{k \in \text{kNN}(x)} \log g_i(x)^{\mathsf{T}} g_i(k). \tag{10}$$

Instead of one assignment matrix $\boldsymbol{S}$ we now obtain multiple matrices $\boldsymbol{S}_0, \boldsymbol{S}_2, \ldots, \boldsymbol{S}_l$. The matrix belonging to the head that shows the lowest loss is used for the aggregation step, thus making sure that only the best clustering is used. For the final stage itself we do not employ multiple heads, so the final outcome of the training is the same as with just one clustering head.

4. Experiments

4.1. Experimental setup

Datasets We evaluate our method on two standard classification datasets, CIFAR10 and CIFAR100 [18]. In addi-

Table 1. Overview of the used benchmarks. Following previous work [25], for CIFAR100 the classes are split into 80 known and 20 novel classes, and evenly split between known and novel classes for the other datasets. The known class samples are evenly split into labeled and unlabeled.

Dataset		CIFAR10 [18]	CIFAR100 [18]	ImageNet-100 [4]	CUB-200 [26]	Stanford Cars [17]	FGVC-Aircraft [21]
# Classes	Known	5	80	50	100	98	50
	Novel	5	20	50	100	98	50
# Images	Labeled	12,500	20,000	31,860	1,498	2,000	1,666
	Unlabeled	37,500	30,000	95,255	4,496	6,144	5,001

Table 2. Accuracy on standard classification datasets. PrCAL stands for PromptCAL [29]. Results for ORCA are taken from [29] and adapted to also use Vit-B/16 for a fair comparison.

	CIFAR10			CIFAR100			ImageNet-100		
Classes	All	Old	New	All	Old	New	All	Old	New
k-means [20]	83.6	85.7	82.5	52.0	52.2	50.8	72.7	75.5	71.3
GCD [25]	91.5	**97.9**	88.2	70.8	77.6	57.0	74.1	89.8	66.3
ORCA [1]	96.9	95.1	97.8	74.2	82.1	67.2	79.2	93.2	72.1
DCCL [22]	96.3	96.5	96.9	75.3	76.8	70.2	80.5	90.5	76.2
PrCAL [29]	**97.9**	96.6	**98.5**	81.2	84.2	75.3	**83.1**	92.7	**78.3**
SimGCD [27]	97.1	95.1	98.1	80.1	81.2	**77.8**	83.0	93.1	77.9
GCD + Ours	92.8	94.4	91.9	76.6	79.5	70.7	82.1	92.6	76.8
PrCAL + Ours	95.5	95.9	95.2	**82.4**	**85.6**	75.9	82.8	**94.1**	77.1

Table 3. Accuracy on fine-grained classification benchmarks. PrCAL stands for PromptCAL [29]. Results for ORCA are taken from [29] and adapted to also use Vit-B/16 for a fair comparison.

	CUB-200			Stanford Cars			FGVC-Aircraft		
Classes	All	Old	New	All	Old	New	All	Old	New
k-means [20]	34.3	38.9	32.1	12.8	10.6	13.8	12.9	12.9	12.8
GCD [25]	51.3	56.6	48.7	39.0	57.6	29.9	45.0	41.1	46.9
ORCA [1]	36.3	43.8	32.6	31.9	42.2	26.9	31.6	32.0	31.4
DCCL [22]	63.5	60.8	64.9	43.1	55.7	36.2	-	-	-
PrCAL [29]	62.9	64.4	62.1	50.2	70.1	40.6	52.2	52.2	**52.3**
SimGCD [27]	60.3	65.6	57.7	53.8	71.9	45.0	**54.2**	**59.1**	51.8
GCD + Ours	62.3	72.0	57.5	45.4	65.5	35.6	47.1	57.1	42.2
PrCAL + Ours	**68.8**	**73.4**	**66.6**	**54.4**	**72.1**	**45.8**	52.0	57.1	49.5

tion, we also evaluate on ImageNet-100, a subset of 100 classes selected from the standard ImageNet [4] dataset. The selection of the ImageNet classes follows [25]. We also evaluate on three fine-grained classification datasets, CUB-200 [26], Stanford Cars [17], and FGCV-Aircraft [21].

We follow the standard data split into known and novel classes used in previous works [22, 25, 29], which means that for CIFAR100, 80 classes are known and 20 are novel, and for all other datasets half the classes are known and half are novel. The known classes are split into 50% labeled and 50% unlabeled samples. A full overview of the used datasets can be found in Tab. 1.

Evaluation Metrics The quality of the final class assignment is quantified using clustering accuracy, which is defined as

$$\mathrm{ACC}(y, \hat{y}) = \max_{m \in \mathrm{perm}(C)} \frac{1}{|y|} \sum_i \mathbb{1}\big(y_i = m(\hat{y}_i)\big), \quad (11)$$

where $\mathrm{perm}(C)$ is the set of permutations of C. Intuitively, it measures the accuracy of the most favorable assignment between predicted classes and ground truth classes. In practice, this matching can be found by using the Hungarian algorithm [19].

Implementation Details Following previous work, we use a Vit-B/16 backbone [5] with all parameters except the last block kept frozen. As pretrained models, we use GCD [25], the original work which introduced the GCD setting, and PromptCAL [29], a more recent method with improved performance. We set the number of first-stage nearest neighbors k_1 to 5, the number of nearest clusters k_2 to 2, and the number of clusters c to 1000. Following SCAN [7], we set λ to 5. The model is trained using the Adam optimizer [16] with a learning rate of 10^{-3}, $\beta_1 = 0.9$ and $\beta_2 = 0.999$. We use a batch size of 512 and train for 200 epochs on the larger datasets (CIFAR10, CIFAR100, and ImageNet-100) or 300 epochs on the smaller ones. The first 100 epochs are used for learning the first-stage overclustering, afterwards, we switch to learning the target classifier. We use faiss [15] to quickly find the nearest neighbors. All experiments were conducted using either Nvidia V100 or Nvidia RTX 3090 GPUs.

4.2. Results

Main results We compare GCA to the original GCD method, as well as an adapted version of ORCA [1] that also uses Vit-B/16 to ensure a fair comparison. The adapted ORCA values are taken from [29]. In addition to these older methods, we also compare to more recent improved approaches such as DCCL [22], PromptCAL [29], and SimGCD [27]. Out of these methods, GCD, DCCL, and PromptCAL use semi-supervised k-means clustering for the final class assignment, whereas ORCA and SimGCD directly output class predictions. For these main results, we

Table 4. Estimated number of classes. The number of classes is estimated using the approach described in [25]. As we do not have access to the original pretrained models, our reproduction predicts slightly different class numbers compared to the original.

	CIFAR100	ImageNet-100	CUB-200	Stanford Cars
Ground truth	100	100	200	196
GCD [25]	100	109	231	230
GCD (repr)	97	112	240	182
PromptCAL	108	110	353	210

assume the number of classes is given beforehand following previous work.

The results on coarse-grained classification tasks in Tab. 2 show that for the GCD pretraining, GCA achieves much better results than the standard GCD method, outperforming it by 8.0% on ImageNet-100, and 4.8% on CIFAR100. However, when using the PromptCAL pretraining, the results are less clear. In this case, GCA does not provide a clear advantage on the coarse-grained tasks, with GCA outperforming standard PromptCAL on CIFAR100, but lagging behind on CIFAR10, with no clear winner on ImageNet-100.

For fine-grained classification tasks, GCA more clearly shows its advantage. As shown in Tab. 3, the GCD-pretrained GCA models outperform GCD considerably, outperforming GCD by 11.0% on CUB-200 and by 6.4% on Stanford Cars. In contrast to the results on coarse-grained datasets, the PromptCAL-pretrained models also perform better than their PromptCAL counterparts on fine-grained tasks, achieving 5.9% better accuracy on CUB-200 and 4.2% higher accuracy on Stanford Cars. For both pretrainings, GCA especially excels at predicting the known classes with the known-class accuracy increasing considerably over the corresponding pretraining. This shows that the proposed cluster aggregation approach is effective at transferring knowledge from labeled to unlabeled samples.

These results show that, while showing strong performance on the fine-grained benchmarks, GCA does not clearly outperform previous work on the coarse-grained benchmarks. A possible explanation for this is that the coarse-grained tasks may be easier to solve due to the larger differences between the clusters, resulting in constrained k-means already obtaining near-optimal results. In contrast, the more difficult fine-grained tasks might be harder for k-means due to the higher similarity between different classes, whereas GCA might be more robust in these cases due to focusing on local neighborhoods in the feature space instead of taking a global view like k-means.

Results with Unknown Class Number While we previously assumed knowledge of the number of classes, we also evaluate our method in a setting where the number of classes is not known. For this case, we first estimate the number of classes using the class number estimation method proposed in [25], then run GCA with the estimated number of classes. Note that while we use the same estimation technique as GCD, we do not have access to the original pretrained models, hence the estimated class numbers are not the same. In addition, the estimates differ between the GCD and the PromptCAL pretraining. The exact estimates are listed in Tab. 4.

We compare GCA to GCD [25] and SimGCD [27] on four different datasets. While GCD also uses the aforementioned class number estimation technique, SimGCD proposed to simply set the number of classes to a large enough number, *e.g.* two times the ground truth class number, and let the model discard unnecessary class prototypes. Consequently, SimGCD reports two numbers, one using the estimation technique from [25] and one using twice the ground truth class number.

The results in Tab. 5 and Tab. 6 mostly mirror the ones for known class numbers, with only a minimal accuracy decrease compared to the known class number setting. The

Table 5. Accuracy on CIFAR100 and ImageNet-100 when the number of classes has to be estimated. The performance with known class number is given for reference. PrCAL stands for PromptCAL [29].

	# of Classes	CIFAR100			ImageNet-100		
Classes		All	Old	New	All	Old	New
GCD [25]	Known	70.8	77.6	57.0	74.1	89.8	66.3
SimGCD [27]	Known	80.1	81.2	77.8	83.0	93.1	77.9
GCD + Ours	Known	76.6	79.5	70.7	82.1	92.6	76.8
PrCAL + Ours	Known	82.4	85.6	75.9	82.8	94.1	77.1
GCD [25]	Estimated	70.8	77.6	57.0	72.7	91.8	63.8
SimGCD [27]	Estimated	80.1	81.2	**77.8**	**81.7**	91.2	**76.8**
SimGCD [27]	2x true	77.7	79.5	74.0	80.9	93.4	74.8
GCD + Ours	Estimated	73.9	77.5	66.8	77.4	92.3	69.9
PrCAL + Ours	Estimated	**81.7**	**86.4**	72.3	81.1	**94.2**	74.5

Table 6. Accuracy on CUB-200 and Stanford Cars when the number of classes has to be estimated. The performance with known class number is given for reference. PrCAL stands for PromptCAL [29].

	# of Classes	CUB-200			Stanford Cars		
Classes		All	Old	New	All	Old	New
GCD [25]	Known	51.3	56.6	48.7	39.0	57.6	29.9
SimGCD [27]	Known	60.3	65.6	57.7	53.8	71.9	45.0
GCD + Ours	Known	62.3	72.0	57.5	45.4	65.5	35.6
PrCAL + Ours	Known	68.8	73.4	66.6	54.4	72.1	45.8
GCD [25]	Estimated	47.1	55.1	44.8	35.0	56.0	24.8
SimGCD [27]	Estimated	61.5	66.4	59.1	49.1	65.1	41.3
SimGCD [27]	2x true	**63.6**	68.9	**61.1**	48.2	64.6	40.2
GCD + Ours	Estimated	61.9	**72.5**	56.6	43.6	64.3	33.7
PrCAL + Ours	Estimated	62.0	65.2	60.4	**54.5**	**71.2**	**46.5**

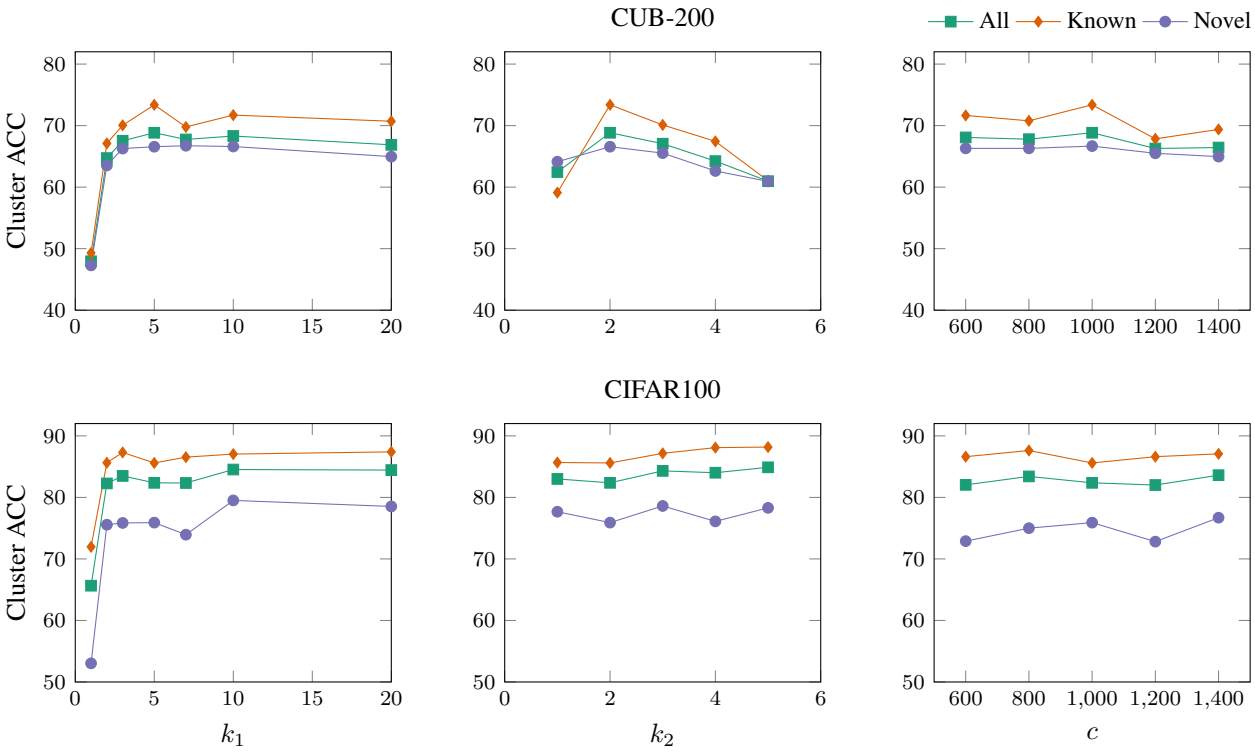

Figure 3. GCA hyperparameter sensitivity analysis. We analyze the sensitivity to changes in the number of first-stage nearest neighbors k_1, the number of second-stage nearest clusters k_2 and the number of first stage clusters c. All experiments are run with PromptCAL [29] as pretraining.

exception to that is the PromptCAL-pretrained model on CUB-200, where performance drops by 6.8% compared to the result in Tab. 3. The most likely cause for this drop is the large discrepancy between predicted and ground truth class number for this combination of dataset and pretraining, with the estimate overshooting the true class number by more than 75%. For the other datasets and pretrainings, the class number estimate is much closer to the ground truth, which results in very little performance degradation.

Further Analysis We analyze GCA's sensitivity to changes in hyperparameters to gain a sense of its robustness. We focus on the number of nearest neighbors k_1, the number of nearest clusters k_2, and the number of first-stage clusters c. The sensitivity analysis is performed on CIFAR100, a coarse-grained dataset, and CUB-200, a fine-grained dataset.

The results in Fig. 3 show that k_1 should not be set to very low values such as 1 or 2, but performance gains quickly plateau afterwards. A closer look at the purity of the first-stage clustering for $k_1 = 1$ and $k_1 = 5$ shows a massive increase in cluster purity from 68% to 82% for CUB-200, and from 70% to 91% for CIFAR100. This shows that the nearest neighbors are essential to achieving a good initial clustering, which is consistent with observations in SCAN [7], the clustering method the first stage of GCA is based on.

For the number of nearest clusters k_2, we see different trends for CUB-200 and CIFAR100. Whereas CIFAR100 shows a slight upward trend with increasing k_2, CUB-200 reaches its best performance at $k_2 = 2$, with a gradual de-

Table 7. Experiments on CIFAR100 with varying numbers of known classes and varying shares of labeled samples. For each setting, the first number signifies the number of known classes, and the second the percentage of known class samples that are labeled. The default setting for CIFAR100 corresponds to C80-L50. PrCAL stands for PromptCAL [29]. Results for GCD, ORCA and PromptCAL are taken from [29].

	C50-L10			C25-L50			C10-L50		
Classes	All	Old	New	All	Old	New	All	Old	New
GCD [25]	60.2	68.9	55.8	56.8	67.6	55.0	48.3	65.1	47.3
ORCA [1]	60.3	66.0	55.3	58.2	79.9	57.5	51.7	78.0	50.2
PrCAL [29]	68.9	77.5	64.7	65.7	76.9	63.9	53.2	**79.3**	51.7
SimGCD [27]	60.5	65.0	56.4	60.6	71.9	58.7	49.6	33.3	50.5
GCD + Ours	69.9	76.4	64.1	64.2	79.4	61.8	**62.7**	69.6	**62.3**
PrCAL + Ours	**74.5**	**81.8**	**67.9**	**67.9**	**81.1**	**65.7**	56.8	55.6	78.6

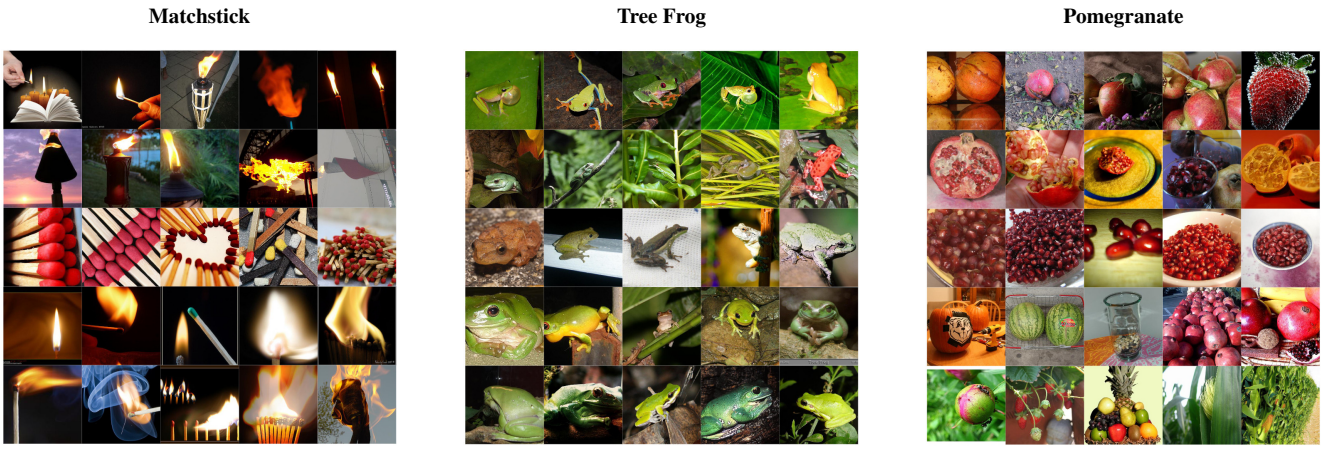

Figure 4. Qualitative examples of our hierarchical clustering approach. Each 5x5 block contains images that were assigned to the same class in the final stage, and each row within a block contains images from one first-stage cluster. The images are sourced from ImageNet-100, a subset of ImageNet [4]. The model used to obtain the clustering and the predictions uses PromptCAL [29] for pretraining.

cline in accuracy for higher values. This could be due to the different number of classes between the two datasets, which results in a different number of first-stage clusters per target class. For example, for each CIFAR100 class, there will be on average 10 first-stage clusters, but only 5 for each CUB-200 class. As a result, increasing k_2 will more quickly lead to unrelated clusters being merged, hence the lower performance. Lastly, the exact value of c does not seem to heavily influence accuracy, with no meaningful differences between $c = 600$ and $c = 1400$ for both datasets. These results show the robustness of GCA, which is especially important since Generalized Category Discovery works with mostly unlabeled data, so in practical applications, searching for optimal hyperparameters is difficult.

We also investigate how well GCA performs when the amount of labeled data is reduced compared to the standard setting. Following [29], we benchmark GCA on three different variations of the CIFAR100 benchmark, varying both the number of known classes and the percentage of known-class samples that are assigned to the labeled set. We compare GCA to GCD, ORCA, PromptCAL, and SimGCD. For SimGCD, we adapt the weight ϵ of the entropy regularization term and set it to 0.5 as the default value for CIFAR100 yields low accuracy in this setting. The results in Tab. 7 show that GCA also achieves strong results in this more extreme setting, with GCA consistently outperforming its corresponding pretrained model. In general, it seems that the number of known classes is more important to the performance of GCD methods than how many samples of these known classes are labeled, which shows that discovering new classes is much harder than recognizing known ones.

Lastly, we provide qualitative examples of the relationship between the first-stage clusters and the target classes. In Fig. 4, we show three different ImageNet-100 classes and for each class, we show five randomly selected examples for five first-stage clusters that were assigned to this class by our PromptCAL-pretrained GCA model. The examples show that the first-stage clusters tend to specialize on particular aspects or perspectives of their superclass, for example, the third cluster of the class "matchstick" contains mostly images that focus on the head of the matchstick, whereas other clusters focus more on the flame. We also see the effect of mispredictions, *e.g.* the second cluster does not consist of matchstick images at all but was mostly likely aggregated into the class due to its visual similarity. This more fine-grained view of the data is an additional benefit of GCA that previous methods do not provide.

5. Conclusion

In this paper, we propose Guided Cluster Aggregation (GCA), a hierarchical approach to GCD that first groups the data into many small clusters before aggregating them into the target classes using the given labels as guidance. Extensive experiments show that GCA is superior to the semi-supervised k-means clustering employed in many previous works. Hierarchical approaches such as GCA are fundamentally not limited to GCD tasks, they could also be applied to other tasks that operate on a mix of labeled and unlabeled data, such as semi-supervised learning. Investigating the potential for such applications is an interesting topic for future work.

Acknowledgements

The authors acknowledge the financial support by the German Federal Ministry for the Environment, Nature Conservation, Nuclear Safety and Consumer Protection (BMUV) within the project EKAPEx 67KI32002A

References

[1] Kaidi Cao, Maria Brbic, and Jure Leskovec. Open-world semi-supervised learning. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. 2, 5, 7

[2] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9630–9640. IEEE, 2021. 1, 2

[3] Haoang Chi, Feng Liu, Wenjing Yang, Long Lan, Tongliang Liu, Bo Han, Gang Niu, Mingyuan Zhou, and Masashi Sugiyama. Meta discovery: Learning to discover novel classes given very limited data. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022. 2

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pages 248–255. IEEE Computer Society, 2009. 5, 8

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. 5

[6] Enrico Fini, Enver Sangineto, Stéphane Lathuilière, Zhun Zhong, Moin Nabi, and Elisa Ricci. A unified objective for novel class discovery. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 9264–9272. IEEE, 2021. 2

[7] Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. SCAN: learning to classify images without labels. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part X*, volume 12355 of *Lecture Notes in Computer Science*, pages 268–285. Springer, 2020. 3, 4, 5, 7

[8] Kai Han, Sylvestre-Alvise Rebuffi, Sébastien Ehrhardt, Andrea Vedaldi, and Andrew Zisserman. Autonovel: Automatically discovering and learning novel visual categories. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(10):6767–6781, 2022. 2

[9] Kai Han, Andrea Vedaldi, and Andrew Zisserman. Learning to discover novel visual categories via deep transfer clustering. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 8400–8408. IEEE, 2019. 2

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 1026–1034. IEEE Computer Society, 2015. 1

[11] Yen-Chang Hsu, Zhaoyang Lv, and Zsolt Kira. Learning to cluster in order to transfer across domains and tasks. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018. 2

[12] Yen-Chang Hsu, Zhaoyang Lv, Joel Schlosser, Phillip Odom, and Zsolt Kira. Multi-class classification without multi-class labels. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019. 2

[13] Jiabo Huang, Shaogang Gong, and Xiatian Zhu. Deep semantic clustering by partition confidence maximisation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*, pages 8846–8855. Computer Vision Foundation / IEEE, 2020. 4

[14] Xu Ji, Andrea Vedaldi, and João F. Henriques. Invariant information clustering for unsupervised image classification and segmentation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 9864–9873. IEEE, 2019. 4

[15] Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Trans. Big Data*, 7(3):535–547, 2021. 5

[16] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. 5

[17] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *2013 IEEE International Conference on Computer Vision Workshops, ICCV Workshops 2013, Sydney, Australia, December 1-8, 2013*, pages 554–561. IEEE Computer Society, 2013. 5

[18] Alex Krizhevsky et al. Learning multiple layers of features from tiny images. Technical report, 2009. 4, 5

[19] Harold W Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2(1-2):83–97, 1955. 5

[20] James MacQueen. Classification and analysis of multivariate observations. In *5th Berkeley Symp. Math. Statist. Probability*, pages 281–297. University of California Los Angeles LA USA, 1967. 2, 5

[21] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew B. Blaschko, and Andrea Vedaldi. Fine-grained visual classification of aircraft. *CoRR*, abs/1306.5151, 2013. 5

[22] Nan Pu, Zhun Zhong, and Nicu Sebe. Dynamic conceptional contrastive learning for generalized category discovery. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 7579–7588. IEEE, 2023. 1, 2, 5

[23] Martin Rosvall, Daniel Axelsson, and Carl T Bergstrom. The map equation. *The European Physical Journal Special Topics*, 178(1):13–23, 2009. 2

[24] Yiyou Sun and Yixuan Li. Opencon: Open-world contrastive learning. *Trans. Mach. Learn. Res.*, 2023, 2023. 2

[25] Sagar Vaze, Kai Han, Andrea Vedaldi, and Andrew Zisserman. Generalized category discovery. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pages 7482–7491. IEEE, 2022. 1, 2, 5, 6, 7

[26] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. Technical report, 2011. 5

[27] Xin Wen, Bingchen Zhao, and Xiaojuan Qi. A simple parametric classification baseline for generalized category discovery. *CoRR*, abs/2211.11727, 2022. 1, 2, 5, 6, 7

[28] Zhitao Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical graph representation learning with differentiable pooling. In Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, editors, *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4805–4815, 2018. 3

[29] Sheng Zhang, Salman H. Khan, Zhiqiang Shen, Muzammal Naseer, Guangyi Chen, and Fahad Shahbaz Khan. Promptcal: Contrastive affinity learning via auxiliary prompts for generalized novel category discovery. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 3479–3488. IEEE, 2023. 1, 2, 5, 6, 7, 8

[30] 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 *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 9462–9470. Computer Vision Foundation / IEEE, 2021. 2