HARD REGULARIZATION TO PREVENT COLLAPSE IN ONLINE DEEP CLUSTERING WITHOUT DATA AUGMENTATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Online deep clustering refers to the joint use of a feature extraction network and a clustering model to assign cluster labels to each new data point or batch as it is processed. While faster and more versatile than offline methods, online clustering can easily reach the collapsed solution where the encoder maps all inputs to the same point and all are put into a single cluster. Successful existing models have employed various techniques to avoid this problem, most of which require data augmentation or which aim to make the average soft assignment across the dataset the same for each cluster. We propose a method that does not require data augmentation, and that, differently from existing methods, regularizes the hard assignments. Using a Bayesian framework, we derive an intuitive optimization objective that can be straightforwardly included in the training of the encoder network. Tested on four image datasets, we show that it consistently avoids collapse more robustly than other methods and that it leads to more accurate clustering. We also conduct further experiments and analysis justifying our choice to regularize the hard cluster assignments.

1 INTRODUCTION

Deep clustering refers to the combination of deep learning and clustering, where the data are first encoded with a deep neural network to a feature space, and then clustering is performed in the feature space. Deep clustering models (and clustering models more generally) can be classified as offline or online. Offline models process the entire dataset, and then assign all cluster labels at once. Online models, by contrast, assign a cluster label to each data point, or each batch, as it is processed. Offline methods tend to produce more accurate clusterings (e.g., (Mahon & Lukasiewicz, 2021; Niu et al., 2021)) reaching close to supervised performance, as they can leverage information from later data points when assigning cluster labels to earlier data points. However, they are more expensive to train, as they must alternate between encoding the entire dataset/training the encoder for some number of epochs, and clustering the encoded data. Online models, on the other hand, can jointly encode and cluster so are less computationally expensive. They are also more versatile, being applicable in real-world settings where new data is constantly becoming available, as opposed to offline methods, which are limited to having a fixed, pre-defined dataset (Silva et al., 2013).

The disadvantage of online methods is that they are more difficult to train. In particular, they run the risk of producing a degenerate solution where the large majority of data points are concentrated in a small number of clusters. In the extreme case, all points are placed into the same cluster. For example, an intuitive way to formulate a training procedure for online deep clustering is to update both the encoder network and the clustering parameters to make the encoding of each point close to its cluster centroid and far from other cluster centroids. (The clustering parameters are, e.g., the centroids in K-means.) However, this training objective is trivially minimized by the encoder network mapping all data points to the same point in feature space, where this point is equal to one of the cluster centroids. Techniques for avoiding collapse we refer to as partition support. Several partition support methods have been proposed, but they mostly require data augmentation (DA), and those that do not are often ad hoc and lack a rigorous technical foundation. Additionally, they take the soft assignments as a measure of partition collapse and propose to regularize the soft assignments to make them more uniform. This paper focuses on deep clustering without DA, which is an advantage

because relying on DA limits a method to domains in which have sufficient prior knowledge to perform class-preserving augmentations. Additionally, we argue that soft assignments are not an accurate measure of collapse, and that we should instead focus on hard assignments.

We propose a DA-free partition support by regularizing hard assignments. Specifically, we consider the problem of how to optimally assign each data point in a batch to the most appropriate cluster. We express this problem probabilistically in a Bayesian framework, where the regularizing element is captured by a uniform prior across clusters. We then use this expression to derive a precise optimization objective, which we also show to be equivalent, up to a small error term, to the objective of maximizing the mutual information of the cluster assignments and the data index. This objective itself is too slow to solve exactly, but we devise a greedy approximation algorithm that can be implemented straightforwardly and results in an intuitive method for fully online clustering, which we term combination assignment. This method outperforms existing online DA-free clustering methods on four popular image clustering datasets. Finally, we analyze the role of hard vs. soft cluster assignments in our partition support method, and in previous methods, and make the case that regularizing hard assignments is a more effective approach. Note, although existing methods can easily convert soft assignments to hard assignments, this is very different from regularizing the hard assignments, as we propose. While the relation between hard and soft clustering has been studied before (Bora et al., 2014; Kearns et al., 1998), its study in the context of regularizing online deep clustering collapse is new.

Our contributions are briefly summarized below.

- We articulate a clear Bayesian framework of the problem of hard assignments in online deep clustering models, which is more theoretically correct than requiring uniformity in each batch, as done by existing methods.
- We use this framework to derive an optimization objective and prove that it is approximately equivalent to an information-theoretic framework that maximizes the mutual information of the cluster assignments and the data index.
- We show empirically that the resulting method significantly outperforms existing partition support methods, both in avoiding partition collapse and resulting clustering accuracy.
- We conduct further analysis of the performance of different partition support methods, which justifies our choice to focus on hard assignments.

The rest of this paper is organized as follows. Section 2 gives an overview of related work, while Section 3 lays the theoretical foundations of our method, describes our greedy algorithm for optimizing the resulting objective and proves the equivalence to mutual information maximization. Section 4 reports our empirical results and analysis, and finally Section 5 summarizes our findings.

2 Related Work

A key component of online deep clustering methods is how they avoid the collapsed solution, where (almost) every data point is placed in the same cluster. Methods designed for contrastive learning, and those clustering models that employ it as a part of the training procedure, are more resistant to collapse, because the negative pairs are encouraged to be represented differently. This can mean that they are to be placed in different clusters (Huang & Gong, 2021), or just that the encodings should be far apart (Zhang et al., 2021; Cai et al., 2021). Either approach helps resist all points having the same representation. Even without negative pairs, data augmentation is essential for some methods to avoid collapse. Grill et al. (2020) perform representation learning without negative pairs, only positive pairs that are encouraged to be similar. Then an "online" network is trained to predict the output of a "target" network, where the target network is a slow-moving average of the online network. Zbontar et al. (2021) use data-augmented pairs to reduce redundancy in the representations by minimizing off-diagonals in the cross-correlation matrix.

Among online clustering models, several partition support methods have been proposed. Kulshreshtha & Guha (2018) freeze the encoder during clustering, but this requires pretraining it on a separate task. Gao et al. (2020) proposes an online autoencoder-based (AE) clustering model, where the reconstruction loss helps to avoid partition collapse, but this limits the method to simple datasets, because the AE's reconstruction loss requires that different images from the same class have

significant pixel overlap. Zhan et al. (2020) continually reweight the loss function to encourage the assignment to smaller clusters. Additionally, when clusters decrease below a certain threshold, they are deleted, and the largest cluster is split in two using K-means. Cai et al. (2021) treat soft cluster assignments as representations, and use data-augmented pairs with the same method as Zbontar et al. (2021). Zhong et al. (2020) minimize the sum of squares of the probability (i.e., soft assignment) of each cluster, marginalized over each training batch. Decomposing the expectation of the square, we see that this also involves minimizing the variance. However, the authors also use contrastive learning and DA to avoid collapse, so they do not fully rely on sum-of-squares minimization.

A similar idea, employed by (Li et al., 2020; Hu et al., 2017; Niu et al., 2021; 2020; Van Gansbeke et al., 2020), is to maximize the entropy of soft assignments, marginalized over each training batch. For an input distribution X with corresponding soft assigned labels Y, both approximated over a batch, an extra term $-H(Y)$ is added to the loss function. As the entropy of a multinomial is maximized at the uniform distribution, this encourages more equally sized clusters. Entropy maximization has the advantage of a solid formal interpretation as part of the maximization of the mutual information between data and cluster assignments: by maximizing $H(Y)$ but minimizing $H(Y|X)$ (the latter is minimized explicitly in (Hu et al., 2017) and implicitly via contrastive learning in (Li et al., 2020)), we are maximizing

$$I(X;Y) = H(Y) - H(Y|X)\,.$$

However, the marginal entropy term tends to only be partially successful at preventing partition collapse, often the entire dataset is still put into only a small number of clusters. As well as being confirmed in our experiments in Section 4, this empirical weakness of entropy maximization for avoiding partition collapse is reported in (Hu et al., 2017), and better results are found by explicitly constraining the soft cluster assignments using non-linear programming. Li et al. (2020) do not rely entirely on entropy maximization, because they employ contrastive learning, which (as explained above) also helps to avoid partition collapse. Thus, while theoretically sound, entropy maximization is not sufficiently effective empirically.

Another approach is to directly impose a constraint on cluster assignments. Based on earlier work by Asano et al. (2019), Caron et al. (2020); Deshmukh et al. (2021); Kumar et al. (2021) proposed to constrain the soft cluster assignments to be marginally uniform across each training batch. Though effective in preventing partition collapse, the constraint of uniformity across each batch is too strict, as noted by Kumar et al. (2021). A given batch is very unlikely to be exactly uniform across classes, so the ground-truth distribution of classes will almost certainly violate this constraint.

3 METHOD

3.1 PROBLEM FORMULATION

We want to simultaneously (a) train the encoder and (b) make hard assignments to each batch of points at a time, based on the features extracted by that encoder. If the encoder is $f_{\theta_1} : \mathbb{R}^n \to \mathbb{R}^m$, our clustering model is $g_{\theta_2} : \mathbb{R}^m \to \{1, \ldots, K\}$, parametrized by θ_2, and our batch size is N, then we seek a function of the form $\Gamma : \mathbb{R}^{N \times m} \to \{1, \ldots, K\}^N$. (The difference between Γ and g_{θ_2} is that the former is used during training to assign an entire batch, while the latter is used at inference time and can assign each point individually.) If we have a method for batchwise assignment, then the training objective can be formulated as minimizing the distance of points from the centroids of their assigned clusters. This objective can be used to train both the encoder and the clustering model:

$$\underset{\theta_1, \theta_2}{\arg\min} \sum_{i=1}^{N} ||f_{\theta_1}(x_i) - \mu_{k_i}||^2\,, \tag{1}$$

where x is the input batch, $k_i = \Gamma(f_{\theta_1}(x))_i$ is the cluster label assigned to the encoding of x_i under f, and μ_k is the kth cluster centroid.

Finding a suitable assignment function Γ is non-trivial. It should be more likely to assign points to the clusters whose centroids are closer. However, using only this rule, and assigning every vector to its closest centroid, we allow a collapsed solution where equation 1 is minimized by f_{θ_1} mapping every point to a single centroid: $\forall i, f_{\theta_1}(x_i) = c_k$, for some $k \in \{1, \ldots, K\}$. The heart of our method is the choice of a suitable Γ, which avoids the collapsed solution. We refer to it as combination assignment, because it uses a prior over the combination of such labels under a multinoulli distribution

3.2 COMBINATION ASSIGNMENT

Combination assignment is based on a Bayesian formulation of the online clustering problem. Let $\mathcal{D}$ be the data distribution, $X \sim \mathcal{D}$ be a sampled batch of data, Z be the random variable defined by applying the feature extraction to X (i.e., Z is the output of a deep encoder network), and let Y be the assigned cluster labels. Here, we consider the encoder to be fixed, and we are just interested in the best hard assignment of cluster labels, given the extracted features. That is, we want to determine the values of Y with the maximum probability under the a posteriori distribution $p(Y|Z)$. We assume a uniform prior over K cluster labels, $p(Y = k) = 1/K$ for $k \in \{1, \ldots, K\}$. Note how this is a weaker and more realistic assumption than each batch containing exactly uniform numbers of each class. Then, the prior probability of a batch containing exactly n_k labels for each cluster $k \in \{1, \ldots, K\}$ is

$$\left(\frac{1}{K}\right)^N \frac{N!}{\prod_{k=1}^K n_k!}\,, \tag{2}$$

where $N = \sum_{k=1}^K n_k$ is the batch size. We model each cluster as a multivariate normal distribution in feature space, so that the likelihood is given by

$$p(Z = z_1, \ldots, z_N | Y = k_1, \ldots, k_N) = \prod_{i=1}^N \frac{\exp(-\frac{1}{2}(z_i - \mu_{k_i})\Sigma_{k_i}^{-1}(z_i - \mu_{k_i}))}{\sqrt{(2\pi)^d |\Sigma_{k_i}|}}\,, \tag{3}$$

where d is the dimension of the feature space, and μ_k and Σ_k are the centroid and covariance matrix of the kth cluster, respectively. If we further assume each cluster is spherical, with the same isotropic variance across all clusters, i.e., $\Sigma_k = \sigma I$, for $k \in \{1, \ldots, K\}$, and we maximize the posterior corresponding to the prior in equation 2 and the likelihood in equation 3, then we get the following optimization problem (details in appendix).

$$\arg\min_Y \sum_{i=1}^N ||z_i - \mu_{k_i}||^2 + 2\sigma \sum_{k=1}^K \log(n_k!)\,. \tag{4}$$

In matrix notation, the objective is

$$\arg\min_{Q \in \mathcal{B}^{N \times K}} \langle Q, \tilde{Q} \rangle + 2\sigma \log(\mathbb{1}_N^T Q!)\mathbb{1}_K \tag{5}$$
$$\text{subject to } Q\mathbb{1}_K = \mathbb{1}_N\,,$$

where $\tilde{Q}$ is the matrix of unnormalized log probabilities, i.e., $\tilde{Q}_{i,j} = ||z_i - \mu_j||^2$, $\langle \cdot, \cdot \rangle$ denotes the Frobenius inner product, $\mathbb{1}_a$ is an a-dimensional vector of all ones, and $\log$ and factorial are applied to each element in the K-dimensional vector $\mathbb{1}_N^T Q$. The requirement that Q be a Boolean matrix specifies that we seek hard assignments, and the constraint $Q\mathbb{1}_K = \mathbb{1}_N$ specifies that the rows of Q are one-hot vectors, i.e., each latent vector is assigned to exactly one cluster.

3.3 SOLVING THE OPTIMIZATION PROBLEM

Unfortunately, it is too slow to solve equation 5 exactly (see appendix for details). However, to motivate an approximation, we can consider the simpler problem of assigning the last data point in a batch, given that the rest have already been assigned. That is, we maximize the conditional probability of the Nth assignment in a batch, conditioned on the $N-1$ previous assignments. This gives the following optimization problem (details in appendix).

$$\arg\max_{k_N = 1, \ldots, K} p(y_N = k | y_1 = k_1, \ldots, y_{N-1} = k_{N-1}; Z) =$$
$$\arg\min_{k_N = 1, \ldots, K} ||z_N - \mu_{k_N}||^2 + 2\sigma \log(n_{k_N} + 1)\,, \tag{6}$$

where n_k is the number of points assigned to cluster k before the Nth assignment.

To approximately solve equation 5, we employ a greedy algorithm that iteratively solves equation 6 with respect to the most confident assignment possible. That is, at each iteration, select the pair (i, k) (corresponding to (N, k_N) above) for which equation 6 is smallest, with i ranging over the indices of still-unassigned points, and k ranging over all clusters, and assign the ith point to cluster k.

To get an intuition on our method, recall that we generally want to assign points to the closest centroid, but that we should try to resist assigning to a cluster that already has lots of points assigned to it. Even if such a cluster has its centroid closest to Nth point, it may be better to instead assign to a smaller, further away cluster. This suggests choosing the cluster assignment so as to minimize a combination of the distance to the centroid and some increasing function of cluster size, which is precisely what equation 6 expresses. The first term says to pick a nearby cluster, and the second term says to penalize clusters that already have many points assigned. It would not be obvious, a priori, what the increasing function of cluster-size should be exactly, but the derivation of equation 6 shows that $\log(n+1)$ is an appropriate choice.

3.4 INTERPRETATION

Here, we show that the greedy algorithm that iteratively solves equation 6 can be interpreted as iteratively making whatever assignment will maximize the mutual information between the batch index i and the cluster labels. First, we show that our method is very closely equivalent to maximizing the entropy of cluster labels in each batch.

Let $H^{(k)}$ be the marginal hard entropy of cluster labels after a new assignment to cluster k:

$$H^{(k)} = \frac{x_k+1}{N+1}\log\frac{x_k+1}{N+1} + \sum_{j=1,j\neq k}^{K}\frac{x_j}{N+1}\log\frac{x_j}{N+1}\,,$$

and consider the difference $H^{(k)} - H^{(k')}$ between the entropy after making assignment k vs. after making a different assignment k'. It can be shown (see appendix for full proof) that

$$H^{(k)} - H^{(k')} \approx \frac{1}{N+1}\left(\log(x_{k'}+1) - \log(x_k+1)\right)\,. \tag{7}$$

Thus, if we were to make each assignment so as to maximize $\log(\mathcal{L}_k(x)) + \lambda H^{(k)}$, where $\mathcal{L}(k)$ is the likelihood of the new data point under cluster k and λ is some hyperparameter, then, subject to the above approximation, for each $k, k' \in \{1,\ldots,K\}$, we would prefer to assign to k iff

$$\begin{aligned}\log\mathcal{L}(k) + \lambda H^{(k)} > \log\mathcal{L}(k') + \lambda H^{(k')} \iff\\ \log\mathcal{L}(k) - \log\mathcal{L}(k') > \frac{\lambda}{N+1}\left(\log(x_{k'}+1) - \log(x_k+1)\right)\,.\end{aligned} \tag{8}$$

Modelling clusters as multivariate normal distributions with isotropic variance σ (as above), and setting $\lambda = N+1$, equation 8 becomes equivalent to equation 6. (See appendix for full proof.)

Thus, our method closely approximates a maximization of the entropy of cluster labels. There is some similarity to those methods, discussed in Section 2, that use an additional loss term to encourage greater entropy of soft assignments in each batch, but an important difference here is that we are maximizing the entropy of hard assignments. This means that the entropy of cluster labels given batch index is automatically zero. Therefore, using the decomposition $I(X;Y) = H(X) - H(X|Y)$, the mutual information of the batch index i and the cluster labels equals the entropy of cluster labels, and so our method is a close approximation to maximizing this mutual information.

3.5 TRAINING PROCEDURE

Our model comprises an encoder network f_{θ_1} and a set of cluster centroids $\theta_2 = \{\mu_1,\ldots,\mu_K\}$. To train on an input batch, we first encode the raw data using f_{θ_1}, then we employ the combination assignment method of Section 3.3, and use these assignments to minimize equation 4 with respect to both θ_1 and θ_2. As the second term in equation 4 has no gradient, the updates are made only with respect to the first term, so equivalently to equation 1. At inference time, we do not perform combination assignment. Instead, we simply assign each point to the cluster with the nearest centroid, so the resulting clustering model can assign points individually, and is not restricted to assigning cluster labels batchwise. The full methods for training and inference are described by the functions TrainOnBatch and PredictBatch, respectively, in Algorithm 1.

Algorithm 1: **(combination assignment):** During training, cluster labels are assigned batchwise, with partition support provided by a uniform prior across clusters. During inference, cluster labels are assigned pointwise, without any explicit partition support.

$f_{\theta_1} \leftarrow$ encoder network;
$\theta_2 = \mu_1, \ldots, \mu_K \leftarrow$ centroids for each of the K clusters;
$\sigma \leftarrow$ isotropic variance of all clusters;
function ASSIGNBATCH(Z)
  $counts \leftarrow K$-dimensional array, initially all 0s;
  $isAssigned \leftarrow N$-dimensional Boolean array, initially all False;
  $D \leftarrow N \times K$ matrix, where $D_{ij} = ||Z_i - \mu_j||^2$;
  for r=1,...,N **do**
    $\tilde{D} \leftarrow N \times K$ matrix, where $\tilde{D}_{ij} = D_{ij} + 2\sigma \log(counts[i] + 1)$;
    $(a, k) \leftarrow$ argmin $\{\tilde{D}_{ij} : \neg isAssigned[i]\}$;
    $counts[k] \leftarrow counts[k] + 1$;
    $isAssigned[a] \leftarrow True$;
    $loss \leftarrow loss + D_{a,k}$
  return $loss$
function TRAINONBATCH(X)
  $Z \leftarrow f_{\theta_1}(X)$, encodings for a batch of N data points;
  $loss \leftarrow AssignBatch(Z)$;
  take a gradient descent step on $loss$ with respect to θ_1, θ_2
function PREDICTDATAPOINT(x)
  $z \leftarrow f_\theta(x)$ encodings for a batch of N data points;
  $assignment = \arg\min_{j=1,\ldots,K} ||z - \mu_j||^2$;
  return assignment

4 EXPERIMENTAL EVALUATION

4.1 DATASETS AND METRICS

We report results on four popular image clustering datasets: CIFAR 10, CIFAR 100, FashionMNIST and STL, with image sizes 32,28 and 96 respectively. We use the standard clustering metrics of accuracy (ACC), normalized mutual information (NMI), and adjusted Rand index (ARI), defined as, e.g., in (Sheng & Huber, 2020). We also report variance of cluster sizes. All datasets have uniform numbers of ground-truth classes, so ideally, this variance should be close to zero.

4.2 CLUSTER SIZES AND CLUSTERING ACCURACY

Table 1 compares our method with three existing methods: sum of squares minimization, denoted "SS" (Zhong et al., 2020), the Sinkhorn-Knopp algorithm for optimal transport, denoted "SK" (Caron et al., 2020; Kumar et al., 2021), and marginal entropy maximization (Li et al., 2020), denoted "Ent". Each is described in Section 2. To make further explicit the phenomenon of partition collapse, we also include a model without any partition support. Our method significantly outperforms others across all datasets and metrics.

Observe that the unregularized model exhibits total collapse in all experiments, placing all points in the same cluster and consequently achieving a cluster performance no better than random guessing. In many of our experiments, the performance of SS is not much better. By making a slight change to the author's original method, we could actually significantly improve its results (see appendix), but it was still unreliable and less accurate than our method. The other two existing partition support methods do a reasonable job of avoiding partition collapse. However, entropy maximization occasionally also reaches the state with all points in the same cluster (this is consistent with previous literature, e.g., (Hu et al., 2017)). The Sinkhorn-Knopp method is more reliable, but by far the most uniform cluster sizes are produced by our method. Note that we do not employ our assignment algorithm at inference time, instead we just assign each point to the cluster with the nearest centroid. This shows that our cluster centroids are well-distributed around the data manifold, each capturing a sizeable subset of the data even when the explicit support is removed. Together, these figures show

Table 1: Effect, on cluster size and clustering performance, of our method compared to two existing methods of preventing partition collapse. "CA" refers to our method of combination assignment, "SK" refers to Sinkhorn-Knopp regularization, as proposed by Caron et al. (2020) and Chen & He (2021), "Ent" refers to entropy maximization, as used by Hu et al. (2017), "SS" is the sum of squares minimization proposed in (Zhong et al., 2020) and others, and "No-Reg" is the model without any partition support component. Along with standard cluster metrics, we report the variance in cluster size (denoted "CVar") to indicate the extent of partition collapse. All figures are the mean of 5 runs, with std dev in parentheses. Best results are in bold.

		CA	SK	Ent	SS	No Reg
Cifar10	Acc	**22.7 (2.07)**	16.7 (0.36)	18.6 (1.36)	11.8 (1.72)	10.0 (0.00)
	NMI	**10.1 (1.68)**	3.8 (0.68)	8.7 (2.63)	1.1 (1.15)	0.0 (0.00)
	ARI	**5.8 (0.93)**	2.7 (0.60)	5.7 (1.76)	0.3 (0.38)	0.0 (0.00)
	CVar	**425 (136)**	8931 (3634)	16240 (2243)	43542 (11424)	54000 (0)
Cifar100	Acc	**6.4 (0.22)**	2.6 (0.21)	2.4 (0.17)	1.2 (0.22)	1.0 (0.00)
	NMI	**13.2 (0.37)**	5.3 (0.41)	6.3 (0.73)	0.6 (1.05)	0.0 (0.00)
	ARI	**1.7 (0.14)**	0.3 (0.04)	0.5 (0.10)	0.0 (0.04)	0.0 (0.00)
	CVar	**1280 (156)**	13451 (4430)	25156 (3283)	55405 (3743)	59400 (0)
FashionMNIST	Acc	**54.5 (6.96)**	25.1 (2.80)	25.5 (5.51)	10.0 (0.04)	10.0 (0.00)
	NMI	**53.2 (4.23)**	17.7 (2.08)	20.9 (10.12)	0.0 (0.04)	0.0 (0.00)
	ARI	**39.1 (6.29)**	9.2 (1.27)	10.6 (5.37)	0.0 (0.00)	0.0 (0.00)
	CVar	**386 (51)**	7087 (3530)	14559 (3203)	53950 (98)	54000 (0)
STL	Acc	**23.5 (1.42)**	15.2 (1.03)	10.9 (1.76)	10.1 (0.20)	10.0 (0.00)
	NMI	**13.7 (1.33)**	2.8 (0.53)	1.3 (2.56)	0.0 (0.08)	0.0 (0.00)
	ARI	**7.1 (0.70)**	1.1 (0.37)	0.2 (0.32)	0.0 (0.00)	0.0 (0.00)
	CVar	**217 (21)**	2319 (466)	11320 (751)	11317 (765)	11700 (0)

that (a) some form of partition support is necessary to learn anything meaningful, (b) our method of combination assignment is better at avoiding partition collapse than previous methods, and (c) this leads to our model producing a better clustering performance.

To better isolate the effect of our proposed partition support method, we do not perform hyperparameter tuning, and use a relatively simple architecture for all datasets. This is the same for all methods being compared. The network consists of two convolutional layers with filter sizes 6 and 16, and ReLU activations, followed by a fully connected layer to a latent space of dimension 128. Training uses Adam, with learning rate 1e-3, $\beta_1 = 0.9$, $\beta_2 = 0.99$, and batch size 256. We follow previous works in setting K, the number of clusters, to the number of ground truth classes.

4.3 Hard vs. Soft Assignment Regularization

A key element of our method is the regularization of hard assignments, whereas previous methods regularize soft cluster assignments. This is a fundamentally different form of regularization, and one which we argue is better able to prevent collapse. For example, assume there are only three clusters and a batch size of four, and consider the following two matrices of assignment probabilities

$$D_1 = \begin{bmatrix} .98 & .01 & .01 \\ .98 & .01 & .01 \\ .49 & .50 & .01 \\ .49 & .01 & .50 \end{bmatrix} \quad D_2 = \begin{bmatrix} .34 & .33 & .33 \\ .34 & .33 & .33 \\ .34 & .33 & .33 \\ .34 & .33 & .33 \end{bmatrix}.$$

The hard and soft entropy (i.e. the entropy of marginal hard and soft assignments respectively) for D_1 are 1.5 and 1.1 respectively, and for D_2 they are 0 and 1.58 respectively. That means D_2 has higher soft entropy than D_1 but a much lower hard entropy. Indeed, despite having nearly maximum soft entropy, D_2 is collapsed with zero hard entropy. A similar scenario is shown for an idealized batch in Figure 1, again revealing that near-uniform soft assignments does not guarantee avoiding collapse, and is in fact quite a different objective.

To investigate whether such differences between hard vs. soft regularization manifest in practice, we empirically measure the variance and entropy of hard and soft assignment probabilities, marginalized across the entire dataset, for each method tested. The results are shown in Table 2.

The most striking difference between hard and soft entropy is in SS. There, the soft entropy is often close to the maximum value (equal to the logarithm of the number of clusters), but the hard entropy

Figure 1: Types of assignments encouraged by different types of regularization, for an idealized batch of size 4, with 5 clusters. **Top**: with no regularization the model collapses, it places most of the probability mass on the same cluster for each data point, and the argmax is the same for each data point. **Middle**: soft regularization forces the soft marginal assignments to be close to uniform, but still the argmax is the same for every data point, so all points are assigned to the same cluster and the model is also collapsed. **Bottom**: regularizing the hard assignments causes the argmax to change. It allows the marginal soft assignment probabilities to deviate from uniform, and instead encourages uniformity in the number assigned to each cluster. This model avoids collapse. In order to make the figure more readable, we cut off the top and middle graphs at 80% and 40% respectively. On full graphs, the hard marginal for cluster 0 would extend up to 100%.

Table 2: Comparison of the hard variance (HV) and entropy (HE), and soft variance (S) and soft entropy (SE). Previous methods regularize the soft assignments, but this does not transfer well to the hard assignments. Ours (CA) is the only method that produces low variance and high entropy for both hard and soft assignments. Best results in bold.

		CA (ours)	SK	Ent	SS
Cifar10	hard variance	**424**	8931	16240	43542
	soft variance	407	**0**	13141	338
	hard entropy	**3.27**	2.51	1.81	0.93
	soft entropy	3.27	**3.32**	2.10	2.58
Cifar100	hard variance	**1280**	13450	25155	55405
	soft variance	190	**0**	4117	120
	hard entropy	**5.44**	3.52	1.63	0.19
	soft entropy	6.47	**6.63**	4.60	6.51
FashionMNIST	hard variance	**386**	7087	14559	53950
	soft variance	368	**0**	10051	376
	hard entropy	**3.27**	2.65	1.89	0.01
	soft entropy	3.27	**3.32**	2.32	3.07
STL	hard variance	**217**	2319	11320	11316
	soft variance	194	**0**	3380	524
	hard entropy	**3.19**	2.31	0.08	0.08
	soft entropy	3.21	**3.32**	1.99	3.06

is consistently close to zero. This shows that this form of regularization produces batch assignment probabilities similar to matrix D_2 above, where the probabilities for each data point are squeezed close to one another, without much change in the order of highest to lowest, in particular the argmax.

A similar discrepancy is found in SK, which produces near-perfect uniformity in the soft assignments, with a variance of zero and the maximum possible entropy (up to rounding) on each dataset. This is because it is a (close approximation to a) hard constraint problem. However, SK's hard assignments still show significant variability, and markedly lower entropy than the soft assignments. This suggests that applying the SK algorithm also produces batch assignment probabilities somewhat similar to D_2 above. Our method, on the other hand, explicitly forces the argmax to be more evenly distributed during training and, as Table 2 shows, this transfers to the testing setting as well. (Recall that, during testing, we simply assign each point to the cluster with the nearest centroid.) Our hard entropy is only slightly lower than our soft entropy, and is consistently higher than that of the other three methods. This supports our argument that the mean soft assignments do not contain sufficient information to determine if the clustering model is learning a meaningful partition, and regularizing this quantity is not an optimal way to prevent collapse. The pattern in the argmax is also important and is lost if we look only at the mean soft assignment across a batch.

Our analysis here of hard vs soft cluster assignments does not contradict (Caron et al., 2020). They report better results using soft assignments as training labels, while we show the superiority of *regularizing*, i.e., encouraging equal numbers of, hard assignments. The two are different contexts of hard and soft labels. We also explored training SK (the method used by Caron et al. (2020)) using soft assignments as targets, but the results were slightly worse than using hard targets.

5 Conclusion

This paper proposed a data-augmentation-free method to prevent collapse in online deep clustering. We frame probabilistically the problem of deciding which clusters to assign a batch of data points to, given the cluster centroids and features of the data points, and we use this framing to derive a concise optimization objective for making hard cluster assignments. We then describe an algorithm to approximately solve this optimization problem and demonstrate empirically on four datasets that this method outperforms existing methods, both in better preventing collapse and in leading to better clustering performance. Finally, we analyze how the cluster assignment distribution is affected by our partition support method and previous comparable methods. The analysis suggests that regularizing the soft assignments, as is done by existing works, is not sufficient to prevent collapse, and that a better approach is to regularize the hard assignments, as is done by our method.

REFERENCES

Yuki Markus Asano, Christian Rupprecht, and Andrea Vedaldi. Self-labelling via simultaneous clustering and representation learning. *arXiv preprint ArXiv:1911.05371*, 2019.

Dibya Jyoti Bora, Dr Gupta, and Anil Kumar. A comparative study between fuzzy clustering algorithm and hard clustering algorithm. *arXiv preprint ArXiv:1404.6059*, 2014.

Yaoming Cai, Zijia Zhang, Yan Liu, Pedram Ghamisi, Kun Li, Xiaobo Liu, and Zhihua Cai. Large-scale hyperspectral image clustering using contrastive learning. *arXiv preprint ArXiv:2111.07945*, 2021.

Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *arXiv preprint ArXiv:2006.09882*, 2020.

Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15750–15758, 2021.

Aniket Anand Deshmukh, Jayanth Reddy Regatti, Eren Manavoglu, and Urun Dogan. Representation learning for clustering via building consensus. *arXiv preprint arXiv:2105.01289*, 2021.

Boyan Gao, Yongxin Yang, Henry Gouk, and Timothy M. Hospedales. Deep clusteringwith concrete k-means. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 4252–4256. IEEE, 2020.

Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in Neural Information Processing Systems*, 33:21271–21284, 2020.

Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In *Proceedings of the International Conference on Machine Learning*, pp. 1558–1567. PMLR, 2017.

Jiabo Huang and Shaogang Gong. Deep clustering by semantic contrastive learning. *arXiv preprint ArXiv:2103.02662*, 2021.

Michael Kearns, Yishay Mansour, and Andrew Y. Ng. An information-theoretic analysis of hard and soft assignment methods for clustering. In *Learning in graphical models*, pp. 495–520. Springer, 1998.

Prakhar Kulshreshtha and Tanaya Guha. An online algorithm for constrained face clustering in videos. In *Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 2670–2674. IEEE, 2018.

Sateesh Kumar, Sanjay Haresh, Awais Ahmed, Andrey Konin, M. Zeeshan Zia, and Quoc-Huy Tran. Unsupervised activity segmentation by joint representation learning and online clustering. *arXiv preprint ArXiv:2105.13353*, 2021.

Junnan Li, Pan Zhou, Caiming Xiong, and Steven CH Hoi. Prototypical contrastive learning of unsupervised representations. *arXiv preprint ArXiv:2005.04966*, 2020.

Louis Mahon and Thomas Lukasiewicz. Selective pseudo-label clustering. In *Proceedings of the German Conference on Artificial Intelligence (Künstliche Intelligenz)*, pp. 158–178. Springer, 2021.

Chuang Niu, Jun Zhang, Ge Wang, and Jimin Liang. Gatcluster: self-supervised gaussian-attention network for image clustering. In *Proceedings of the European Conference on Computer Vision*, pp. 735–751. Springer, 2020.

Chuang Niu, Hongming Shan, and Ge Wang. Spice: Semantic pseudo-labeling for image clustering. *arXiv preprint ArXiv:2103.09382*, 2021.

Taoran Sheng and Manfred Huber. Unsupervised embedding learning for human activity recognition using wearable sensor data. In *Proceedings of the The Thirty-Third International Flairs Conference*, 2020.

Jonathan A. Silva, Elaine R. Faria, Rodrigo C. Barros, Eduardo R. Hruschka, André CPLF de Carvalho, and João Gama. Data stream clustering: A survey. *ACM Computing Surveys (CSUR)*, 46 (1):1–31, 2013.

Wouter Van Gansbeke, Simon Vandenhende, Stamatios Georgoulis, Marc Proesmans, and Luc Van Gool. Scan: learning to classify images without labels. In *Proceedings of the European Conference on Computer Vision*, pp. 268–285. Springer, 2020.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: self-supervised learning via redundancy reduction. In *Proceedings of the International Conference on Machine Learning*, pp. 12310–12320. PMLR, 2021.

Xiaohang Zhan, Jiahao Xie, Ziwei Liu, Yew-Soon Ong, and Chen Change Loy. Online deep clustering for unsupervised representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6688–6697, 2020.

Dejiao Zhang, Feng Nan, Xiaokai Wei, Shangwen Li, Henghui Zhu, Kathleen McKeown, Ramesh Nallapati, Andrew Arnold, and Bing Xiang. Supporting clustering with contrastive learning. *arXiv preprint ArXiv:2103.12953*, 2021.

Huasong Zhong, Chong Chen, Zhongming Jin, and Xian-Sheng Hua. Deep robust clustering by contrastive learning. *arXiv preprint ArXiv:2008.03030*, 2020.