Improving Unsupervised Image Clustering With Robust Learning

Sungwon Park∗1,2  Sungwon Han∗1,2  Sundong Kim2  Danu Kim1,2  Sungkyu Park2  Seunghoon Hong1  Meeyoung Cha2,1
1School of Computing, KAIST  2Data Science Group, Institute for Basic Science

Abstract

Unsupervised image clustering methods often introduce alternative objectives to indirectly train the model and are subject to faulty predictions and overconfident results. To overcome these challenges, the current research proposes an innovative model RUC that is inspired by robust learning. RUC’s novelty is at utilizing pseudo-labels of existing image clustering models as a noisy dataset that may include misclassified samples. Its retraining process can revise misaligned knowledge and alleviate the overconfidence problem in predictions. The model’s flexible structure makes it possible to be used as an add-on module to other clustering methods and helps them achieve better performance on multiple datasets. Extensive experiments show that the proposed model can adjust the model confidence with better calibration and gain additional robustness against adversarial noise.

1. Introduction

Unsupervised clustering is a core task in computer vision that aims to identify each image’s class membership without using any labels. Here, a class represents the group membership of images that share similar visual characteristics. Many studies have proposed deep learning-based algorithms that utilize distance in a feature space as the similarity metric to assign data points into classes [11, 44]. Training without ground-truth guidance, however, is prone to finding trivial solutions that are learned from low-level visual traits like colors and textures [22]. Several studies have introduced innovative ways to guide the model’s training indirectly by setting alternative objectives. For example, Hu et al. [20] proposed to maximize the mutual information between input and its hidden representations, and Ji et al. [22] proposed to learn invariant features against data augmentation. Entropy-based balancing has often been adopted to prevent degenerate solutions [17, 22, 42].

Nevertheless, these alternative objectives are bound to producing overconfident results, i.e., low-entropy predictions, due to the dense grouping among clusters. When uncertain samples are added to a wrong cluster at an early stage of training, the model gradually becomes overconfident in its later predictions as the noise from misclassification accumulates and degrades the overall performance.

This paper introduces a novel robust learning training method, RUC (Robust learning for Unsupervised Clustering), that runs in conjunction with existing clustering models to alleviate the noise discussed above. Utilizing and treating the existing clustering model’s results as a noisy dataset that may include wrong labels, RUC updates the model’s misaligned knowledge. Bringing insights from the literature, we filter out unclean samples and apply loss correction as in Fig. 1. This process is assisted by label smoothing and co-training to reduce any wrong gradient signals from unclean labels. This retraining process with revised pseudo-labels further regularizes the model and prevents overconfident results.

RUC comprises two key components: (1) extracting clean samples and (2) retraining with the refined dataset. We propose confidence-based, metric-based, and hybrid strategies to filter out misclassified pseudo-labels. The first strategy considers samples of high prediction confidence from the original clustering model as a clean set; it filters out low confidence samples. This strategy relies on the
model’s calibration performance. The second strategy utilizes similarity metrics from unsupervised embedding models to detect clean samples with non-parametric classifiers by checking whether the given instance shares the same labels with top $k$-nearest samples. The third strategy combines the above two and selects samples that are credible according to both strategies.

The next step is to retrain the clustering model with the sampled dataset. We use MixMatch [5], a semi-supervised learning technique, which uses clean samples as labeled data and unclean samples as unlabeled data. We then adopt label smoothing to leverage strong denoising effects on the label noise [29] and block learning from overconfident samples [22, 42]. Finally, a co-training architecture with two networks is used to mitigate noise accumulation from the unclean samples during training and increase performance.

We evaluate RUC with rigorous experiments on datasets, including CIFAR-10, CIFAR-20, STL-10, and ImageNet-50. Combining RUC to an existing clustering model outperforms the state-of-the-art results with the accuracy of 90.3% in CIFAR-10, 54.3% in CIFAR-20, 86.7% in STL-10, and 78.5% in ImageNet-50 dataset. RUC also enhances the baseline model to be robust against adversarial noise. Our contributions are as follows:

- The proposed algorithm RUC aids existing unsupervised clustering models via retraining and avoiding overconfident predictions.
- The unique retraining process of RUC helps existing models boost performance. It achieves a 5.3pp increase for the STL-10 dataset when added to the state-of-the-art model (81.4% to 86.7%).
- The ablation study shows every component in RUC is critical, including the three proposed strategies (i.e., confidence-based, metric-based, and hybrid) that excel in extracting clean samples from noisy pseudo-labels.
- The proposed training process is robust against adversarial noise and can adjust the model confidence with better calibrations.

Implementation details of the model and codes are available at https://github.com/deu30303/RUC.

### 2. Related Work

#### 2.1. Unsupervised Image Clustering

The main objective of clustering is to group the data points into distinct classes of similar traits [21]. Most real-world problems deal with high dimensional data (e.g., images), and thereby, setting a concrete notion of similarity while extracting low-dimensional features becomes key components for setting appropriate standards for grouping [49]. Likewise, unsupervised clustering is a line of research aiming to tackle both dimensionality reduction and boundary identification over the learned similarity metric [17]. Existing research can be categorized into sequential, joint, and multi-step refinement approach.

**Sequential approach.** Sequential approach extracts features, then sequentially applies the conventional distance or density-based clustering algorithm for class assignments. For example, Ding et al. [11] use principal component analysis to extract low-dimensional features and then apply $k$-means clustering to assign classes. For feature extraction, autoencoder structures are often used to extract latent features before grouping, types of autoencoder include stacked [44], boolean [3], or variational autoencoder [24]. However, these models tend to produce features with little separation among clusters due to the lack of knowledge on subsequent assignment processes.

**Joint approach.** The joint approach’s characteristic is to use an end-to-end pipeline that concurrently performs feature extraction and class assignment. An example is Yang et al. [51], which adopt the concept of clustering loss to guarantee enough separations among clusters. End-to-end CNN pipelines are used widely to iteratively identify clusters while refining extracted features [6, 8, 49]. Recent studies have shown that a mutual information-based objective is an effective measure to improve classification accuracy [20, 22]. Nonetheless, those models still bear the problem of generating unintended solutions that depend on trivial low-level features from random initialization [17].

**Multi-step refinement approach.** To mitigate the unintended trivial solutions, recent approaches leverage the power of unsupervised embedding learning models to provide better initialization for downstream clustering tasks [9, 48, 50]. These methods generate feature representations to gather data points with similar visual traits and push away the rest in an embedding space. With the initialization, clustering results are elaborated in a refinement step, bringing significant gain in its class assignment quality [17, 42]. In particular, SCAN [42] first obtains high-level feature representations by feature similarity then clusters those representations by nearest neighbors, and this model has shown remarkable performance on unsupervised clustering.

**Add-on modules to improve unsupervised clustering.** The proposed retraining process with sample selection strategy improves off-the-shelf unsupervised clustering algorithms (e.g., sequential, joint, multi-step refinement) by acting as an add-on module. Our module’s main objective is to revise the misaligned knowledge of trained clustering models via label cleansing and retraining with the refined labels. This method has not been well investigated before but has begun to be proposed recently. Gupta et al. [14] show that semi-supervised retraining improves unsupervised clustering. They draw a graph where data samples are nodes, and the confidence from ensemble models
between the samples is an edge. Then, a dense sub-graph is considered as a clean set.

The main difference between Gupta et al. and ours is in how we treat the pseudo-labels obtained by the clustering. Gupta et al. treats the pseudo-label as a ground-truth for semi-supervised learning, which produces sub-optimal result if the pseudo-label is noisy (i.e., memorization). In contrast, we introduce the robust learning concept of label smoothing and co-training to mitigate the memorization of noisy samples, which leads to substantial improvements in the calibration and clustering performance.

2.2. Robust Learning With Label Noise

A widely used setting for robust learning is where an adversary has deliberately corrupted the labels, which otherwise arise from some clean distribution [33, 40]. According to the literature, deep networks easily overfit to the label noise during training and get a low generalization power [28]. In this light, models that prevent overfitting in a noise label environment have been studied.

Loss correction. The first representative line of work is a loss correction, which relabels unclean samples explicitly or implicitly. For example, Patrini et al. [35] estimate the label transition probability matrix to correct the loss and retrain the model. To estimate the transition matrix more accurately, the gold loss correction approach [19] is proposed to utilize trusted labels as additional information.

Loss reweighting. The next line of work is loss reweighting, which aims to give a smaller weight to the loss of unclean samples so that model can reduce the negative effect of label noise during training. One work computes the importance as an approximated ratio of two data distributions; clean and unclean [46]. On the other hand, the active bias approach [7] calculates the inconsistency of predictions during training and assigns a weight to penalize unclean data.

Sample selection. Relabeling the misclassified samples may cause a false correction. In this context, recent works introduce a sample selection approach that filters out misclassified samples and only selects clean data for training [30, 32]. Notably, the small loss trick, which regards the sample with small training loss as clean, effectively separates true- and false-labeled data points [2, 23, 28]. Also, recent studies suggest diverse ways to lead additional performance by maintaining two networks to avoid accumulating sampling bias [16, 54], adopting refurbishment of false-labeled samples [39], or using a semi-supervised approach to utilize false-labeled sample maximally [27]. Our model advances some of these sample selection approaches to filter out unclean samples out of clustering results and utilize clean samples only during retraining.

3. Method

RUC is an add-on method that can be used in conjunction with the existing unsupervised clustering methods to refine mispredictions. Its key idea is at utilizing the initial clustering results as noisy pseudo-labels and learning to refine them with a mild clustering assumption [41] and techniques from the robust learning [27, 29].

Figure 2 and Algorithm 1 illustrate the overall pipeline of the proposed algorithm. Given the initial pseudo-labels, we first divide the training data into the two disjoint sets: clean and unclean (Section 3.1). Then treating these sets each as labeled and unlabeled data, we train a classifier in a semi-supervised manner while refurbishing the labeled and unlabeled data (Section 3.2). We guide the semi-supervised class assignment with robust learning techniques, such as co-training and label smoothing, to account for inherent label noises. These techniques are useful in handling label noises and calibrating the model’s prediction score. Below we describe the model in details.

3.1. Extracting Clean Samples

Let $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ denote the training data, where $x_i$ is an image and $y_i = g_\theta(x_i)$ is a pseudo-label from
an unsupervised classifier $g_\psi$. The model first divides the pseudo-labels into two disjoint sets as $D = X \cup U$ with a specified sampling strategy. We consider $X$ as clean, whose pseudo-labels are moderately credible and thus can be used as a labeled dataset $(x, y) \in X$ for refinement. In contrast, we consider $U$ as unclean, whose labels we discard $u \in U$. Designing an accurate sampling strategy is not straightforward, as there is no ground-truth to validate the pseudo-labels directly. Inspired by robust learning’s clean set selection strategy, we explore three different approaches: (1) confidence-based, (2) metric-based, and (3) hybrid.

Confidence-based strategy. This approach selects clean samples based on the confidence score of the unsupervised classifier. Given a training sample $(x, y) \in D$, we consider the pseudo-label $y$ is credible if $\max(y) > \tau_1$, and add it to the clean set $X$. Otherwise, it is assigned to $U$. This is motivated by the observation that the unsupervised classifier tends to generate overconfident predictions; thus, we trust only the most typical examples from each class while ignoring the rest. The threshold $\tau_1$ is set substantially high to eliminate as many uncertain samples as possible.

Metric-based strategy. The limitation of the above approach is that the selection strategy still entirely depends on the unsupervised classifier. The metric-based approach leverages an additional embedding network $h_\psi$ learned in an unsupervised manner (e.g., SimCLR [9]) and measures the credibility of the pseudo-label based on how well it coincides to the classification results using $h_\psi$. For each $(x, y) \in D$, we compute its embedding $h_\psi(x)$, and apply the non-parametric classifier based on $k$-Nearest Neighbor ($k$-NN) by $y' = k$-NN($h_\psi(x)$). We consider that the pseudo-label is credible if $\arg\max(y) = \arg\max(y')$ and add it to the clean set $X$. Otherwise, it is assigned to the unclean set $U$.

Hybrid strategy. This approach will add a sample to the clean set $X$ only if it is considered credible by both the confidence-based and metric-based strategies. All other samples are added to $U$.

3.2. Retraining via Robust Learning

Given the clean set $X$ and the unclean set $U$, our next step aims to train the refined classifier $f_\theta$ that revises incorrect predictions of the initial unsupervised classifier.

Vanilla semi-supervised learning. A naive baseline is to consider $X$ as labeled data and $U$ as unlabeled data each to train a classifier $f_\theta$ using semi-supervised learning techniques. We utilize MixMatch [5] as such a baseline, which is a semi-supervised algorithm that estimates low-entropy mixed labels from unlabeled examples using MixUp augmentation [56].

Note that our method is not dependent on the particular choice of the semi-supervised learning method and can incorporate the others.
bring additional resistance against noisy labels since a large amount of extra virtual examples from MixUp interpolation makes memorization hard to achieve [27, 56]. Specifically, given a pair of labeled data \((x_1, y_1)\) and \((x_2, y_2)\) sampled from either labeled or unlabeled data, it augments the data using the following operations.

\[
\lambda \sim \text{Beta}(\alpha, \alpha) \quad (1) \\
X' = \max(\lambda, 1 - \lambda) \quad (2) \\
x' = \lambda' x_1 + (1 - \lambda') x_2 \quad (3) \\
y' = \lambda' y_1 + (1 - \lambda') y_2. \quad (4)
\]

In the case of unlabeled data \(u \in \mathcal{U}\), MixMatch is employed such that a surrogate label \(y = q\) is obtained by averaging the model’s predictions over multiple augmentations after sharpening [5]. Later, we will show that using the labels \(y\) and \(q\) directly in semi-supervised learning leads to a sub-optimal solution and digest how it improves robustness.

For \(X\) and \(U\) after MixMatch (Eq. (5)), a vanilla semi-supervised learning model trains with two separate losses: the cross-entropy loss for the labeled set \(X\) (Eq. (6)), and the consistency regularization for the unlabeled set \(U\) (Eq. (7)).

\[
H(p, q) \text{ denotes the cross-entropy between } p \text{ and } q. 
\]

\[
\hat{X}, \hat{U} = \text{MixMatch}(X, U) \\
\mathcal{L}_{\hat{X}} = \frac{1}{|X|} \sum_{x, \hat{y} \in \hat{X}} H(\hat{y}, f_{\theta}(\hat{x})) \quad (6) \\
\mathcal{L}_{\hat{U}} = \frac{1}{|U|} \sum_{u, \hat{q} \in \hat{U}} ||\hat{q} - f_{\theta}(\hat{u})||_2^2 \quad (7)
\]

**Label smoothing.** To regularize our model from being overconfident to noisy predictions, we apply label smoothing along with vanilla semi-supervised learning. Label smoothing prescribes soft labels by adding uniform noise and improves the calibration in predictions [29]. Given a labeled sample with its corresponding label \((x, y) \in X\), we inject uniform noise into all classes as follows:

\[
\hat{y} = (1 - \epsilon) \cdot y + \frac{\epsilon}{(C - 1)} \cdot (1 - y) \quad (8)
\]

where \(C\) is the number of class and \(\epsilon \sim \text{Uniform}(0, 1)\) is the noise. We compute cross-entropy using the soft label \(\hat{y}\) and the predicted label of the strongly augmented sample \(\phi_A(x)\) via RandAugment [10]. We find that strong augmentations minimize the memorization from noise samples.

\[
\mathcal{L}_{X'} = \frac{1}{|X'|} \sum_{x, \hat{y} \in X'} H(\hat{y}, f_{\theta}(\phi_A(x))) \quad (9)
\]

Our final objective for training can be written as:

\[
\mathcal{L}(\theta; X, \hat{X}, \hat{U}) = \mathcal{L}_{X'} + \mathcal{L}_{\hat{X}} + \lambda_{\mathcal{U}} \mathcal{L}_{\hat{U}}. \quad (10)
\]

where \(\lambda_{\mathcal{U}}\) is a hyper-parameter to control the effect of the unsupervised loss in MixMatch.

**Co-training.** Maintaining a single network for learning has a vulnerability of overfitting to incorrect pseudo-labels since the initial error from the network is transferred back again, and thereby, accumulated [16]. To avoid this fallacy, we additionally introduce a co-training module where the two networks \(f_{\theta(1)}, f_{\theta(2)}\) are trained in parallel and exchange their guesses for teaching each other by adding a co-refinement step on top of MixMatch.

Co-refinement is a label refinement process that aims to produce reliable labels by incorporating both networks’ predictions. Following the previous literature [27], we apply co-refinement both on the label set \(X\) and the unlabeled set \(U\) for each network. Here, we explain the co-refinement process from the perspective of \(f_{\theta(1)}\). For the labeled data point \(x\), we calculate the linear sum between the original label \(y\) in \(X\) and the prediction from the counter network \(f_{\theta(2)}\) (Eq. (11)) and apply sharpening on the result to generate the refined label \(\bar{y}\) (Eq. (12)).

\[
\bar{y} = (1 - w^{(2)}) \cdot y + w^{(2)} \cdot f_{\theta(2)}(x) \quad (11)
\]

\[
\bar{y} = \text{Sharpen}(\bar{y}, T), \quad (12)
\]

where \(w^{(2)}\) is the counter network’s confidence value of \(x\), and \(T\) is the sharpening temperature. For the unlabeled set \(U\), we apply an ensemble of both networks’ predictions to guess the pseudo-label \(q\) of data sample \(u\) as follows:

\[
\bar{q} = \frac{1}{2M} \sum_{m} (f_{\theta(1)}(u_m) + f_{\theta(2)}(u_m)) \quad (13)
\]

\[
\bar{q} = \text{Sharpen}(\bar{q}, T), \quad (14)
\]

where \(u_m\) is \(m\)-th weak augmentation of \(u\).

In place of \(X\) and \(U\), co-refinement produces the refined dataset \((x, \bar{y}) \in \hat{X}^{(1)}\), and \((u, \bar{q}) \in \hat{U}^{(1)}\) through Eq. (11) to (14). We utilize those datasets as an input for MixMatch, and the model is eventually optimized as follows:

\[
\hat{X}^{(1)}, \hat{U}^{(1)} = \text{Co-refinement}(X, U, \theta^{(1)}, \theta^{(2)}) \quad (15)
\]

\[
\hat{X}^{(1)}, \hat{U}^{(1)} = \text{MixMatch}(\hat{X}^{(1)}, \hat{U}^{(1)}) \quad (16)
\]

\[
\theta^{(1)} \leftarrow \arg \min_{\theta^{(1)}} \mathcal{L}(\theta^{(1)}; X, \hat{X}^{(1)}, \hat{U}^{(1)}), \quad (17)
\]

where \(\mathcal{L}\) is the loss defined in Eq. (10). This process is also conducted for \(f_{\theta(2)}\) in the same manner.

**Co-refurbishing.** Lastly, we refurbish the noisy samples at the end of every epoch to deliver the extra clean samples across the training process. If at least one of the networks’ confidence on the given unclean sample \(u \in U\) is over the threshold \(\tau_2\), the corresponding sample’s label is updated with the network’s prediction \(p\). The updated sample is then regarded as clean and appended to the labeled set \(\hat{X}\).

\[
p = f_{\theta(k)}(u), \text{ where } k = \arg \max_{k} (\max(f_{\theta(k)}(u))) \quad (18)
\]

\[
\hat{X} \leftarrow X \cup \{(u, 1_p) \mid \max(p) > \tau_2\}, \quad (19)
\]
where \( \mathbf{1}_p \) is a one-hot vector of \( p \) whose \( i \)-th element is 1, considering \( i = \arg \max(p) \).

4. Experiments

For evaluation, we first compared the performance of our model against other baselines over multiple datasets. Then, we examined each component’s contribution to performance improvement. Lastly, we investigated how RUC helps improve existing clustering models in terms of their confidence calibration and robustness against adversarial attacks.

4.1. Unsupervised Image Clustering Task

**Settings.** Four benchmark datasets were used. The first two are CIFAR-10 and CIFAR-100, which contain 60,000 images of 32x32 pixels. For CIFAR-100, we utilized 20 superclasses following previous works [42]. The next is STL-10, containing 100,000 unlabeled images and 13,000 labeled images of 96x96 pixels. For the clustering problem, only 13,000 labeled images were used. Lastly, we test the model with the large-scale ImageNet-50 dataset, containing 65,000 images of 256x256 pixels.

Our model employed the ResNet18 [18] architecture following other baselines [17,22,42] and the model was trained for 200 epochs. Initial confidence threshold \( \tau_1 \) was set as 0.99, and the number of neighbors \( k \) to divide the clean and noise samples was set to 100. The threshold \( \tau_2 \) for refurbishing started from 0.9 and increased by 0.02 in every 40 epochs. The label smoothing parameter \( \epsilon \) was set to 0.5. For evaluating the class assignment, the Hungarian method [25] was used to map the best bijection permutation between the predictions and ground-truth.

**Result.** Table 1 shows the overall performance of clustering algorithms over three datasets: CIFAR-10, CIFAR-20, and STL-10. For these datasets, the proposed model RUC, when applied to the SCAN [42] algorithm, outperforms all other baselines. Particularly for STL-10, the combined model shows a substantial improvement of 5.3 pp. Table 2 reports ImageNet-50 result on the confidence based sampling strategy, which demonstrates RUC’s applicability to large-scale dataset. Furthermore, RUC achieves consistent performance gain over another clustering model, TSUC [17]. These results confirm that our model can be successfully applied to existing clustering algorithms and improve them. We also confirm that all three selection strategies (i.e., confidence-based, metric-based, and hybrid) bring considerable performance improvement.

4.2. Component Analyses

To evaluate the model’s efficacy, we conduct an ablation study by repeatedly assessing its performance after removing each component. We also evaluate the accuracy of different selection and refurbishing strategies based on precision, recall, and the F1 score.

**Ablation study.** The proposed model utilizes two robust learning techniques to cope with unclean samples: co-training and label smoothing. We remove each component from the full model to assess its efficacy. Table 3 shows the classification accuracy of each ablation on the STL-10 dataset. RUC with all components performs the best, implying that dropping any component results in performance degradation. We also compare a variant, which drops both label smoothing and co-training (i.e., MixMatch only). The effect of co-training is not evident in Table 3. Nevertheless, it improved the performance from 36.3% to 39.6% for the lowest noise ratio CIFAR-20 pseudo-labels when we set
Figure 3: Visualization of clustering results on STL-10. The leftmost figure shows the results from the SCAN, while the right three figures display the intermediate results of our model on top of SCAN at different training epochs. This result demonstrates that RUC effectively alleviates the overconfident predictions while enhancing the clustering quality.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>CIFAR-10</th>
<th>CIFAR-20</th>
<th>STL-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCAN</td>
<td>88.7</td>
<td>50.6</td>
<td>81.4</td>
</tr>
<tr>
<td>SCAN + Gupta et al.</td>
<td>88.3 / 89.5</td>
<td>53.2 / 53.3</td>
<td>84.2 / 84.3</td>
</tr>
<tr>
<td>SCAN + DivideMix</td>
<td>86.5 / 87.9</td>
<td>53.5 / 53.6</td>
<td>80.6 / 83.9</td>
</tr>
<tr>
<td>SCAN + M-correction</td>
<td>81.6 / 88.6</td>
<td>48.5 / 50.4</td>
<td>81.1 / 81.3</td>
</tr>
<tr>
<td>SCAN + P-correction</td>
<td>87.7 / 88.7</td>
<td>49.5 / 50.5</td>
<td>81.2 / 81.4</td>
</tr>
<tr>
<td>SCAN + RUC (ours)</td>
<td>90.1 / 90.1</td>
<td>54.3 / 54.5</td>
<td>86.6 / 86.7</td>
</tr>
</tbody>
</table>

Table 4: Performance comparison with other possible add-on modules (Last / Best accuracy)

As an alternative of RUC, one may combine the extant robust learning algorithms (e.g., M-correction [1], P-correction [53], and DivideMix [27]) or another previously proposed add-on module (e.g., Gupta et al. [14]) on top of SCAN [42]. Table 1 summarizes the comparisons to four baselines. For a fair comparison, we employed SCAN as the initial clustering method and applied each baseline on top of it. As shown in the results, improving noisy clustering is non-trivial as some baselines show even worse results after the refinement (e.g., DivideMix, M-correction, P-correction). While Gupta et al. effectively refines the results, its improvement is limited when the initial clustering is reasonably accurate (e.g., CIFAR-10). In contrast, RUC achieves consistent improvements in all datasets with non-trivial margins, showing that a carefully designed robust learning strategy is critical to the performance gain.

4.3. In-Depth Performance Analysis

So far, we showed that RUC improves existing baselines substantially, and its components contribute to the performance gain. We now examine RUC’s calibration effect and present a qualitative analysis by applying it to the state-of-the-art baseline model, SCAN [42]. We will also demonstrate the role of RUC in handling adversarially crafted noise.

Confidence calibration. Many unsupervised clustering algorithms are subject to overconfident results due to their entropy-based balancing [17, 42]. If a model is overconfident to noisy samples, separating the clean and unclean set becomes challenging, and this can induce overall performance degradation. Figure 4 shows the calibration effect of RUC. SCAN’s confidence is highly concentrated near 1, while our model’s confidence is widely distributed over [0.6, 0.8]. We also report the degree of calibration quality using Expected Calibration Error (ECE) [13]:

\[
ECE = \frac{1}{n} \sum_{m=1}^{M} |B_m| \left( \frac{\text{acc}(B_m) - \text{conf}(B_m)}{M} \right),
\]
where \( n \) is the number of data, \( B_m \) is the \( m \)-th group from equally spaced buckets based on the model confidence over the data points; \( \text{acc}(B_m) \) and \( \text{conf}(B_m) \) are the average accuracy and confidence over \( B_m \). Lower ECE of RUC in Figure 4 implies that our approach led to better calibrations.

To observe this effect more clearly, we visualize the clustering confidence result at different training epochs in Figure 3. Unlike the result of SCAN in which the overly clustered sample and the uncertain sample are mixed, the result of SCAN+RUC shows that the sample’s class distribution has become smoother, and uncertain samples disappeared quickly as training continues.

**Qualitative analysis.** We conducted a qualitative analysis to examine how well RUC corrects the initial misclassification in pseudo-labels. Figure 5 compares the confusion matrices of SCAN and the SCAN+RUC for STL-10. A high concentration of items on the diagonal line confirms the advanced correction effect of RUC for every class. Figure 6 compares how the two models interpreted class traits based on the Grad-CAM [38] visualization on example images. The proposed model shows a more sophisticated prediction for similar images.

**Robustness to adversarial noise.** Clustering algorithms like SCAN introduce pseudo-labels to train the model via the Empirical Risk Minimization (ERM) method [43]. ERM is a learning principle that minimizes the averaged error over the sampled training data (i.e., empirical risk) to find the model with a small population risk (i.e., true risk). However, ERM is known to be vulnerable to adversarial examples, which are crafted by adding visually imperceptible perturbations to the input images [31, 56].

Here, we show that RUC improves robustness against the adversarial noise. We conduct an experiment on STL-10 using adversarial perturbations of FGSM [12] and BIM [26] attacks, whose directions are aligned with the gradient of the loss surface of given samples. Figure 7 compares the model’s ability to handle the adversarial noise. Models based on MixMatch (Gupta et al., DivideMix, RUC) out-perform the rest, probably because the calibration effect of MixUp prevents overconfident predictions. Among them, RUC achieves superior improvement, demonstrating that robust learning components, such as careful filtering, label smoothing, and co-training, can also handle the adversarial noise (see the supplementary material for further details).

**5. Conclusion**

This study presented RUC, an add-on approach for improving existing unsupervised image clustering models via robust learning. Retraining via robust training helps avoid overconfidence and produces more calibrated clustering results. As a result, our approach achieved a meaningful gain on top of two state-of-the-art clustering methods. Finally, RUC helps the clustering models to be robust against adversarial attacks. We expect robust learning will be a critical building block to advance real-world clustering solutions.

**Acknowledgements.** This work was supported in part by the IBS (IBS-R029-C2) and the Basic Science Research Program through the NRF funded by the Ministry of Science and ICT in Korea (No. NRF-2017R1E1A1A01076400), Institute of Information & communications Technology Planning & Evaluation (IITP) grant (2020-0-00153 and 2016-0-00464), Samsung Electronics, HPC support funded by MSIT & NIPA.
References


